Ф. ГИЛЛ
У. МЮРРЕЙ
М. РАЙТ

ПРАКТИЧЕСКАЯ ОПТИМИЗАЦИЯ

Ф. ГИЛЛ, У. МЮРРЕЙ,
М. РАЙТ

ПРАКТИЧЕСКАЯ ОПТИМИЗАЦИЯ

ПЕРЕВОД С АНГЛИЙСКОГО
В. Ю. Лебедева

ПОД РЕДАКЦИЕЙ
А. А. Петрова

МОСКВА «МИР» 1985

PRACTICAL OPTIMIZATION

Philip E. Gill
Walter Murray
Margaret H. Wright

Systems Optimization Laboratory
Department of Operations Research
Stanford University
California, USA

Academic Press
A Subsidiary of Harcourt Brace Jovanovich, Publishers
London New York Toronto Sydney San Francisco
1981

ББК 22.143
Г 47
УДК 681.3

Гилл Ф., Мюррей У., Райт М.

Г 47 Практическая оптимизация: Пер. с англ.— М.: Мир, 1985.—509 с., ил

Книга американских специалистов, знакомых советскому читателю по переводу «Численных методов условной оптимизации» (М.: Мир, 1977), представляет собой пособие по математическому программированию. Авторы тщательно отобрали и изложили только те алгоритмы, которые эффективны при решении практических задач.

Для математиков-прикладников, научных работников, специалистов, студентов, изучающих или применяющих в своей работе оптимизационные методы.

Г $\frac{2405000000-086}{041\ (01)-85}$ 6-85, ч. I

ББК 22.143
517.1

Редакция литературы по математическим наукам

© 1981 by Academic Press Inc. (London) Ltd.
© Перевод на русский язык, «Мир», 1985

Читатель знаком с Ф. Гиллом и У. Мюрреем не только по их многочисленным статьям в научных журналах. Они редактировали сборник трудов конференции по методам условной оптимизации, проведенной Национальной физической лабораторией (Великобритания, Тэддингтон) в январе 1974 года. Сборник был переведен на русский язык [1]. Это был обзор тогдашнего состояния численных методов отыскания экстремума функции при ограничениях. Он имел четкую прикладную направленность: концентрировал внимание читателя на трудностях, возникающих при практическом решении задач, и способах преодоления этих трудностей.

Предлагаемая монография Ф. Гилла, У. Мюррея и М. Райт «Практическая оптимизация» имеет столь же острую практическую направленность. Как и упомянутый сборник статей, она отражает современное — теперь уже спустя десятилетие — состояние методов и техники решения задач оптимизации. Общая картина предстает перед читателем преломленной через призму опыта и, если угодно, научных вкусов авторов — больших знатоков своего дела. Это придает изложению и ясность, и логическую стройность, и оригинальность. Правда, отдельные детали получились не совсем удачными и потребовали от переводчика дополнительных усилий, чтобы точно передать существо дела.

Авторы старались написать книгу так, чтобы даже не слишком подготовленный читатель мог понять ее, не обращаясь к учебникам. Для ее чтения достаточно знать только основы математического анализа и линейной алгебры. Изложение начинается со сведений о способах представления чисел в ЭВМ, о возникающих при этом погрешностях и о погрешностях, сопутствующих вычислениям. Показано, как ошибки могут влиять на результаты работы алгоритмов. Так, с самого начала внимание читателя привлекается к практическим вычислениям. Затем сообщаются нужные сведения из линейной алгебры и выводятся необходимые и достаточные условия минимума функции как для случая, когда на независимые переменные не наложено никаких условий, так и для случая, когда условия наложены.

---

[1] См. «Численные методы условной оптимизации». Ред. Ф. Гилл и У. Мюррей. — М.: Мир, 1977.

Теперь читатель подготовлен к изучению алгоритмов отыскания минимумов. Сначала он знакомится с алгоритмами безусловной оптимизации и уяснит, как сильно свойства гладкости функции и информация об этих свойствах влияют на структуру алгоритмов и их эффективность. Затем наступает очередь алгоритмов вычисления минимума функции при линейных ограничениях. Алгоритмы решения задач с ограничениями-равенствами конструируются на базе алгоритмов безусловной оптимизации, задачи с ограничениями-неравенствами сводятся к последовательностям задач с равенствами при помощи правил построения наборов активных ограничений, т. е. ограничений, которые на текущем этапе вычислений считают равенствами. Последними предстают алгоритмы вычисления минимума функции при нелинейных ограничениях: методы штрафов, методы проектирования и методы модифицированных функций Лагранжа. Они включают в себя предыдущие алгоритмы.

Последние главы книги вновь обращают читателя к сугубо практическим вопросам: как реализовать изложенные методы, где и как их лучше использовать? Авторы не убоялись обсуждать плохо формализуемые (и поэтому обычно не обсуждаемые) вопросы, от решения которых во многом зависит успех применения того или другого алгоритма, например масштабирование задачи или критерии остановки вычислений. Многочисленные примеры и иллюстрации, систематически сопровождающие изложение материала, помогают читателю понять суть дела.

Книга «Практическая оптимизация» может служить и учебным пособием по математическому программированию и численным методам, и руководством к применению наиболее надежных из имеющихся сейчас универсальных алгоритмов оптимизации. Ясность изложения основных принципов и практические советы, основанные на богатом опыте авторов, привлекут к книге всех, кто начинает изучать численные методы оптимизации. Общий взгляд авторов на алгоритмы, высказываемые ими суждения не оставят равнодушными и специалистов в этой области.

А. А. Петров

Будучи сотрудниками Национальной физической лаборатории и Стэнфордского университета и участвуя в создании фонда программ Группы численных алгоритмов (NAG), мы уже многие годы занимаемся разработкой и программным воплощением методов оптимизации. За последние двадцать лет в этой области были достигнуты большие успехи. Значительно улучшены показатели эффективности и надежности математического обеспечения почти всех категорий оптимизационных задач. Однако возросла и сложность алгоритмов: качество повышалось за счет привлечения более тонких идей численной линейной алгебры и теории вычислений с ограниченной точностью. Лучшие из современных программ поиска экстремума весьма сложны и действуют далеко не очевидным образом.

В этой книге мы рассматриваем — по необходимости в общих чертах — предмет *практической оптимизации*. Эпитет «практическая» здесь и в названии книги употреблен, чтобы подчеркнуть, что внимание будет уделено не только формальным схемам, но также содержанию и особенностям их машинных реализаций. В частности, исследована чувствительность алгоритмов к ошибкам вычислений с ограниченной точностью и обсуждены их алгебраические блоки.

Ради замкнутости изложения мы включили в книгу две предварительные главы: в одной представлены используемые в дальнейшем результаты вычислительной линейной алгебры и элементарные сведения относительно последствий ошибок округления при расчетах на ЭВМ, а другая посвящена условиям оптимальности.

Избранные алгоритмы безусловной оптимизации и оптимизации при линейных и нелинейных ограничениях описаны в трех главах. Приводятся соображения, лежащие в основах алгоритмов, рассматриваются их теоретические и численные свойства. Для пояснения везде, где это уместно, даются примеры и иллюстрации. В основном мы представили алгоритмы, которые не раз успешно применяли на практике; иные обсуждаются лишь постольку, поскольку это способствует пониманию некоторых важных моментов или служит основой для последующих построений. Более подробные сведения, а также не попавшие в наш обзор алгоритмы можно найти по ссылкам, вынесенным в замечания, завершающие каждый большой раздел. Материал книги излагается достаточно подробно, и это позволяет рекомендовать ее в качестве учебника по курсу математического программирования.

Две последние главы посвящены менее формализованной, но отнюдь не второстепенной тематике; их можно расценивать как «наставления пользователям». Например, здесь даны некоторые рекомендации в отношении оптимизационного моделирования: по нашим наблюдениям, учет алгоритмических возможностей при разработке моделей часто оказывается весьма полезным. Кроме того, мы подробно обсуждаем вопросы выбора программы для конкретной задачи, интерпретации численного решения, диагностики (а иногда и устранения) причин плохой работы или отказа алгоритма.

При написании этой книги мы получали советы и помощь от многих людей. Прежде всего, хотим поблагодарить нашего друга и коллегу Майкла Сондерса не только за множество ценных комментариев, но и за его добрый юмор и терпение, которые так помогли нам в минуты, когда казалось, что работа над книгой продлится вечность. Он играл ведущую роль в разработке представленных в главах 5 и 6 алгоритмов решения задач большой размерности.

Мы очень признательны Дэвиду Мартину за его постоянную поддержку группы оптимизации в Национальной физической лаборатории и Джорджу Данцигу за его усилия по созданию алгоритмической группы в Лаборатории оптимизации систем Стэнфордского университета.

Мы благодарим Брайена Хинде, Сузан Ходсон, Инид Лонг и Дэвида Рида за их участие в создании и испытаниях многих представленных в этой книге алгоритмов.

Разнообразную помощь при подготовке книги оказывали Грег Добсон, Дэвид Фукс, Стефания Гэй, Ричард Стоун и Уэс Виклер. Отдельные места удалось изложить яснее благодаря замечаниям Паулю Беневедес-Соареса, Энди Конна, Лауреано Эскудеро, Дона Игл харта, Джеймса Лайнесса, Хорхе Море, Майкла Овертона и Дэнни Соренсена.

Мы благодарим Клива Холла за воспроизведение сгенерированных машиной рисунков на лазерном графопостроителе в Национальной физической лаборатории. Прочие рисунки были мастерски исполнены Нэнси Симина.

Эта книга была набрана с помощью программной системы TEX, созданной Доном Кнутом [1]. Мы благодарим его за то, что он предоставил нам различные макросы этой системы, позволившие улучшить качество окончательного текста. Крис Туччи очень помог в подготовке последнего варианта.

Наконец, мы хотим принести глубочайшую благодарность нашим близким, ободрявшим и поддерживавшим нас в течение долгой работы над книгой.

Стэнфордский университет
Ма 1981 г.

Ф. Э. Г.
У. М.
М. Г. Р.

[1] D. E. Knuth, TEX and METAFONT: New Directions in Typesetting, American Mathematical Society and Digital Press, Bedford, Massachusetts (1979).

ГЛАВА 1

ВВЕДЕНИЕ

Человечество ставит перед собой только те задачи, которые оно может разрешить
К. Маркс Из предисловия к «Критике политической экономии» (1859)

1.1. ПОСТАНОВКА ЗАДАЧИ ОПТИМИЗАЦИИ

Постановка любой задачи оптимизации начинается с определения набора независимых переменных и обычно включает условия, которые характеризуют их приемлемые значения. Эти условия называют *ограничениями* задачи. Еще одной обязательной компонентой описания является скалярная мера «качества», именуемая *целевой функцией* и зависящая каким-то образом от переменных. Решение оптимизационной задачи — это приемлемый набор значений переменных, которому отвечает оптимальное значение целевой функции. Под оптимальностью обычно понимают *максимальность* или *минимальность*; например, речь может идти о максимизации прибыли или минимизации массы.

К задачам на поиск оптимума сводятся многие из проблем математики, системного анализа, техники, экономики, медицины и статистики. В частности, они возникают при построении математических моделей. Когда для изучения какого-нибудь сложного явления конструируется математическая модель, к оптимизации прибегают для того, чтобы определить такую структуру и такие параметры последней, которые обеспечивали бы наилучшее согласование с реальностью. Другой традиционной областью применения оптимизации являются процедуры принятия решений, так как большинство из них нацелено именно на то, чтобы сделать «оптимальный» выбор. Помимо оптимизационных задач, представляющих самостоятельный интерес, на практике часто возникают задачи, которые «встроены» в некоторые вычислительные процессы, где они играют хотя и существенную, но все же вспомогательную роль. Это настолько типичная ситуация, что при описании процесса в целом далеко не всегда указывают на наличие подобных задач; к примеру, сказав, что процесс предполагает определение точки, в которой какая-то функция достигает своего критического значения, могут и не добавить, что эта точка будет найдена решением оптимизационной задачи.

Данная книга посвящена вопросам *численного поиска* оптимума и в том числе оптимизационным алгоритмам. При описании таких алгоритмов всегда используют стандартные формы представления задач. В частности, полезно ввести некую универсальную форму,

подходящую для большинства задач, встречающихся на практике. В качестве такой формы мы будем использовать следующую:

$$\text{NCP} \quad \text{найти} \quad \min_{x \in R^n} F(x)$$
$$\text{при ограничениях} \quad c_i(x) = 0, \ i = 1, 2, \ldots, m';$$
$$c_i(x) \geqslant 0, \ i = m'+1, \ldots, m.$$

Здесь целевая функция F и функция ограничений $\{c_i\}$ суть вещественнозначные скалярные функции. Говоря в дальнейшем о *функциях задачи*, мы будем иметь в виду F и $\{c_i\}$ одновременно.

В подтверждение тезиса о многообразии прикладных оптимизационных задач мы рассмотрим две из них. Решение первой определило расположение материала на страницах данной книги. Эту работу выполнила программная система машинного набора TEX. Она предназначена для того, чтобы, соблюдая заданные размеры полей страниц, размещать на них вводимый текст в форме, удобной для чтения. Достигается это с помощью двух средств: система подбирает расстояние между буквами, словами, строками и параграфами и может разбивать последние слова строк в соответствии с правилами переноса. Каждое из упомянутых расстояний имеет «идеальное значение» и определенные границы варьирования. При этом за уклонение от идеала начисляется «штраф». Различные штрафы вводятся также, чтобы избежать нежелательных ситуаций типа появления смежных строк, заканчивающихся переносимым словом, или размещения в начале страниц формулы, вынесенной в отдельную строку. Суммарный штраф минимизируется. Таким образом, в задаче поиска хорошего расположения текста, решаемой системой TEX, есть все элементы общей оптимизационной постановки: скалярная функция измерения качества; переменные, значения которых подбираются из соображений минимизации этой функции; ограничения на характер и величину разрешенных вариаций.

Следующий пример представляет собой упрощенный вариант задачи выбора профиля носовой части летательного аппарата с минимальным сопротивлением в потоке газа. Здесь критерием качества является лобовое сопротивление тела при заданной скорости, а переменные, значения которых надо подобрать,— это конструктивные параметры. Чтобы формализовать задачу, надо прежде всего выделить эти параметры, определив структуру математической модели носовой части. Мы будем считать, что последнюю можно описать как набор нескольких конических секций и одной сферической, причем радиус основания R полагается заданным. Эта модель изображена на рис. 1а, где перечислены и параметры, значения которых предстоит выбрать. Хотя реальный профиль должен иметь более сложную форму, огрубления, допущенные в модели, будут приемлемыми, если число секций взять достаточно большим.

После того как структура модели носовой части зафиксирована, надо найти зависимость лобового сопротивления от восьми перемен-

ных: $\alpha_1, \ldots, \alpha_4, r_1, \ldots, r_4$. Для этого потребуется соответствующая математическая модель обтекания. Мы ее строить не собираемся и отметим только, что, какова бы она ни была, результатом ее применения будет функция $D(\alpha_1, \ldots, \alpha_4, r_1, \ldots, r_4)$, оценивающая лобовое сопротивление по значениям переменных. Она и станет целевой функцией нашей задачи.

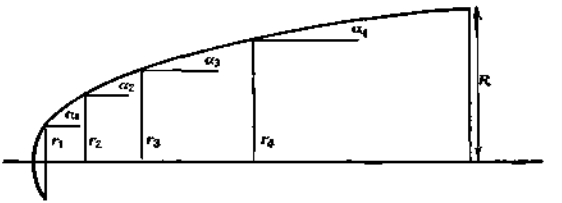

Рис. 1а. Модель профиля носовой части.

Для завершения формализации задачи о наилучшем профиле осталось выставить ограничения на величины переменных, вытекающие из физического смысла проблемы и гарантирующие, что полученное решение будет иметь смысл. В частности, первую группу ограничений составят условия неотрицательности радиусов $r_1, \ldots, r_4$:

$$r_i \geqslant 0, \quad i=1, \ldots, 4.$$

Происхождение задачи требует также, чтобы значения углов $\{\alpha_i\}$ лежали в диапазоне $[0, \pi/2]$ и чтобы каждый следующий угол не превосходил предыдущего. Значит, в задачу следует включить ограничения вида

$$0 \leqslant \alpha_i \leqslant \pi/2, \quad i=1, \ldots, 4;$$
$$\alpha_4 \leqslant \alpha_3 \leqslant \alpha_2 \leqslant \alpha_1.$$

В носовой части летательного аппарата обычно устанавливаются всевозможные приборы. Их список может быть определен заранее, и тогда носовая часть должна иметь объем, достаточный для их размещения. Кроме того, разумная конструкция должна иметь длину не больше заданной. Таким образом, в задаче будут присутствовать следующие ограничения:

$$\text{объем } (\alpha_1, \ldots, \alpha_4, \quad r_1, \ldots, r_4) \geqslant V;$$
$$\text{длина } (\alpha_1, \ldots, \alpha_4, \quad r_1, \ldots, r_4) \leqslant L.$$

Наконец, какие-то ограничения могли бы возникнуть из соображений прочности и на основании иных стандартов.

Итак, мы свели проблему оптимизации профиля к математической задаче вида

$$\text{найти} \min_{\alpha_1, \ldots, r_4} D(\alpha_1, \ldots, r_4)$$

$$\text{при ограничениях } 0 \leqslant r_i \leqslant R, \quad i = 1, \ldots, 4;$$

$$0 \leqslant \alpha_i \leqslant \pi/2, \quad i = 1, \ldots, 4;$$

$$\alpha_4 \leqslant \alpha_3 \leqslant \alpha_2 \leqslant \alpha_1;$$

$$0 \leqslant \text{объем } (\alpha_1, \ldots, \alpha_4, r_1, \ldots, r_4) - V,$$

$$0 \leqslant L - \text{длина } (\alpha_1, \ldots, \alpha_4, r_1, \ldots, r_4).$$

Характер и последовательность использованных при этом рассуждений типичны. Выделение представительного набора переменных, определение функции цели, описание требуемых качеств решения в терминах соотношений между переменными — таковы этапы математической формализации любой прикладной оптимизационной проблемы. Теперь остается только выбрать какой-нибудь из алгоритмов поиска минимума в задачах типа NCP и с его помощью рассчитать наилучший профиль.

1.2. Классификация оптимизационных задач

Подавляющее большинство оптимизационных задач, возникающих на практике, приводится к виду NCP. Даже те задачи, которые сами по себе в эти рамки не укладываются, часто могут быть сведены к последовательности стандартных. Однако существование столь универсальной формы представления вовсе не означает, что различиями между отдельными задачами следует пренебрегать. Наоборот, имея дело с конкретной постановкой, всегда надо постараться использовать ее *особенности* для того, чтобы организовать поиск решения самым эффективным образом. К примеру, усилия можно сэкономить, отказавшись от *каких-то* проверок, необходимых в общем случае, но лишних в конкретном, или избежав повторных вычислений величин, являющихся константами. Ниже рассматривается классификация разнообразных задач типа NCP по тем их особенностям, которые наиболее существенны для выбора алгоритма решения.

Определение признаков, по которым разумно классифицировать оптимизационные задачи,— проблема не простая. Самым подробным способом «классификации» было бы считать каждую задачу уникальной. Если бы любое изменение в постановке задачи существенно влияло на форму рациональной организации ее решения, это было бы оправданно и означало бы, например, что введение в задачу одной дополнительной переменной требует пересмотра алгоритма поиска оптимума. Однако, к счастью, подобной чувствительности алгоритмов к вариациям постановок не наблюдается, что и позволяет говорить о классификации в подлинном смысле этого слова. Хотя набо-

ра признаков, идеального для всех случаев жизни, не существует, достаточно разумный список составить можно. Поскольку имеется в виду, что разным классам задач будут отвечать разные алгоритмы решения, этот список должен быть результатом соразмерения выгод от эксплуатации выделяемых свойств и затрат на разработку соответствующего математического обеспечения.

Наиболее очевидные различия между задачами связаны с математическими характеристиками их функций. Например, в одном случае целевая функция будет гладкой, а в другом — разрывной; иногда функции задачи просты и свойства их понятны, а иногда явные выражения для функций отсутствуют и расчет их значений требует решения каких-то достаточно сложных подзадач.

В приведенной ниже таблице дана стандартная схема классификации оптимизационных задач по типам их функций. Каждый из перечисленных признаков существен для выбора алгоритма решения:

Типы $F(x)$	Типы $\{c_i(x)\}$
Функция одной переменной	Ограничения отсутствуют
Линейная функция	Простые ограничения на переменные
Сумма квадратов линейных функций	Линейные функции
Квадратичная форма	Линейные функции с разреженной матрицей коэффициентов
Сумма квадратов нелинейных функций	Гладкие нелинейные функции
Гладкая нелинейная функция	Гладкие нелинейные функции с разреженной матрицей Якоби
Нелинейная функция с разреженной матрицей Гессе	Негладкие нелинейные функции
Негладкая нелинейная функция	

В соответствии с данной таблицей выделяется, например, категория задач на поиск минимума гладкой нелинейной функции при простых ограничениях на переменные.

Помимо названных существуют и другие признаки классификации оптимизационных задач. Среди них обязательно следует упомянуть *размерность*. От нее зависит, сколько памяти и вычислений потребуется для поиска решения тем или иным методом. Как правило, методы, эффективные для задач с небольшим числом переменных, не пригодны в случаях, когда переменные исчисляются сотнями или тысячами. Классификация по размерности всегда относительна: считать ли задачу большой или маленькой, определяется тем, какие вычислительные средства имеются в распоряжении. Одна

и та же задача для мини-ЭВМ может оказаться большой, а для обычной машины — маленькой.

Еще один показатель, который может существенно различаться для разных задач и всегда учитывается при выборе алгоритмов,— это доступность производных. В одних задачах аналитические значения первых и вторых производных целевой функции вычисляются легко, а в других вычислению поддаются точные значения лишь самой функции. Когда говорят о доступности производных, то имеют в виду не только возможность построения процедуры расчета их точных значений, но и приемлемую трудоемкость этой процедуры. Последнее означает, что затраты на расчет производных сопоставляются с прочими затратами на реализацию поиска решения задачи.

Наконец, выбор алгоритма может определяться природой задачи и нуждами исследования, в рамках которого она возникла. Эти «внешние» факторы часто диктуют условия, никоим образом не вытекающие из математической постановки задачи. Например, по какой-то причине может оказаться необходимым, чтобы некоторые ограничения были соблюдены без исключения на всех итерациях. Суть задачи помимо прочего определяет и точность, с которой ее надо решить; если, к примеру, результаты решения используются как второстепенные данные для некой внешней итерации, то тратить усилия на достижение максимальной точности бессмысленно.

Разбор методов оптимизации в последующих главах включает сведения о том, как различные свойства задач влияют на эффективность методов. Будут даны также рекомендации, как выбрать способы решения и анализа результатов в зависимости от категории задачи.

1.3. КРАТКИЙ ОБЗОР СОДЕРЖАНИЯ

Когда-то арсенал методов оптимизации был небогат, эти методы были простыми и казалось естественным положение, когда каждый, кому нужно было решить оптимизационную задачу, шел в библиотеку, подыскивал описание подходящей схемы в каком-нибудь журнале (а то и сочинял свою схему) и самостоятельно программировал ее. Однако времена меняются, и сегодня подобное положение было бы неприемлемо.

Во-первых, стало намного сложнее разбираться в литературе, сильно разросшейся за последние годы. Эти годы были порой бурного развития аппарата оптимизации, и нет такой категории задач, для которой не появилось бы новых алгоритмов. Сейчас уже трудно надеяться, что рядовой пользователь, полистав журналы, сможет найти самый подходящий алгоритм.

Во-вторых, современные алгоритмы оптимизации в большинстве своем довольно сложны. Поэтому, даже отыскав тот из них, который следовало бы применить, пользователь скорее всего не станет его

программировать. К тому же, как показывают последние результаты численного анализа, кустарная реализация не только сложных, но и внешне простых вычислений может приводить к большим ошибкам и численной неустойчивости.

Короче говоря, сегодняшний пользователь не хочет (и, с нашей точки зрения, не должен) конструировать свои процедуры поиска экстремума и писать свои программы решения оптимизационных задач, начиная «с нуля». Ему нужны не ссылки на журнальные статьи, а хорошие библиотеки стандартных программ. Это, однако, не означает, что он может пребывать в полном неведении относительно устройства алгоритмов, с которыми ему предстоит работать, и основных особенностей реализующих их пакетов.

Данная книга предназначена в помощь тому, кто хочет в полной мере использовать возможности доступного программного обеспечения оптимизационных задач. Точнее, мы надеемся, что она поможет наиболее эффективно применять имеющиеся методы в случаях, когда они пригодны, а также успешно адаптировать и модифицировать их, когда это необходимо. Наряду с обсуждением всевозможных процедур оптимизации в книге затрагивается ряд смежных вопросов, возникающих при решении большинства прикладных задач. В частности, даются рекомендации по поводу того, как ставить задачи, чтобы шансы на успешное решение были максимальны, и как разбираться в причинах отказов алгоритмов.

В гл. 2 приведен обзор избранных результатов численного анализа. Знакомый с этим предметом читатель может ее пропустить. Особое внимание в ней уделено ошибкам машинной арифметики и некоторым из разделов вычислительной линейной алгебры. Этот материал образует основу для понимания дальнейшего изложения.

В остальных главах рассматриваются различные методы оптимизации, способы постановки задач, вопросы использования стандартных программ и анализа результатов вычислений. Мы хотим подчеркнуть, что изложение весьма сжато и содержит только самые необходимые сведения. Поэтому читатель не должен рассчитывать на то, что, одолев книгу, он станет экспертом по затронутым в ней проблемам. Однако для понимания сути дела материала достаточно.

ГЛАВА 2

ОСНОВЫ

Единственный способ противостоять природе — основательно познать ее.

Джон Локк (1693)

2.1. ВВЕДЕНИЕ В ТЕОРИЮ ОШИБОК ВЫЧИСЛЕНИЙ

2.1.1. ИЗМЕРЕНИЕ ОШИБКИ

Вычисляя какую-нибудь величину на ЭВМ, мы, как правило, получаем лишь ее приближенное значение, и надо уметь измерять степень его уклонения от точного значения. Понятно, что разумная мера такого уклонения должна быть равна нулю, если вычисленное значение совпадает с точным, должна быть «малой», если они «близки», и «большой», если они «существенно различны». Однако, чтобы по этому правилу построить строгий критерий точности, надо наполнить конкретным содержанием слова, выделенные кавычками, а это задача нетривиальная.

Очевидной мерой ошибки аппроксимации искомой величины служит разность между ее точным и приближенным значениями. Обозначим через x первое, а через $\tilde{x}$ второе. Тогда ошибка будет равна $\tilde{x}-x$, а неотрицательную величину $|\tilde{x}-x|$ принято называть *абсолютной ошибкой* приближения $\tilde{x}$.

Принимая во внимание только абсолютную ошибку, далеко не всегда можно правильно оценить качество приближения. Например, если $x=2$ и $\tilde{x}=1$, абсолютная ошибка равна 1, и в данном случае ее, по-видимому, следует считать «большой», так как приведенные значения вряд ли можно назвать близкими. Однако, если $x=10^{10}$, а $\tilde{x}=10^{10}+1$, ту же — единичную — абсолютную ошибку мы скорее всего сочтем «малой» и решим так потому, что разность между x и $\tilde{x}$ мала по сравнению с x. Эти рассуждения приводят к понятию *относительной* ошибки, которая определена как

$$\frac{|\tilde{x}-x|}{|x|},$$

если x отличается от нуля, и не определена в противном случае; таким образом, при вычислении относительной ошибки учитывается модуль точного значения величины. В приведенных примерах относительные ошибки равны 0.5 и 10^{-10} соответственно, так что вторая из них действительно «мала».

Когда точное значение рассчитываемой величины близко к нулю, пользоваться относительной ошибкой следует с осторожностью. На практике часто оказывается удобным измерять качество прибли-

жения величиной

$$\frac{|\tilde{x}-x|}{1+|x|},$$

которая объединяет в себе черты абсолютной и относительной ошибок. Она близка к первой при $|x|\ll 1$ и мало отличается от второй при $|x|\gg 1$.

2.1.2. ПРЕДСТАВЛЕНИЕ ЧИСЛА В МАШИНЕ

Уяснив смысл термина «ошибка», мы рассмотрим далее источники возникновения ошибок *при* расчетах на ЭВМ. Их несколько, и первым мы обсудим тот, который обусловлен самим способом представления чисел в машине. Технические детали последнего для разных классов машин различны, но основные принципы всегда одни и те же.

Общепринятый способ записи числовой информации состоит в представлении ее упорядоченной последовательностью цифр. Этот принцип используется и в известной всем десятичной системе счисления. Мы изображаем вещественное число цепочкой символов, которая начинается со знака плюса или минуса и продолжается списком цифр из набора 0, 1, ..., 9, возможно разделенных на две части десятичной запятой (иногда она предполагается заданной неявно). При расшифровке такой записи позиции, которые цифры занимают относительно запятой, определяют соответствующие степени числа 10.

В точности на тех же идеях строятся и способы записи чисел в ЭВМ. При этом электронные устройства памяти можно представлять себе как цепочки элементов, у каждого из которых есть только два состояния — «включено» и «выключено». Эти состояния интерпретируются как значения двоичного числа, которое может быть либо нулем, либо единицей и за которым закрепилось название «бит». А коль скоро всякая информация превращается в ЭВМ в цепочки битов, то и основания машинных систем счисления обычно бывают степенями числа 2: три наиболее распространенных основания — 2 (бинарная арифметика), 8 (восьмеричная арифметика с цифрами 0, 1, ..., 7) и 16 (шестнадцатеричная арифметика с цифрами 0, 1, ..., 9, A, ..., F).

Обычно для хранения одного числа в памяти машины выделяется поле, достаточное для записи фиксированного количества цифр. Его принято называть *словом*. Интерпретация цифр в пределах слова определяется специальными правилами — форматами записи. Два из них мы сейчас рассмотрим.

Первый формат называется представлением с *фиксированной запятой*. При хранении числа в этом формате позиция «запятой», отделяющей целую часть вещественного числа от дробной, предполагается неизменной. Проще говоря, фиксируются количества

разрядов, выделяемых для хранения целой и дробной частей числа. Если, например, под запись целых чисел отводятся слова с четырьмя десятичными разрядами, то целые от 0 до 9999 будут представлены в машине фактически так же, как они обычно записываются. Разница лишь в том, что в машинном представлении нули в старших разрядах не отбрасываются (т. е. число 20 будет представлено цепочкой 0020).

Чтобы хранить в представлении с фиксированной запятой числа разных знаков, можно ввести так называемое *смещение* — число, которое должно быть добавлено к запоминаемому перед записью в ЭВМ. Например, полагая смещение равным 4999, мы сможем в четырех десятичных разрядах хранить числа от —4999 до 5000, причем число —4999 будет представлено как 0000. С другой стороны, можно предусмотреть хранение знака в специально выделенном для этого бите слова.

Представление с фиксированной запятой приемлемо лишь тогда, когда известно, что все числа, которые придется хранить, лежат в определенном диапазоне. Это условие оказывается слишком жестким, потому что в большинстве исследовательских вычислительных работ данные, как правило, характеризуются большим разбросом значений данных. Здесь естественнее использовать представление числа в формате *с плавающей запятой*, аналогичное его записи в виде произведения десятичной дроби со знаком на степень числа 10.

В представлении с плавающей запятой ненулевое число x задается в виде

$$x=m\beta^e, \tag{2.1}$$

где β — основание системы счисления машины, e — целое со знаком, именуемое *порядком числа* x, а m — *мантисса* x. При конкретном значении β это представление будет единственным, если потребовать, чтобы мантисса была *нормализована*, т. е. удовлетворяла неравенствам

$$1/\beta\leqslant|m|<1.$$

Поскольку порядок e по определению есть целое со знаком, его естественно хранить в каком-нибудь подходящем формате с фиксированной запятой. Что же касается мантиссы, то для представления ее знака выделяют специальный бит, а модуль мантиссы хранят в виде строки из t цифр $m_1, m_2, m_3, \ldots, m_t$, где $0\leqslant m_i\leqslant\beta-1$, которая интерпретируется как запись дроби

$$\beta^{-t}(m_1\beta^{t-1}+m_2\beta^{t-2}+\ \ldots+m_t).$$

Если мантисса m нормализована, в этой записи $m_1\neq0$. В нормализованном представлении максимум величины $|m|$ равен $1-\beta^{-t}$, что соответствует $m_i=\beta-1$, $i=1, \ldots, t$; минимальное значение $|m|$ равно β^{-1} и получается при $m_1=1$, $m_2=\ldots=m_t=0$.

Поскольку нуль — ненормализованное число, любая схема хранения данных в формате с плавающей запятой должна предусматривать особый способ записи нуля.

Рассмотрим теперь два примера, иллюстрирующие сказанное. Начнем с гипотетической машины, которая оперирует десятичной арифметикой и имеет 8-разрядные слова, причем эти разряды используются следующим образом: левый («первый») разряд отведен под запись знака мантиссы m; следующие два разряда содержат порядок со смещением на 50, так что диапазоном его значений будут числа от —50 до +49; наконец, в последних пяти разрядах хранится модуль нормализованной мантиссы. В этой машине число $+.12345\times 10^{-3}$ будет записано так, как показано на рис. 2а.

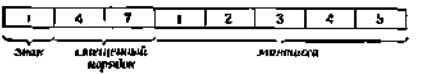

Рис. 2а. Десятичное слово с записью числа 0.12345×10^{-3}.

Более сложный пример взят из жизни и относится к машинам серий IBM 360/370 с шестнадцатеричной арифметикой. Для записи чисел ординарной точности в представлении с плавающей запятой в этих машинах отводятся слова, состоящий из 32 битов, группируемых в четыре 8-битовых *байта* или в восемь шестнадцатеричных разрядов. Первый бит слова содержит знак мантиссы (0 означает плюс, 1 — минус). Следующие семь битов (остаток первого байта) содержат порядок со смещением на 64. В байтах со второго по четвертый хранится модуль нормализованной мантиссы.

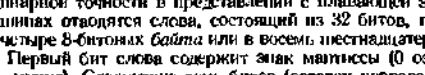

Рис. 2b. Шестнадцатеричное слово с записью числа $-42/4096$.

Рассмотрим представление числа $-42/4096=-16^{-1}(2/16+10/256)$. Истинный порядок в данном случае равен -1, а его смещенным значением будет 63. Модуль нормализованной мантиссы равен .2A (основание 16). Таким образом, запись числа в машине будет выглядеть так, как показано на рис. 2b.

2.1.3. ОШИБКИ ОКРУГЛЕНИЯ

Множества чисел, которые можно записать словом заданной длины в форматах с фиксированной и плавающей запятой, конечны. Если слово некоторой машины содержит t разрядов по основа-

нию β, то в ней можно оперировать не более чем β^t различными числами. Эти β^t чисел формируют так называемое *представимое* множество машины. Все иные, не попавшие в это множество числа не могут быть представлены в ней точно, и запись любого из них в память будет сопровождаться некоторой ошибкой. Такие ошибки обычно называют *ошибками округления* или *ошибками представления*.

Некоторые числа непредставимы в машине стандартным способом из-за того, что их величины лежат за границами диапазона значений, которые можно хранить в машинном слове. Эти границы определяются следующим образом. Если e_{max} есть максимальный разрешенный порядок, то значение максимального модуля числа, представимого в формате с плавающей запятой, равно $K=\beta^{e_{max}}(1-\beta^{-t})$. Если e_{min} есть минимальный разрешенный порядок, то наименьший модуль числа, представимого в нормализованном виде, равен $k=\beta^{(e_{min}-1)}$. Попытка записать число, по модулю превосходящее K, приведет к *переполнению*, а число, по модулю меньшее, чем k,— к *антипереполнению*.

Помимо рассмотренных есть и другие числа, которые нельзя представить в машине (без ошибки) обычным путем. Это — числа, имеющие в представлении (2.1) мантиссу с более чем t значащими цифрами. Например, число $\pi=3.14159...$ не может быть представлено точно никаким конечным набором цифр. Заметим, что, кроме всего прочего, возможность представления конкретного числа определяется и тем, каково основание β. Так, число 1/10 представимо в одноразрядной десятичной арифметике и не представимо никаким конечным набором цифр в арифметиках с основаниями 2, 8 и 16.

Коль скоро задано непредставимое в формате с плавающей запятой число x, которое тем не менее не приводит ни к переполнению, ни к антипереполнению, возникает вопрос о выборе представимого числа $\tilde{x}$, аппроксимирующего x наилучшим образом; при этом $\tilde{x}$ обычно обозначают через $fl(x)$. Поскольку x заведомо лежит между двумя представимыми числами, минимизация ошибки округления в таком случае сводится к выбору в качестве $\tilde{x}$ ближайшего представимого «соседа» числа x. Этот выбор, если он единствен, определяется следующим правилом: оставьте m_t без изменения, если модуль отбрасываемой части мантиссы составляет менее половины значения единицы в последнем сохраняемом разряде (т. е. менее $^1/_2\beta^{-t}$), а в противном случае добавьте единицу к m_t (и, если потребуется, заново нормализуйте мантиссу). При такой схеме округления в машине с десятичной арифметикой и шестиразрядной мантиссой числа 3.14159265... и —20.98999 будут преобразованы в 3.14159 и —20.9900 соответственно.

Теперь осталось определить, как будет округляться непредставимое число, лежащее в точности посередине между двумя представимыми. Есть несколько способов автоматически разрешить данную неоднозначность. Например, часто используется правило

округления до ближайшего четного. В соответствии с ним непредставимое число округляется до того из ближайших представимых, чей последний разряд четен. При этом числа .98435 и .98445 будут округлены в десятичной машине с четырехразрядной арифметикой до одного и того же числа .9844.

Описанные схемы округления называют «корректными». Такое округление приводит к ошибкам, не превосходящим по абсолютной величине половины значения единицы в младшем сохраняемом разряде мантиссы. Пусть x — ненулевое число, а $\hat{x}$ — его t-разрядное округленное значение, причем их порядки по основанию β равны. Тогда, обозначив через m и $\hat{m}$ мантиссы x и $\hat{x}$ соответственно, получим

$$|m-\hat{m}| \leqslant \frac{1}{2}\beta^{-t},$$

а так как относительная ошибка приближения $\hat{x}$ в рассматриваемом случае равна

$$\frac{|\hat{x}-x|}{|x|} = \frac{|m-\hat{m}|}{|m|},$$

из предыдущего неравенства видно, что величину этой ошибки можно оценить по формуле

$$\frac{|fl(x)-x|}{|x|} \leqslant \frac{1}{2}\beta^{1-t}. \qquad (2.2)$$

Здесь учтено, что $1/\beta \leqslant |m| < 1$. Фигурирующее в полученной оценке (2.2) число β^{1-t} играет в анализе погрешностей вычислений с плавающей запятой важную роль. Его принято называть *относительной точностью* машины (или просто машинной точностью). Впредь в этой книге оно будет обозначаться через ε_M.

В некоторых машинах для приближения непредставимых чисел используются и другие схемы округления, например правило усечения, в соответствии с которым все цифры мантиссы m после младшей сохраняемой просто отбрасываются независимо от их «номиналов». Тогда в десятичной машине с четырехразрядной арифметикой все числа между .98340 и .983499...9 будут округлены до .9834. При этом относительную ошибку можно оценить аналогичной (2.2) формулой вида

$$\frac{|fl(x)-x|}{|x|} \leqslant \beta^{1-t},$$

в соответствии с которой абсолютная ошибка теперь может быть равна единице последнего разряда.

2.1.4. ОШИБКИ ПРИ ВЫПОЛНЕНИИ АРИФМЕТИЧЕСКИХ ОПЕРАЦИЙ

Выполнение арифметических операций на ЭВМ сопровождается дополнительными ошибками округления, связанными с необходимостью запоминания непредставимых результатов вычислений

Даже если числа x_1 и x_2 сами по себе представимы, их точная сумма или произведение могут оказаться непредставимыми. Чтобы проиллюстрировать потерю точности в процессе арифметических преобразований, рассмотрим один из способов выполнения операций сложения и вычитания в формате с плавающей запятой.

Вычисление суммы или разности двух чисел с плавающей запятой можно организовать следующим образом. Мантиссу большего по модулю числа поместим в регистр r_1, а меньшего — в регистр r_2 и сдвинем последний вправо на столько позиций, на сколько отличаются порядки чисел. После этого выполним сложение или вычитание, записав результат в «удлиненный» регистр R. Схематически ситуация изображена на рис. 2с.

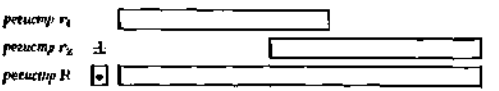

Рис. 2с. Операнды машинного сложения и вычитания.

Регистр R имеет специальный разряд переполнения, отмеченный на рис. 2с жирной точкой. Он необходим на случай появления (в результате сложения или вычитания мантисс) значащей цифры перед десятичной запятой. Если это произойдет, регистр R должен быть нормализован сдвигом вправо. Если же разряд переполнения не понадобился и первые после десятичной запятой разряды получились нулевыми, для нормализации R нужен сдвиг влево.

Ошибка при сложении и вычислении появляется из-за того, что мантисса, записанная в регистр R, вообще говоря, будет содержать более, чем τ, значащих цифр, и соответственно ее придется округлять. Кроме того, в некоторых машинах при сдвиге r_2 вправо сохраняются не все разряды r_2, которые младше самого младшего разряда r_1. В данном случае при нормализации R сдвигом влево может возникнуть дополнительная потеря точности.

Вычисленный результат операции с плавающей запятой удовлетворяет соотношению

$$fl(a\ op\ b)=(a\ op\ b)(1+\eta).$$

Здесь a и b — два представимых числа, op — одна из операций «+», «—», «×» или «÷», а η — величина, зависящая от a, b, машинной точности и способа реализации в машине арифметики с плавающей запятой.

Ошибка η не может быть меньше той, которая диктуется необходимостью однократного округления результата операции, и, как правило, ее оценка сверху пропорциональна машинной точности с небольшим множителем.

2.1.5. ОШИБКИ КОМПЕНСАЦИИ

Среди арифметических операций есть такие, которые могут приводить к появлению относительных ошибок, превышающих величину машинной точности во много раз. Это — вычисления разностей почти совпадающих округленных чисел. Связанные с ними ошибки принято называть *ошибками компенсации*.

Рассмотрим два числа x_1 и x_2, чьи представления в формате с плавающей запятой равны $\hat{x}_1=x_1(1+\varepsilon_1)$ и $\hat{x}_2=x_2(1+\varepsilon_2)$ соответственно, а ε_1, ε_2 ограничены по модулю сверху относительной машинной точностью. Точную разность $\hat{x}_1$ и $\hat{x}_2$ можно записать следующим образом:

$$\Delta\hat{x}=x_1(1+\varepsilon_1)-x_2(1+\varepsilon_2)=(x_1-x_2)(1+\eta), \qquad (2.3)$$

где величина η будет относительным уклонением значения $\Delta\hat{x}$ от точного значения разности исходных чисел.

Если $x_1=x_2$, мы говорим о *полной компенсации*. В противном случае из (2.3) получается выражение для η вида

$$\eta=\frac{\varepsilon_1 x_1-\varepsilon_2 x_2}{x_1-x_2}. \qquad (2.4)$$

Соответственно для относительной ошибки приближения $\Delta\hat{x}$ справедлива такая оценка сверху:

$$|\eta|=\frac{|\varepsilon_1 x_1-\varepsilon_2 x_2|}{|x_1-x_2|}=\frac{|\varepsilon_2(x_1-x_2)+x_1(\varepsilon_1-\varepsilon_2)|}{|x_1-x_2|}\leqslant \leqslant e_M\left(1+2\frac{|x_1|}{|x_1-x_2|}\right). \qquad (2.5)$$

Из (2.5) видно, что, когда модуль $|x_1-x_2|$ мал по сравнению с $|x_1|$ (т. е. число x_1 «примерно равно» x_2), относительная ошибка в $\Delta\hat{x}$ может быть много больше машинной точности e_M. При этом она окажется большой не из-за того, что вычитание $\hat{x}_2$ из $\hat{x}_1$ выполнено не точно (ведь $\Delta\hat{x}$ есть *точная* разность между $\hat{x}_1$ и $\hat{x}_2$), а потому, что сами величины $\hat{x}_1$ и $\hat{x}_2$ получены округлением x_1, x_2 и включают соответствующие ошибки; заметим, что если ε_1 и ε_2 равны нулю, то η также будет нулем. Когда x_1 и x_2 взаимно близки, их старшие разряды во время вычитания взаимно уничтожаются, и потому младшие разряды, отброшенные при округлении x_1 и x_2 до $\hat{x}_1$ и $\hat{x}_2$, приобретают вес. Таким образом, компенсация в данном случае «вскрывает предыдущие огрехи». Если же x_1 и x_2 существенно различны, оценка сверху для ошибки компенсации при вычитании оказывается величиной того же порядка, что и оценки ошибок других операций с плавающей запятой, т. е. здесь она никакого особого значения иметь не будет.

В качестве примера рассмотрим вычисление разности величин

$$x_1=.2946796847, \quad x_2=.2946782596 \qquad (2.6)$$

ка машине, в которой мантисса представления числа с плавающей запятой содержит шесть десятичных цифр ($\varepsilon_M = 10^{-6}$). При корректном округлении значения $\tilde{x}_1$ и $\tilde{x}_2$ будут равны .294680 и .294678 соответственно, и их разность (вычисленная без ошибки) равна 2×10^{-6}. В то же время разность между *точными* значениями x_1 и x_2 равна $.14251 \times 10^{-6}$, откуда следует, что $\Delta\tilde{x}$ имеет относительную ошибку компенсации $\eta \sim .40341$. Из (2.5) видно, что верхняя граница относительной ошибки компенсации снижается при уменьшении ε_M, если вычитание x_2 из x_1 выполнить на машине с восьмиразрядной мантиссой, то относительная ошибка компенсации будет равна $.357 \times 10^{-6}$.

Не зная точных значений x_1, x_2 и не имея возможности делать точных арифметических вычислений, истинного значения ошибки (2.4) не определить. Поэтому приходится пользоваться приближенными значениями, которые получаются оцениванием правой части неравенства (2.5). Их мы и будем называть ошибками компенсации, причем речь, разумеется, идет о вычислимых оценках. Методы расчета таких оценок обсуждаются в гл. 8.

2.1.6. ТОЧНОСТЬ ПРИ ПОСЛЕДОВАТЕЛЬНЫХ ВЫЧИСЛЕНИЯХ

В данном разделе вводится терминология, используемая при анализе точности величин, получающихся в результате длинных цепочек вычислений с привлечением других расчетных или измеряемых значений.

Обозначим через f точное значение искомой величины; это — значение, которое было бы получено, если бы все промежуточные вычисления выполнялись точно и с точными значениями всех привлекаемых данных. Пусть далее $fl(f)$ — реальный конечный результат вычислений. Тогда, если

$$fl(f) = f + \sigma,$$

то $|\sigma|$ в соответствии с приведенным ранее определением (см. разд. 2.1.1) есть абсолютная *ошибка приближения* $fl(f)$. Мы будем использовать термин *абсолютная точность* (или *уровень шума*) для обозначения положительного числа ε_A, которое является верхней границей для абсолютной ошибки, т. е. $|\sigma| \leqslant \varepsilon_A$. При ненулевом значении f качество приближения $fl(f)$ иногда удается выразить в терминах *относительной точности*, так как погрешности расчетов в рамках арифметики с плавающей запятой и в обычных методах вычисления стандартных функций изначально проявляются именно как относительные. Так, например, для большинства машин расчетное значение $\sqrt{x}$ будет содержать ошибку, не превосходящую единицы в младшем разряде мантиссы его представления в формате с плавающей запятой. Введя относительную ошибку, мы можем записать $fl(f)$ в виде

$$fl(f) = f(1+\delta).$$

При этом *относительной точностью* будем называть положительное число ε_R, для которого $|\delta| \leqslant \varepsilon_R$.

Связь между ошибками δ и σ очевидна и выражается равенством $|\delta f| = |\sigma|$. Оно выполнено по определению, т. е. справедливо вне зависимости от того, что представляет собой величина f. К сожалению, сказать то же самое о вычислимых границах ε_A, ε_R нельзя. Правда, абсолютная и относительная точности нередко удовлетворяют соотношению $\varepsilon_R \sim \varepsilon_A |f|$. Если f — стандартная функция, то обычно $\varepsilon_R = \varepsilon_M$ и $\varepsilon_A = \varepsilon_M |f|$, но в общем случае связь между ε_A и ε_R оказывается значительно более сложной, особенно при малых $|f|$.

2.1.7. АНАЛИЗ ОШИБОК ДЛЯ АЛГОРИТМОВ

Когда какая-то вычислительная задача решается на ЭВМ, результат представляет собой набор представимых чисел, как правило, генерируемых последовательностью операций в арифметике с плавающей запятой. Эти операции связаны с ошибками, и потому ответ будет приближенным даже в том случае, если используемый алгоритм теоретически является точным. Разные алгоритмы реагируют на условность машинной арифметики по-разному, и в данном разделе мы рассмотрим вопрос о том, как можно классифицировать их по отношению к этой реакции.

В идеале всегда хочется иметь гарантию «близости» численного решения к точному, т. е. располагать удовлетворительной оценкой вида

$$\|s - \bar{s}\| \leqslant \delta, \tag{2.7}$$

где s — точное решение, $\bar{s}$ — численное, $\|\cdot\|$ — разумная мера уклонения. Исходя из этого, хорошим следовало бы считать такой алгоритм, который всегда обеспечивает соблюдение неравенства (2.7) с достаточно малым δ. Однако при данном подходе к оценке качества алгоритмов, именуемом *прямым анализом ошибок*, большинство из них пришлось бы признать плохими.

Дело в том, что многим задачам присуще свойство *плохой обусловленности*. Это внутреннее свойство, и определяется оно вне какой-либо привязки к дефектам машинной арифметики. Плохо обусловленная задача характерна тем, что малое возмущение ее параметров может привести к большому возмущению точного решения (соответствующие примеры даны в разд. 2.2.4). В силу этого обстоятельства прямой анализ ошибок для таких задач всегда дает пессимистичные оценки.

Допустим, например, что все расчеты по некоторому алгоритму могут быть выполнены точно и лишь одну-единственную ошибку округления устранить не удается. По любым разумным меркам этот алгоритм должен считаться хорошим. Однако, применив его к очень плохо обусловленной задаче, мы рискуем получить численное ре-

шение, которое будет сильно отличаться от настоящего. Соответственно величина δ в (2.7) с необходимостью окажется большой, т. е. с точки зрения прямого анализа ошибок этот алгоритм плох.

Итак, возможности классификации алгоритмов на основе прямого анализа ошибок ограничены. Значительно более плодотворной является иная форма анализа ошибок — так называемый *обратный анализ*. В рамках этого подхода численное решение трактуется как точное, но не для исходной, а для некоторой возмущенной задачи. Если алгоритм гарантирует «близость» этих задач, он считается хорошим.

Обозначим через P исходную задачу, и пусть ее численное решение является точным для некоторой задачи $\tilde{P}$. Результатом обратного анализа ошибок обычно является оценка вида

$$\|P-\tilde{P}\|\leqslant\Delta, \tag{2.8}$$

где Δ зависят от машинной точности и от задачи P.

В отличие от прямого анализа ошибок обратный анализ показывает, что для большинства общепризнанных алгоритмов величина Δ «мала» независимо от P. Такие алгоритмы принято называть *численно устойчивыми*, так как ошибки выполняемых в соответствии с ними вычислений приводят к небольшому уклонению от исходной задачи. Заметим, кстати, что Δ может быть ненулевой уже в силу необходимости аппроксимации начальных данных задачи представимыми числами.

Замечания и избранная библиография к разделу 2.1

Впервые роль ошибок округления в алгебраических процедурах была проанализирована Уилкинсоном (1963). Тому, кто хочет поподробнее ознакомиться со способами реализации арифметики с плавающей запятой на современных машинах, рекомендуем работу Кахана (1973). Хорошим универсальным пособием по численному анализу является книга Дальквиста и Бьёрка (1974).

2.2. ВВЕДЕНИЕ В ВЫЧИСЛИТЕЛЬНУЮ ЛИНЕЙНУЮ АЛГЕБРУ

В этом разделе рассматриваются те элементы вычислительной линейной алгебры, которые для оптимизационных приложений представляются наиболее важными. Изложение их будет весьма кратким и не претендует на полноту; желающим ознакомиться с предметом более детально следует обратиться к цитируемым ниже источникам.

2.2.1. ПРЕДВАРИТЕЛЬНЫЕ СВЕДЕНИЯ

2.2.1.1. Скаляры. Одиночные количественные характеристики и результаты измерений обычно формализуются понятием «скаляр» или «вещественное число». Вещественные числа обладают мно-

ними замечательными свойствами по отношению к операциям сложения и умножения, используемыми в дальнейшем при определении операций над векторами и матрицами.

Впредь скаляры, как правило, будут обозначаться малыми греческими буквами: α, β, δ и т. д.

2.2.1.2. Векторы. Формально вектор может быть определен как *упорядоченный набор скаляров*, причем и значения чисел в этом наборе, и их порядок существенны. Подобно тому как вещественное число служит мерой отдельного свойства какого-либо объекта (скажем, высоты стола), векторы используются для описания упорядоченных совокупностей свойств. Предположим, к примеру, что высота стола рассматривается как его первая характеристика, длина — как вторая, а ширина — как третья. Тогда стол, имеющий 3 фута высоты, 3.5 фута длины и 2 фута ширины, может быть описан вектором

$$\text{стол} = \begin{pmatrix} 3.0 \\ 3.5 \\ 2.0 \end{pmatrix}.$$

Стандартная форма записи вектора опирается на привычный порядок чтения текстов — сверху вниз. Соответственно векторы изображают вертикальными столбцами чисел с первой компонентой в верхней позиции.

Количество скалярных числовых величин, образующих вектор, называют *размерностью* вектора, а сами числа — его *компонентами*. Последние, как правило, нумеруются индексом, пробегающим значения от 1 до n, где n — размерность вектора.

В дальнейшем векторы будут обозначаться малыми латинскими буквами: a, b, x, ... Обозначением отдельной компоненты вектора будет служить его имя с нижним индексом, указывающим номер этой компоненты. Так, через x_3 принято обозначать вторую компоненту вектора x.

Два вектора считаются *одинаковыми*, если их соответственные компоненты равны. Это определение предполагает, что проверяемые на совпадение векторы имеют одинаковую размерность.

Иногда удобно использовать представление вектора как *строки*, т. е. горизонтального списка чисел. Если вектор-столбец x задан в виде

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ x_n \end{pmatrix},$$

то через x^T будем обозначать вектор-строку, компоненты которой упорядочены в соответствии с общепринятым правилом чтения слева направо:

$$x^T=(x_1, x_2, \ldots, x_n).$$

2.2.1.3. Матрицы. Переход от одной скалярной величины к упорядоченному списку таких величин отражается понятием «вектор». Еще один шаг обобщения в том же направлении приводит к понятию «матрица». Это — результат двойного упорядочения набора чисел.

Как и в случае с векторами, обычная форма записи матриц опирается на стандартный порядок чтения текстов — сверху вниз и слева направо; двум направлениям записи элементов отвечают два типа их упорядочения. «Первый» элемент матрицы располагается в верхнем левом углу записи. «Первый» тип упорядочения соответствует записи сверху вниз и выделяет «строки» матрицы, в каждой из которых содержится одно и то же количество скалярных элементов. Число строк называют *строковой размерностью*. «Второй» тип упорядочения отвечает правилу чтения слева направо и выделяет столбцы матрицы. Их число называется *столбцовой размерностью*. Если количество строк и столбцов одинаково, говорят, что матрица является *квадратной*.

Матрицы впредь будут обозначаться заглавными латинскими буквами (т. е. A, B, ..., W). Ссылку на одиночный элемент матрицы принято задавать ее именем (соответствующей малой или заглавной буквой) с двумя нижними индексами, первый из которых есть индекс строки, а второй — столбца. Таким образом, через A_{ij} или a_{ij} обозначают элемент, стоящий в матрице A на пересечении строки i и столбца j.

Приведенные определения иллюстрируются следующей записью матрицы с тремя строками и двумя столбцами:

$$B=\begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{pmatrix}.$$

Вектор-столбец мы теперь можем рассматривать как частный случай матрицы со столбцовой размерностью, равной единице. Все элементы такой матрицы имеют один и тот же единичный второй индекс, поэтому его можно отбросить.

Равенство между двумя матрицами означает равенство между любыми их соответственными элементами. Оно предполагает совпадения строковых и столбцовых размерностей сравниваемых матриц.

Транспонированной называют матрицу, которая получится из исходной, если столбцовые и строковые индексы поменять ролями. Матрица, транспонированная к A, обозначается через A^T и определяется следующим равенством: $(A^T)_{ij}=A_{ji}$. Например, транспони-

рование матрицы

$$A=\begin{pmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{pmatrix}$$

дает матрицу

$$A^T=\begin{pmatrix}1 & 4\\ 2 & 5\\ 3 & 6\end{pmatrix}.$$

Матрицу называют *симметричной*, если $A=A^T$. Это означает, во-первых, что A – квадратная матрица и, во-вторых, что при всех i и j выполняется равенство $a_{ij}=a_{ji}$.

Элементы a_{ii} матрицы A называются *диагональными*, а все остальные элементы — *внедиагональными*. Элементы a_{ij}, для которых $j>i$, принято называть *наддиагональными*, а те, для которых $j<i$, – *поддиагональными*.

2.2.1.4. Операции над векторами и матрицами. Подобно тому как векторы и матрицы представляют собой упорядоченные совокупности скалярных величин, операции над векторами и матрицами являются упорядоченными совокупностями скалярных операций.

Простейшая операция — умножение матрицы (или вектора) на число. Ее результат есть матрица (или вектор), элементы которой равны произведениям этого числа на соответствующие элементы исходной матрицы (или вектора). Таким образом,

$$\alpha\begin{pmatrix}x_1\\ x_2\\ x_3\end{pmatrix}=\begin{pmatrix}\alpha x_1\\ \alpha x_2\\ \alpha x_3\end{pmatrix};\qquad -4\begin{pmatrix}1 & 0\\ 3 & -4\end{pmatrix}=\begin{pmatrix}-4 & 0\\ -12 & 16\end{pmatrix}.$$

Следующая операция — *сложение* двух матриц или векторов. Она опирается на обычную операцию скалярного сложения. Определение таково: элементами суммы двух матриц (векторов) являются суммы соответственных элементов слагаемых. Это определение подразумевает совпадение размерностей суммируемых матриц (векторов). Следующий пример иллюстрирует смысл операции сложения в векторном случае:

$$\begin{pmatrix}a_1\\ a_2\\ a_3\end{pmatrix}+\begin{pmatrix}b_1\\ b_2\\ b_3\end{pmatrix}=\begin{pmatrix}a_1+b_1\\ a_2+b_2\\ a_3+b_3\end{pmatrix}.$$

Матрица или вектор, все элементы которых нулевые, играют во вновь введенной операции сложения ту же роль, что и нуль в скалярном суммировании: сложение произвольной матрицы (вектора) с нулевой сохраняет ее неизменной. Нулевые матрицы и векторы принято обозначать стандартным символом 0. Их размерности всегда определяются из контекста.

Легко убедиться, что сложение матриц (векторов) обладает теми же свойствами, что и сложение скаляров, а именно

ассоциативность: $A+(B+C)=(A+B)+C$;
коммутативность: $A+B=B+A$.

Для двух векторов a и b одинаковой размерности n вводят понятие *скалярное произведение*, величина которого γ определяется равенством

$$\gamma=a_1b_1+a_2b_2+\ldots+a_nb_n=\sum_{i=1}^{n}a_ib_i.$$

Впредь мы будем обозначать его через a^Tb. Отметим, что при подсчете скалярного произведения компоненты векторов перемножаются в соответствии с их упорядочением. В операции вычисления скалярного произведения сохраняются два следующих свойства скалярного умножения:

коммутативность: $a^Tb=b^Ta$;
дистрибутивность по векторному сложению: $a^T(b+c)=a^Tb+a^Tc$.

Хотя скалярное произведение на первый взгляд аналогично простому произведению двух чисел, между ними есть принципиальное отличие, о котором следует сказать особо. Оно состоит в том, что скалярное произведение двух ненулевых векторов может быть нулем, в то время как перемножение двух ненулевых чисел всегда дает ненулевой результат. Например, для

$$a=\begin{pmatrix}1\\-1\\2\end{pmatrix},\qquad b=\begin{pmatrix}1\\-1\\-1\end{pmatrix}$$

получим $a^Tb=1+1-2=0$.

Когда $a^Tb=0$, говорят, что векторы a и b *ортогональны* друг другу. Заметим также, что из равенства $a^Tc=b^Tc$ в общем случае *не следует* равенство $a=b$. В этом легко убедиться на тройке векторов:

$$a=\begin{pmatrix}4\\1\end{pmatrix},\quad b=\begin{pmatrix}2\\3\end{pmatrix},\quad c=\begin{pmatrix}1\\1\end{pmatrix}.$$

Однако некая связь между a, b и c все же есть: в силу свойства дистрибутивности можно утверждать, что $(a-b)^Tc=0$, т. е. разность $a-b$ ортогональна c.

Понятие ортогональности позволяет определить нулевой вектор как вектор, ортогональный *всем* векторам.

Обобщением операции взятия скалярного произведения векторов является операция перемножения матрицы. Произведение C матрицы A на B определяется как матрица, (i, j)-й элемент которой есть скалярное произведение i-й *строки* A на j-й *столбец* B. Это

определение имеет смысл, если число столбцов матрицы A совпадает с числом строк матрицы B: только в этом случае можно говорить о соответствующих скалярных произведениях. Строковая размерность C совпадает со строковой размерностью A, а ее столбцовая размерность будет равна столбцовой размерности B.

Обозначив через a_i^T i-ю строку матрицы A, а через b_j j-й столбец B, произведение A на B можно записать так:

$$C = AB = \begin{pmatrix} a_1^T \\ a_2^T \\ \cdot \\ \cdot \\ \cdot \end{pmatrix} \widehat{\ } (b_1 b_2 \ldots) = \begin{bmatrix} a_1^T b_1 & a_1^T b_2 & \cdot & \cdot \\ a_2^T b_1 & a_2^T b_2 & \cdot & \cdot \\ \cdot & & \cdot & \cdot \\ \cdot & & \cdot & \cdot \\ \cdot & & \cdot & \cdot \end{bmatrix}.$$

Например, для

$$A = \begin{pmatrix} 1 & 1 \\ 2 & -3 \end{pmatrix}, \quad B = \begin{pmatrix} 4 & 1 \\ 0 & 2 \end{pmatrix}$$

получим

$$AB = \begin{pmatrix} 4 & 3 \\ 8 & -4 \end{pmatrix}.$$

Важно отметить явную несимметричность определения произведения матриц по отношению к строкам и столбцам сомножителей. Именно в силу этой несимметричности порядок сомножителей становится существенным.

В матричном произведении AB матрицу A называют *левым* множителем B и говорят об умножении B на A *слева*. В результате такого умножения столбцы B преобразуются независимо — каждый столбец B определяет соответствующий столбец произведения, так что, например, изменение первого столбца B приведет к изменению только первого столбца AB. Аналогично матрицу B называют правым сомножителем A и говорят об умножении A на B *справа*. При умножении справа независимо преобразуются строки умножаемой матрицы, так что изменение в i-й строке A повлечет за собой изменение только в i-й строке AB.

Операция матричного умножения обладает следующими свойствами:

ассоциативность: $(AB)C = A(BC)$;

дистрибутивность по матричному сложению:

$$A(B+C) = AB + AC.$$

При этом в силу упомянутой выше асимметрии матричное умножение в общем случае *не* коммутативно. Даже для квадратных матриц (единственный случай, когда определены оба произведения AB и BA), вообще говоря, будет $AB \neq BA$. Еще одно свойство матричного умножения состоит в том, что $(AB)^T = B^T A^T$.

Определенное ранее скалярное произведение можно рассматривать как частный случай матричного произведения, когда матрица,

состоящая из одной строки, умножается справа на матрицу, состоящую из одного столбца. Следующий частный случай — произведение матрицы на вектор, т. е. перемножение двух матриц, вторая из которых состоит из единственного столбца. Такое произведение определено для матрицы A и вектора x, если размерность x совпадает с числом столбцов A.

Особую роль в операции умножения играют так называемые *единичные* матрицы. Единичная матрица порядка n, которую принято обозначать через I_n, есть квадратная матрица вида

$$I_n = \begin{pmatrix} 1 & 0 & 0 & \cdot & 0 & 0 \\ 0 & 1 & 0 & \cdot & 0 & 0 \\ 0 & 0 & 1 & \cdot & 0 & 0 \\ & & & & & \\ 0 & 0 & \cdot & \cdot & 1 & 0 \\ 0 & 0 & \cdot & \cdot & 0 & 1 \end{pmatrix}.$$

Ее свойства подобны свойствам числа 1 в скалярном умножении. Если A есть $m \times n$-матрица, то $AI_n = A$ и $I_m A = A$. Когда размерность единичной матрицы ясна из контекста, указывающий ее нижний индекс часто опускают. Столбец единичной матрицы с номером i обычно обозначается через e_i.

2.2.1.5. Матрицы специальной структуры. В этом разделе будут выделены некоторые особые типы матриц с характерными распределениями нулевых элементов или наличием специфичных связей между элементами.

Матрицу называют *диагональной*, если все ее недиагональные элементы равны нулю. Такие матрицы обычно обозначаются буквой D; например,

$$D = \begin{pmatrix} -2 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Диагональную матрицу удобно задавать списком ее диагональных элементов. Для этого используется следующая форма записи: $D = \mathrm{diag}(d_1, d_2, \ldots, d_n)$.

Квадратная матрица называется *верхней* (или *правой*) *треугольной*, если все ее поддиагональные элементы — нули, т. е.

$$a_{ij} = 0, \quad \text{если} \quad i > j.$$

Верхние треугольные матрицы обычно обозначают буквами R или U; например,

$$R = \begin{pmatrix} 1 & 2 & 4 \\ 0 & -7 & 1 \\ 0 & 0 & 3 \end{pmatrix}.$$

Если речь идет о матрице, удовлетворяющей условию (2.9) и имеющей больше столбцов, чем строк, говорят, что она является верхней трапециевидной. Это — матрица типа

$$T = \begin{pmatrix} 1 & 2 & 4 & 6 \\ 0 & -3 & 5 & 2 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Аналогично вводятся понятия *нижняя треугольная* и *нижняя трапециевидная матрицы*. Это — матрицы, элемент которых $t_{ij}=0$, если $i<j$. Нижнюю треугольную матрицу обычно обозначают через L. Квадратную треугольную матрицу (верхнюю или нижнюю) называют *единичной треугольной*, если все ее диагональные элементы равны единице.

Набор векторов $\{q_1, q_2, \ldots, q_k\}$ называется *ортогональным*, если $q_i^T q_j=0$, $i \neq j$. При соблюдении дополнительного равенства $q_i^T q_i=1$ говорят, что векторы *ортонормальны*. Квадратную матрицу называют *ортогональной* (*ортонормальной*), если ее столбцы ортогональны (ортонормальны). Для ортонормальной матрицы $Q=(q_1, q_2, \ldots, q_n)$ по определению выполнено равенство $Q^TQ=I_n$.

Матрица вида uv^T, где u и v — векторы, называется *матрицей ранга один*. Каждый столбец матрицы uv^T пропорционален вектору u, а каждая ее строка — вектору v^T. Специальный выбор векторов u и v приводит к специальным матрицам ранга один. Например, если $v=e_i$ (i-й столбец единичной матрицы), то все столбцы произведения uv^T, за исключением i-го, будут нулевыми, а i-й столбец совпадет с вектором u. Матрицы вида $I+\alpha uv^T$, где α — число, принято называть *элементарными*.

2.2.2. ВЕКТОРНЫЕ ПРОСТРАНСТВА

В этом разделе нижний индекс у буквы, обозначающей вектор, используется для ссылки на определенный элемент некоторого набора векторов.

2.2.2.1. Линейные комбинации. Имея набор, состоящий из k векторов $\{a_1, a_2, \ldots, a_k\}$, и набор из k чисел $\{\alpha_1, \alpha_2, \ldots, \alpha_k\}$, можно составить *линейную комбинацию* векторов $\{a_i\}$ с коэффициентами $\{\alpha_i\}$. Для этого надо i-й вектор умножить на i-й скаляр и все полученные таким образом произведения сложить, т. е. линейной комбинацией векторов $\{a_i\}$ с коэффициентами $\{\alpha_i\}$ называется вектор b, определенный равенством

$$b=\alpha_1 a_1+\alpha_2 a_2+\ldots+\alpha_n a_n. \qquad (2.10)$$

Процедура формирования линейной комбинации выполняется над упорядоченными наборами векторов и чисел, что наводит на мысль представить ее в виде матрично-векторной операции. Действительно, эта процедура есть не что иное, как умножение

матрицы A со столбцами $\{a_i\}$ на вектор с компонентами $\{\alpha_i\}$, т. е.

$$b = (a_1 a_2 \ldots a_k) \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \cdot \\ \cdot \\ \cdot \\ \alpha_k \end{pmatrix}.$$

Таким образом, любое произведение матрицы и вектора есть линейная комбинация столбцов матрицы с коэффициентами, равными компонентам вектора.

Линейную комбинацию с нулевыми коэффициентами принято называть *тривиальной*, а если хотя бы один из коэффициентов в линейной комбинации отличен от нуля, ее называют *нетривиальной*.

2.2.2.2. **Линейная зависимость и независимость.** Пусть $\{a_1, a_2, \ldots, a_k\}$ — некоторый набор векторов и вектор a_{k+1} можно представить в виде их линейной комбинации с некоторыми коэффициентами. Тогда говорят, что a_{k+1} *линейно зависим* от векторов $\{a_1, a_2, \ldots, a_k\}$. Так, например, вектор

$$a_3 = \begin{pmatrix} 3 \\ 2 \\ 3 \end{pmatrix} \tag{2.11a}$$

является линейной комбинацией векторов

$$a_1 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \quad a_2 = \begin{pmatrix} 0 \\ 1/2 \\ 0 \end{pmatrix} \tag{2.11b}$$

(поскольку $a_3 = 3a_1 + 4a_2$), т. е. a_3 линейно зависим от $\{a_1, a_2\}$.

Линейную зависимость можно определить и по-другому — как свойство набора векторов. Соответствующее определение звучит так: векторы $\{a_1, a_2, \ldots, a_k, a_{k+1}\}$ линейно зависимы, если нуль представим в виде их нетривиальной линейной комбинации. Так, например, векторы (2.11) линейно зависимы, поскольку $3a_1 + 4a_2 - a_3 = 0$.

Схожими определениями описывается и обратная к рассмотренной ситуация. Если вектор a_{k+1} *не может* быть представлен в виде линейной комбинации векторов набора $\{a_1, a_2, \ldots, a_k\}$, то говорят, что a_{k+1} *линейно независим* от $\{a_1, a_2, \ldots, a_k\}$. Аналогично система векторов $\{a_1, a_2, \ldots, a_k, a_{k+1}\}$ называется *линейно независимой*, если любая их нетривиальная линейная комби-

нация отлична от нуля. К примеру, вектор

$$a_4=\begin{pmatrix}1\\1\\-1\end{pmatrix}$$

непредставим линейной комбинацией a_1 и a_2, в силу чего система $\{a_1, a_2, a_4\}$ оказывается линейно независимой.

Если столбцы некоторой матрицы A линейно независимы, то говорят, что A имеет *полный столбцовый ранг*; матрица A имеет *полный строковый ранг*, если линейно независимы ее строки. Заметим, что при линейной независимости столбцов A из равенства $Ax=0$ с необходимостью следует, что $x=0$, поскольку нетривиальная линейная комбинация независимых столбцов не может быть нулевой.

2.2.2.3. Векторные пространства; подпространства, базис. Возьмем из множества всевозможных векторов размерности n какой-нибудь вектор x. Для любого числа α вектор αx также будет принадлежать этому множеству; ему будет принадлежать и любая сумма вида $x+y$, где y — еще один n-мерный вектор. Следовательно, множество всех n-мерных векторов замкнуто относительно операций суммирования и умножения на скаляр, а потому и относительно операции построения линейных комбинаций. Множества, обладающие данными свойствами, называют *линейными пространствами*. Таким образом, множество всех n-мерных векторов является *линейным векторным пространством*. Его принято обозначать через $\mathbb{R}^n$ (или E^n).

Имея некоторый набор, содержащий k векторов из $\mathbb{R}^n$, скажем $\{a_1, a_2, \ldots, a_k\}$, рассмотрим множество $\mathfrak{L}$ всех векторов, представимых линейными комбинациями от $\{a_1, a_2, \ldots, a_k\}$. Через A обозначим матрицу, чей l-й столбец равен a_l. Тогда вектор b будет принадлежать множеству $\mathfrak{L}$ (этот факт принято отражать записью $b\in\mathfrak{L}$), если $b=Ax$ для некоторого k-мерного вектора x.

Нетрудно проверить, что любая линейная комбинация векторов из множества $\mathfrak{L}$ также является его элементом. Чтобы убедиться в этом, заметим, что из включения $b\in\mathfrak{L}$ следует, что

$$\alpha b=\alpha Ax=A\alpha x, \tag{2.12}$$

т. е. $\alpha b\in\mathfrak{L}$. Далее, если $b\in\mathfrak{L}$ и $c\in\mathfrak{L}$ ($b=Ax$ и $c=Ay$), то

$$b+c=Ax+Ay=A(x+y) \tag{2.13}$$

(использовано свойство дистрибутивности матричного умножения). Значит, сумма $(b+c)$ — тоже вектор из $\mathfrak{L}$. Ну, а коль скоро множество $\mathfrak{L}$ замкнуто относительно операций суммирования и умножения на скаляр, оно будет замкнутым и относительно операции линейного комбинирования. Таким образом, $\mathfrak{L}$ представляет собой *линейное подпространство* пространства $\mathbb{R}^m$.

Подпространство всевозможных линейных комбинаций векторов $\{a_1, a_2, \ldots, a_k\}$ принято называть *натянутым* на них. Когда эти векторы интерпретируются как столбцы некоторой матрицы A, это пространство называют также *столбцовым пространством* A. Про векторы, принадлежащие подпространству, часто говорят, что они *лежат* в нем.

Для каждого нетривиального подпространства $\mathcal{V}$ определено положительное целое число r, именуемое его *рангом* или *размерностью*. Это минимальное число векторов, которых достаточно, чтобы сгенерировать $\mathcal{V}$. Понятно, что соответствующие векторы должны составлять линейно независимую систему, и эту систему принято называть *базисом* подпространства $\mathcal{V}$. В действительности базисом k-мерного подпространства будут *любые* k лежащих в нем линейно независимых векторов. Таким образом, базис определяется неоднозначно. Проиллюстрируем этот факт на примере двумерного подпространства, натянутого на векторы

$$a_1 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \quad a_2 = \begin{pmatrix} 0 \\ 2 \\ 0 \end{pmatrix}. \tag{2.14}$$

Они линейно независимы и потому сами образуют базис. При этом подпространство представляет собой совокупность всевозможных векторов вида

$$b = \begin{pmatrix} \alpha \\ \beta \\ \alpha \end{pmatrix},$$

где α, β — произвольные числа, и ясно, что его можно рассматривать как множество линейных комбинаций от

$$a_1 = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}, \quad a_2 = \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix}.$$

Соответственно эти векторы также являются базисом.

Пусть векторы $\{a_1, a_2, \ldots, a_k\}$ образуют базис подпространства $\mathcal{V}$. Тогда любой вектор из $\mathcal{V}$ представим в виде их линейной комбинации единственным образом. Чтобы доказать это, допустим, что некий вектор b можно представить двумя линейными комбинациями от $\{a_i\}$:

$$b = \alpha_1 a_1 + \alpha_2 a_2 + \ldots + \alpha_k a_k = \beta_1 a_1 + \beta_2 a_2 + \ldots + \beta_k a_k.$$

Это равенство эквивалентно следующему:

$$(\alpha_1 - \beta_1) a_1 + (\alpha_2 - \beta_2) a_2 + \ldots + (\alpha_k - \beta_k) a_k = 0. \tag{2.15}$$

Последнее же в силу линейной независимости векторов $\{a_i\}$ может выполняться только если $\alpha_i-\beta_i$ для всех i. Значит, коэффициенты разложения по базису действительно определены однозначно.

Некоторые вычисления значительно упрощаются, когда векторы базиса взаимно ортогональны. В частности, ортогональность базиса существенно облегчает расчет коэффициентов разложения по нему произвольного вектора b из $\mathfrak{L}$. В самом деле, если

$$b=\alpha_1 a_1+\alpha_2 a_2+\ldots+\alpha_k a_k,$$

то

$$a_i^T b=\alpha_1 a_i^T a_1+\alpha_2 a_i^T a_2+\ldots+\alpha_k a_i^T a_k=\alpha_i a_i^T a_i,$$

потому что все слагаемые справа, за исключением одного, оказываются нулями в силу ортогональности. Так как a_i есть вектор из базиса, он не может быть равным нулю. Соответственно не равно нулю скалярное произведение $a_i^T a_i$, и из последнего равенства мы находим, что коэффициент α_i равен $a_i^T b/a_i^T a_i$.

В рассмотренном ранее примере (2.14) векторы a_1, a_2 взаимно ортогональны. Поэтому коэффициенты разложения вектора

$$b=\begin{pmatrix}3\\-1\\3\end{pmatrix}$$

по a_1, a_2 определяются формулами

$$\alpha_1=\frac{a_1^T b}{a_1^T a_1}=\frac{3+3}{2}=3,$$

$$\alpha_2=\frac{a_2^T b}{a_2^T a_2}=\frac{-2}{4}=-\frac{1}{2},$$

и, действительно,

$$\begin{pmatrix}3\\-1\\3\end{pmatrix}=3\begin{pmatrix}1\\0\\1\end{pmatrix}-\frac{1}{2}\begin{pmatrix}0\\2\\0\end{pmatrix}.$$

2.2.2.4. Нуль-пространство. Для любого подпространства $\mathfrak{L}$ из $\mathfrak{R}^n$ определено его дополнение $\mathfrak{L}'$ в $\mathfrak{R}^n$. Это — множество таких векторов y, для каждого из которых $x^T y=0$ при любом x из $\mathfrak{L}$, т. е. векторов, ортогональных всем x из $\mathfrak{L}$. Множество $\mathfrak{L}'$ является подпространством, поскольку определяющее его свойство ортогональности не нарушается операцией построения линейных комбинаций. Подпространство $\mathfrak{L}'$ называют *ортогональным дополнением* $\mathfrak{L}$ в $\mathfrak{R}^n$. У двух подпространств $\mathfrak{L}$ и $\mathfrak{L}'$ есть только одна общая точка — нуль. Если размерность $\mathfrak{L}$ равна k, его ортогональное дополнение $\mathfrak{L}'$ будет иметь размерность $n-k$.

Любой вектор в $\mathfrak{R}^n$ может быть представлен линейной комбинацией векторов из $\mathfrak{L}$ и $\mathfrak{L}'$, т. е., объединив какие-нибудь базисы

этих двух подпространств, мы тем самым получим базис в $\mathfrak{R}^n$. Когда $\mathfrak{L}$ определяется как подпространство, натянутое на векторы $\{a_1, a_2, \ldots, a_k\}$, его ортогональное дополнение называют также *нуль-пространством* системы $\{a_1, a_2, \ldots, a_k\}$.

Рассмотрим для примера подпространство $\mathfrak{L}$ с базисом $\{a_1, a_2\}$ из (2.14). Тогда вектор

$$a_3 = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix},$$

ортогональный a_1 и a_2, будет лежать в $\mathfrak{L}$ (и будет базисом этого подпространства, так как размерность последнего равна единице).

2.2.3. ЛИНЕЙНЫЕ ПРЕОБРАЗОВАНИЯ

2.2.3.1. Матрицы как преобразования. До сих пор мы рассматривали вещественные матрицы как двумерные наборы чисел либо как объединения столбцовых (или строковых) векторов. Однако на матрицу можно смотреть и как на средство *преобразования векторов* — ведь, умножая некоторый вектор слева на матрицу, мы получаем в результате новый, в общем случае отличный от первого вектор. В этом контексте уместно говорить о *применении* матрицы к вектору. В дальнейшем мы не будем акцентировать внимание на том, что матрица есть не само преобразование, а лишь средство его формального представления, и будем отождествлять матрицы с преобразованиями.

Преобразование вектора матрицей является *линейным*. Это следует из рассмотренных ранее свойств матричного умножения. Для любой матрицы A и любых векторов x, y и скаляров α, β справедливо равенство

$$A(\alpha x + \beta y) = A(\alpha x) + A(\beta y) = \alpha(Ax) + \beta(Ay),$$

так что матричное преобразование линейной комбинации векторов как целого дает линейную комбинацию отдельно преобразованных векторов с теми же коэффициентами.

2.2.3.2. Свойства линейных преобразований. Имея матрицу (преобразование) A, естественно задаться вопросом о том, существуют ли ненулевые векторы, которые A преобразует в нуль, т. е. можно ли найти такой ненулевой x, чтобы выполнялось равенство

$$Ax = 0. \qquad (2.16)$$

Как уже говорилось выше, это возможно лишь в том случае, когда столбцы матрицы A линейно зависимы. Квадратные матрицы с линейно зависимыми столбцами называют *вырожденными*; квадратные матрицы, столбцы которых линейно независимы, называют *невырожденными*.

Коль скоро некоторый некулевой вектор x удовлетворяет равенству (2.16), для любого числа α получим

$$A(y+\alpha x)=Ay+\alpha Ax=Ay, \tag{2.17}$$

т. е. результаты преобразований суммы $y+\alpha x$ и одного вектора y матрицей A идентичны. Соответственно в данном случае невозможно определить по результату, какой вектор преобразовывался.

Предположим теперь, что равенство (2.16) возможно только при нулевом x. Тогда столбцы матрицы A должны быть линейно независимыми, и при этом результаты преобразования разных векторов никогда не совпадут. Действительно, линейная независимость столбцов A служит гарантией того, что равенства $Ax=Ay$ и соответственно $Ax-Ay=A(x-y)=0$ будут выполняться лишь при $x=y$. Тем самым устанавливается правомерность «сокращения» на матрицу полного столбцового ранга уравнений (матричных или векторных), в которых эта матрица состоит левым множителем и в правой, и в левой частях.

2.2.3.3. Обратные матрицы. Рассмотрим преобразования некоторого вектора x невырожденной матрицей A в вектор $b=Ax$. Так как A невырождена, x определен этим равенством однозначно, т. е. существует однозначная обратная к преобразованию A зависимость x от b. Можно показать, что эта зависимость также является матричным преобразованием. Соответствующую матрицу, которая переводит Ax в x, называют *обратной* к A и обозначают через A^{-1}. Она единственна и невырождена.

По определению матрица A^{-1} при любых x обеспечивает равенство

$$A^{-1}(Ax)=x,$$

или, что то же самое,

$$(A^{-1}A-I)x=0.$$

В силу произвольности x отсюда следует, что должно быть $A^{-1}A=I$ (здесь I — единичная матрица). Легко убедиться также, что если A и B — невырожденные матрицы, то справедливо соотношение $(AB)^{-1}=B^{-1}A^{-1}$.

Когда исходная невырожденная матрица обладает специальной структурой, обратная к ней также будет выглядеть специфичной. В частности, для ортонормальной матрицы Q выполняется равенство $Q^{-1}=Q^T$. Матрица, обратная к нижней (верхней) треугольной, также оказывается нижней (верхней) треугольной. При обращении невырожденной элементарной матрицы $I+\alpha uv^T$ мы снова получим элементарную матрицу, причем с теми же двумя векторами: если $\alpha u^Tv\neq -1$, то

$$(I+\alpha uv^T)^{-1}=I+\beta uv^T,$$

где $\beta=-\alpha/(1+\alpha u^Tv)$.

2.2.3.4. Собственные значения; собственные векторы. Для любой квадратной матрицы найдется хотя бы одно специальное число (возможно, комплексное) и соответствующий ему ненулевой вектор u, такие, что

$$Au=\lambda u, \tag{2.18}$$

т. е. преобразование вектора u матрицей A сводится к умножению на λ. Число λ называют *собственным значением* матрицы A, вектор u – ее *собственным вектором*, отвечающим собственному значению λ. Например, число $\lambda=2$ является собственным значением матрицы

$$A=\begin{pmatrix}1 & 2\\ 0 & 2\end{pmatrix},$$

поскольку

$$A\begin{pmatrix}2\\ 1\end{pmatrix}=\begin{pmatrix}4\\ 2\end{pmatrix}=2\begin{pmatrix}2\\ 1\end{pmatrix}.$$

Собственный вектор u всегда определен с точностью до произвольного множителя: умножение его на любое число не нарушит равенства (2.18). Последнее можно переписать следующим образом:

$$(A-\lambda I)u=0.$$

Отсюда видно, что если λ — собственное значение матрицы A, то матрица $A-\lambda I$ оказывается вырожденной. Набор всех собственных значений матрицы A часто обозначают через $\{\lambda_i|A|\}$.

Полезной характеристикой является функция $\Pi(A)$ — произведение всех собственных чисел матрицы A. Эта функция обладает следующим важным свойством: для произвольных матриц A и B выполняется равенство $\Pi(AB)=\Pi(A)\Pi(B)$.

Две матрицы называют *подобными*, когда их наборы собственных значений совпадают. Если W — некоторая невырожденная матрица, произведение WAW^{-1} будет подобным матрице A, поскольку для ее любого собственного значения λ и соответствующего собственного вектора x имеем

$$Ax=\lambda x,\quad WAW^{-1}(Wx)=\lambda(Wx).$$

Вещественная $n\times n$-матрица имеет n собственных значений (не обязательно различных) и максимум n линейно независимых собственных векторов. В общем случае собственные значения вещественной матрицы – комплексные числа. Нас, однако, за редким исключением, будут интересовать собственные значения симметричных матриц (матриц, удовлетворяющих равенству $A=A^T$), а последние обладают следующими двумя свойствами:

(i) все собственные значения симметричной матрицы вещественны;

(ii) симметричная $n\times n$-матрица имеет n различных линейно независимых собственных векторов.

Из собственных векторов симметричной матрицы всегда можно составить ортонормальную систему $\{u_i\}$, $i=1, 2, \ldots, n$, т. е. выбрать их так, чтобы $u_i^T u_j=0$, $i\neq j$; $u_i^T u_i=1$. Иначе говоря, из них можно сформировать ортонормальный базис для $\mathfrak{R}^n$.

Если A – невырожденная матрица, все ее собственные значения будут ненулевыми, а обратные к ним числа будут собственными значениями обратной матрицы A^{-1}.

Максимальное и минимальное собственные числа симметричной матрицы A удовлетворяют равенствам

$$\lambda_{max}[A]=\max_{x\neq 0}\frac{x^TAx}{x^Tx}, \qquad \lambda_{min}[A]=\min_{x\neq 0}\frac{x^TAx}{x^Tx}.$$

Спектральным радиусом A называют число $\rho(A)=\max|\lambda_i[A]|$

2.2.3.5. Знакоопределенность. Если все собственные значения симметричной матрицы A положительны, ее называют *положительно определенной*. Если A – положительно определенная матрица, для любого ненулевого вектора x обеспечено неравенство $x^TAx>0$. Отсюда, в частности, следует, что все диагональные элементы положительно определенной матрицы должны быть положительными. Аналогично положительной определенности вводится понятие *отрицательная определенность*. Это – свойство симметричной матрицы, состоящее в том, что все ее собственные числа отрицательны. Если собственные значения симметричной матрицы A неотрицательны, говорят, что она является *положительно полуопределенной*. Когда среди собственных значений симметричной матрицы A есть и положительные, и отрицательные числа, A называют *знакоопределенной*.

2.2.4. ЛИНЕЙНЫЕ УРАВНЕНИЯ

2.2.4.1. Свойства линейных уравнений. Одной из фундаментальных задач вычислительной линейной алгебры является задача поиска решения системы линейных уравнений: заданы $m\times n$-матрица A и m-мерный вектор b; найти n-мерный вектор x, такой, что

$$Ax=b. \tag{2.19}$$

В этой постановке вектор b часто называют (по очевидным причинам) *правой частью*, а x – *вектором неизвестных*. Как уже говорилось выше, вектор x в (2.19) можно интерпретировать либо как список коэффициентов совпадающей с b линейной комбинации столбцов матрицы A, либо как вектор, который преобразование A переводит в b.

Для того чтобы система (2.19) имела решение, нужно, чтобы вектор b лежал в подпространстве, натянутом на столбцы A. Если b принадлежит этому подпространству, систему называют *совместной*.

Например, система

$$\begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

совместна при любом b, потому что столбцы A линейно независимы, и соответственно натянутое на них подпространство совпадает с $\mathbb{R}^2$.

Система

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \\ 2 \end{pmatrix} \tag{2.20}$$

также совместна, поскольку b есть линейная комбинация столбцов A, причем ее решение единственно.

Если вектор b *не лежит* в подпространстве, натянутом на столбцы A, систему (2.19) называют *несовместной*, и в этом случае никакой вектор x равенства (2.19) не обеспечит. Так, заменив правую часть в (2.20), мы получим *несовместную* систему вида

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}, \tag{2.21}$$

и теперь никакая линейная комбинация столбцов A не совпадает с b.

Если вектор b совместим с A, система (2.19) будет иметь единственное решение тогда и только тогда, когда столбцы A линейно независимы. Если же в этой ситуации столбцы A линейно зависимы, у системы (2.19) *будет* бесконечное множество решений. Чтобы понять, почему это так, вспомним, что в силу линейной зависимости найдется ненулевой вектор z, такой, что $Az=0$. Соответственно, если x — некоторое решение системы (2.19), то для любого числа δ получим

$$A(x+\delta z) = Ax+\delta Az = Ax = b,$$

т. е. вектор $x+\delta z$ тоже будет решением (2.19). К примеру, вектор $x=(3,\ 0)^T$ есть решение совместной системы

$$\begin{pmatrix} 1 & -2 \\ 0 & 0 \\ 1 & -2 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 3 \\ 0 \\ 3 \end{pmatrix}, \tag{2.22}$$

и, так как

$$\begin{pmatrix} 1 & -2 \\ 0 & 0 \\ 1 & -2 \end{pmatrix}\begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix},$$

ее решениями будут также векторы вида

$$\begin{pmatrix}3\\0\end{pmatrix}+\delta\begin{pmatrix}2\\1\end{pmatrix}.$$

2.2.4.2. **Нормы векторов и матрицы.** Впоследствии нам часто будет необходимо как-то «измерить» векторы и матрицы для того, чтобы можно было говорить, что один вектор (матрица) «больше» или «меньше» другого (другой). Правило, по которому с векторами или матрицами связываются некоторые неотрицательные числа, имеющие смысл их мер, дается определением «нормы».

Норма вектора $\|\ \|$ — это некоторая скалярная функция, обладающая тремя свойствами:

(i) $\|x\|\geqslant 0$ для любого вектора x и $\|x\|=0$ в том и только в том случае, если $x=0$;

(ii) для любого вещественного числа δ выполняется равенство $\|\delta x\|=|\delta|\|x\|$;

(iii) для любых двух векторов x и y справедливо неравенство $\|x+y\|\leqslant\|x\|+\|y\|$ (*неравенство треугольника*).

Целое семейство векторных норм, именуемых *p-нормами* и обозначаемых через $\|x\|_p$, определяется формулой вида

$$\|x\|_p=\left(\sum_i^n|x_i|^p\right)^{1/p}$$

При любом целом p она задает функцию, удовлетворяющую требованиям (i) — (iii).

Наиболее часто используются p-нормы с p 1, 2, ∞. Квадратичную норму ($p=2$) иногда называют *евклидовой*; заметим, что $\|x\|_2^2=x_1^2+\ldots+x_n^2=x^Tx$. При p ∞ получаем $\|x\|_\infty=\max_i|x_i|$. Например, если $x^T=(1, -2, -3)$, то $\|x\|_\infty=3$, $\|x\|_1=6$ и $\|x\|_2=\sqrt{14}$.

Существует несколько полезных неравенств, связывающих значение скалярного произведения двух векторов с их нормами. В частности, рассматривая векторы x, y и $y-x$ как стороны треугольника в $\mathbb{R}^n$ и обозначив через θ угол между x и y, по формуле косинусов получим

$$\|y+x\|_2^2=\|y\|_2^2+\|x\|_2^2-2\|y\|_2\|x\|_2\cos\theta.$$

Заменив выражение $\|y-x\|_2^2$ скалярным произведением $(y-x)^T(y-x)$ и раскрыв скобки, отсюда придем к равенству

$$\cos\theta=\frac{y^Tx}{\|x\|_2\|y\|_2},$$

а поскольку значение $\cos\theta$ лежит между -1 и $+1$, это позволяет утверждать, что

$$|y^Tx|\leqslant\|x\|_2\|y\|_2.$$

Данное соотношение называют *неравенством Шварца*.

Научившись измерять векторы, желательно научиться приписывать нормы и матрицам. *Норма матрицы*, которую, как и норму вектора, принято обозначать через $\|\cdot\|$,— это некоторая скалярная функция, обладающая тремя свойствами:

(i) $\|A\| \geqslant 0$ для любой матрицы A и $\|A\| = 0$ в том и только в том случае, если A — нулевая матрица;

(ii) $\|\delta A\| = |\delta| \|A\|$;

(iii) $\|A+B\| \leqslant \|A\| + \|B\|$.

Поскольку матрицы можно перемножать, получая тем самым новые матрицы, целесообразно подчинить матричную норму еще одному условию, а именно потребовать, чтобы

(iv) $\|AB\| \leqslant \|A\| \|B\|$.

Матричные нормы удобно определять через векторные. Делается это так. Задавшись какой-нибудь векторной нормой $\|\cdot\|$, рассмотрим для матрицы A значения $\|Ax\|$ при всевозможных x, удовлетворяющих равенству $\|x\| = 1$. Среди них обязательно найдется максимальное. Его и возьмем в качестве нормы матрицы A. Такую матричную норму принято называть *индуцированной* или *подчиненной* векторной. Итак, $\|A\|$ можно определять по формуле

$$\|A\| = \max_{\|x\|=1} \|Ax\|, \tag{2.23}$$

где справа стоит векторные норма и максимум достигается при каком-то (или каких-то) x.

Три нормы $m \times n$-матрицы A, подчиненные трем введенным ранее векторным нормам, таковы:

$\|A\|_1 = \max\limits_{1 \leqslant i \leqslant n} \left(\sum\limits_{j=1}^{m} |a_{ij}| \right)$ — максимум суммы модулей элементов в столбце;

$\|A\|_2 = (\lambda_{max}(A^T A))^{1/2}$ — квадратный корень максимального собственного значения симметричной матрицы $A^T A$;

$\|A\|_\infty = \max\limits_{1 \leqslant i \leqslant m} \left(\sum\limits_{j=1}^{n} |a_{ij}| \right)$ — максимум суммы модулей элементов в строке.

Квадратичную норму $\|A\|_2$ иногда называют еще *спектральной нормой* матрицы.

Помимо трех рассмотренных часто используется так называемая *норма Фробениуса* $\|\cdot\|_F$, которая не подчинена векторной. Она возникает, если интерпретировать $m \times n$-матрицу A как вектор с mn компонентами, и представляет собой его евклидову норму:

$$\|A\|_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}^2 \right)^{1/2}.$$

Векторная норма $\|\cdot\|$ и матричная норма $\|\cdot\|'$ называются *согласованными*, если для любых A и x выполняется неравенство

$$\|Ax\| \leqslant \|A\|' \|x\|. \tag{2.24}$$

В частности, оно соблюдено (по определению) для любой пары векторной и подчиненной ей матричной норм, причем на некотором (некоторых) x достигается равенство. Евклидова векторная норма и матричная норма Фробениуса также являются согласованными.

2.2.4.3. Теория возмущений; число обусловленности. Выше было установлено, что линейная система

$$Ax=b \tag{2.25}$$

с квадратной невырожденной матрицей A однозначно разрешима для любой правой части b. О том, как искать решения таких систем, речь пойдет ниже. Однако прежде, чем заняться соответствующими методами, полезно рассмотреть вопрос о влиянии на x из (2.25) малых изменений (возмущений) правой части и элементов матрицы.

Точное решение (2.25) дается формулой $x=A^{-1}b$. Точным решением системы с возмущенной правой частью $b+\delta b$ будет вектор $x+\delta x$, удовлетворяющий равенству

$$A(x+\delta x)=b+\delta b.$$

(«Приставкой» δ здесь и в дальнейшем помечаются малые приращения векторов или матриц.) Соответственно

$$x+\delta x=A^{-1}(b+\delta b),$$

а так как $x=A^{-1}b$, отсюда ясно, что

$$\delta x=A^{-1}\delta b.$$

Для измерения δx воспользуемся какой-нибудь парой совместимых векторной и матричной норм. Это приведет к оценке

$$\|\delta x\|\leqslant\|A^{-1}\|\,\|\delta b\|, \tag{2.26}$$

где при некотором δb возможно равенство. Таким образом, возмущение точного решения может превосходить возмущение правой части не более, чем в $\|A^{-1}\|$ раз.

Для определения *относительного* эффекта того же самого возмущения заметим, что

$$\|b\|\leqslant\|A\|\,\|x\|. \tag{2.27}$$

С учетом этого неравенства из (2.26) легко получить такое соотношение:

$$\frac{\|\delta x\|}{\|x\|}\leqslant\|A\|\|A^{-1}\|\frac{\|\delta b\|}{\|b\|}. \tag{2.28}$$

Предположим теперь, что матрица системы (2.25) возмущается на δA. Тогда возмущенное решение (2.25) $x+\delta x$ будет обеспечивать равенство $(A+\delta A)(x+\delta x)=b$, откуда следует, что

$$\delta x=-A^{-1}\delta A\,(x+\delta x)$$

и соответственно

$$\|\delta x\| \leqslant \|A^{-1}\| \, \|\delta A\| \, \|x+\delta x\|,$$

или, что то же самое,

$$\frac{\|\delta x\|}{\|x+\delta x\|} \leqslant \|A^{-1}\| \|A\| \frac{\|\delta A\|}{\|A\|}.$$

Вводя относительное возмущение матрицы, это неравенство можно переписать так:

$$\frac{\|\delta x\|}{\|x+\delta x\|} \leqslant \|A\| \|A^{-1}\| \frac{\|\delta A\|}{\|A\|}. \qquad (2.29)$$

И в (2.28), и в (2.29) верхней границей относительного изменения точного решения служит произведение порождающего его относительного возмущения данных (правой части или матрицы) на одно и то же число $\|A\| \, \|A^{-1}\|$. Это число называют *числом обусловленности матрицы* A в процедуре решения линейной системы (указание на процедуру обычно опускается) и обозначают через $\operatorname{cond}(A)$.

Поскольку для любой подчиненной матричной нормы выполняется равенство $\|I\|=1$ и по определению $I=A^{-1}A$, справедлива оценка

$$1=\|I\| \leqslant \|A\| \, \|A^{-1}\|,$$

так что $\operatorname{cond}(A) \geqslant 1$ для любой матрицы.

Число обусловленности матрицы A характеризует *максимальный* эффект от возмущений в b и A при решении (2.25). Из оценок (2.28) и (2.29) следует, что при «большом» $\operatorname{cond}(A)$ точное решение системы может существенно измениться даже при малом изменении данных. Матрицы A с «большими» $\operatorname{cond}(A)$ принято называть *плохо обусловленными*, а с «малыми» $\operatorname{cond}(A)$ — *хорошо обусловленными*.

Чтобы проиллюстрировать сказанное, рассмотрим систему с матрицей A и правой частью b вида

$$A=\begin{pmatrix} .550 & .423 \\ .484 & .372 \end{pmatrix}, \quad b=\begin{pmatrix} .127 \\ .112 \end{pmatrix}. \qquad (2.30)$$

Точное решение уравнения $Ax=b$ есть

$$x=\begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$

При возмущенной правой части

$$b+\delta b=\begin{pmatrix} .12707 \\ .11228 \end{pmatrix}, \quad \delta b=\begin{pmatrix} .00007 \\ .00028 \end{pmatrix}$$

точным решением становится вектор

$$x+\delta x=\begin{pmatrix} 1.7 \\ -1.91 \end{pmatrix}, \quad \delta x=\begin{pmatrix} .7 \\ -.91 \end{pmatrix}.$$

В данном случае в норме $\|\cdot\|_\infty$ получим $\|\delta b\|/\|b\| = .0022$, в то время как $\|\delta x\|/\|x\| = .91$. Видно, что относительное изменение решения намного превышает относительное изменение правой части.

Проварьируем теперь (2, 1)-й элемент матрицы A так, чтобы

$$A + \delta A = \begin{pmatrix} .530 & .423 \\ .483 & .372 \end{pmatrix}.$$

При этом точное решение возмущенной системы

$$(A + \delta A)(x + \delta x) = b,$$

округленное до четырех знаков, выглядит следующим образом:

$$x + \delta x = \begin{pmatrix} -.4535 \\ .8809 \end{pmatrix}.$$

И здесь относительное изменение решения намного превышает относительное изменение исходных данных (матрицы A). Объяснение в обоих случаях одно и то же — плохая обусловленность матрицы из (2.30). Обратная к ней матрица A^{-1} (с точностью до четырех знаков) имеет вид

$$A^{-1} = \begin{pmatrix} -2818 & 3205 \\ 3667 & -4167 \end{pmatrix}.$$

и соответственно в норме $\|\cdot\|_\infty$ будет $\|A\| = .973$, $\|A^{-1}\| = 7834$, $\mathrm{cond}(A) = 7622$. Таким образом, неожиданно большие на первый взгляд вариации решения еще очень далеки от тех, которые могли бы получиться в худшем случае.

Следует подчеркнуть, что представленный анализ возмущений проводился в терминах *точных решений* и, значит, касался только внутренних свойств рассмотренной математической задачи. Плохая обусловленность была определена безотносительно к проблемам вычислений с конечной точностью и по сути дела представляет собой чисто геометрическую характеристику матричного преобразования. Она говорит о том, что коэффициенты растяжения разных векторов преобразованием A сильно разбросаны.

2.2.4.4. Треугольные линейные системы. Рассмотрим задачу численного поиска решения невырожденной линейной системы вида

$$Ax = b. \tag{2.31}$$

В каких случаях это решение может быть получено простой процедурой? Предположим, что матрица A является треугольной (скажем, нижней треугольной). Так что система (2.31) на самом деле

выглядит так:

$$\begin{pmatrix} l_{11} & & & & \\ l_{21} & l_{22} & & & \\ l_{31} & l_{32} & l_{33} & & \\ \cdot & \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ l_{n1} & l_{n2} & l_{n3} & \cdot\ \cdot & l_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \cdot \\ b_n \end{pmatrix}.$$

В данном случае первое уравнение содержит единственную неизвестную x_1, значение которой сразу определяется посредством одного деления:

$$x_1 = \frac{b_1}{l_{11}}.$$

Во второе уравнение входят две искомые величины x_1 и x_2, но первая нам уже известна, и поэтому этого уравнения достаточно, чтобы найти вторую. Она вычисляется по формуле

$$x_2 = \frac{b_2 - l_{21}x_1}{l_{22}}.$$

Продолжая в том же духе, мы на каждой итерации будем извлекать из очередного уравнения значение очередной неизвестной, все время имея дело с уравнениями, в которых все искомые величины, кроме одной, уже найдены. Таким образом, треугольная система легко решается последовательным вычислением неизвестных в определенном порядке. Невырожденность матрицы здесь существенна, так как в ее отсутствие хотя бы один из элементов l_{ii} будет нулем, и вычисление придется прервать на соответствующем шаге.

Процедура, аналогичная рассмотренной, может быть использована и для поиска решения системы с верхней треугольной матрицей, только тогда надо вычислить сначала x_n, затем x_{n-1} и т. д. Ее называют *подстановкой назад*, а процедуру решения нижней треугольной системы — *подстановкой вперед*.

Количество арифметических операций, необходимых для решения треугольной системы общего вида описанным способом, оценивается так: для вычисления первой неизвестной нужно одно деление; расчет второй неизвестной включает одно умножение, одно сложение и одно деление; при вычислении k-й неизвестной потребуется выполнить $k-1$ сложений, $k-1$ умножений и одно деление и т. д. вплоть до $k=n$. Просуммировав все это, получим, что полное число сложений и полное число умножений равны $\frac{1}{2}n^2$, а полное число делений равно n.

Еще проще решаются системы (2.31) с *диагональной* матрицей A (которую можно рассматривать как особую разновидность треугольной). В этом случае каждое уравнение содержит в точности одну неизвестную, значение которой никак не связано со значе-

ниями других неизвестных. Для решения диагональной линейной системы нужно выполнить только n делений и ничего больше.

2.2.4.5. Анализ ошибок. Выше (разд. 2.1.7) было введено понятие *обратного анализа ошибок* — подхода, который применительно к любой вычислительной процедуре состоит в интерпретации *численного решения* исходной задачи как *точного решения* некоторой возмущенной задачи и в оценке близости между этими задачами. Такой анализ очень важен для понимания численных алгоритмов, но, к сожалению, его детали всегда ужасно утомительны. Поэтому стандартная техника будет проиллюстрирована на весьма кратко изложенной процедуре обратного анализа ошибок для алгоритма решения нижней треугольной системы линейных уравнений подстановкой вперед. При этом будут использованы сведения о вычислениях с плавающей запятой, данные в разд. 2.1.4.

Первый шаг решения системы $Lx = b$, где L — нижняя треугольная матрица, состоит в расчете первой компоненты вектора неизвестных. Ее численно найденное значение будет равно

$$x_1 = fl\left(\frac{b_1}{l_{11}}\right) = \frac{b_1}{l_{11}(1+\delta_1)}, \tag{2.32}$$

где величина $|\delta_1|$ ограничена сверху малым числом ε_M, отражающим точность машинных вычислений. Следовательно, x_1 окажется точным решением уравнения

$$l_{11}(1+\delta_1)x_1 = b_1.$$

Далее, расчетным значением второй компоненты x будет

$$x_2 = fl\left(\frac{b_2 - l_{21}x_1(1+\delta_1)}{l_{22}}\right) = \frac{b_2 - l_{21}x_1(1+\xi_{21})(1+\eta_{21})+b_2(1+\eta_{22})}{l_{22}(1+\delta_2)}, \tag{2.33}$$

где ξ_{21}, η_{22}, δ_2 обусловлены погрешностями умножения, сложения и деления соответственно. Модуль каждой из этих величин ограничен сверху константой ε_M. После объединения слагаемых, связанных с разными ошибками, (2.33) можно переписать так:

$$l_{21}(1+\beta_{21})x_1 + l_{22}(1+\beta_{22})x_2 = b_2.$$

Здесь $|\beta_{21}|$ и $|\beta_{22}|$ не превосходят произведения некоторого фиксированного числа на ε_M.

Продолжая эти рассуждения, нетрудно убедиться, что расчетное значение x_r, найденное на r-м шаге, будет удовлетворять уравнению

$$l_{r1}(1+\beta_{r1})x_1 + l_{r2}(1+\beta_{r2})x_2 + \ldots + l_{rr}(1+\beta_{rr})x_r = b_r,$$

где каждая из величин $|\beta_{ij}|$ ограничена сверху произведением ε_M на некоторую линейную функцию от номера итерации r. Таким образом, показано, что численное решение x является *точным*

решением возмущенной треугольной системы вида

$$(L+\delta L)x=b, \tag{2.34}$$

в которой модуль каждого элемента r-й строки матрицы δL не превосходит выражения, содержащего машинную точность, номер r и соответствующий элемент исходной матрицы L. Значит, хотя истинное возмущение δL, возникающее при конкретной правой части b, является функцией генерируемого численного решения, оценки для $|(\delta L)_{ij}|$ сверху не зависят ни от b, ни от x, ни от числа обусловленности матрицы L. Равенство (2.34) в силу этих оценок означает, что *численное решение* исходной системы оказывается *точным решением системы, близкой к ней*.

2.2.5. РАЗЛОЖЕНИЯ МАТРИЦ

Во многих задачах вычислительной линейной алгебры оказывается полезным представлять исходные матрицы произведениями или суммами других матриц специального вида. Такие представления называют *разложениями* или *факторизациями*.

Несколько разложений, наиболее часто употребляемых в оптимизационных алгоритмах, будут более или менее подробно рассмотрены в данном разделе.

2.2.5.1. LU-разложение; гауссовы исключения. Один из подходов к решению невырожденной линейной системы общего вида

$$Ax=b \tag{2.35}$$

состоит в преобразовании ее в эквивалентную систему с треугольной матрицей. Для этого надо построить такую невырожденную матрицу M, чтобы произведение MA было треугольным. Затем можно искать x как решение эквивалентной (2.35) системы вида $MAx=Mb$ методом подстановки вперед или назад. Интересно отметить, что наиболее часто употребляемая формальная процедура построения M в значительной степени совпадает с известной всем со школьной скамьи процедурой решения небольших систем линейных алгебраических уравнений с помощью карандаша и бумаги.

Рассмотрим линейную систему

$$\begin{aligned} 2x_1+2x_2+\ x_3&=5,\\ x_1+3x_2+2x_3&=6,\\ 4x_1-2x_2+3x_3&=5, \end{aligned} \tag{2.36}$$

у которой

$$A=\begin{pmatrix}2 & 2 & 1\\ 1 & 3 & 2\\ 4 & -2 & 3\end{pmatrix},\quad b=\begin{pmatrix}5\\6\\5\end{pmatrix}. \tag{2.37}$$

В качестве первого шага ее решения «исключим» переменную x_1 из второго и третьего уравнений, вычитая из второй строки матрицы системы первую, поделенную на 2, а из третьей — первую, умноженную на 2. В результате придем к матрице с нулями во (2, 1)-й и (3, 1)-й позициях. Полностью она будет выглядеть так:

$$A^{(2)} = \begin{pmatrix} 2 & 2 & 2 \\ 0 & 2 & 3/2 \\ 0 & -6 & 1 \end{pmatrix}.$$

Тем самым один шаг преобразования исходной матрицы в верхнюю треугольную проделан. Чтобы завершить преобразование, надо умножить вторую строку вновь полученной матрицы на 3 и сложить результат с третьей строкой. Это приведет к искомой верхней треугольной матрице вида

$$A^{(3)} = \begin{pmatrix} 2 & 2 & 2 \\ 0 & 2 & 3/2 \\ 0 & 0 & 11/2 \end{pmatrix}.$$

Изложенный способ последовательной триангуляризации матрицы поочередным обнулением поддиагональных элементов называется *гауссовыми исключениями*. После выполнения $k-1$-го шага этой процедуры первые $k-1$ столбцов матрицы системы уже преобразованы к верхнему треугольному виду, и значения их компонент в дальнейшем не изменяются. На k-м шаге исключения обнуляют все (j, k)-е элементы с $j>k$, для чего k-я строка поочередно вычитается из j-х с весами m_{jk}, где

$$m_{jk} = \frac{a_{jk}^{(k)}}{a_{kk}^{(k)}}, \quad j = k+1, \ \ldots, \ n.$$

Диагональный элемент $a_{kk}^{(k)}$ частично преобразованной матрицы называется *ведущим элементом* k-го шага. Он имеет особое значение, так как, если на каком-то шаге окажется нулевым, вычисления придется прервать. Числа $\{m_{jk}\}$, $j=k+1, \ \ldots, \ n$, называют *множителями* k-го шага.

После того как матрица преобразована в верхнюю треугольную, все действия, совершенные по ходу этого преобразования с ее строками, надо проделать с компонентами правой части. Тем самым построение треугольной системы, эквивалентной исходной, будет завершено, и останется только одно — найти ее решение методом подстановки назад. В примере (2.36) вектор b трансформируется следующим образом:

$$b = \begin{pmatrix} 5 \\ 6 \\ 5 \end{pmatrix} \to b^{(2)} = \begin{pmatrix} 5 \\ 7/2 \\ -5 \end{pmatrix} \to b^{(3)} = \begin{pmatrix} 5 \\ 7/2 \\ 11/2 \end{pmatrix},$$

а метод подстановки назад дает $x_3=1$, $x_2=1$, $x_1=1$.

С формальной точки зрения каждый шаг исключения представляет собой умножение слева матрицы системы на специфическую элементарную матрицу (разд. 2.2.1.5). Например, элементарная матрица, осуществляющая первый шаг исключения для системы (2.36), такова:

$$M_1 = I - \begin{pmatrix} 0 \\ 1/2 \\ 2 \end{pmatrix} (1\ 0\ 0) = \begin{pmatrix} 1 & 0 & 0 \\ -1/2 & 1 & 0 \\ -2 & 0 & 1 \end{pmatrix}.$$

Заметим, что M_1 — единичная нижняя треугольная матрица, которая отличается от диагональной только первым столбцом (столбцом с номером шага исключения), где есть ненулевые субдиагональные элементы.

Итак, если в случае с n неизвестными описанный процесс гауссовых исключений не прервется из-за того, что какой-то ведущий элемент окажется нулем, то этот процесс можно будет проинтерпретировать как поэтапное умножение A слева на пакет $\{M_i\}$ из $n-1$ единичных нижних треугольных матриц с целью получить в результате верхнюю треугольную матрицу U:

$$M A \cdots M_{n-1} \ldots M_2 M_1 A = U. \tag{2.38}$$

Каждая M_i преобразует i-й столбец частично триангуляризованной матрицы к желаемому виду. Следует подчеркнуть, что формализм описания процедуры исключений с помощью матриц $\{M_i\}$ вовсе не предполагает запоминания их целиком при реализации алгоритма на машине; i-я матрица полностью задается $n-i$ множителями, так что исчерпывающие данные о преобразовании поместятся, например, в освобождающийся поддиагональный блок записи A.

После того как A триангуляризована, значение x ищется как решение верхней треугольной системы

$$MAx = Ux = Mb, \tag{2.39}$$

правая часть которой Mb вычисляется применением к вектору b тех же преобразований, которым подвергались строки A.

Поскольку матрица M невырождена, равенство (2.38) можно переписать так:

$$A = M^{-1}U = M_1^{-1}M_2^{-1} \ldots M_{n-1}^{-1}U. \tag{2.40}$$

В силу специальной структуры матриц M_i обратные к ним матрицы также будут единичными нижними треугольными. Читатель без труда убедится, что M_i^{-1} отличается от M_i только знаками элементов i-го столбца. Соответственно, если переобозначить M^{-1} через L, равенство (2.40) принимает вид представления A произведением единичной нижней треугольной матрицы L на

верхнюю треугольную матрицу U:

$$A=LU.$$

Это представление называется *LU-разложением* A.

Резюмируя сказанное, мы можем теперь представить процедуру решения системы $Ax=b$ гауссовыми исключениями как трехэтапный алгоритм, в котором требуется

(i) вычислить L и U, такие, что $A=LU$ (ведущие элементы предполагаются ненулевыми);

(ii) решить систему $Ly=b$ (что эквивалентно формированию $y=Mb$);

(iii) решить систему $Ux=y$.

Если не принимать во внимание малых n, то можно сказать, что основной объем вычислений по методу гауссовых исключений приходится на этап формирования L и U. Для этого требуется выполнить $^1/_3n^3$ умножений, в то время как сложность второго и третьего этапов характеризуется n^2 умножениями.

До сих пор один весьма существенный недостаток метода гауссовых исключений упоминался лишь мимоходом. Вспомним, что описанную выше процедуру придется прервать, если на каком-то шаге ведущий элемент окажется нулем, а это возможно даже для невырожденных матриц. Возьмем, к примеру, матрицу

$$A=\begin{pmatrix}0 & 1 & 1\\ 2 & 3 & 4\\ 1 & 0 & 7\end{pmatrix}.$$

Обнулить поддиагональные элементы ее первого столбца вычитанием взвешенной первой строки из второй и третьей не удастся. Правда, эту трудность преодолеть легко: достаточно *переставить* строки 1 и 2 (или 1 и 3) — и (1, 1)-й элемент станет ненулевым. При невырожденной матрице подобные перестановки найдутся всегда. Регламентирующие их правила называют стратегиями выбора ведущего элемента.

С точки зрения теории отсутствие нулевых ведущих элементов — это единственное, что нужно для успешной работы процедуры гауссовых исключений. Жизнь, однако, устроена сложнее. Для вычислителей давно уже стало аксиомой (или почти аксиомой), что процесс, неопределенный при нулевом значении какой-то величины, будет численно неустойчивым (т. е. может сопровождаться большими ошибками), когда она «мала». Применительно к гауссовым исключениям эта мудрость подтверждается вполне строго, и вывод таков: с помощью перестановок надо избегать *не только нулевых, но и малых ведущих элементов*.

Среди различных стратегий выбора ведущего элемента чаще всего используют так называемое *частичное упорядочение*. Суть

в том, что при выборе очередной, k-й ведущей строки предпочтение отдается той из еще не побывавших ведущими строк, в которой модуль элемента на k-го столбца максимален. Она и становится k-й, «поменявшись местами» со строкой, которая занимала это место ранее (а указанный элемент становится ведущим). Например, если бы частичное упорядочение использовалось на первом шаге триангуляризации матрицы (2.37), то строки 1 и 3 следовало бы переставить, получив тем самым матрицу

$$A=\begin{pmatrix}4 & -2 & 3\\ 1 & 3 & 2\\ 2 & 2 & 1\end{pmatrix}.$$

Подобные перестановки возможны на каждом шаге. Формально они описываются умножением на *перестановочную* матрицу, элементы которой могут быть только нулями или единицами, причем в каждом столбце и каждой строке есть в точности по одной единице. Умножение некоторой матрицы на перестановочную слева меняет местами строки, справа — столбцы. Например, перестановка строк 1 и 3 матрицы A в (2.37) достигается умножением A слева на

$$P_{13}=\begin{pmatrix}0 & 0 & 1\\ 0 & 1 & 0\\ 1 & 0 & 0\end{pmatrix}.$$

Дополнение гауссовых исключений описанной выше стратегией выбора ведущей строки означает, что всякий раз, когда потребуется поменять строки местами, частично триангуляризованная матрица будет умножена слева на соответствующую перестановочную. В результате это приведет к LU-разложению матрицы, которая получается из исходной перестановкой строк. Таким образом, будет справедливо равенство $PA=LU$, где P — некоторая перестановочная матрица. При этом модули всех поддиагональных элементов матрицы L будут лежать между нулем и единицей. Последнее вытекает из способа выбора ведущего элемента: его модуль всегда максимален в рассматриваемой части очередного столбца, и, стало быть, модули всех множителей (отношений прочих элементов этой части столбца к ведущему) меньше или равны единице.

Обратный анализ ошибок для гауссовых исключений с частичным упорядочением — довольно громоздкая процедура. Поэтому ограничимся сводкой результатов: *численно найденные* L и U будут *точными треугольными сомножителями* разложения некоторой матрицы $P(A+E)$. Введем величину

$$g=\frac{\max\limits_{k}|a_{ij}^{(k)}|}{\max|a_{ij}|},$$

которую называют *фактором роста*. Она отражает «рост» элементов промежуточных, частично триангуляризованных матриц по отношению к соответственным элементам исходной матрицы. Тогда справедливо неравенство

$$\|E\|_\infty \leqslant n^3 \gamma^2 \varepsilon_M g \|A\|_\infty,$$

где γ — константа порядка единицы, не зависящая ни от A, ни от n, а ε_M — машинная точность.

Представленная оценка точности показывает, что основным источником неустойчивости алгоритма является рост элементов промежуточных матриц. В отсутствие разумной стратегии выбора ведущего элемента этому росту ничто не препятствовало бы, и тогда численно найденные L, U могли бы определить матрицу, не имеющую никакого отношения к исходной.

2.2.5.2. LDL^T-разложение и разложение по Холескому. Любая положительно определенная матрица A может быть представлена произведением вида

$$A = LDL^T, \tag{2.41}$$

где L — *единичная* нижняя треугольная матрица, а D — диагональная матрица со строго положительными диагональными элементами. Это представление называется *LDL^T-разложением*.

Поскольку диагональ D положительна, равенство (2.41) можно переписать так:

$$A = LD^{1/2}D^{1/2}L^T = \tilde{L}\tilde{L}^T = R^TR. \tag{2.42}$$

Здесь L — невырожденная нижняя треугольная матрица общего вида, а R — невырожденная верхняя треугольная матрица общего вида. Это представление принято называть *разложением по Холескому*, а матрицу R — фактором Холецкого или «квадратным корнем» из A (по аналогии со скалярным случаем). Впредь, учитывая близость (2.41) и (2.40), мы будем называть разложением по Холескому *и то и другое* представление.

Факторы Холецкого могут быть вычислены путем прямого поэлементного сопоставления. По определению имеем

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{12} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{13} & a_{23} & a_{33} & \cdots & a_{3n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{1n} & a_{2n} & a_{3n} & \cdots & a_{nn} \end{pmatrix} = \begin{pmatrix} r_{11} & & & & \\ r_{12} & r_{22} & & & \\ r_{13} & r_{23} & r_{33} & & \\ \cdots & \cdots & \cdots & \cdots & \\ r_{1n} & r_{2n} & r_{3n} & \cdots & r_{nn} \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & \cdots & r_{1n} \\ & r_{22} & r_{23} & \cdots & r_{2n} \\ & & r_{33} & \cdots & r_{3n} \\ & & & \cdots & \cdots \\ & & & & r_{nn} \end{pmatrix} \tag{2.43}$$

Соответственно, приравнивая (1, 1)-й элемент матрицы слева и отвечающий ему элемент произведении справа, получим $a_{11} = r_{11}^2$, $r_{11} = \sqrt{a_{11}}$. После этого, сравнивая первые строки левой и правой

частей (2.43), найдем все оставшиеся элементы первой строки R. В самом деле, это сравнение показывает, что

$$a_{12}=r_{11}r_{12},\quad a_{13}=r_{11}r_{13},\ \ldots,\quad a_{1n}=r_{11}r_{1n}.$$

Далее, элементы (2, 2) матриц R и A удовлетворяют равенству $a_{22}=r_{12}^2+r_{22}^2$. Отсюда, зная r_{12}, находим r_{22}, затем вторую строку R и т. д. Приведенный алгоритм иногда называют *построчной* факторизацией Холесского. Элементы матрицы R можно вычислить и в таком порядке: r_{11}, r_{12}, r_{22}, r_{13}, r_{23}, r_{33}, ... По понятным причинам соответствующую процедуру принято называть *столбцовой* факторизацией.

Для построения разложения по Холесскому $n\times n$-матрицы требуется выполнить около $\frac{1}{6}n^3$ операций умножения (сложения) и вычислить n квадратных корней (последнее — только для формы (2.42)).

Замечательная особенность алгоритма Холесского состоит в том, что в *отличие* от гауссовых исключений он численно устойчив *без каких-либо перестановок*. Это свойство определяется следующим соотношением между элементами A и R:

$$r_{1k}^2+r_{2k}^2+\ldots+r_{kk}^2=a_{kk},\quad k=1,\ \ldots,\ n.$$

В силу него существует априорное ограничение сверху на модули элементов R: каждый из них не превосходит максимальной величины a_{kk}. Соответственно «лавинообразный рост» элементов R невозможен независимо от того, будут ведущие элементы (в данном случае естественно назвать так элементы r_{ii}) малыми или нет. Обратный анализ ошибок дает для численной реализации алгоритма Холесского формулу $RR^T=A+E$ с ограничением на $\|E\|$ сверху, аналогичным полученному для гауссовых исключений, но не содержащим фактора роста.

Было бы заманчиво предположить, что любую симметричную матрицу можно представить в виде (2.41), если не оговаривать знаки диагональных элементов D. К сожалению, что следует подчеркнуть особо, данная гипотеза ошибочна. Для произвольной наперед взятой знакоопределенной симметричной матрицы LDL^T-представление может оказаться невозможным; например, его не существует для матрицы

$$A=\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}.$$

Если же оно и возможно, то гарантировать численную устойчивость приведенного выше алгоритма уже не удается, поскольку никаких априорных ограничений на субдиагональные элементы L в рассматриваемом случае не будет. Так, при малых ε элементы

разложения

$$\begin{pmatrix} \varepsilon & 1 \\ 1 & \varepsilon \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1/\varepsilon & 1 \end{pmatrix} \begin{pmatrix} \varepsilon & 0 \\ 0 & \varepsilon - 1/\varepsilon \end{pmatrix} \begin{pmatrix} 1 & 1/\varepsilon \\ 0 & 1 \end{pmatrix}$$

становятся очень большими.

2.2.5.3. *QR*-разложение. В разд. 2.2.5.1 был изложен способ преобразования матрицы в верхнюю треугольную применением последовательности элементарных нижних треугольных матриц. Аналогичного результата можно достичь и по-другому, используя набор элементарных *ортогональных* матриц.

Для любого ненулевого вектора w определено так называемое преобразование *Хаусхолдера* с элементарной симметричной матрицей

$$H = I - \frac{1}{\beta} ww^T,$$

где $\beta = \frac{1}{2}\|w\|_2^2$. Эта матрица ортонормальна и, следовательно, преобразует векторы с сохранением евклидовой нормы. Для двух произвольных несовпадающих векторов a и b одинаковой евклидовой длины всегда можно подобрать матрицу Хаусхолдера, которая переведет один вектор в другой, т. е. обеспечит равенство

$$Ha = \left(I - \frac{1}{\beta} ww^T\right) a = a - w\left(\frac{w^T a}{\beta}\right) = b. \qquad (2.44)$$

Легко проверить, что оно справедливо при любом w, коллинеарном разности $a - b$.

Заметим также, что воздействие преобразования H на вектор a сводится к вычитанию из a вектора w с некоторым множителем. Соответственно те компоненты исходного вектора, которым отвечают нулевые компоненты вектора Хаусхолдера w, при этом преобразовании не изменяются. Более того, преобразование Хаусхолдера вообще не изменит вектора a, если окажется, что $w^T a = 0$.

Среди всевозможных ортогональных преобразований выделяют так называемые *плоские повороты*. Последние часто используются для обнуления какой-нибудь одной компоненты вектора. Плоский поворот задается матрицей Хаусхолдера с вектором w, у которого есть только две ненулевые компоненты. Если они стоят в i-й и j-й позициях, эта матрица будет отличаться от единичной только i-й и j-й строками и столбцами. При i, меньшем j, ее элементы (i, i), (i, j), (j, i) и (j, j) образуют конфигурацию вида

$$\begin{pmatrix} c & s \\ s & -c \end{pmatrix},$$

где $c^2 + s^2 = 1$ (так что $c = \cos\theta$ и $s = \sin\theta$ для некоторого θ).

Умножение вектора слева на рассматриваемую матрицу плоского поворота изменит только i-ю и j-ю компоненты вектора.

Обычно 0 подбирают так, чтобы в j-й позиции получить нуль. Если сам вектор обозначен через x, для этого надо положить $s=-x_j/r$, $c=\pm x_i/r$, где $r=(x_i^2+x_j^2)^{1/2}$. Из двух возможных знаков обычно выбирают верхний. Компактное описание любого плоского поворота представляет собой четверку чисел: i, j, s и c.

Поскольку преобразованием Хаусхолдера можно осуществить любой поворот, причем векторы, ортогональные w, при этом останутся неизменными, с помощью n таких преобразований любую $m\times n$-матрицу A ранга n можно переделать в верхнюю треугольную. Очередное, i-е преобразование H_i подбирается так, чтобы обнулить в i-м столбце элементы с $i+1$-го по m-й, сохранив первые $i-1$ столбцов нетронутыми. (Когда $m=n$, на все потребуется $n-1$ преобразований.)

В частности, преобразование H_1 должно перевести столбец a_1 матрицы A в вектор, коллинеарный единичному. Поскольку норма при этом сохранится, последний будет иметь вид $(r_{11}, 0, \ldots, 0)^T$, где $|r_{11}|=\|a_1\|_2$. Соответственно в качестве w для H_1 можно взять $(a_{11}-r_{11}, a_{21}, \ldots, a_{m1})^T$. Чтобы избежать ошибок компенсации при подсчете первой компоненты этого вектора, знак r_{11} берут противоположным знаку a_{11}.

После n преобразований Хаусхолдера получим

$$H_n \ldots H_2 H_1 A = Q A = \begin{pmatrix} R \\ 0 \end{pmatrix}, \tag{2.45}$$

где R есть невырожденная верхняя треугольная $n\times n$-матрица, а Q — ортонормальная матрица, равная произведению $H_n \ldots H_1$. Представление (2.45) называют *QR-разложением* A.

Предположение о полном столбцовом ранге A существенно при выводе (2.45), поскольку оно гарантирует, что преобразуемая часть очередного столбца всегда будет ненулевой, т. е. определения всех n преобразований H_i будут корректны. Если же ранг A равен $r<n$, надо переставить столбцы так, чтобы первые r стали линейно независимыми. Тогда результатом r соответствующих преобразований над ними будет представление вида

$$QAP = \begin{pmatrix} T \\ 0 \end{pmatrix},$$

где P — перестановочная, а T — верхняя трапециевидная $r\times n$-матрица. После этого дополнительными r преобразованиями Хаусхолдера *над строками* можно обнулить последние $n-r$ столбцов, сохранив треугольную структуру (но не сами элементы!) левой верхней части матрицы. Окончательно получим

$$Q_r \ldots Q_1 Q A P = Q A V = \begin{pmatrix} R & 0 \\ 0 & 0 \end{pmatrix}.$$

где R — это $r\times r$-невырожденная верхняя треугольная, а V — $n\times n$-ортонормальная матрица. Это представление называют *полным ортогональным разложением* матрицы A.

Ортогональные факторы приведенного разложения несут информацию о пространствах, натянутых на строки и столбцы A. Расщепим матрицу Q^T на две составляющие следующим образом:

$$Q^T = (Q_1 Q_2),$$

где Q_1 и Q_2 суть $m\times r$- и $m\times (m-r)$-матрицы соответственно. Тогда столбцы Q_1 будут ортонормальным базисом столбцового пространства A, а столбцы Q_2 сформируют ортонормальный базис его ортогонального дополнения. Кроме того, если $r\leqslant n$, последние $n-r$ строк матрицы V образуют ортонормальный базис для множества векторов, ортогональных строкам A.

Число операций, требуемых для расчета Q, R и V, в прямоугольном случае определяется значениями m, n и r. Если $m\geqslant n\geqslant r$, то количества сложений и умножений будут приблизительно равными $mn^2 - {}^1/_3 n^3 + (n-r)r^2$.

Когда разложение, аналогичное рассмотренному, требуется для матрицы A с полным строковым рангом (с линейно независимыми строками), то, как правило, его удобнее строить в форме

$$AQ = (L\,0),$$

известной под названием «LQ-разложение».

Чаще всего ортогональные разложения используются при решении *линейных задач о наименьших квадратах*: найти x, такой, чтобы величина $\|Ax-b\|_2^2$ была минимальной. Поскольку ортонормальное преобразование сохраняет евклидову длину вектора, исходная невязка (вектор $Ax-b$) будет иметь ту же норму, что и преобразованная матрицей Q. Таким образом, справедливо равенство

$$\|Ax-b\|_2 = \|Q(Ax-b)\|_2 = \|\tilde{R}x - Qb\|_2,$$

или, что то же самое,

$$\|Ax-b\|_2 = \left\|\begin{pmatrix} R \\ 0 \end{pmatrix} x - Qb\right\|_2.$$

Из выражения в правой части видно, что норма преобразованного вектора невязок минимальна, когда первые n компонент векторов $\tilde{R}x$ и Qb совпадают. Соответственно при $n=r$ искомый вектор x будет единственным решением невырожденной линейной системы вида $Rx = Q^T b$.

2.2.5.4. Спектральное разложение симметричной матрицы. Известно, что все собственные числа симметричной матрицы A вещественны, а из ее собственных векторов можно составить ортонормальный базис в $\mathbb{R}^n$. Обозначим эти числа $(\lambda_1, \ldots, \lambda_n)$, а векторы

через $\{u_1, \ldots, u_n\}$. Тогда по определению

$$Au_i=\lambda_i u_i, \quad i=1, \ldots, n. \tag{2.46}$$

Введя матрицу U, i-й столбец которой равен u_i, эти векторные равенства можно заменить одним матричным $AU=U\Lambda$, где $\Lambda=\operatorname{diag}(\lambda_1, \ldots, \lambda_n)$. Так как U по построению ортонормальна, отсюда следует, что $A=U\Lambda U^T$, или, в более понятной форме,

$$A=\sum_{i=1}^{n} \lambda_i u_i u_i^T. \tag{2.47}$$

Это представление называется *спектральным разложением* матрицы A.

Итеративный алгоритм построения собственной системы симметричной матрицы требует около $3n^3$ сложений и умножений.

2.2.5.5. Разложение по особым числам. Любая $m\times n$-матрица может быть представлена в виде

$$A=U\Sigma V^T,$$

где U — ортонормальная $m\times m$-матрица, V — ортонормальная $n\times n$-матрица, а $\Sigma=\operatorname{diag}(\sigma_1, \ldots, \sigma_n)$ — диагональная $m\times n$-матрица с $\sigma_i\geqslant 0$ при любом i.

Величины $\{\sigma_i\}$ называют *особыми числами* матрицы A, и обычно предполагают, что они упорядочены по убыванию: $\sigma_1\geqslant\sigma_2\geqslant\ldots\geqslant 0$. Если ранг матрицы A равен r, то $\sigma_r>0$, $\sigma_{r+1}=0$, т. е. у нее будет ровно r ненулевых особых чисел. При этом особые числа обладают следующими свойствами:

(i) $\sigma_i^2(A)=\lambda_i[A^TA]$;

(ii) $\sigma_1(A)=\|A\|_2$ (максимальное особое число равно квадратичной норме матрицы).

Для квадратной невырожденной матрицы A минимальное особое число σ_n равно расстоянию (измеренному с помощью квадратичной нормы) от A до ближайшей вырожденной матрицы: $\sigma_n=\min\|A-S\|_2$, где S — вырожденная матрица. Разложение по особым числам для симметричной матрицы легко получается из ее спектрального разложения, причем $\sigma_i=|\lambda_i|$.

2.2.5.6. Псевдообратные матрицы. Для любой невырожденной квадратной матрицы A однозначно определена обратная матрица A^{-1}, такая, что при произвольной правой части b решением системы $Ax=b$ будет вектор $x=A^{-1}b$. Если же A — вырожденная или прямоугольная матрица, обратной к ней не существует; более того, в этих случаях система $Ax=b$ может оказаться несовместной. Здесь естественно пользоваться обобщением понятия обратного преобразования, которое формулируется в терминах соответствующей задачи минимизации суммы квадратов невязок.

Псевдообратной к $m \times n$-матрице A называют матрицу A^+, однозначно определенную следующим требованием: при любой правой части b формула $x=A^+b$ должна давать самый «короткий» (в смысле евклидовой нормы) вектор из множества решений задачи минимизации значения $\|b - Ax\|_2$.

Существует несколько математически эквивалентных выражений для матрицы, псевдообратной к A. Если A невырожденная, то $A^+=A^{-1}$. Когда A имеет полный столбцовый ранг, ее псевдообратную матрицу можно представить так:

$$A^+ = (A^TA)^{-1}A^T.$$

В этом же случае, располагая QR-разложением (2.45), можно воспользоваться формулой $A^+=R^{-1}Q^T$, причем именно она рекомендуется для конкретных вычислений. При неполном ранге A наиболее удобная форма представления A^+ вытекает из разложения по особым числам. Если $A=U\Sigma V$ с r ненулевыми особыми числами, то

$$A^+=V\Omega U^T,$$

где $\Omega=\mathrm{diag}(\omega_i)$ с $\omega_i=1/\sigma_i$, $i=1, \ldots, r$, и $\omega_i=0$, $i \geqslant r+1$.

2.2.5.7. Пересчет матричных разложений. При составлении многих алгоритмов приходится решать проблему организации перехода от известного разложения некоторой матрицы A к неизвестному разложению тесно связанной с ней матрицы $\hat{A}$. Естественно, что при этом хочется как-то использовать старое разложение, с тем чтобы сократить объем вычислений, необходимых для построения нового.

Кардинальный способ экономного планирования вычислений в этой ситуации состоит в том, чтобы найти метод *модификации* или *пересчета* разложения исходной матрицы. В большинстве случаев модификация потребует существенно меньших усилий, чем выполнение полного цикла расчетов, необходимых, чтобы построить разложение, начиная «с нуля». Соответствующие эффективные методы модификации имеются для всех рассмотренных ранее матричных разложений. Однако их детальный разбор выходит за рамки данной книги, и поэтому здесь будут кратко описаны только два из них. Оба относятся к наиболее часто встречающемуся в оптимизационных приложениях случаю, когда $\hat{A}$ отличается от A на матрицу ранга один (см. разд. 2.2.1.5).

Простейшая модификация ранга один состоит в присоединении к матрице столбца. Относительно нее любую процедуру факторизации, в которой столбцы обрабатываются поочередно, можно рассматривать как последовательность пересчетов. Таким образом, здесь нет нужды придумывать специальные методы модификации: они уже содержатся в основных процедурах. Рассмотрим, например, обновление QR-разложения для матрицы A, получающейся

присоединением справа к $m\times n$-матрице A нового столбца a. При этом

$$Q\bar{A}=(QA\ Qa)=\begin{pmatrix} R & v_1 \\ 0 & v_2 \end{pmatrix},$$

где v_1, v_2 — составляющие m-мерного вектора $v=Qa$, а Q, R — факторы разложения A. Обозначим через H_{n+1} матрицу Хаусхолдера, которая обнуляет компоненты вектора v с $(n+2)$-й по m-ю, оставляя первые n компонент v неизменными, так что

$$H_{n+1}v=\begin{pmatrix} v_1 \\ \gamma \\ 0 \end{pmatrix},$$

где $|\gamma|=\|v_2\|_2$. Тогда, применив H_{n+1} к $Q\bar{A}$, получим

$$H_{n+1}Q\bar{A}=\begin{pmatrix} R & v_1 \\ 0 & \gamma \\ 0 & 0 \end{pmatrix},$$

или, что то же самое,

$$\overline{QA}=\begin{pmatrix} \overline{R} \\ 0 \end{pmatrix}.$$

Это — формула QR-разложения для A. Оно получено преобразованиями, которые по сути есть не что иное, как один шаг общей процедуры QR-факторизации, и включают построение ровно одной матрицы Хаусхолдера.

Некоторые другие процедуры факторизации также удается проинтерпретировать в терминах обновления факторов при последовательной модификации матрицы. После этого методы пересчета становятся естественным продолжением исходных процедур, и их нетрудно получать в форме, допускающей не только более сложные прямые, но и обратные преобразования матриц.

Еще один метод пересчета факторов, который мы рассмотрим в этом разделе, это метод вывода разложения по Холесскому для положительно определенной матрицы $\bar{B}$, получающейся из положительно определенной матрицы B модификацией вида $\bar{B}=B\pm vv^T$.

В случае с верхним знаком имеем

$$\bar{B}=LDL^T+vv^T=L(D+pp^T)L^T,$$

где p — решение треугольной системы $Lp=v$. При этом структура матрицы $D+pp^T$ настолько проста, что для элементов ее факторов Холесского (которые обозначим через $\bar{L}$ и $\bar{D}$) легко выписать нерекуррентные формулы. Таким образом,

$$\bar{B}=L\bar{L}\bar{D}\bar{L}^TL^T=\bar{L}\bar{D}\bar{L}^T,$$

где через L обозначено произведение $L\bar{L}$, а D—$\bar{D}$. Поскольку результатом перемножения двух единичных нижних треугольных матриц является матрица того же вида, полученное равенство и есть искомое разложение. Вычисления здесь можно организовать так, что расчет компонент вектора p к умножение L на $\bar{L}$ будут вестись параллельно, причем потребуют всего лишь порядка n^2 операций. Соответствующий экономный алгоритм вычисления L и $\bar{D}$ состоит в следующем:

(i) положить $l_0=1$, $v^{(0)}=v$;

(ii) для $j=1, 2, \ldots, n$ найти

$$\left.\begin{aligned} &p_j=v_j^{(j)},\ l_j=l_{j-1}+p_j^2/d_j,\\ &\bar{d}_j=d_j l_j/l_{j-1},\ \beta_j=p_j/(d_j l_j),\\ &v_i^{(j+1)}=v_i^{(j)}-p_j l_{ij},\\ &\bar{l}_{ij}=l_{ij}+\beta_j v_i^{(j+1)}, \end{aligned}\right\} i=j+1,\ \ldots,\ n.$$

Если матрица $\bar{D}$ возникает в результате вычитания, при обновлении факторов должны быть приняты особые меры предосторожности, исключающие возможность потери положительной определенности (появления на диагонали $\bar{D}$ неположительных элементов) из-за ошибок округления. Поэтому алгоритм в данном случае будет несколько сложнее предыдущего:

(i) решить уравнение $Lp=v$ и вычислить $t_{n+1}=1-p^TD^{-1}p$; если $t_{n+1}\leqslant\varepsilon_M$, положить $t_{n+1}=\varepsilon_M$, где ε_M — относительная машинная точность;

(ii) для $j=n, n-1, \ldots, 1$ вычислить

$$\left.\begin{aligned} &t_j=t_{j+1}+p_j^2/d_j,\ \bar{d}_j=d_j t_{j+1}/t_j,\\ &\beta_j=-p_j/(d_j t_{j+1}),\ v_i^{(j)}=p_j,\\ &\bar{l}_{ij}=l_{ij}+\beta_j v_i^{(j+1)},\\ &v_i^{(j)}=v_i^{(j+1)}+p_j l_{ij}, \end{aligned}\right\} i=j+1,\ \ldots,\ n.$$

Поскольку предполагается, что теоретически положительная определенность матрицы $\bar{B}$ гарантирована, *точное* значение величины t_{n+1} должно быть положительным. Заменяя слишком малое значение t_{n+1} на ε_M, мы рискуем сделать ошибку того же порядка, что и ошибка, возникающая при вычислении разности $\bar{B}-B=vv^T$. При этом нетрудно убедиться, что все значения $\bar{d}_j$, $j=1, 2, \ldots, n$, вычисляемые по представленной выше схеме, будут положительны независимо от того, какие ошибки округления возникнут в расчетах.

Аналогичные методы применимы и для пересчета факторов разложения (2.42).

2.2.6. МНОГОМЕРНАЯ ГЕОМЕТРИЯ

В этом разделе рассматриваются геометрические характеристики векторов x, удовлетворяющих равенству

$$a^T x = \beta, \tag{2.48}$$

где $a = (a_1, a_2, \ldots, a_n)$ — некоторый ненулевой вектор, а β — число.

Введем множество векторов z, для которых $a^T z = 0$. Они формируют линейное векторное пространство размерности $n-1$ (нуль-пространство вектора a). Через $Z = \{z_1, z_2, \ldots, z_{n-1}\}$ обозначим его ортонормальный базис. Поскольку вектор a линейно независим от $\{z_1, z_2, \ldots, z_{n-1}\}$, присоединение его к Z дает базис для $\mathcal{R}^n$. Соответственно каждый вектор x представим в виде

$$x = \alpha_1 z_1 + \alpha_2 z_2 + \ldots + \alpha_{n-1} z_{n-1} + \alpha_n a, \tag{2.49}$$

а для x, удовлетворяющих (2.48), получим $a^T x = \alpha_n a^T a = \beta$, откуда

$$\alpha_n = \frac{\beta}{a^T a} = \frac{\beta}{\|a\|_2^2}.$$

Множество всевозможных векторов вида (2.49), где значение α_n фиксировано, называют *гиперплоскостью*, вектор a — ее *нормалью*, а вектор $a/\|a\|_2^2$ единичной евклидовой длины — *единичной нормалью*.

Гиперплоскость можно представлять себе как результат *переноса* $(n-1)$-мерного подпространства, ортогонального вектору a, в направлении, параллельном этому вектору. Заметим, что при $\beta = 0$ гиперплоскость проходит через начало координат, т. е. совпадает с этим подпространством. В противном случае расстояние от произвольной точки x гиперплоскости до начала координат ограничено снизу положительной величиной. По определению квадрат этого расстояния равен

$$x^T x = \alpha_1^2 + \alpha_2^2 + \ldots + \alpha_n^2 (a^T a),$$

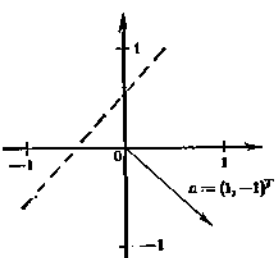

Рис. 2d. Гиперплоскость в двумерном случае.

т. е. минимальное значение последнего равно $|\beta|/\|a\|_2$ и достигается при $\alpha_1 = \ldots = \alpha_{n-1} = 0$ (когда x пропорционален a).

Для наглядной иллюстрации рассмотрим множество двумерных векторов, определенное равенством

$$a^T x = a_1 x_1 + a_2 x_2 = \beta,$$

где $a_1 = 1$, $a_2 = -1$, $\beta = -1/2$. Оно изображено на рис. 2d штриховой линией. Заметьте, что вектор $a = (1, -1)^T$ ортогонален гиперпло-

скости (в данном случае она представляет собой прямую), и кратчайшее расстояние от нее до начала координат равно

$$\frac{|\beta|}{\|a\|} = \frac{\left|\frac{1}{2}\right|}{\sqrt{2}} = \frac{\sqrt{2}}{4}.$$

На рис. 2е изображена гиперплоскость в трехмерном случае. Обратите внимание, что векторы $z_1, z_2, \ldots, z_{n-1}$ могут интерпретироваться как оси ортогональной системы координат, лежащей в гиперплоскости и имеющей началом ее точку, ближайшую к исходному началу координат. Различные значения β при фиксированном a будут давать параллельные гиперплоскости.

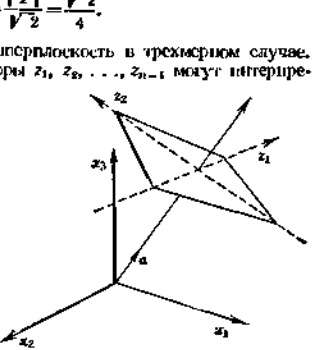

Рис. 2е. Гиперплоскость в трехмерном случае.

Преобразование переноса можно применить к любому подпространству $\mathscr{L}$, определив множество всех векторов вида $x_0 + y$, где $y \in \mathscr{L}$, а x_0 — фиксированный вектор. Такие множества называют *линейными многообразиями*. Гиперплоскость есть особый случай линейного многообразия, характерный тем, что размерность $\mathscr{L}$ равна $n-1$.

Замечания и избранная библиография к разделу 2

Есть много вводных курсов по вычислительной линейной алгебре; см., например, Стюарт (1973), Стренг (1976) и соответствующие главы в книге Лоусона и Хенсона (1974). Более специальные и сложные вопросы рассматриваются в книгах Форсайта и Молера (1967) и Лоусона и Хенсона (1974). Подробности методов обновления матричных разложений пока еще не просочились в учебники. Однако заинтересованный читатель может найти их в статьях Гилла, Голуба, Мюррея и Сондерса (1974), Дэнисла, Грэга, Кауфмана и Стюарта (1976).

Зрелому математику мы рекомендуем в качестве источника, содержащего полный и тщательный обзор основополагающих сведений по вычислительной алгебре, книгу Уилкинсона (1965).

К настоящему времени разработано большое количество высококачественных программ для решения задач линейной алгебры. Первоклассными источниками их могут послужить книги Уилкинсона и Райнша (1971), Смита и др. (1974), Донгарра, Банча, Молера и Стюарта (1979).

2.3. ЭЛЕМЕНТЫ МНОГОМЕРНОГО АНАЛИЗА

2.3.1. ФУНКЦИИ МНОГИХ ПЕРЕМЕННЫХ; ЛИНИИ УРОВНЯ

В последующих разделах рассматриваются основные свойства скалярных функций многих переменных, используемые при построении оптимизационных алгоритмов. Эти функции будем обозначать через $F(x)$, где x — вещественный n-мерный вектор $(x_1, x_2, \ldots, x_n)^T$, во всех случаях, кроме одного, когда $n=1$. В этом случае для обозначения функции будет использован символ $f(x)$, а индекс единственной компоненты аргумента x, как правило, будет опускаться. Ниже встретятся также векторные функции — упорядоченные наборы скалярных функций векторного аргумента x, компоненты которых независимо от размерности x будут обозначаться через $f_i(x)$.

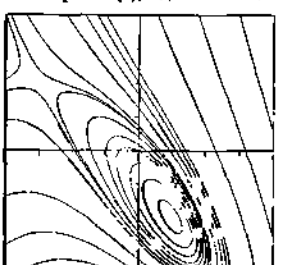

Рис. 2.1. Контурный график функции $F(x) = e^{x_1}(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1)$.

В дальнейшем везде, где это целесообразно, обсуждение функций многих переменных сопровождается иллюстративными примерами для случаев малой размерности. В частности, используются так называемые контурные графики.

Для произвольной функции $F(x)$ векторного аргумента $x \in \mathcal{R}^n$ уравнение $z = F(x)$ определяет в $(n+1)$-мерном пространстве $\mathcal{R}^{n+1}$ некоторую гиперповерхность. Когда $n=2$, это — обычная поверхность в трехмерном пространстве (z, x_1, x_2). При этом для каждого c из множества значений функции $F(x_1, x_2)$ уравнение $F(x_1, x_2) = c$ неявно задает кривую в координатах x_1, x_2 на плоскости $z=c$. Если несколько таких кривых, соответствующих некоторой выборке значений параметра c, изобразить на одной плоскости, получится контурный график функции. Образующие его кривые принято называть линиями уровня. На рис. 2.1 представлен контурный график функции $F(x) = e^{x_1}(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1)$. Показаны линии уровня, отвечающие значениям c, равным 0.2, 0.4, 0.7, 1., 1.7, 1.8, 2.8, 4, 5, 6 и 20.

2.3.2. НЕПРЕРЫВНЫЕ ФУНКЦИИ И ИХ ПРОИЗВОДНЫЕ

В основе подавляющего большинства методов решения оптимизационных задач лежит идея использования информации о функции в точке для прогноза ее поведения в целом. Мы рассмотрим сей-

час некоторые свойства функций, позволяющие делать подобные прогнозы. Центральным среди них является *непрерывность*.

Функцию одной переменной называют *непрерывной* в точке x, если для любого наперед заданного числа $\varepsilon(\varepsilon>0)$ можно найти число $\delta(\delta>0)$, такое, что при всех y, удовлетворяющих неравенству $|y-x|<\delta$, будет справедливо неравенство $|f(y)-f(x)|<\varepsilon$. Геометрически непрерывность f в точке x означает, что график f выглядит «над x» как слитная кривая. Функцию f, не удовлетворяющую данному определению, называют *разрывной* в x. Значение такой функции в x может сильно отличаться от ее значений в соседних с x точках (см., например, рис. 2 g).

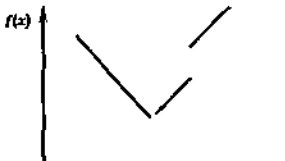

Рис. 2g. Разрывная функция.

Чтобы дать строгое определение непрерывности в многомерном случае, введем понятие *окрестности* точки n-мерного пространства. Пусть δ — некоторое положительное число. Тогда *δ-окрестностью* точки x назовем множество всех точек y, удовлетворяющих неравенству $\|y-x\|\leqslant\delta$. Какую именно норму использовать в этом определении, как правило, не существенно, хотя от этого, разумеется, зависит «форма» окрестности. Чаще всего берут евклидову норму. В трехмерном случае определение δ-окрестности с евклидовой нормой задает шар радиуса δ с центром в x. Когда ясно, о какой величине δ идет речь либо эта величина не играет роли, вместо «δ-окрестность» говорят просто «окрестность».

Приведенное выше определение непрерывности функции одной переменной непосредственно обобщается на многомерный случай переходом от интервалов значений скалярного аргумента к δ-окрестностям значений векторного аргумента. Соответственно скалярная функция $F(x)$ векторного аргумента x называется *непрерывной* в точке x, если для любого $\varepsilon>0$ найдется $\delta>0$, такое, что при всех y, удовлетворяющих неравенству $\|y-x\|<\delta$ (т. е. лежащих в δ-окрестности x), будет $|F(x)-F(y)|<\varepsilon$. Если же непрерывной называют векторную функцию, то имеют в виду, что непрерывны все ее скалярные компоненты.

Отметим, что заключение о непрерывности или разрывности функции F в точке x не зависит от того, какая норма будет использована для измерения разности $y-x$. Это может повлиять только на выбор конкретных значений δ.

На контурном графике функции двух переменных ее разрыв проявляется «слиянием» разных линий уровня (например, в точке «вертикального обрыва» поверхности $z=F(x_1, x_2)$).

Следующее важное свойство функций — *дифференцируемость*. Начнем с определения его в одномерном случае. Для этого рассмотрим график функции f на интервале, содержащем некоторую точку $\bar{x}$. Выберем на этом же интервале еще одну точку y ($y \neq \bar{x}$) и построим отрезок, соединяющий в плоскости графика точки с координатами $(\bar{x}, f(\bar{x}))$ и $(y, f(y))$. Тангенс угла наклона этого отрезка к горизонтали равен

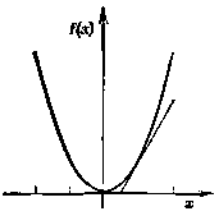

Рис. 2h. Функция $f(x) = x^2$.

$$m(y) = \frac{f(y) - f(\bar{x})}{y - \bar{x}}.$$

Если он имеет предел при стремлении y к $\bar{x}$, этот предел, равный тангенсу угла между горизонталью и касательной к графику в точке $\bar{x}$, называют *первой производной* или *градиентом* от f в $\bar{x}$. Для обозначения первой производной используют символы $f'(\bar{x})$, $f^{(1)}(\bar{x})$ или $df/dx|_{\bar{x}}$. Формально

$$f'(\bar{x}) = \lim_{y \to \bar{x}} m(y) = \lim_{h \to 0} \frac{f(\bar{x}+h) - f(\bar{x})}{h}. \tag{2.50}$$

Когда $f'(\bar{x})$ существует, функцию f называют *дифференцируемой* в точке $\bar{x}$. Рис. 2h иллюстрирует сказанное на примере функции $f(x) = x^2$, точки $\bar{x} = 1$ и интервала $[-2, 2]$. При любом h для $f(x)$ имеем

$$\frac{f(\bar{x}+h) - f(\bar{x})}{h} = \frac{\bar{x}^2 + 2\bar{x}h + h^2 - \bar{x}^2}{h} = 2\bar{x} + h. \tag{2.51}$$

Предел (2.51) при $h \to 0$ существует при всех $\bar{x}$ и равен $f'(\bar{x}) = 2\bar{x}$. Следовательно, $f'(1) = 2$.

Существование производной, определенной соотношением (2.50), означает, что величина $m(y)$ стремится к одному и тому же пределу независимо от того, с какой стороны y приближается к $\bar{x}$. Однако может случиться так, что предел для $m(y)$ будет существовать только при стремлении y к $\bar{x}$ с одной стороны либо значения пределов при приближении к $\bar{x}$ слева и справа будут различными. Возьмем, например, непрерывную функцию $f(x) = |x|$, изображенную на рис. 2i. Очевидно, что ее наклон равен -1 для $x < 0$ и равен $+1$ для $x > 0$. В точке $\bar{x} = 0$ будет $m(y) = -1$, если $y < \bar{x}$, и $m(y) = +1$, если $y > \bar{x}$. Соответственно предела (2.50) для функции $f(x) = |x|$ при $\bar{x} = 0$ нет, т. е. в начале координат она недифференцируема.

Чтобы полнее описывать рассмотренные случаи недифференцируемости, вводят понятие *односторонние производные*. Когда существует предел

$$\lim_{\substack{h\to 0\\ h>0}} \frac{f(\bar{x}+h)-f(\bar{x})}{h}, \tag{2.52}$$

его называют *производной справа* от функции f в точке $\bar{x}$ и обозначают через $f'_+(\bar{x})$. Аналогично определяется *производная слева*. Это — предел, отличающийся от (2.52) заменой условия $h>0$ на $h<0$. В предыдущем примере с $f(x)=|x|$ обе односторонние производные существуют во всех точках и совпадают везде, за исключением начала координат.

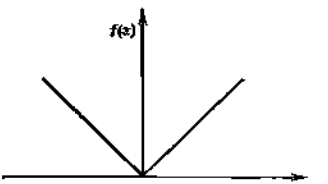

Рис. 2.1. Функция $f(x)=|x|$.

Коль скоро функция f дифференцируема в любой точке некоторого интервала, ее производную f' можно рассматривать на нем как новую функцию скалярного аргумента x. Последнюю естественно обозначить через $f'(x)$; в этом контексте запись „(x)" означает не какую-то фиксированную точку, как прежде, а независимую переменную. Если функция $f'(x)$ в свою очередь окажется дифференцируемой в точке $\bar{x}$, ее первую производную называют *второй производной* от f в $\bar{x}$ и обозначают через $f''(\bar{x})$, $f^{(2)}(\bar{x})$ или $\frac{d^2f}{dx^2}\Big|_{\bar{x}}$. Возвращаясь к примеру с $f(x)=x^2$, легко убедиться, что вторая производная f'' определена для всех x и постоянна:

$$\frac{f'(\bar{x}+h)-f'(\bar{x})}{h}=\frac{2\bar{x}+2h-2\bar{x}}{h}=2.$$

Понятно, что начатый процесс дифференцирования производных можно продолжать и дальше, пока определены соответствующие пределы. Тем самым вводится понятие n-й производной функции f в точке $\bar{x}$, которую принято обозначать через $f^{(n)}(\bar{x})$ или $\frac{d^nf}{dx^n}\Big|_{\bar{x}}$.

Читателю, не знакомому с основами дифференцирования, следует заглянуть в какой-нибудь учебник по математическому анализу, где он найдет правила дифференцирования суммы, произведения и частного. Здесь мы ограничимся тем, что приведем без доказательства общее правило дифференцирования сложной функции $g(f(x))$. Оно заключается в том, что при определенных необременительных ограничениях на f, g и x справедливо равенство

$$\frac{d}{dx}g(f(x))=g'(f(x))f'(x).$$

Идеи дифференцируемости функции одной переменной легко обобщаются на случай многомерного аргумента — достаточно рассмотреть вариации функции для его покомпонентных изменений. Например, рассмотрим в точке $\bar{x}=(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)^T$ отклик F на приращение первой переменной, в то время как остальные $n-1$ переменных остаются неизменными. Если окажется, что существует предел

$$\left.\frac{\partial F}{\partial x_1}\right|_{\bar{x}}=\lim_{h\to 0}\frac{F(\bar{x}_1+h, \bar{x}_2, \ldots, \bar{x}_n)-F(\bar{x})}{h},$$

назовем его *частной производной* от F по x_1. Это число представляет собой наклон касательной к F *вдоль направления* x_1.

Аналогичным образом определяются частные производные от F по другим координатам, причем частную производную от F по x_i принято обозначать через $\partial F/\partial x_i|_{\bar{x}}$. Когда в точке $\bar{x}$ и в некоторой ее окрестности существуют все n частных производных и в $\bar{x}$ они непрерывны, то говорят, что F *дифференцируема* в $\bar{x}$.

При определенных условиях (подобных рассмотренным ранее в одномерном случае) на каждую частную производную от F можно смотреть как на новую функцию вектора x. Из этих функций можно составить n-мерный вектор, который принято называть *градиентом* от F и обозначать через $\nabla F(x)$ или $g(x)$:

$$\nabla F(x)\equiv g(x)=\begin{pmatrix}\frac{\partial F}{\partial x_1}\\ \cdot\\ \cdot\\ \cdot\\ \frac{\partial F}{\partial x_n}\end{pmatrix}.$$

К примеру, для функции $F(x)$ вида

$$F(x)=x_1x_2^2+x_2\cos x_1$$

имеем

$$g(x)=\begin{pmatrix}\frac{\partial F}{\partial x_1}\\ \frac{\partial F}{\partial x_2}\end{pmatrix}=\begin{pmatrix}x_2^2-x_2\sin x_1\\ 2x_1x_2+\cos x_1\end{pmatrix}.$$

Функцию F, градиент которой не зависит от x, называют *линейной функцией* от x; в этом случае $F(x)\equiv c^Tx+\alpha$, где c, α — некоторые фиксированные вектор и число соответственно, причем $\nabla F(x)\equiv c$.

В одномерном случае производная определяет направление линии, касательной к кривой, заданной функцией $F(x)$. Точно так же в многомерном случае градиент от F в точке $\bar{x}$ задает ориентацию *касательной гиперплоскости* к F в $\bar{x}$. Он является

ве нормалью, а расстояние от этой гиперплоскости до начала координат определяется значением $F(\bar{x})$. В многомерном случае, как и в одномерном, есть функции, для которых условие дифференцируемости в некоторых точках не соблюдается. На рис. 2.9 изображены линии уровня одной из таких функций: $F(x) = \max\{|x_1|, |x_2|\}$. В точках, координаты которых связаны равенством $|x_1| = |x_2|$, частных производных от F не существует. Эти точки являются «угловыми» на контурном графике. Возможны также ситуации, когда частные производные по одним координатам существуют, а по другим — нет.

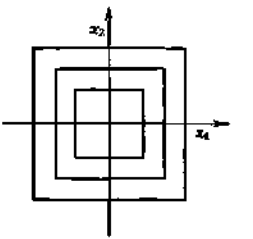

Рис. 2.9. Контурный график функции $F(x) = \max\{|x_1|, |x_2|\}$.

Продолжая аналогию с одномерным случаем, читатель без труда построит для функции многих переменных определения производных высших порядков. При этом число производных следующего порядка в n раз больше числа предыдущего. Так, «первая производная» функции n переменных представляет собой n-мерный вектор, а ее «вторая производная» имеет уже n^2 компонент, каждая из которых есть частная производная от соответствующей компоненты градиента по соответствующей переменной:

$$\frac{\partial}{\partial x_i}\left(\frac{\partial F}{\partial x_j}\right), \quad i = 1, \ldots, n; \quad j = 1, \ldots, n. \tag{2.53}$$

Величины (2.53) обычно записывают так:

$$\frac{\partial^2 F}{\partial x_i \partial x_j}, \quad i \neq j; \quad \frac{\partial^2 F}{\partial x_i^2}, \quad i = j.$$

Если частные производные $\partial F/\partial x_i$, $\partial F/\partial x_j$ и $\partial^2 F/\partial x_i \partial x_j$ существуют и непрерывны, то существуют и $\partial^2 F/\partial x_j \partial x_i$, причем $\partial^2 F/\partial x_i \partial x_j = \partial^2 F/\partial x_j \partial x_i$. В этом случае все n^2 вторых частных производных принято сводить в квадратную симметричную матрицу, именуемую *матрицей Гессе* функции $F(x)$. Впредь эту матрицу будем обозначать через $\nabla^2 F(x)$ или через $G(x)$:

$$\nabla^2 F(x) \equiv G(x) = \begin{pmatrix} \frac{\partial^2 F}{\partial x_1^2} & \cdots & \frac{\partial^2 F}{\partial x_1 \partial x_n} \\ & \vdots & \\ \frac{\partial^2 F}{\partial x_n \partial x_1} & & \frac{\partial^2 F}{\partial x_n^2} \end{pmatrix}.$$

Если матрица Гессе некоторой функции F постоянна, эту функцию называют квадратичной. Соответствующую F можно записать

так:

$$F(x) = \frac{1}{2}x^T Gx + c^T x + \alpha, \tag{2.54}$$

где G, c и α — не зависящие от x матрица, вектор и число (множитель 1/2 перед квадратичным слагаемым вводится для того, чтобы скомпенсировать двойку, возникающую при дифференцировании). Заметим, что выражения для производных от функции (2.54) таковы: $\nabla F(x) = Gx + c$, $\nabla^2 F(x) = G$.

Дифференцирование вторых производных функции n переменных дает n^3 третьих производных и т. д. Частные производные от F r-го порядка обозначают через

$$\frac{\partial^r F}{\partial x_{i_1} \partial x_{i_2} \ldots \partial x_{i_r}}, \quad i_j = 1, \ldots, n,$$

или через $\nabla^r F(x)$. Надо, однако, отметить, что производные порядков выше второго редко используются в оптимизационных процедурах.

Определения производных, согласованные с данными выше, легко построить для векторнозначной функции. Эти производные надо рассматривать как результат независимого дифференцирования ее скалярных компонент. Один шаг такого дифференцирования дает *матрицу Якоби*. Для m-мерной функции $f(x) = (f_1(x), \ldots, f_m(x))^T$ n-мерного аргумента x — это $m \times n$-матрица, (i, j)-й элемент которой есть производная от f_i по x_j; соответственно i-я строка этой матрицы представляет собой вектор, получающийся транспонированием градиента функции f_i. Заметим, что матрица Гессе скалярной функции $F(x)$ совпадает с матрицей Якоби ее градиента.

Класс функций, имеющих непрерывные производные порядков с первого по k-й, обозначают через C^k. Класс C^2 принято называть множеством дважды непрерывно дифференцируемых функций. Дифференцируемые функции иногда называют также «гладкими». Этот термин оправдан приведенной выше интерпретацией точек разрывов производных как «угловых» точек функции.

Обозначим через g функцию n переменных, и пусть f — векторная функция, такая, что $f = (f_1(0), \ldots, f_n(0))^T$, где каждая f_i есть скалярная функция одного аргумента 0. Тогда производная от $g(f(0))$ по 0 определяется следующей формулой, обобщающей приведенное ранее правило дифференцирования сложных функций:

$$\frac{\partial}{\partial 0} g(f(0)) = (f_1'(0) f_2'(0) \ldots f_n'(0)) \nabla g(f).$$

2.3.3. ПОРЯДОК ФУНКЦИИ

Пусть $f(h)$ — некоторая функция скалярного аргумента h. Говорят, что $f(h)$ имеет *порядок* h^p (и пишут при этом $f(h) = O(h^p)$), если существует конечное число M ($M > 0$), не зависящее от h,

такое, что при достаточно малых $|h|$ выполняется неравенство

$$|f(h)| \leqslant M|h|^p. \qquad (2.55)$$

Значение подобных оценок состоит в том, что на их основе сопоставляют скорость стремления к нулю разных функций. Считают, что если $p > q$, то величина $O(h^p)$ имеет „более высокий порядок малости", чем величина $O(h^q)$, и ее модуль при малых $|h|$ сходится к нулю быстрее. Надо, однако, иметь в виду, что гарантировать правильность этого вывода можно лишь в том случае, если в качестве показателей p и q берутся максимально возможные.

Когда выбирают схему аппроксимации какой-то величины, то обычно предпочтение отдают той схеме, ошибки которой имеют наибольший порядок. В этом, безусловно, есть рациональное зерно, но тем не менее некоторая осторожность здесь не помешает — ведь истинное значение $O(h^p)$ определяется помимо p константой M и величиной h. Поэтому, например, функция $f_1(h) = 10^{-4}h$ оказывается при $h > 10^{-10}$ ближе к нулю, чем $f_2(h) = 10^6 h^2$, хотя последняя имеет более высокий порядок малости.

2.3.4. ТЕОРЕМА ТЕЙЛОРА

В оптимизации находят применение многие результаты классического анализа, но чаще всего используются следствия так называемых «теорем Тейлора» или «теорем о среднем». Эти теоремы важны потому, что показывают, как по значениям функции и ее производных в точке строить приближения ее значений в окрестности.

В одномерном случае справедлива следующая

Теорема Тейлора. *Если $f \in C^r$, то для любых x, h существует число θ ($0 \leqslant \theta \leqslant 1$), такое, что*

$$f(x+h) = f(x) + hf'(x) + \frac{1}{2}h^2 f''(x) + \ldots + \frac{1}{(r-1)!}h^{r-1}f^{(r-1)}(x) + \frac{1}{r!}h^r f^{(r)}(x+\theta h), \qquad (2.56)$$

где через $f^{(k)}(x)$ обозначается k-я производная от f, вычисленная в точке x.

Используя формализм, введенный в предыдущем разделе, тейлоровское разложение (2.56) можно переписать так:

$$f(x+h) = f(x) + hf'(x) + \frac{1}{2}h^2 f''(x) + \ldots + \frac{1}{(r-1)!}h^{r-1}f^{(r-1)}(x) + O(h^r).$$

Здесь учтено, что величина $|f^{(r)}|$ ограничена на отрезке $[x, x+h]$.

От рассмотренного скалярного случая легко перейти к векторному. Пусть x — некоторая точка, а p — направление в n-мерном пространстве, и пусть $F \in C^r$. Тогда величина $F(x+hp)$ будет r раз

непрерывно дифференцируемой функцией скалярного аргумента h, и в соответствии с теоремой Тейлора ее представление типа (2.56) в окрестности точки $h=0$ выглядит так:

$$F(x+hp) = F(x) + hg(x)^T p + \frac{1}{2} h^2 p^T G(x) p + \ldots$$
$$\ldots + \frac{1}{(r-1)!} h^{r-1} D^{r-1} F(x) + \frac{1}{r!} h^r D^r F(x+\theta hp).$$

Здесь θ — некоторое число из отрезка $[0, 1]$, а

$$D^q F(x) = \sum_{i_1=1}^{n} \sum_{i_2=1}^{n} \ldots \sum_{i_q=1}^{n} \left\{ p_{i_1} p_{i_2} \ldots p_{i_q} \frac{\partial^q F(x)}{\partial x_{i_1} \partial x_{i_2} \ldots \partial x_{i_q}} \right\}.$$

В оптимизационных алгоритмах, как правило, используются только три первых члена тейлоровских разложений. Таким образом, в дальнейшем мы ограничимся представлениями вида

$$F(x+hp) \quad F(x) + hg(x)^T p + \frac{1}{2} h^2 p^T G(x) p + O(h^3).$$

Заметим, что скорость изменения функции F при движении вдоль направления p из точки x задается величиной $g(x)^T p$. Последнюю принято называть *первой производной по направлению*. Аналогично число $p^T G(x) p$ представляет собой вторую производную функции F по направлению p. Ее обычно называют *кривизной* F вдоль p. Если для некоторого p выполнено неравенство $p^T G(x) p > 0$ (< 0), то говорят, что p — направление *положительной кривизны (отрицательной кривизны)*.

Из разложения функции F около точки $\hat{x}$ по Тейлору вытекают простые способы ее аппроксимации в окрестности x. Например, пренебрегая в этом разложении всеми величинами по h, получим

$$F(\hat{x}+p) \approx F(\hat{x}) + g(\hat{x})^T p.$$

Выражение $F(\hat{x}) + g(\hat{x})^T p$ задает линейную функцию n-мерного вектора p (смещения из $\hat{x}$) и аппроксимирует значение F в точке $x+p$ с ошибкой порядка $\|p\|^2$. Учет еще одного члена тейлоровского разложения приводит к квадратичной аппроксимации:

$$F(\hat{x}+p) \approx F(\hat{x}) + g(\hat{x})^T p + \frac{1}{2} p^T G(\hat{x}) p.$$

Ее ошибка имеет порядок $\|p\|^3$.

2.3.5. КОНЕЧНО-РАЗНОСТНАЯ АППРОКСИМАЦИЯ ПРОИЗВОДНЫХ

Из тейлоровских разложений вытекают не только способы аппроксимации неизвестных значений функции в окрестности по известным значениям ее самой и ее производных в точке, но и способы решения обратной задачи, а именно задачи аппроксимации неиз-

шестных значений производных в точке по известным значениям функции в окрестности. В одномерном случае представление (2.56) дважды непрерывно дифференцируемой функции f в окрестности точки x можно переписать так:

$$\frac{f(x+h)-f(x)}{h} = f'(x)+\frac{1}{2}hf''(x+\theta_1 h), \tag{2.57}$$

где $0\leqslant\theta_1\leqslant 1$. Отсюда следует, что

$$\frac{f(x+h)-f(x)}{h} = f'(x)+O(h). \tag{2.58}$$

Соответственно, используя в качестве приближения величины $f'(x)$ левую часть этого равенства, мы допускаем ошибку порядка h. Эту ошибку принято называть *ошибкой отбрасывания*. В общем случае параметр θ_1 остается неизвестным, и поэтому точное значение последней, равное $^1/_2 hf''(x+\theta_1 h)$, определить не удается. Приходится удовлетворяться рассчитываемым или оцениваемым значением ее верхней границы.

Величину h в (2.58) принято называть *конечно-разностным интервалом*. Когда модуль $|h|$ много меньше частного $|f'(x)|/|f''(x+\theta_1 h)|$, ошибка отбрасывания будет «малой». Левая часть равенства (2.58), определяющая наклон хорды, которая соединяет на графике функции f точки $(x, f(x))$ и $(x+h, f(x+h))$, называется *правой конечно-разностной аппроксимацией* производной $f'(x)$.

Аналогично для точки $x-h$ имеем

$$f(x-h) = f(x)-hf'(x)+\frac{1}{2}h^2 f''(x-\theta_2 h), \tag{2.59}$$

где $0\leqslant\theta_2\leqslant 1$. Из этого представления следует формула *левой конечно-разностной аппроксимации*:

$$f'(x) \approx \frac{f(x)-f(x-h)}{h}. \tag{2.60}$$

Здесь ошибка равна $^1/_2 hf''(x-\theta_2 h)$.

Если в формулах (2.57), (2.59) выполнить еще по одному шагу тейлоровского разложения, результатами будут равенства вида

$$f(x+h) = f(x)+hf'(x)+\frac{1}{2}h^2 f''(x)+O(h^3), \tag{2.61}$$

$$f(x-h) = f(x)-hf'(x)+\frac{1}{2}h^2 f''(x)+O(h^3). \tag{2.62}$$

Вычтем второе из первого и поделим полученную разность на h; это приведет к равенству

$$\frac{f(x+h)-f(x-h)}{2h} = f'(x)+O(h^2), \tag{2.63}$$

в котором нет слагаемых с $f''(x)$, так как они сокращаются при вычитании. Выражение в его левой части называют *центральной конечно-разностной аппроксимацией* для $f'(x)$; соответствующая

ошибка имеет второй порядок малости и определяется значениями $f^{(3)}$ в некоторых точках отрезка $[x-h, x+h]$. Заметим, что для расчета нового приближения производной требуются значения функции в двух дополнительных точках, а не в одной, как для правой и левой аппроксимаций. Таким образом, повышение точности аппроксимации достигнуто ценой привлечения дополнительной информации, и это — общее правило.

Чтобы проиллюстрировать применение конечно-разностных формул, рассмотрим функцию $f(x)=x^3$ с производной $f'(x)=3x^2$. Правая аппроксимация дает в данном случае оценку $f'(x)$, равную

$$\frac{f(x+h)-f(x)}{h}=3x^2+3xh+h^2,$$

так что ошибкой отбрасывания будет $3xh+h^2$. Если же воспользоваться центральной аппроксимацией, то в качестве оценки величины $3x^2$ получим

$$\frac{f(x+h)-f(x-h)}{2h}=\frac{6x^2h+2h^3}{2h}=3x^2+h^2,$$

и здесь ошибка есть просто h^2.

Тейлоровское разложение функции f позволяет строить формулы аппроксимации конечными разностями не производных не только первого, но и более высоких порядков. Например, сложение (2.61) и (2.62) дает равенство

$$\frac{f(x+h)-2f(x)+f(x-h)}{h^2}=f''(x)+O(h),$$

где слагаемое $O(h)$ включает $f^{(4)}$. Его левая часть может служить приближением второй производной с ошибкой порядка h, а оценка величины $f''(x)$ с ошибкой $O(h^2)$ вытекает из равенства

$$\frac{1}{4h^2}(f(x+2h)-f(x+h)-f(x-h)+f(x-2h))=f''(x)+O(h^2).$$

Для того чтобы строить формулы конечно-разностной аппроксимации высших производных, нужны определения разностей высших порядков. Конструкцию таких формул и определений проиллюстрируем на простейшей схеме аппроксимации k-й производной. Для этого введем *оператор* Δ *вычисления правой разности*:

$$\Delta f(x)=f(x+h)-f(x).$$

Здесь ради краткости записи зависимость Δ от h не указывается. Результат двукратного последовательного применения этого оператора назовем разностью второго порядка. Если и для нее ввести свой оператор, то его естественно обозначить через Δ^2:

$$\Delta^2 f(x)=\Delta(\Delta f(x))=\Delta f(x+h)-\Delta f(x)=f(x+2h)-2f(x+h)+f(x).$$

Аналогично вводятся и разности более высоких порядков. Чтобы получить наглядную интерпретацию этих формул, обозначим через

f_j величины $f(x+jh)$ и сведем значения $\Delta^k f_j$ в *таблицу разностей* вида

$$
\begin{array}{ccccc}
f_0 & & & & \\
 & \Delta f_0 & & & \\
f_1 & & \Delta^2 f_0 & & \\
 & \Delta f_1 & & \Delta^3 f_0 & \\
f_2 & & \Delta^2 f_1 & & \Delta^4 f_0 \\
 & \Delta f_2 & & \Delta^3 f_1 & \\
f_3 & & \Delta^2 f_2 & & \\
 & \Delta f_3 & & & \\
f_4 & & & &
\end{array}
$$

Тогда правило их вычисления можно сформулировать так: для подсчета очередной разности в таблице надо взять две смежные разности из предыдущего столбца и вычесть верхнюю из нижней.

Можно показать, что справедливо равенство $\Delta^k f_0 = h^k f^{(k)}(x) + O(h^{k+1})$. Соответственно, построив таблицу разностей по значениям $f(x+jh)$, $j = 0, 1, \ldots, k$, мы можем оценить $f^{(k)}(x)$ так: $f^{(k)}(x) \approx (1/h^k)\Delta^k f$.

Формулы конечно-разностной аппроксимации производных известны и для функций многих переменных. Пусть h_j — интервал конечной разности, связанный с j-й компонентой аргумента x, и e_j — вектор, j-я компонента которого равна единице, а все остальные — нулю. Тогда

$$F(x+h_j e_j) = F(x) + h_j g(x)^T e_j + \frac{1}{2} h_j^2 e_j^T G(x+\theta_j h_j e_j) e_j,$$

где $0 \leqslant \theta_j \leqslant 1$. Слагаемые $g(x)^T e_j$ и $e_j^T G(x+\theta_j h_j e_j) e_j$ суть не что иное, как j-я компонента вектора $g(x)$ и j-й диагональный элемент матрицы $G(x+\theta_j h_j e_j)$ соответственно. Поэтому

$$g_j(x) = \frac{1}{h_j}(F(x+h_j e_j) - F(x)) + O(h_j),$$

и это — аналог равенства (2.58), порождающий формулы правой аппроксимации компонент градиента. Левая и центральная конечно-разностные аппроксимации величины $g_j(x)$ равны

$$\frac{1}{h_j}(F(x) - F(x-h_j e_j)), \quad \frac{1}{2h_j}(F(x+h_j e_j) - F(x-h_j e_j)).$$

Как и в одномерном случае, можно построить конечно-разностные приближения вторых производных. В частности, приближением первого порядка для $G_{ij}(x)$ будет

$$\frac{1}{h_i h_j}(F(x+h_j e_j + h_i e_i) - F(x+h_j e_j) - F(x+h_i e_i) + F(x)).$$

Эта формула есть результат аппроксимации j-го столбца матрицы $G(x)$ вектором $(1/h_j)(g(x+h_j e_j) - g(x))$ с последующей заменой компонент векторов $g(x+h_j e_j)$ и $g(x)$ соответствующими разностями первого порядка.

2.3.6. Скорости сходимости последовательностей

Подавляющее большинство методов оптимизации относится к классу *итеративных*. Это означает, что с их помощью генерируются бесконечные последовательности приближений $\{x_k\}$ искомого решения x^*. При этом работоспособность метода еще не гарантирована доказательством сходимости соответствующей последовательности — нужна определенная скорость сходимости. В данном разделе будут кратко рассмотрены формальные средства, которыми можно характеризовать эти скорости. Последующее изложение не претендует на полноту, но самые важные понятия теории сходимости в нем освещены (более глубокие знания можно почерпнуть из книг, указанных в библиографии). Хотелось бы сразу оговориться, что следствия общей теории сходимости на практике должны использоваться с большой осторожностью. Так, например, нельзя оценивать алгоритмы только по величинам теоретических скоростей сходимости генерируемых ими последовательностей — хотя эти скорости в определенной степени отражают эффективность методов, условия, при которых они достижимы, реализуются редко. Точно так же нельзя пренебрегать алгоритмом лишь по той причине, что теорема о скорости его сходимости не доказана: это может объясняться низким качеством метода, но не исключено, что доказательства нет просто потому, что провести его очень сложно.

Дальнейшие рассмотрения относятся к последовательности $\{x_k\}$, сходящейся к x^*. Чтобы упростить изложение, мы предположим, что все ее элементы различны и ни одна из точек x_k не совпадает с x^*.

Наиболее эффективный способ оценивания скорости сходимости состоит в сопоставлении качества приближения x_{k+1} с качеством приближения x_k, т. е. в измерении отношения расстояния между x_{k+1} и x^* к расстоянию между x_k и x^*.

Последовательность $\{x_k\}$ называется сходящейся *с порядком* r, если r — максимальное число, для которого

$$0 \leqslant \lim_{k\to\infty} \frac{\|x_{k+1}-x^*\|}{\|x_k-x^*\|^r} < \infty.$$

Поскольку величина r определяется предельными свойствами $\{x_k\}$, ее иногда называют *асимптотической скоростью сходимости*. При $r=1$ говорят о *линейной сходимости*, а при $r=2$ — о *квадратичной*.

Для последовательности $\{x_k\}$ с порядком сходимости r вводится понятие «асимптотический параметр ошибки» — это число вида

$$\gamma = \lim_{k\to\infty} \frac{\|x_{k+1}-x^*\|}{\|x_k-x^*\|^r}. \qquad (2.64)$$

Когда $r=1$, параметр γ должен быть строго меньше единицы; иначе сходимости не будет.

Чтобы проиллюстрировать данные определения, рассмотрим две характерные последовательности. Первая задается формулой

$$x_k = c^{2^k}, \tag{2.65}$$

где c — константа, удовлетворяющая условию $0 \leqslant c < 1$. Каждый следующий член этой последовательности равен квадрату предыдущего, а ее предел равен нулю. При этом

$$\frac{|x_{k+1} - 0|}{|x_k - 0|^2} = c,$$

так что скорость сходимости r в данном случае равна двум. Квадратичная сходимость, грубо говоря, означает, что с каждым шагом число правильных цифр у x_k удваивается. Первые четырнадцать членов последовательности (2.65) с параметром $c = 0.99$ выписаны во второй столбец табл. 2а. Обратите внимание, что удвоение числа правильных цифр начинается не сразу, а, точнее, при $k \geqslant 7$.

Вторая модельная последовательность определяется формулой

$$y_k = c^{-2^k}, \tag{2.66}$$

где $c \geqslant 0$. Каждый член этой последовательности является квадратным корнем предыдущего. Ее предел при любом $c > 0$ равен 1, причем

$$\lim_{k \to \infty} \frac{c^{-(k+1)} - 1}{c^{-k} - 1} = \lim_{k \to \infty} \frac{1}{c^{-(k+1)} + 1} = \frac{1}{2}.$$

Таким образом, эта последовательность сходится *линейно*. Ее начальные члены при $c = 2.2$ выписаны в третьей колонке табл. 2а.

Таблица 2а. Примеры квадратичной, линейной и сверхлинейной сходимостей

k	x_k	y_k	z_k
0	0.99	2.2	—
1	0.9801	1.4832397	1.0
2	0.96059601	1.2178833	0.25
3	0.92274469	1.1035775	0.03703704
4	0.85145777	1.0505130	0.00390625
5	0.72498033	1.0249454	0.00032
6	0.52559649	1.0123958	0.00002143
7	0.27625167	1.0061788	0.00000121
8	0.07631498	1.0030847	0.0000000596
9	0.00582398	1.0015411	0.0000000026
10	0.00003392	1.0007703	
11	0.11515×10^{-8}	1.0003851	
12	0.13236×10^{-17}	1.0001925	
13	0.17519×10^{-34}	1.0000963	
14	0.30692×10^{-72}	1.0000481	

Заметим, что примерно через каждые три итерации в приближении y_k появляется дополнительная правильная цифра.

Для линейно сходящейся последовательности пошаговое уменьшение нормы $\|x_k - x^*\|$ существенно зависит от величины асимптотического параметра ошибки. Если этот параметр оказывается нулевым, говорят, что последовательность сходится *сверхлинейно*. Вообще сходящейся сверхлинейно называют любую последовательность, для которой предел (2.64) равен нулю. Ясно, что под это определение попадают все последовательности со скоростями сходимости больше единицы. Проиллюстрировать сверхлинейную скорость сходимости при $r \to 1$ можно на примере такой последовательности:

$$z_k = \frac{1}{k^k}. \tag{2.67}$$

Предел z_k равен нулю, и

$$\lim_{k\to\infty} \frac{z_{k+1}}{z_k} = \lim_{k\to\infty} \frac{1}{k\,(1+1/k)^{k+1}} = 0.$$

Первые девять членов этой последовательности представлены в четвертой колонке табл. 2а.

Замечания и избранная библиография к разделу 2.3

Полное изложение материала, которому был посвящен данный раздел, можно найти в любом учебнике по математическому анализу, см., например, Курант (1936) и Апостол (1957). Формализм скоростей сходимости подробно рассмотрен в книге Ортега и Райнболдта (1973).

ГЛАВА 3

УСЛОВИЯ ОПТИМАЛЬНОСТИ

Изложение вопроса будет неполным, пока в той или иной форме мы не сведем всех условий
Джон Стюарт Милль (1846)

3.1. ОПРЕДЕЛЕНИЕ МИНИМУМА

Как было сказано в гл. 1, задачей оптимизации называется задача поиска минимума скалярной функции на множестве значений ее аргумента, удовлетворяющих некоторым ограничениям. В основном речь пойдет об ограничениях, которые выражаются в терминах соотношений на значение непрерывных функций от переменных задачи; другие типы ограничений будут рассмотрены в гл. 7.

Нам предстоит обсудить весьма широкий класс задач, именуемых *нелинейными задачами условной оптимизации*. Ставятся они следующим образом:

$$\text{NCP} \quad \text{найти} \quad \min_{x \in \mathbb{R}^n} F(x)$$

$$\text{при ограничениях} \quad c_i(x) = 0, \quad i = 1, 2, \ldots, m';$$

$$c_i(x) \geqslant 0, \quad i = m'+1, \ldots, m.$$

Произвольную точку $\hat{x}$, удовлетворяющую всем ограничениям NCP, будем называть *допустимой*. Множества всех допустимых точек называют *допустимой областью*. Например, в двумерном случае единственное ограничение $x_1 + x_2 = 0$ задает допустимую область, которая представляет собой прямую, изображенную на рис. 3а штриховой линией; если это ограничение заменить неравенством $x_1^2 + x_2^2 \leqslant 1$, допустимой областью станет также изображенный на рис. 3а круг единичного радиуса (выше проиллюстрирован и принятый в данной книге способ пометки внешней стороны границы множества, заданного неравенством: с этой стороны наносится штриховка). В дальнейшем используется термин „недопустимая задача". Это — задача, у которой нет допустимых точек.

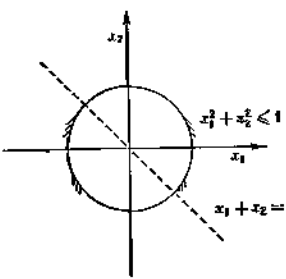

Рис. 3а. Ограничения $x_1 + x_2 = 0$ и $x_1^2 + x_2^2 \leqslant 1$.

Как уже было сказано в гл. 1, класс задач NCP разбивают на подклассы, и делается это по ряду признаков, существенных для выбора метода решения. Эти признаки упоминались в гл. 1, а различные методы (с указанием ориентации) будут представлены в дальнейшем. Однако, прежде чем говорить о методе, необходимо определить, что понимается под «решением» задачи NCP, и выяснить, какими свойствами оно обладает. Мы будем называть решением NCP точку *локального минимума* целевой функции в допустимой области, т. е. точку, удовлетворяющую ограничениям задачи и характерную тем, что между значениями целевой функции в ней и в *соседних допустимых точках* имеются специфические отношения. Формальные определения звучат следующим образом. Пусть x^* - допустимая точка NCP и через $N(x^*, \delta)$ обозначено множество допустимых точек, содержащихся в δ-окрестности x^*.

Определение А. *Точка x^* является точкой сильного локального минимума в задаче* NCP, *если существует число $\delta > 0$, такое, что*
А1. $F(x^*) < F(y)$ *для всех* $y \in N(x^*, \delta)$, $y \neq x^*$.

Заметим, что под это определение подпадает, в частности, особый случай, когда $N(x^*, \delta)$ состоит из *единственной* точки x^*.

Определение В. *Точка x^* является точкой слабого локального минимума в задаче* NCP, *если существует число $\delta > 0$, такое, что*
В1. $F(x^*) \leqslant F(y)$ *для всех* $y \in N(x^*, \delta)$;
В2. x^* *не удовлетворяет определению сильного локального минимума.*

В соответствии с данными определениями x^* *не будет* точкой локального минимума, если в любой ее окрестности найдется допустимая точка с меньшим значением F.

Определение решения задачи NCP как точки *локального минимума* на первый взгляд может показаться довольно искусственным - ведь ясно, что наибольший интерес, как правило, представляет *глобальный минимум*, т. е. точка, в которой значение целевой функции не хуже, чем в *любой* допустимой, а не только в близлежащих. Однако если мы хотим называть рассматриваемые в последующих главах процедуры методами решения задач NCP, то решение надо определять именно так, поскольку все они являются методами локальной минимизации. Конечно, с их помощью можно искать и точки глобальных минимумов, но, за исключением ряда специальных случаев, когда локальное решение автоматически оказывается глобальным, успеха здесь гарантировать нельзя. Что же касается других методов, которые надежно сходились бы к точкам глобальных минимумов в мало-мальски интересных многоэкстремальных задачах, то их просто не существует. Последнее, впрочем, не должно удручать практиков, поскольку для большинства прикладных задач удовлетворительные решения все-таки находятся.

Три выделенные разновидности минимума проиллюстрированы на рис. 3b. Как видно из этого рисунка, бывают задачи, в которых

присутствуют минимумы всех сортов; с другой стороны, попадаются и такие, в которых минимумов нет вообще. В частности, значения функции $F(x)$ в допустимых точках могут оказаться не ограниченными снизу. Это весьма вероятно, к примеру, если $F(x)=x_1 \cdot 1 x_2^2$ и допустимая область не ограничена. Минимума может не существовать и при ограниченной снизу $F(x)$, если она монотонно убывает при $\|x\|$, стремящейся в бесконечность. Примером служит $f(x)=e^{-x}$.

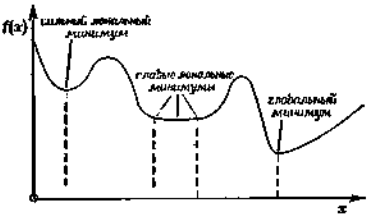

Рис. 3b. Примеры минимумов в одномерном случае.

Оставшаяся часть данной главы посвящена условиям оптимальности. Интерес к этим условиям определяется двумя обстоятельствами: во-первых, для разработки правил останова в алгоритмах оптимизации желательно иметь конструктивные средства, позволяющие определять, является точка оптимальной или нет; во-вторых, стратегию движения к оптимуму можно конструировать, исходя из стремления выполнить соотношения условий оптимальности.

Определения А и В сильного и слабого локальных минимумов сами по себе практической ценности не имеют. Использование их для проверки точки на оптимальность предполагало бы подсчет значений F для всех допустимых точек из ее δ-окрестности, и, даже учитывая, что на цифровой машине потребуется перебрать лишь конечный набор *представимых* соседних точек, соответствующий объем вычислений все равно пришлось бы признать неприемлемым. К счастью, если $F(x)$ и $\{c_i(x)\}$ обладают некоторыми свойствами гладкости, удается найти иные, более конструктивные способы описания оптимальных точек.

Впредь везде, где не оговорено противное, будет предполагаться, что целевая функция и функции ограничений дважды непрерывно дифференцируемы. Мы получим несколько разновидностей условий оптимальности, двигаясь от простых задач типа NCP к сложным. Схема вывода каждый раз будет включать два основных элемента — анализ поведения целевой функции вблизи исследуемой точки и (для задач с ограничениями) описание ее допустимой окрестности.

3.2. БЕЗУСЛОВНАЯ ОПТИМИЗАЦИЯ

3.2.1. ОДНОМЕРНЫЙ СЛУЧАЙ

Мы начнем изучение условий оптимальности с простейшей задачи о безусловной минимизации функции одной переменной:

$$\text{найти } \min_{x\in\mathbb{R}^1} f(x).$$

Будет показано, что если f всюду дважды непрерывно дифференцируема и x^* — точка ее локального минимума, то должны выполняться следующие соотношения:

Необходимые условия минимума в одномерной задаче без ограничений

A1. $f'(x^*)=0$.

A2. $f''(x^*)\geqslant 0$.

Равенство A1 доказывается от противного. Мы установим, что при ненулевой производной $f'(x^*)$ в каждой окрестности x^* нашлись бы точки с меньшим значением f, а поскольку никаких ограничений на x нет, это противоречило бы определению x^* как точки минимума. В силу гладкости f справедливо следующее разложение.

$$f(x^*+\varepsilon)=f(x^*)+\varepsilon f'(x^*)+\frac{1}{2}\varepsilon^2 f''(x^*+\theta\varepsilon), \qquad (3.1)$$

где θ $(0\leqslant\theta\leqslant 1)$ — некоторое число, вообще говоря зависящее от ε. Допустим, что производная $f'(x^*)$ отрицательна. Тогда должно существовать число $\bar{\varepsilon}$ $(\bar{\varepsilon}>0)$, такое, что $\varepsilon f'(x^*)+\varepsilon^2\frac{1}{2}f''(x^*+\theta\varepsilon)<0$ для всех ε из диапазона $0<\varepsilon\leqslant\bar{\varepsilon}$. При каждом из таких ε в силу (3.1) получим, что $f(x^*+\varepsilon)<f(x^*)$. Следовательно, в каждой окрестности x^* найдутся точки с меньшим значением функции. Это противоречит оптимальности x^*, и, значит, $f'(x^*)$ не может быть отрицательной величиной. Аналогично устанавливается, что производная $f'(x^*)$ не может быть положительной. Стало быть, если x^* — точка локального минимума, то производная $f'(x^*)$ равна нулю.

Любую точку $\bar{x}$, в которой выполнено равенство $f'(\bar{x})=0$, называют *стационарной* точкой функции f. Только что было показано, что стационарность — это одно из свойств точек минимумов гладких функций; ясно, однако, что стационарными будут и точки их локальных *максимумов*. Кроме того, производная может обращаться в нуль в точках, где нет ни минимума, ни максимума; их называют точками *перегиба*. Например, начало координат будет стационарной, но не экстремальной точкой функции $f(x)=x^3$ (рис. 3с).

Требование равенства нулю производной $f(x^*)$ называют необходимым *условием оптимальности первого порядка*. В названии

отражено, что это условие касается только первой производной. Из аналогичных соображений неравенство А2 принято называть *условием второго порядка*. Его мы также докажем от противного. Пусть x^* — точка локального минимума. Тогда в силу А1 $f'(x^*)=0$, и разложение (3.1) принимает вид

$$f(x^*+\varepsilon) = f(x^*)+\frac{1}{2}\varepsilon^2 f''(x^*+\theta\varepsilon). \quad (3.2)$$

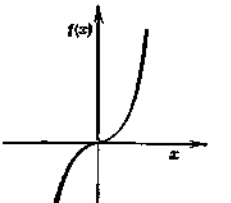

Рис. 3с. Функция $f(x)=x^3$.

Если величина $f''(x^*)$ отрицательна, то функция $f''(x)$, являясь по предположению непрерывной, будет отрицательной в любой точке из некоторой окрестности x^*. Соответственно при любом ненулевом ε, достаточно малом, чтобы точка $x^*+\varepsilon$ попала в эту окрестность, из (3.2) получим $f(x^*+\varepsilon)<f(x^*)$. Но это противоречило бы определению x^* как точки локального минимума. Значит, неравенство $f''(x^*)<0$ невозможно.

Необходимые условия часто привлекают для того, чтобы установить *неоптимальность* точки. Если же требуется доказать ее *оптимальность*, обращаются к *достаточным* условиям, соблюдение которых является гарантией оптимума. В частности, существование в точке x^* сильного локального минимума обеспечено, если справедливы два следующих соотношения:

Достаточные условия минимума в одномерной задаче без ограничений

B1. $f'(x^*)=0$.

B2. $f''(x^*)>0$.

Равенство B1 уже знакомо нам как необходимое условие оптимальности. Когда оно выполнено, можно пользоваться разложением (3.2). Если же выполняется и условие B2, то непрерывность $f''(x)$ позволяет утверждать, что фигурирующая в этом разложении вторая производная $f''(x^*+\theta\varepsilon)$ при достаточно малых по модулю ε будет положительной. Тогда для ненулевых малых ε из (3.2) получим неравенство $f(x^*+\varepsilon)>f(x^*)$, т. е. $f(x^*)$ меньше значения f в любой не совпадающей с x^* точке из некоторой окрестности x^*. Это и означает, что x^* — точка сильного локального минимума.

При разрывных $f(x)$ или $f'(x)$ дополнить исходное определение сильного локального минимума практически ценными условиями оптимальности непросто, но кое-что сказать все же можно. Так, например, если x^* не является точкой разрыва f и в некоторой окрестности x^* функция f дважды непрерывно дифференцируема, то можно утверждать, что для x^* сохраняют силу все доказанные выше условия оптимальности. Если же $f(x)$ непрерывна, но x^* —

точка разрыва $f'(x)$, то нетрудно доказать такое утверждение: для того чтобы в x^* существовал сильный локальный минимум, достаточно наличия неравенств $f'_+(x^*)>0$, $f'_-(x^*)<0$.

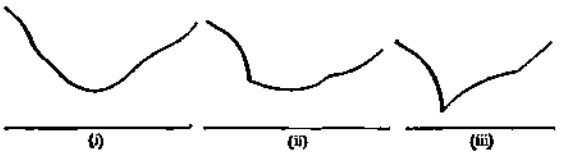

Рис. 3d. Три типа минимума в одномерном случае.

На рис. 3d показаны три ситуации с точками минимума: (i) функция f всюду дважды непрерывно дифференцируема; (ii) f' разрывна, но не в точке x^*; (iii) x^* — точка разрыва производной f'.

3.2.2. МНОГОМЕРНЫЙ СЛУЧАЙ

В данном разделе рассматривается задача минимизации функции n переменных:

UCP $\qquad$ найти $\min\limits_{x\in R^n} F(x)$.

Как и в одномерном случае, начнем с необходимых условий оптимальности. Эти условия для точки x^*, где реализуется локальный минимум функции $F(x)$, состоят в следующем:

Необходимые условия минимума для задачи UCP

C1. $\|g(x^*)\|=0$, *т. е.* x^* *является стационарной точкой.*

C2. *Матрица* $G(x^*)$ *положительно полуопределена.*

Для доказательства снова воспользуемся тейлоровским разложением целевой функции в окрестности точки x^*:

$$F(x^*+\varepsilon p)=F(x^*)+\varepsilon p^T g(x^*)+\frac{1}{2}\varepsilon^2 p^T G(x^*+\varepsilon\theta p)p. \qquad (3.3)$$

Здесь ε и θ — числа, причем $0\leqslant\theta\leqslant 1$, а p — вектор размерности n. Не умаляя общности, можно предполагать, что ε в (3.3) есть величина положительная.

Необходимость равенства C1 докажем от противного. Допустим, что x^* — точка локального минимума, но градиент $g(x^*)$ не равен нулю. Тогда найдется вектор p, такой, что

$$p^T g(x^*)<0. \qquad (3.4)$$

Например, этому неравенству удовлетворит $p=-g(x^*)$. Существуют и другие подходящие p, причем всех их принято называть

направлениями спуска в точке x^*. Зафиксировав какое-нибудь из указанных p, мы всегда сможем подобрать положительное число ε так, что для любого положительного ε, удовлетворяющего неравенству $\varepsilon \leqslant \bar{\varepsilon}$, в правой части (3.3) будет $\varepsilon p^T g(x^*) + \frac{1}{2}\varepsilon^2 p^T G(x^* + \varepsilon\theta p)\, p < 0$. Соответственно для таких ε и p получим $F(x^* + \varepsilon p) < F(x^*)$.

Итак, если градиент $g(x^*)$ не равен нулю, в любой окрестности x^* найдутся точки со значениями функции F, меньшими, чем $F(x^*)$. Последнее противоречило бы определению x^*, и, значит, точка локального минимума должна быть стационарной.

Как и в одномерном случае, градиент может обращаться в нуль в точках, где нет локального минимума, и в том числе в точках, где нет ни минимума, ни максимума. Последние принято называть *седловыми*. Характер поведения функции в окрестности седловой точки показан на рис. 3е.

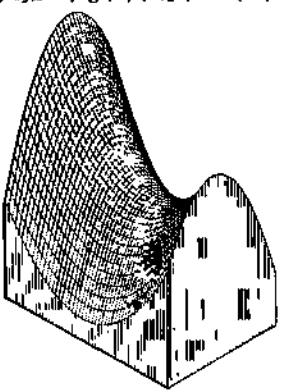

Рис. 3е. Седловая точка в двумерном случае.

Доказательство необходимости условия второго порядка С2 также построим на противоречии. Прежде всего заметим, что в силу С1 справедливо разложение

$$F(x^* + \varepsilon p) = F(x^*) + \frac{1}{2}\varepsilon^2 p^T G(x^* + \varepsilon\theta p)\, p. \qquad (3.5)$$

При этом из непрерывности матричной функции $G(x)$ следует, что для любого p величина $p^T G(x^* + \varepsilon\theta p)\, p$ будет мало отличаться от $p^T G(x^*)\, p$, если только взять $|\varepsilon|$ достаточно малым. Значит, при наличии у $G(x^*)$ хотя бы одного отрицательного собственного значения, полагая p равным соответствующему собственному вектору, для всех малых по модулю ненулевых ε получим неравенство $p^T G(x^* + \varepsilon\theta p)\, p < 0$. Но это с учетом (3.5) означало бы, что в каждой окрестности x^* найдутся точки с меньшими значениями F. Последнее невозможно в силу оптимальности x^*, что и доказывает справедливость В2.

Докажем теперь условия, *достаточные* для того, чтобы точка x^* была точкой сильного локального минимума в задаче UCP. Они состоят в следующем:

Достаточные условия минимума для задачи UCP

D1. $\|g(x^*)\|=0$.

D2. *Матрица* $G(x^*)$ *положительно определена.*

Из того, что в некоторой точке x^* выполнено условие D1, следует возможность разложения (3.5). Если к тому же матрица $G(x^*)$ положительно определена, можно (опираясь на непрерывность функции $G(x)$) утверждать, что матрица в правой части (3.5) тоже будет положительно определенной, если только точка $x^*+\varepsilon p$ попадет в некую окрестность x^*. Но она наверняка попадет туда при любом p и достаточно малом по модулю ε. Поэтому для любого p при малых ненулевых ε справедливо неравенство $p^TG(x^*+\varepsilon p)p>0$. С учетом (3.5) последнее означает, что $F(x^*)$ меньше, чем значение F в любой несовпадающей с x^* точке из некоторой окрестности x^*, т. е. x^* — точка сильного локального минимума.

3.2.3. СВОЙСТВА КВАДРАТИЧНЫХ ФУНКЦИЙ

Тейлоровское разложение (3.3) для гладкой функции F указывает на то, что в малой окрестности любой точки эту функцию можно хорошо приближать некоторой квадратичной формой. Значит, алгоритмы минимизации гладких функций можно строить, ориентируясь на их квадратичные приближения. По этой причине свойства квадратичных функций представляют для нас особый интерес, и имеет смысл рассмотреть их подробнее.

Пусть $\Phi(x)$ — квадратичная функция вида

$$\Phi(x)=c^Tx+\frac{1}{2}x^TGx, \tag{3.6}$$

где c и G — фиксированные вектор и симметричная матрица. Тогда значения $\Phi(\bar{x})$ и $\Phi(\bar{x}+\alpha p)$ при произвольных векторах $\bar{x}$, p и любом числе α связаны между собой так:

$$\Phi(\bar{x}+\alpha p)=\Phi(\bar{x})+\alpha p^T(G\bar{x}+c)+\frac{1}{2}\alpha^2p^TGp. \tag{3.7}$$

В соответствии с общим определением точку x^* назовем стационарной точкой функции Φ, если $\nabla\Phi(x^*)=Gx^*+c=0$. Таким образом, любая стационарная точка x^* является решением системы линейных уравнений вида

$$Gx^*=-c. \tag{3.8}$$

Несовместность системы (3.8), т. е. невозможность представить вектор c линейной комбинацией столбцов G, означает, что функция Φ не имеет стационарных точек. В данном случае она не ограничена ни сверху, ни снизу. Если же система (3.8) совместна, то по крайней мере одна стационарная точка найдется, причем она будет единственной, если матрица G невырождена.

Если x^*— стационарная точка Φ, из (3.7) и (3.8) следует, что

$$\Phi(x^*+\alpha p) = \Phi(x^*)+\frac{1}{2}\alpha^2 p^T G p, \qquad (3.9)$$

т. е. поведение Φ в окрестности x^* зависит только от матрицы G. Обозначим через λ_j и u_j ее j-е собственное значение и отвечающий ему собственный вектор. Тогда $Gu_j=\lambda_j u_j$ (причем в силу симметричности G векторы $\{u_j\}$, $j=1, \ldots, n$, всегда можно подобрать так, чтобы они были ортонормальными; см. разд. 2.2.3.4), и, положив в (3.9) вектор p равным u_j, получим

$$\Phi(x^*+\alpha u_j) = \Phi(x^*)+\frac{1}{2}\alpha^2\lambda_j.$$

Таким образом, характер изменения Φ при движении из x^* вдоль направления u_j полностью определяется знаком λ_j. Если λ_j больше нуля, с ростом $|\alpha|$ значение Φ будет увеличиваться. Если λ_j меньше нуля, то с увеличением $|\alpha|$ значение Φ будет убывать. Наконец, если λ_j равно нулю, при движении из x^* вдоль u_j функция Φ будет постоянной; более того, в этом случае из (3.7) видно, что Φ ведет себя как *линейная функция* на любой прямой, параллельной вектору u_j.

Когда матрица G является положительно определенной, у функции Φ будет одна-единственная стационарная точка x^*, и в этой точке реализуется ее глобальный минимум. Линии уровня такой квадратичной функции представляют собой концентрические эллипсы с главными осями, параллельными собственным векторам G, причем длины осей каждого эллипса обратно пропорциональны квадратным корням соответствующих собственных значений. Когда матрица G положительно *полуопределена*, то у Φ либо вообще нет стационарных точек, либо они образуют целое линейное многообразие. В последнем случае значения Φ во всех стационарных точках будут одинаковыми и все эти точки будут точками глобального минимума. При знаконеопределенной G стационарные точки Φ (если таковые найдутся) являются седловыми, и, как уже было сказано ранее, Φ при этом не ограничена ни сверху, ни снизу.

На рис. 3.1 изображены линии уровня трех квадратичных функций с тремя разными комбинациями знаков собственных чисел G:

(i) два положительных собственных числа

$$G=\begin{pmatrix}5 & 3\\ 3 & 2\end{pmatrix},\quad c=\begin{pmatrix}-5.5\\ -3.5\end{pmatrix};$$

(ii) одно положительное собственное число и одно нулевое

$$G=\begin{pmatrix}4 & 2\\ 2 & 1\end{pmatrix},\quad c=\begin{pmatrix}-4\\ -2\end{pmatrix};$$

(iii) одно положительное собственное число и одно отрицательное

$$G=\begin{pmatrix}3 & -1\\ -1 & -8\end{pmatrix},\quad c=\begin{pmatrix}0.5\\ 8.5\end{pmatrix}.$$

Локальное сходство любой гладкой функции F с квадратичной позволяет использовать представленные результаты для предсказания поведения F в окрестности ее стационарной точки x^* по собственным числам матрицы $G(x^*)$. Например, если одно из них

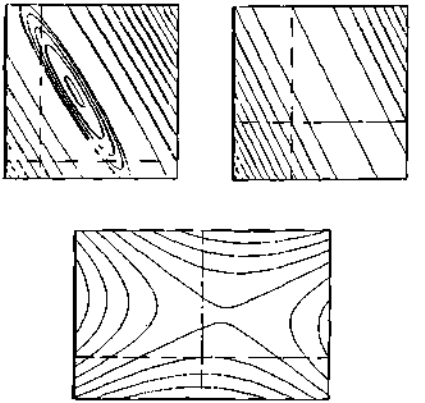

Рис. 38. Линии уровня: (i) положительно определенной квадратичной функции; (ii) положительно полуопределенной квадратичной функции; (iii) знаконеопределенной квадратичной функции.

окажется близким к нулю, то можно ожидать, что при движении из x^* вдоль соответствующего собственного вектора функция F будет почти постоянной.

3.3. Оптимизация при линейных ограничениях

В большинстве прикладных оптимизационных постановок приемлемыми являются не все мыслимые значения переменных. Соответственно приходится вводить какие-то ограничения. В том числе нередко используются ограничения в виде требований, чтобы некоторые *линейные функции* переменных были равны нулю, неотрицательны или неположительны. При этом линейной мы называем функцию вида $l(x) = a^T x - \beta$, где a^T — некоторая вектор-строка, а β — число. Градиент от $l(x)$ постоянен и равен a.

Хотя выше упоминается три типа линейных ограничений, в действительности достаточно рассмотреть только два из них:

(i) $a^T x - \beta = 0$ (ограничение-равенство);

(ii) $a^T x - \beta \geq 0$ (ограничение-неравенство).

Понятно, что ограничение $a^Tx-\beta\leqslant 0$ третьего типа всегда можно заменить эквивалентным неравенством $-a^Tx+\beta\geqslant 0$. В соответствии с общепринятой формой записи линейные ограничения (i) и (ii) в дальнейшем будут фигурировать в виде соотношений $a^Tx=\beta$ и $a^Tx\geqslant\beta$.

Простейшими среди линейных являются ограничения, задаваемые функциями $l(x)$, зависящими только от одной переменной. В этом случае, если соответствующая переменная есть x_i, возможны такие варианты:

(iii) $x_i=\beta$ (x_i фиксируется на β);
(iv) $x_i\geqslant\beta$ (β есть верхняя граница для x_i);
(v) $x_i\leqslant\beta$ (β есть нижняя граница для x_i).

Условия (iv) и (v) принято называть *простыми ограничениями* на переменную x_i.

3.3.1. Задачи с ограничениями типа линейных равенств

В этом разделе мы рассмотрим условия оптимальности для задачи, все ограничения которой описываются линейными равенствами:

LEP $$\text{найти } \min_{x\in\mathbb{R}^n} F(x)$$
$$\text{при ограничениях } \hat{A}x=\hat{b}.$$

Здесь $\hat{A}$ есть $m\times n$-матрица, а $\hat{b}$ является m-мерным вектором. Коэффициентами i-го ограничения являются компоненты i-й строки матрицы $\hat{A}$. Впредь эту строку будем обозначать через $\hat{a}_i^T$:

$$\hat{a}_i^Tx=\hat{a}_{i1}x_1+\ldots+\hat{a}_{in}x_n=\hat{b}_i.$$

В соответствии с определениями разд. 3.1 допустимая точка x^* является точкой локального минимума задачи LEP, если $F(x^*)\leqslant F(x)$ на множестве всех допустимых x из некоторой окрестности x^*. Чтобы вывести отсюда условия оптимальности x^*, необходимо прежде всего найти удобные средства описания подобных множеств.

Когда ограничения задачи LEP несовместны, то допустимых точек не существует, и тут говорить не о чем. Поэтому будем предполагать, что вектор $\hat{b}$ принадлежит области значений преобразования $\hat{A}$, т. е. система уравнений в LEP разрешима. В данном случае есть откуда выбирать x^*, причем если t строк матрицы $\hat{A}$ линейно независимы, то t степеней свободы при этом выборе будут заблокированы. Так, в двумерной задаче с одним ограничением $x_1+x_2=0$ решение выбирается среди точек прямой, изображенной на рис. 3.8 штриховой линией.

Следует подчеркнуть, что важен *ранг* системы ограничений, а не их число. Например, если к ограничению $x_1+x_2=0$ добавить ограничение $2x_1+2x_2=0$ (линейно зависимое от первого), то ранг сох-

рается, и допустимое множество при этом никак не изменится. Все будет выглядеть совсем иначе, если добавить равенство $x_1 - x_2 = 1$ (выделяемое им множество точек изображено на рис. 3g точечной линией). В данном случае ранг системы ограничений увеличится на единицу, и допустимое множество выродится в точку.

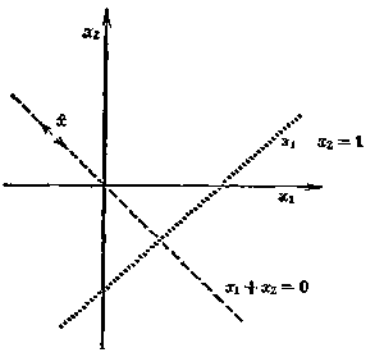

Рис. 3g. Ограничения $x_1 + x_2 = 0$ и $x_1 - x_2 = 1$.

Чтобы получить для задачи LEP простое описание всех возможных перемещений из допустимой точки, надо воспользоваться самыми элементарными свойствами линейных подпространств. Рассмотрим вектор $\bar{x} - \hat{x}$, соединяющий две допустимые точки $\bar{x}$ и $\hat{x}$. Так как $A\bar{x} = b$ и $A\hat{x} = b$, этот вектор будет удовлетворять равенству $A(\bar{x} - \hat{x}) = 0$. Поскольку от $\bar{x}$ и $\hat{x}$ ничего, кроме допустимости, не требовалось, тем самым установлено, что вектор перемещения p из одной произвольной допустимой точки в другую всегда ортогонален всем строкам матрицы A, т. е.

$$Ap = 0. \tag{3.10}$$

Такие векторы p называются *возможными направлениями* относительно ограничений-равенств задачи LEP и образуют линейное подпространство (см. разд. 2.2.2.4). Любой шаг α из любой допустимой точки $\hat{x}$ вдоль любого из возможных направлений p не нарушит ни одного из ограничений, так как $A(\hat{x} + \alpha p) = A\hat{x} = b$. Равенство (3.10) полностью характеризует возможные направления — даже бесконечно малый шаг вдоль вектора p, для

которого $\hat{A}p \neq 0$, выведет за пределы допустимого множества. На рис. 3g стрелками указаны возможные направления для ограничения $x_1 + x_2 = 0$.

Выберем в подпространстве векторов, удовлетворяющих равенству (3.10), какой-нибудь базис (см. разд. 2.2.2.4), и пусть Z—матрица, столбцами которой являются векторы этого базиса. Тогда $\hat{A}Z = 0$, и все возможные направления будут линейными комбинациями столбцов Z. Таким образом, простое описание множества допустимых перемещений получено: это шаги вдоль всевозможных направлений вида $p = Zp_Z$, где p_Z—некоторый вектор соответствующей размерности.

Для вывода условий оптимальности допустимой точки x^* воспользуемся тейлоровским разложением функции F в окрестности x^* вдоль возможного направления p ($p = Zp_Z$):

$$F(x^* + \varepsilon Zp_Z) = F(x^*) + \varepsilon p_Z^T Z^T g(x^*) + \frac{1}{2}\varepsilon^2 p_Z^T Z^T G(x^* + \varepsilon\theta p) Zp_Z. \quad (3.11)$$

Здесь θ удовлетворяет неравенствам $0 \leqslant \theta \leqslant 1$, а ε—число произвольного знака. Рассуждая так же, как и в разд. 3.2.2, на основании (3.11) легко показать, что, когда величина $p_Z^T Z^T g(x^*)$ не равна нулю, каждая окрестность x^* будет содержать допустимые точки с меньшими, чем $F(x^*)$, значениями функции F. Значит, *необходимым условием* существования в x^* локального минимума LEP является равенство нулю произведения $p_Z^T Z^T g(x^*)$ *при любом* p_Z, что возможно лишь в том случае, если

$$Z^T g(x^*) = 0. \quad (3.12)$$

Вектор $Z^T g(x^*)$ принято называть *спроектированным градиентом от* F в точке x^*. Любую точку, в которой спроектированный градиент обращается в нуль, называют *условной стационарной*.

Опираясь на равенство (3.12), можно показать, что градиент $g(x^*)$ должен быть линейной комбинацией строк матрицы $\hat{A}$, т. е.

$$g(x^*) = \sum_{i=1}^{m} \hat{a}_i \lambda_i^* = \hat{A}^T \lambda^* \quad (3.13)$$

для некоторого вектора λ^*, именуемого вектором *множителей Лагранжа*. *Множители Лагранжа определяются из* (3.13) *однозначно только тогда, когда строки* $\hat{A}$ линейно независимы.

Вывод (3.13) из (3.12) несложен. В самом деле, известно, что любой вектор представим в виде линейной комбинации строк $\hat{A}$ и столбцов Z (см. разд. 2.2.2.4). В частности, при некоторых λ^* и g_Z имеем $g(x^*) = \hat{A}^T\lambda^* + Zg_Z$. Умножив это равенство на Z^T, при соблюдении (3.12) получим $Z^T Z g_Z = 0$, а поскольку матрица $Z^T Z$ по определению невырождена, отсюда следует, что $g_Z = 0$, т. е. $g(x^*) = \hat{A}^T\lambda^*$.

Займемся теперь необходимыми условиями оптимальности второго порядка. Так как в точке минимума x^* должно выполняться равенство $Z^T g(x^*) = 0$, тейлоровское разложение (3.11) принимает вид

$$F(x^* + \varepsilon Z p_Z) - F(x^*) + \frac{1}{2}\varepsilon^2 p_Z^T Z^T G(x^* + \varepsilon\theta p) Z p_Z. \qquad (3.14)$$

Соответственно, рассуждая тем же путем, что и в случае с безусловным минимумом, нетрудно показать, что наличие у матрицы $Z^T G(x^*) Z$ отрицательных собственных значений приводило бы к существованию в любой окрестности x^* допустимых точек с меньшими значениями F. Следовательно, в оптимальной точке x^* задачи LEP матрица $Z^T G(x^*) Z$ должна быть положительно полуопределенной. Это и есть искомое условие. Фигурирующую в нем матрицу принято называть *спроектированной матрицей Гессе. Важно отметить, что положительной полуопределенности самой матрицы Гессе $G(x^*)$ здесь не требуется.*

Чтобы проиллюстрировать применение условий второго порядка, рассмотрим двумерную задачу минимизации функции $-x_1^2 + x_2^2$ при ограничении $x_2 = 1$. В этом случае $g(x) = (-2x_1, 2x_2)^T$, $\hat{A} = (0, 1)$ и

$$G(x) = \begin{pmatrix} -2 & 0 \\ 0 & 2 \end{pmatrix}.$$

Легко проверить, что равенство (3.13) выполнено с $\lambda = 1/2$ в точке $\hat{x} = (0, 1)^T$, которая тем самым претендует на оптимальность. Однако для любого вектора $p = (\delta, 0)^T$, где δ — произвольное ненулевое число, имеем $\hat{A}p = 0$ и $p^T G(\hat{x}) p = -2\delta^2 < 0$. Таким образом, в точке $\hat{x}$ локального минимума нет.

Итак, получены следующие необходимые условия существования в точке x^* локального минимума задачи LEP.

Необходимые условия оптимальности для задачи LEP

E1. $\hat{A}x^* = \hat{b}$.

E2. $Z^T g(x^*) = 0$, *или, что то же самое,* $g(x^*) = \hat{A}^T\lambda^*$.

E3. $Z^T G(x^*) Z$ *положительно полуопределена.*

Достаточные условия оптимальности для задачи LEP выводятся почти тем же способом и выглядят почти так же, как полученные ранее условия для задачи без ограничений. Только теперь в них будут фигурировать не обычные, а спроектированные градиент и матрица Гессе.

Достаточные условия минимума для задачи LEP

F1. $\hat{A}x^* = \hat{b}$.

F2. $Z^T g(x^*) = 0$, *или, что то же самое,* $g(x^*) = \hat{A}\lambda^*$.

F3. $Z^T G(x^*) Z$ *положительно определена.*

3.3.2. ЗАДАЧИ С ОГРАНИЧЕНИЯМИ ТИПА ЛИНЕЙНЫХ НЕРАВЕНСТВ

3.3.2.1. Общие условия оптимальности. Рассмотрим задачу, все ограничения которой заданы линейными неравенствами:

LIP найти $\min_{x \in \mathbb{R}^n} F(x)$

при ограничениях $Ax \geqslant b$.

Как и в случае с ограничениями-равенствами, вывод условий оптимальности начнем с поиска удобного способа описания всех допустимых точек, лежащих в окрестности исследуемой. При этом будем проводить четкое различие между теми из ограничений, которые выполняются как равенства, и теми, которые выполнены как строгие неравенства. Ограничение $a_i^T x \geqslant b_i$ называют *активным* (или *сдерживающим*) в допустимой точке $\hat{x}$, если $a_i^T \hat{x} = b_i$, и *неактивным*, если $a_i^T \hat{x} > b_i$. В том и другом случае говорят, что ограничение *соблюдено*. Если же $a_i^T \hat{x} < b_i$, ограничение называют *нарушенным* в $\hat{x}$.

Активные ограничения представляют особый интерес потому, что только они определяют множество возможных перемещений около допустимой точки. Если i-е ограничение неактивно в точке $\hat{x}$, нулевой шаг из $\hat{x}$, не нарушающий этого ограничения, можно сделать в *любом* направлении, т. е. при любом p и достаточно малом $|\alpha|$ точка $\hat{x}+\alpha p$ будет удовлетворять неактивному ограничению. Таким образом, неактивное ограничение может препятствовать только «большим» перемещениям.

Иначе дело обстоит с активным ограничением: оно сокращает возможности перемещений в *любой* окрестности допустимой точки. Допустим, что i-е ограничение оказалось активным в $\hat{x}$, т. е. $a_i^T \hat{x} = b_i$. Тогда по отношению к нему есть две категории возможных направлений. Во-первых, возможными являются все направления p, удовлетворяющие равенству $a_i^T p = 0$. Их называют *удерживающими* относительно i-го ограничения, так как последнее сохраняет активность в любой точке вида $\hat{x}+\alpha p$, где α — произвольное число. Говорят, что точка, движущаяся вдоль удерживающего направления, остается «на» ограничении. Во-вторых, возможны движения вдоль p, такие, что $a_i^T p > 0$. Эти направления называют *неудерживающими* по отношению к i-му ограничению. Поскольку $a_i^T(\hat{x}+\alpha p) - b_i = a_i^T \hat{x} - b_i + \alpha a_i^T p \geqslant b_i$ при $\alpha > 0$, i-е ограничение становится *неактивным* в точке $\hat{x}+\alpha p$. Соответственно говорят, что шаг вдоль неудерживающего направления уводит «с» ограничения.

На рис. 3.9 изображены некоторые из возможных относительно ограничения $x_1 + x_2 \geqslant 1$ направлений в допустимой точке $\hat{x} = (1/2, 1/2)^T$.

Чтобы выяснить, является ли точка x^* оптимальной в задаче LIP, необходимо прежде всего идентифицировать активные в ней ограничения. Определяющие их l строк матрицы A структурируем в новую матрицу $\hat{A}$, и пусть $\hat{b}$—вектор, полученный выборкой соответствующих компонент b. Тогда $\hat{A}x^* = \hat{b}$, причем строки $\hat{A}$ расположим с учетом их первоначального упорядочения в матрице A так, что элементами $\hat{a}_1^T$ будут коэффициенты «первого» активного ограничения. Для упрощения выкладок предположим, что эти строки линейно независимы; тем не менее условия, которые мы получим, сохранят силу и в отсутствие у $\hat{A}$ полного строкового ранга. Через Z обозначим матрицу, чьи столбцы формируют базис в подпространстве векторов, ортогональных строкам $\hat{A}$. Тогда векторы p, удовлетворяющие равенству $\hat{A}p = 0$, можно будет представлять в виде линейных комбинаций столбцов Z.

Рис. 3h. Возможные направления для линейного ограничения $x_1 + x_2 \geqslant 1$.

Рассмотрим тейлоровское разложение функции F около точки x^* вдоль удерживающего направления p ($p = Zp_Z$):

$$F(x^* + \varepsilon Z p_Z) = F(x^*) + \varepsilon p_Z^T Z^T g(x^*) + \frac{1}{2}\varepsilon^2 p_Z^T Z^T G(x^* + \varepsilon\theta p) Z p_Z. \quad (3.15)$$

Здесь θ удовлетворяет неравенствам $0 \leqslant \theta \leqslant 1$, а ε—любое число. Как и в случае с ограничениями-равенствами, это разложение позволяет установить, что если произведение $p_Z^T Z^T g(x^*)$ окажется ненулевым при *каком-нибудь* p_Z, то в точке x^* локального минимума не будет. Значит, необходимым условием оптимальности x^* является равенство $Z^T g(x^*) = 0$ или, что эквивалентно,

$$g(x^*) = \hat{A}^T\lambda^*. \quad (3.16)$$

Иногда множители Лагранжа приписывают всем ограничениям задачи, полагая их по определению равными нулю для неактивных ограничений. Мы же условимся вводить множители Лагранжа только для активных ограничений.

Условие (3.16) обеспечивает стационарность функции F в точке x^* вдоль всех удерживающих направлений. Однако возможны также движения вдоль неудерживающих направлений, и x^* заведомо не сможет быть оптимальной точкой, если среди них найдется хотя бы одно направление убывания функции F.

Ведь, если такое направление существует, любой достаточно малый положительный шаг вдоль него, не выводя из допустимого множества, обеспечит строгое убывание значения F. Значит, если мы хотим более полно описать оптимальную точку x^*, надо найти условие, которое гарантировало бы, что для всех p, удовлетворяющих неравенству $\hat{A}p \geqslant 0$, получим $g(x^*)^T p \geqslant 0$. Поскольку уже известно, что вектор $g(x^*)$ есть линейная комбинация строк $\hat{A}$ с множителями Лагранжа, можно переформулировать задачу: требуется найти условие, при котором

$$g(x^*)^T p = \lambda_1^* \hat{a}_1^T p + \ldots + \lambda_t^* \hat{a}_t^T p \geqslant 0, \tag{3.17}$$

если только $\hat{a}_i^T p \geqslant 0$, $i = 1, \ldots, t$.

Оказывается, что неравенство (3.17) выполняется при всех интересующих нас p только в том случае, когда $\lambda_i^* \geqslant 0$, $i = 1, \ldots, t$. Таким образом, точка x^* не может быть оптимальной, если среди множителей Лагранжа есть отрицательные. Чтобы убедиться в этом, допустим, что x^* — точка локального минимума (так что (3.16) выполнено), но $\lambda_j^* < 0$ при некотором j. Тогда в силу линейной независимости строк $\hat{A}$ найдется неудерживающее направление p, такое, что

$$\hat{a}_j^T p = -1; \quad \hat{a}_i^T p = 0, \quad i \neq j. \tag{3.18}$$

Для него

$$g(x^*)^T p = -\lambda_j^* \hat{a}_j^T p = -\lambda_j^* < 0,$$

и, следовательно, p является возможным направлением спуска, существование которого противоречит оптимальности x^*. Значит, из оптимальности x^* в задаче LIP следует *неотрицательность всех множителей Лагранжа*. Применение этого правила знаков проиллюстрировано на рис. 3i.

Рассматривая тейлоровские разложения функции F в окрестности x^* вдоль удерживающих направлений, можно получить условие оптимальности второго порядка, состоящее в том, что спроектированная матрица Гессе $Z^T G(x^*) Z$ должна быть положительно полуопределенной. Это условие в точности аналогично условию оптимальности второго порядка для задач с ограничениями-равенствами.

Суммируя сказанное выше, выпишем полный набор необходимых условий оптимальности для задачи LIP:

Необходимые условия минимума для задачи LIP

G1. $Ax^* \geqslant b$, *причем* $\hat{A}x^* = b$.

G2. $Z^T g(x^*) = 0$, *или, что эквивалентно,* $g(x^*) = \hat{A}^T \lambda^*$.

G3. $\lambda_i^* \geqslant 0$, $i = 1, \ldots, t$.

G4. $Z^T G(x^*) Z$ *положительно полуопределена.*

Перейдем теперь к выводу достаточных условий оптимальности для LIP. В сравнении со случаем, когда все ограничения имеют вид

равенства, здесь появляется одно усложнение, и связано оно с возможностью обращения в нуль множителей Лагранжа активных ограничений. Вообще множитель Лагранжа, соответствующий j-му активному ограничению, представляет собой оценку первого порядка для скорости изменения целевой функции при движении вдоль

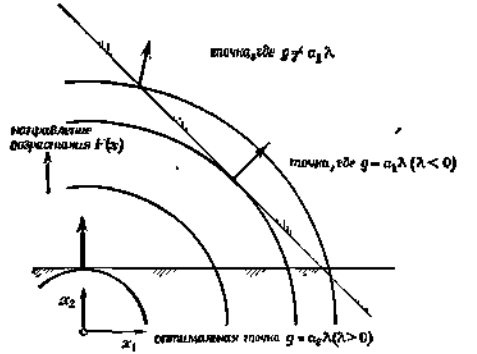

Рис. 31. Применение условий оптимальности первого порядка.

направления, неудерживающего относительно j-го ограничения и удерживающего относительно всех остальных активных ограничений. Поэтому, например, положительность множителя Лагранжа означает, что при движении вдоль соответствующего направления целевая функция будет возрастать. И поэтому же равенство множителя нулю ничего не говорит об истинном знаке изменения функции при указанном перемещении, гарантируя лишь ее постоянство в линейном приближении. Отсюда ясно, что любой набор достаточных условий оптимальности для задачи LIP должен включать какие-то дополнительные требования, которые касаются направлений, неудерживающих по отношению к активным ограничениям с нулевыми множителями Лагранжа.

В подтверждение сказанного рассмотрим пример, когда из-за наличия нулевого множителя Лагранжа положительная определенность спроектированной матрицы Гессе оказывается недостаточным основанием для вывода об оптимальности точки. Возьмем двумерную задачу минимизации разности $x_1^2 - x_2^2$ при ограничении $x_2 \geqslant 0$. В точке $x^* = (0, 0)^T$ имеем $g(x^*) = (0, 0)^T$, $\hat{A} = (0, 1)$, $Z = (1, 0)^T$ и $Z^T G(x^*) Z = 2$. Соответственно $g(x^*) = \hat{A}^T \lambda^*$, где $\lambda^* = 0$. При этом несмотря на то, что спроектированная матрица

Гессе положительно определена, локального минимума в x^* нет, поскольку любой положительный шаг из x^* вдоль направления $(0, \delta)^T$, где δ – положительное число, не нарушит ограничения задачи и приведет к уменьшению значения функции F. В данном случае начало координат не является точкой минимума потому, что, хотя F стационарна вдоль неудерживающего направления $(0, \delta)^T$, ее кривизна вдоль этого направления отрицательна.

Самый простой набор достаточных условий оптимальности для задачи LIP получается, если потребовать, чтобы все множители Лагранжа были положительны. Тогда возрастание F вдоль каждого из неудерживающих направлений гарантировано, и полная сводка условий, обеспечивающих наличие в x^* сильного локального минимума, будет выглядеть так:

Достаточные условия минимума для задачи LIP

H1. $Ax^* \geqslant b$, *причем* $\hat{A}x^* = \hat{b}$.
H2. $Z^T g(x^*) = 0$, *или, что эквивалентно,* $g(x^*) = \hat{A}^T\lambda^*$.
H3. $\lambda_i^* > 0$, $i = 1, \ldots, t$.
H4. $Z^T G(x^*) Z$ *положительно определена.*

Потребовать положительности множителей Лагранжа — не единственный прием вывода достаточных условий оптимальности для задачи LIP. От этого требования можно и отказаться, но тогда надо как-то усилить условие на матрицу $G(x^*)$, чтобы обеспечить положительность кривизны F вдоль тех из возможных направлений, которые являются удерживающими только относительно ограничений с ненулевыми множителями. Например, можно выставить требование положительности кривизны F вдоль любых (независимо от допустимости) направлений такого сорта. Соответствующие условия приведены ниже. Если через $\hat{A}_+$ обозначить матрицу активных ограничений с *положительными* множителями Лагранжа, а через Z_+ матрицу, столбцами которой являются векторы базиса подпространства ортогонального строкам $\hat{A}_+$, то эти условия будут выглядеть следующим образом

Достаточные условия минимума для задачи LIP (с нулевыми множителями Лагранжа)

I1. $Ax^* \geqslant b$, *причем* $\hat{A}x^* = \hat{b}$.
I2. $Z^T g(x^*) = 0$, *или, что эквивалентно,* $g(x^*) = \hat{A}^T\lambda^*$.
I3. $\lambda_i^* \geqslant 0$, $i = 1, \ldots, t$.
I4. $Z_+^T G(x^*) Z_+$ *положительно определена.*

Заметим, что в рассмотренном выше примере данные условия не выполняются. Это естественно, поскольку исследуемая на оптимальность точка в действительности не оптимальна, но она станет таковой, если заменить целевую функцию на $x_1^4 + x_2^4$. В этом случае вновь полученные достаточные условия окажутся выполненными.

Для задач с линейными ограничениями, среди которых есть и

равенства, и неравенства, условия оптимальности получаются комбинированием условий данного и предыдущего разделов. Точнее говоря, условия для смешанной задачи будут отличаться от рассмотренных в этом разделе тем, что «активными» надо будет считать не только удерживающие неравенства, но и все ограничения-равенства, причем знаки их множителей Лагранжа не оговариваются.

3.3.2.2. **Линейное программирование.** Особый класс оптимизационных задач с линейными ограничениями составляют задачи, в которых линейными являются и целевые функции. Их принято называть *задачами линейного программирования* или сокращенно LP-задачами. Таким образом, для LP-задачи

$$F(x) = c^T x, \qquad (3.19)$$

где c — некоторый фиксированный n-мерный вектор (это не самая общая форма задания линейной функции, поскольку здесь нет постоянного слагаемого; однако оно не влияет на решение, и потому его обычно опускают). Градиент от функции (3.19) равен c, а ее матрица Гессе тождественно равна нулю.

Если c — ненулевой вектор, функция (3.19) не ограничена снизу: ее значение может быть сделано сколь угодно большим по модулю отрицательным числом за счет движения вдоль любого ненулевого $\hat{x}$, для которого $c^T\hat{x} < 0$. Следовательно, чтобы задача минимизации такой функции имела конечное решение, в ней обязательно должны присутствовать какие-то ограничения.

Поскольку компоненты вектора c и матриц ограничений обычно выбираются независимо, равенство $c = \hat{A}^T\lambda^*$ из необходимых условий оптимальности чаще всего выполнимо только при невырожденной матрице $\hat{A}$ (она составлена из коэффициентов ограничений-равенств и сдерживающих неравенств). В таких случаях искомый минимум достигается в единственной точке x^*, представляющей собой решение невырожденной системы линейных уравнений вида $\hat{A}x^* = b$. Если же условие $c = \hat{A}^T\lambda^*$ (при неотрицательности множителей неравенств) выполнится для матрицы $\hat{A}$, строковый ранг которой меньше n, то одно и то же минимальное значение функции c^Tx будет достигаться на целом множестве точек x^*. Например, в задаче минимизации $F(x) = 3x_1 + 3x_2$ при единственном ограничении $x_1 + x_2 = 1$ оптимальное значение целевой функции равно 3, а решением будет любой вектор вида $(^1/_2 - \delta,\ ^1/_2 + \delta)^T$.

В гл. 5, где будут рассматриваться методы линейного программирования, фигурирует понятие *двойственная задача*. Смысл его состоит в следующем. Имея оптимизационную (не обязательно линейную) задачу P (*прямую задачу*), в некоторых случаях можно определить связанную с ней задачу D того же типа (*двойственную*

задачу) так, что множители Лагранжа в P будут частью решения D, а множители Лагранжа из D будут содержаться в решении P.

Сейчас нас интересует следующая прямая задача линейного программирования:

$$\text{найти } \min_{x \in \mathbb{R}^n} c^T x$$
$$\text{при ограничениях } Ax \geqslant b,$$

где A есть $m \times n$-матрица. Ниже показано, что отвечающая ей двойственная задача выглядит так:

$$\text{найти } \max_{y \in \mathbb{R}^m} b^T y$$
$$\text{при ограничениях } A^T y = c, \quad y \geqslant 0.$$

Ранее мы условились связывать множители Лагранжа только с активными ограничениями, но здесь при анализе двойственности удобно приписывать множители Лагранжа и неактивным ограничениям, полагая их по определению нулевыми. Не умаляя общности, можно считать, что активными в решении x^* прямой задачи являются первые t ограничений и их множители Лагранжа положительны. Тогда расширенный вектор y^* множителей Лагранжа равен $(\lambda^*, 0)^T$, где $\lambda^* > 0$.

Покажем, что y^* — решение двойственной задачи. Для этого нужно доказать, что y^* есть допустимая точка последней и что в y^* выполнены необходимые условия оптимальности первого порядка. Дело в том, что при нелинейных F эти условия являются лишь необходимыми только из-за того, что в них не учитываются все слагаемые тейлоровских разложений F порядка выше первого. Для линейных F, в разложениях которых эти слагаемые попросту отсутствуют, данные условия становятся и достаточными.

Пусть $\bar{A}$ — матрица активных, а $\bar{\bar{A}}$ — матрица неактивных ограничений прямой задачи в ее решении x^*. Тогда из условий оптимальности этого решения имеем

$$c = \bar{A}^T \lambda^* = (\bar{A}^T \bar{\bar{A}}^T) \begin{pmatrix} \lambda^* \\ 0 \end{pmatrix} = A^T y^*.$$

Следовательно, ограничения-равенства двойственной задачи для y^* выполнены. Кроме того, в силу тех же условий оптимальности $y^* \geqslant 0$, т. е. выполнены и двойственные неравенства. Значит, y^* — допустимая точка двойственной задачи.

Теперь осталось установить, что y^* удовлетворяет условиям оптимальности первого порядка. Для этого прежде всего отметим, что каждой положительной компоненте вектора y^* (т. е. каждому элементу λ^*) отвечает неактивное ограничение неотрицательности в двойственной задаче, а каждой нулевой — активное. Следовательно, все активные в y^* двойственные ограничения образуют систему

равенств вида

$$\begin{pmatrix} \hat{A}^T & \bar{A}^T \\ 0 & I \end{pmatrix}\begin{pmatrix} \lambda^* \\ \bar{0} \end{pmatrix} = \begin{pmatrix} c \\ 0 \end{pmatrix}.$$

Учитывая полученное описание набора активных в y^* ограничений и имея в виду эквивалентность задачи максимизации функции b^Ty задаче минимизации $-b^Ty$, можно утверждать, что необходимым и достаточным условием оптимальности y^* будет соблюдение векторного равенства

$$\begin{pmatrix} \hat{A} & 0 \\ \bar{A} & I \end{pmatrix}\begin{pmatrix} \hat{\sigma} \\ \bar{\sigma} \end{pmatrix} = -b = -\begin{pmatrix} \hat{b} \\ \bar{b} \end{pmatrix}$$

при некоторых $\hat{\sigma}$, $\bar{\sigma}$, причем вектор $\bar{\sigma}$ должен быть неотрицательным. Здесь $\hat{\sigma}$ состоит из множителей Лагранжа ограничений-равенств двойственной задачи, а $\bar{\sigma}$—из множителей активных в y^* неравенств. Проверяемое соотношение распадается на два равенства. Первое имеет вид $\hat{A}\hat{\sigma} = -\hat{b}$ и выполняется при $\hat{\sigma} = -x^*$. Чтобы удовлетворить и второму, надо взять $\bar{\sigma} = \bar{A}x^* - \bar{b}$. Поскольку x^* удовлетворяет всем (в частности, неактивным) ограничениям прямой задачи, в данном случае $\bar{\sigma} \geqslant 0$. Таким образом, достаточные условия оптимальности y^* выполнены, т. е. y^*—решение двойственной задачи. При этом x^* оказывается вектором множителей Лагранжа двойственных ограничений-равенств.

3.3.2.3. Квадратичное программирование. Еще один специальный класс задач с линейными ограничениями составляют *задачи квадратичного программирования*. В них целевыми функциями являются квадратичные формы вида

$$F(x) = c^Tx + \frac{1}{2}x^TGx, \tag{3.20}$$

где c и G суть фиксированные вектор и симметричная матрица. Градиент функции (3.20) равен $Gx + c$, а ее матрица Гессе равна G.

К квадратичным задачам, как и к линейным, применимо понятие «двойственность». Пусть исходная (прямая) задача ставится так:

$$\text{найти } \min c^Tx + \frac{1}{2}x^TGx$$

$$\text{при ограничениях } Ax \geqslant b_0.$$

Тогда двойственной к ней задачей будет следующая:

$$\text{найти } \max_{y \in \mathbb{R}^m,\ w \in \mathbb{R}^n} \Psi(y, w) = b^Ty - \frac{1}{2}w^TGw$$

$$\text{при ограничениях } A^Ty - Gw + c,$$

$$y \geqslant 0.$$

Если y^* - расширенный вектор множителей Лагранжа, отвечающих решению x^* прямой задачи, то пара (y^*, w^*), где $w^* = x^*$, будет допустимой и удовлетворит необходимым условиям оптимальности первого порядка для двойственной задачи. Показывается это в точности по той же схеме, что и в линейном случае. Для того чтобы можно было гарантировать оптимальность (y^*, w^*), нужно потребовать две вещи: невырожденность матрицы G и положительную полуопределенность $AG^{-1}\hat{A}^T$.

3.3.2.4. Оптимизация при простых ограничениях на переменные. На практике нередко встречаются задачи ЦР, в которых допустимые значения вектора x определяются только простыми ограничениями на его компоненты:

$$l_i \leqslant x_i \leqslant u_i, \quad i=1, \ldots, n. \tag{3.21}$$

В этой записи любую из границ можно считать бесконечной, моделируя тем самым односторонние неравенства.

Специфика рассматриваемых задач приводит к значительным упрощениям в условиях оптимальности. Формально эта специфика выражается в том, что матрица активных ограничений $\hat{A}$ всегда будет состоять только из столбцов единичной матрицы, взятых со знаками «плюс» или «минус». При этом активность ограничения означает, что соответствующая переменная принимает свое граничное значение, а потому вместо термина «активное ограничение» здесь часто употребляют термин *зафиксированная переменная*. Векторы, составленные из величин, относящихся к переменным, зафиксированным на нижних границах, принято помечать нижним индексом L, а на верхних — индексом U. Так, например, через x_L и x_U обозначают векторы, набранные из тех компонент x, которые зафиксированы на нижних и верхних границах соответственно. В силу специфики матрицы $\hat{A}$ множитель Лагранжа для активного ограничения на переменную x_i равен $g_i(x^*)$, если $x_i^* = l_i$, и равен $-g_i(x^*)$, если $x_i^* = u_i$ (поскольку ограничение $x_i \leqslant u_i$ можно переписать как $-x_i \geqslant -u_i$).

Далее, матрицу Z можно составить из столбцов единичной матрицы, отвечающих тем переменным, которые не приняли граничных значений (их называют *свободными*, обозначая набранный из них вектор через x_{FR}). Тогда спроектированный градиент $Z^T g(x^*)$ окажется не чем иным, как вектором, составленным из компонент $g(x^*)$, отвечающих свободным переменным, а спроектированной матрицей Гессе будет матрица, скомпонованная из соответствующих им элементов $G(x^*)$. Поэтому, обозначив вектор через g_{FR}^*, а матрицу через G_{FR}^*, мы можем выписать следующие достаточные условия существования в x^* сильного локального минимума функции $F(x)$ при ограничениях (3.21):

Достаточные условия минимума при простых ограничениях
J1. $l \leqslant x^* \leqslant u$, $l < x^*_{FR} < u$.
J2. $g'_{FR} = 0$.
J3. $g''_l > 0$, $g''_u < 0$.
J4. G_{FR} *положительно определена.*

Легче всего проверить эти условия в одномерном случае, когда требуется найти минимум функции одной переменной f на отрезке $[l, u]$. Например, для того чтобы этот минимум достигался в точке $x^* = l$, достаточно соблюдение неравенства $f'(l) \geqslant 0$ (замстьте, что свободных переменных здесь нет).

3.4. ОПТИМИЗАЦИЯ ПРИ НЕЛИНЕЙНЫХ ОГРАНИЧЕНИЯХ

Ограничения, налагаемые на переменные, далеко не всегда содержат только линейные функции. Часто равенства и неравенства, определяющие допустимую область задачи, содержат *нелинейности*. Такими задачами мы и займемся в данном разделе. Точнее говоря, речь пойдет о задачах с ограничениями вида

(i) $c_i(x) = 0$ (ограничение-равенство),
(ii) $c_i(x) \geqslant 0$ (ограничение-неравенство),

где $c_i(x)$ — некоторая нелинейная функция. Понятно, что неравенства $c_i(x) \leqslant 0$ можно не рассматривать, заменяя их эквивалентными неравенствами $-c_i(x) \geqslant 0$.

Содержание данного раздела относится к задачам, *все* ограничения которых нелинейны. Условие оптимальности для задач, включающих и линейные, и нелинейные ограничения, читатель без труда получит сам, скомбинировав результаты предыдущего раздела с представленными ниже.

3.4.1. ЗАДАЧИ С ОГРАНИЧЕНИЯМИ ТИПА НЕЛИНЕЙНЫХ РАВЕНСТВ

Анализ условий оптимальности при нелинейных ограничениях мы начнем со случая, когда все они имеют вид равенств:

NEP найти $\min\limits_{x \in \mathcal{R}^n} F(x)$

при ограничениях $\hat{c}_i(x) = 0$, $i = 1, \ldots, t$.

Вектор градиента функции $\hat{c}_i(x)$ будем обозначать через $\hat{a}_i(x)$, а ее матрицу Гессе — через $\hat{G}_i(x)$. Кроме того, нам потребуется *матрица Якоби* для набора функций ограничений $\{c_i(x)\}$, $i = 1, \ldots, t$. Так называют $t \times n$-матрицу $\hat{A}(x)$, i-я строка которой равна $\hat{a}_i^T(x)$.

Для исследования допустимой точки x^* задачи NEP на оптимальность необходимо конструктивно описать не нарушающие ограничений перемещения из x^*. Тогда на основе анализа пове-

дения функции F при таких перемещениях можно будет сделать какие-то выводы. Когда ограничения были линейными, допустимые перемещения описывались в терминах подпространства возможных направлений (см. разд. 3.3.1). Для нелинейных ограничений столь простое описание невозможно, так как если функция $\hat{c}_i$ нелинейна, то даже при наличии равенства $\hat{c}_i(x^*)=0$ направлений p, таких, что $\hat{c}_i(x^*+\alpha p)=0$ при достаточно малых $|\alpha|$, вообще говоря, не существует.

Чтобы сохранить допустимость, надо двигаться из x^* по *допустимой дуге* (дуга — это направленная кривая в $\mathcal{R}^n$, параметризованная скалярной переменной 0). Например, в случае с двумя переменными и единственным ограничением роль такой дуги играет линия уровня функции ограничения $\hat{c}$, отвечающая ее нулевому значению. Впредь дугу будем обозначать через $\alpha(\theta)$, считая, что $\alpha(0)=x^*$; производная от α по θ в точке x^*, представляющая собой касательный к дуге вектор, ниже обозначается через p.

Движение из x^* вдоль некоторой дуги будет допустимым относительно i-го ограничения, если во всех точках этой дуги функция $\hat{c}_i$ равна нулю. При таком движении нулевой будет и скорость изменения $\hat{c}_i$, по определению равная производной от $\hat{c}_i$ вдоль дуги. В частности, эта производная будет равна нулю в x^*, что в силу правила дифференцирования сложной функции (см. разд. 2.3.2) дает

$$\frac{d}{d\theta}\hat{c}_i(\alpha(\theta))|_{\theta=0}=\nabla\hat{c}_i(\alpha(0))^T p=\hat{a}_i(x^*)^T p=0.$$

Значит, если дуга допустима относительно всех ограничений задачи NEP, то касательный к ней вектор p должен удовлетворять равенству

$$\hat{A}(x^*)\,p=0. \qquad (3.22)$$

В ситуации с линейными ограничениями аналогичное (3.22) равенство (3.10) *полностью* характеризовало допустимые перемещения: любое возможное направление должно было удовлетворять (3.10) и все направления, удовлетворяющие (3.10), были возможными. Теперь же условие (3.22) служит только необходимой характеристикой допустимых перемещений и не является достаточной: оно должно выполняться для любого вектора p, касательного к какой-нибудь допустимой дуге, но не любому удовлетворяющему (3.22) вектору p отвечает допустимая дуга, по отношению к которой он был бы касательным.

Поясним сказанное на простом примере. Для этого рассмотрим в двумерном случае два ограничения $\hat{c}_1(x)=(x_1-1)^2+x_2^2-1$ и $\hat{c}_2(x)=(x_1+1)^2+x_2^2-1$. Оба они выполнены в начале коорди-

нат, и в этой точке равенству (3.22) удовлетворит любой вектор вида $p=(0, \delta)^T$. В то же время допустимых дуг здесь попросту не существует, так как начало координат—единственная точка, где $\hat{c}_1$ и $\hat{c}_2$ обращаются в нуль одновременно.

Равенство (3.22) будет исчерпывающей характеристикой множества векторов p, касательных к допустимым дугам в точке x^*, если в этой точке выполнятся определенные требования к функциям ограничений. Эти требования обычно называют *условиями регулярности*. Существует довольно много различных типов таких условий, причем большинство их имеет сугубо теоретическое значение. С практической же точки зрения наиболее важным является простое условие, состоящее в требовании *линейной независимости градиентов функций ограничений в точке* x^*, или, другими словами, полноты строкового ранга матрицы $\hat{A}(x^*)$. Соблюдение его гарантирует, что любой вектор p, удовлетворяющий равенству (3.22), будет касательным к некоторой допустимой дуге. Отсюда в свою очередь удается получить эффективные необходимые условия оптимальности для NEP. Поэтому ниже будет предполагаться, что матрица $\hat{A}(x^*)$ имеет полный строковый ранг. Отметим, что если все ограничения линейны, то это условие никакой роли не играет, поскольку тогда (3.22) полностью характеризует возможные направления независимо от того, выполнено оно или нет.

Начнем с необходимых условий первого порядка. Понятно, что x^* может быть оптимальной точкой только тогда, когда функция F стационарна в x^* вдоль любой допустимой дуги, т. е. когда $\nabla F(\alpha(\theta))_{\theta=0}=0$. Отсюда по правилу дифференцирования сложной функции получим, что

$$g(x^*)^T p=0 \tag{3.23}$$

при любом p, удовлетворяющем (3.22).

Пусть далее $Z(x^*)$—матрица, чьи столбцы формируют базис подпространства векторов, ортогональных всем строкам $\hat{A}(x^*)$. Тогда из того, что равенство (3.23) справедливо при любых p, получившихся (3.22), следует, что

$$Z(x^*)^T g(x^*)=0. \tag{3.24}$$

Это условие аналогично условию (3.12) для задач с линейными ограничениями, но только теперь матрица Z зависит от x^*. Вектор $Z(x^*)^T g(x^*)$, как и в линейном случае, называют *спроектированным градиентом* от F в x^*.

Рассуждая точно так же, как это делалось при выводе необходимых условий для задач с линейными равенствами, нетрудно показать, что равенство (3.24) эквивалентно требованию, чтобы вектор $g(x^*)$ был линейной комбинацией строк матрицы $\hat{A}(x^*)$:

$$g(x^*)=\hat{A}(x^*)^T\lambda^*. \tag{3.25}$$

Здесь λ^* — некоторый l-мерный вектор множителей Лагранжа.

Введем *функцию Лагранжа*

$$L(x, \lambda) = F(x) + \lambda^T \hat{c}(x) \tag{3.26}$$

с независимыми аргументами x и l-мерным λ. Тогда условие (3.25) становится утверждением о стационарности (по x) этой функции в точке x^* при $\lambda = \lambda^*$.

Полный вывод необходимых условий оптимальности второго порядка (требующий довольно громоздких выкладок) мы приводить не будем и ограничимся лишь краткими заметками доказательства. По аналогии с линейным случаем понятно, что x^* может быть решением задачи NEP только тогда, когда функция F имеет в x^* неотрицательную кривизну вдоль любой из допустимых дуг. Точнее говоря, для каждой допустимой $\alpha(0)$ должно выполняться неравенство

$$\frac{d^2}{d0^2} F(\alpha(0))\Big|_{0=0} \geqslant 0. \tag{3.27}$$

Чтобы вывести отсюда конструктивное условие оптимальности, надо найти выражение производной $d^2F/d0^2$ через исходные функции задачи. При этом в соответствии с правилом дифференцирования сложных функций формула вычисления производной $d^2F/d0^2$ будет содержать два слагаемых: одно — с матрицей Гессе функции F, а второе — со второй производной дуги $d^2\alpha/d0^2$, и именно это второе слагаемое надо заменить чем-то без $d^2\alpha/d0^2$. Здесь надо привлечь равенство (3.25) и учесть постоянство всех функций $\hat{c}_i$ на любой допустимой дуге. В результате этих преобразований из (3.27) получается следующее необходимое условие оптимальности второго порядка: для всех p, удовлетворяющих (3.22), должно выполняться соотношение

$$p^T \left(G(x^*) - \sum_{i=1}^{l} \lambda_i^* \hat{G}_i(x^*)\right) p \geqslant 0. \tag{3.28}$$

Отметим, что центральную роль в этом условии минимума для задач с нелинейными ограничениями играет матрица, которая есть не что иное, как матрица Гессе функции Лагранжа. Ее принято обозначать через $W(x, \lambda)$, т. е.

$$W(x, \lambda) \equiv G(x) - \sum_{i=1}^{l} \lambda_i \hat{G}_i(x).$$

Требование соблюдения неравенства (3.28) для всех p, подчиняющихся (3.22), равносильно требованию положительной полуопределенности матрицы

$$Z(x^*)^T \left(G(x^*) - \sum_{i=1}^{l} \lambda_i^* \hat{G}_i(x^*)\right) Z(x^*). \tag{3.29}$$

Ее принято называть *спроектированной матрицей Гессе* функции Лагранжа.

Еще раз подчеркнем, что в условии оптимальности второго порядка для задачи с нелинейными ограничениями фигурирует специфическая комбинация матриц Гессе для целевой функции и функций ограничений. Таким образом, существенное значение имеет кривизна последних, и это проиллюстрировано на рис. 3]. Он относится к задаче с линейной целевой функцией, кривизна которой по определению равна нулю. Видно, что точка $\hat{x}$ удовлетворяет (8.24), но не является оптимальной именно потому, что кривизна функции ограничения имеет не тот знак.

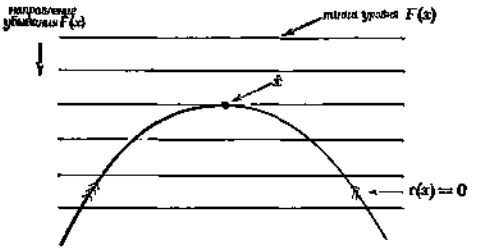

Рис. 3]. Роль кривизны ограничения в условиях оптимальности.

Итак, если ограничения в x^* удовлетворяют требованию регулярности, необходимые условия оптимальности x^* для задачи NEP будут состоять в следующем:

Необходимые условия минимума для задачи NEP

К1. $\hat{c}(x^*) = 0$.

К2. $Z(x^*)^T g(x^*) = 0$, *или, что эквивалентно,* $g(x^*) = \hat{A}(x^*)^T \lambda^*$.

К3. *Матрица* $Z(x^*)^T W(x^*, \lambda^*) Z(x^*)$ *положительно полуопределена.*

Используя аналогичную технику доказательства, можно получить также условия, достаточные для того, чтобы точка x^* была точкой сильного локального минимума для NEP. Они формулируются следующим образом:

Достаточные условия минимума для задачи NEP

L1. $\hat{c}(x^*) = 0$.

L2. $Z(x^*)^T g(x^*) = 0$, *или, что эквивалентно,* $g(x^*) = \hat{A}(x^*)^T \lambda^*$.

L3. *Матрица* $Z(x^*)^T W(x^*, \lambda^*) Z(x^*)$ *положительно определена.*

3.4.2. ЗАДАЧИ С ОГРАНИЧЕНИЯМИ ТИПА НЕЛИНЕЙНЫХ НЕРАВЕНСТВ

В заключение мы рассмотрим условия оптимальности для задач, все ограничения которых заданы нелинейными неравенствами:

$$\text{NIP} \qquad \text{найти } \min_{x \in \mathbb{R}^n} F(x)$$
$$\text{при ограничениях } c_i(x) \geqslant 0, \quad i = 1, \ldots, n.$$

Как и в случае с линейными неравенствами, надо выделять те из ограничений, которые обращаются в анализируемой точке x^* в равенства. Их будем называть *активными* в x^*, а остальные ограничения — *неактивными*. Таким образом, i-е ограничение активно в x^*, если $c_i(x^*) = 0$, и неактивно, если $c_i(x^*) > 0$. Возможности перемещения в окрестности точки x^* определяются только активными ограничениями, а неактивные будут выполнены при любых достаточно малых уклонениях от x^*.

Вывод условий оптимальности для задачи NIP опирается на понятия удерживающих и неудерживающих перемещений (разд. 3.3.2.1) и на понятие дуги, допустимой относительно нелинейного ограничения (разд. 3.4.1). На этой основе строится определение допустимых удерживающих и неудерживающих дуг, которое играет центральную роль в соответствующих доказательствах.

Чтобы получить конструктивные условия оптимальности для задачи NIP, снова, как и в предыдущем разделе, требуется предположение о регулярности ограничений в x^*. Только теперь оно будет относиться не ко всем, а лишь к активным ограничениям. Составленный из их функций t-мерный вектор ниже обозначается через $\hat{c}(x^*)$, а матрица, строками которой являются транспонированные векторы градиентов этих функций, будет обозначаться через $\hat{A}(x^*)$. В качестве условия регулярности ограничений в x^* можно выставить требование полноты строкового ранга этой матрицы. Здесь опять же надо отметить, что это требование существенно только для случая с нелинейными ограничениями.

В предположении, что строки $\hat{A}(x^*)$ линейно независимы, можно, комбинируя доказательства из разд. 3.3.2.1 и 3.4.1, вывести следующие необходимые условия существования в x^* локального минимума:

Необходимые условия минимума для задачи NIP

М1. $c(x^*) \geqslant 0$, *причем* $\hat{c}(x^*) = 0$.

М2. $Z(x^*)^T g(x^*) = 0$, *или, что эквивалентно,* $g(x^*) = \hat{A}(x^*)^T \lambda^*$.

М3. $\lambda_i^* \geqslant 0$, $i = 1, \ldots, t$.

М4. *Матрица* $Z(x^*)^T W(x^*, \lambda^*) Z(x^*)$ *положительно полуопределена.*

Достаточные условия оптимальности, как и в случае с линейными неравенствами, представим в двух вариантах. Это снова связано с проблемой нулевых множителей Лагранжа. Первый набор условий получается если снять эту проблему, потребовав, чтобы все множители Лагранжа были положительны. Тогда *достаточными* условиями существования в x^* сильного локального минимума в задаче NIP будут следующие соотношения:

Достаточные условия минимума для задачи NIP

N1. $c(x^*) \geqslant 0$, *причем* $\hat{c}(x^*) = 0$.

N2. $Z(x^*)^T g(x^*) = 0$, *или, что эквивалентно,* $g(x^*) = \hat{A}(x^*)^T \lambda^*$.

N3. $\lambda_i^* > 0$, $i = 1, \ldots, t$.

N4. *Матрица* $Z(x^*)^T W(x^*, \lambda^*) Z(x^*)$ *положительно определена.*

Можно получить и другие, допускающие наличие нулевых множителей Лагранжа достаточные условия. В них требования к матрице Гессе должны быть усилены таким образом, чтобы была гарантия положительности кривизны F вдоль любой допустимой дуги, которая является удерживающей относительно активных ограничений с ненулевыми множителями и может быть неудерживающей по отношению к остальным активным ограничениям. Условия данного типа представлены ниже. В них через $Z_+(x^*)$ обозначена матрица, чьи столбцы образуют базис подпространства, ортогонального векторам градиентов функций активных ограничений с положительными множителями Лагранжа.

Достаточные условия минимума для задачи NIP (с нулевыми множителями Лагранжа)

O1. $c(x^*) \geqslant 0$, *причем* $\hat{c}(x^*) = 0$.

O2. $Z(x^*)^T g(x^*) = 0$, *или, что эквивалентно* $g(x^*) = \hat{A}(x^*)^T \lambda^*$.

O3. $\lambda_i^* \geqslant 0$, $i = 1, \ldots, t$.

O4. *Матрица* $Z_+(x^*)^T W(x^*, \lambda^*) Z_+(x^*)$ *положительно определена.*

Замечания и избранная библиография к разд. 3.4

Условия оптимальности для задачи с ограничениями, рассмотренные в этой главе, часто называют условиями Куна — Таккера (см. Кун и Таккер (1951) и Кун (1976)). Их более полное и подробное изложение читатель найдет, например, в книгах Фиакко и Мак-Кормика (1968), Авриела (1976) и в работе Пауэлла (1974).

Выше были представлены только такие условия оптимальности в нелинейных задачах, которые либо необходимы, либо достаточны. Условия же, являющиеся и необходимыми, и достаточными, в общем случае необычайно сложны — даже для задач на безусловный экстремум. Поэтому они не были включены в текст книги, тем более что их практическая проверка возможна лишь в исключительных ситуациях. Интересное обсуждение условий такого сорта для задач без ограничений дано в книге Гё и Томаса (1968).

ГЛАВА 4

МЕТОДЫ БЕЗУСЛОВНОЙ МИНИМИЗАЦИИ

Ла, если Муза не может бежать, когда с нее сняли путы, значит, ей просто не хватает резвости.
Джон Драйден (1697)

В гл. 4, 5 и 6 представлена некоторая выборка из обширного семейства методов оптимизации, разработанных к настоящему времени для различных категорий задач. Изложение носит обзорный характер, причем приводятся мотивы, по которым казалось целесообразным включить в него тот или иной метод. Кроме того, обсуждаются отдельные теоретические и вычислительные особенности рассматриваемых схем.

Набор методов, описанных в этой и двух последующих главах, довольно ограничен. Дело в том, что исчерпывающий обзор занял бы слишком много места; только методов безусловной минимизации хватило бы на отдельную книгу. К тому же наряду с обзором оптимизационных схем мы хотели обсудить целый ряд сугубо практических вопросов, связанных с их эксплуатацией. Они вынесены в гл. 7 и 8, где даны советы, как формализовать нестандартные задачи, как анализировать причины отказа или неэффективной работы алгоритмов и как добиваться от оптимизационных программ математического обеспечения наилучших показателей. Некоторые из этих рекомендаций имеют общий характер и могут быть отнесены ко всем методам. Однако наибольшее значение они все же имеют для тех алгоритмов, к которым адресуются.

Обзор содержит главным образом методы, которые неоднократно и успешно применялись авторами данной книги на практике. Прочие алгоритмы даны лишь постольку, поскольку это способствует пониманию каких-то принципиальных вещей или служит основой для последующих построений. Ссылки на литературу, где можно почерпнуть дополнительные сведения, вынесены в специальные разделы, завершающие изложение каждой группы методов.

4.1. МЕТОДЫ ДЛЯ ФУНКЦИЙ ОДНОЙ ПЕРЕМЕННОЙ

4.1.1. ПОИСК НУЛЯ ФУНКЦИИ ОДНОЙ ПЕРЕМЕННОЙ

В разд. 3.2.1 установлено, что для существования в точке x^* безусловного минимума гладкой функции $F(x)$ необходимо наличие равенства $F'(x)^*=0$. Таким образом, задачи поиска минимума и нуля

оказываются тесно связанными (хотя и неэквивалентными). Во многом близки и пути их решения, причем методы поиска нуля несколько проще. С них мы и начнем. Точнее говоря, речь пойдет о задаче отыскания на конечном отрезке такой точки x^*, в которой некоторая нелинейная функция $f(x)$ принимает нулевое значение и меняет знак.

Первый вопрос, возникающий в связи с рассматриваемой задачей,—это вопрос о том, что называть ее численным решением. Машина считает с ограниченной точностью и оперирует конечным набором представимых значений аргумента x. В этом наборе скорее всего не найдется такого $\bar{x}$, для которого вычисляемое значение $fl(f(\bar{x}))$ величины $f(\bar{x})$ было бы нулевым, т. е. рассчитывать на точный нуль бессмысленно. Значит, надо довольствоваться тем, что найден отрезок $[a, b]$, удовлетворяющий условиям

$$f(a)\,f(b) < 0, \quad |a-b| < \delta,$$

где δ — некоторый «малый» допуск. Поскольку истинный корень лежит в данном случае где-то в пределах $[a, b]$, в качестве его окончательной оценки можно взять любую точку этого отрезка.

В приведенных ниже описаниях алгоритмов поиска нуля используются для специальных термина. Мы будем называть *интервалом неопределенности* отрезок, содержащий искомую точку x^* и имеющий минимальную длину среди всех включающих x^* отрезков, оперированных алгоритмом. Мы будем говорить также, что нуль *локализован* на некотором интервале, если значения f на его границах имеют разные знаки.

4.1.1.1. Метод деления пополам. Этот метод состоит в *последовательном сокращении интервала неопределенности на основе сопоставления знаков функции.* Допустим, каким-то образом выявлен исходный интервал $[a, b]$, в крайних точках которого знаки f различны, т. е. $f(a)f(b)<0$. Вычислим значение f в середине этого интервала. Если оно окажется нулевым, задача решена; в противном случае заменим a или b средней точкой в зависимости от того, совпадает ли знак функции в ней со знаком $f(a)$ или $f(b)$ соответственно. Тем самым интервал неопределенности будет уменьшен вдвое. После этого вычисляется значение f в новой средней точке и т. д. Работа описанного алгоритма проиллюстрирована на рис. 4.я.

Деление пополам обеспечивает локализацию x^* с любым допуском δ, *если только точности вычисления f будет достаточно для правильного определения ее знаков.* При этом локализация корня на отрезке длины, не превосходящей δ, достигается вычислением около $\log_2((b-a)/\delta)$ значений функции f в разных точках. Теоретически деление пополам является «оптимальным» алгоритмом поиска нуля для функций, меняющих знак на $[a, b]$. Точнее говоря, ему отвечает *минимальная гарантированная* оценка длины интервала неопреде-

ленности, получающегося ценой *заданного* количества обращений к процедуре расчета функции.

Одной из характеристик быстродействия метода поиска нуля является скорость сходимости в точку генерируемой им последо-

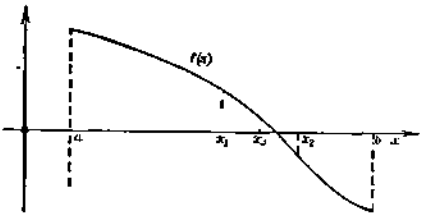

Рис. 4а. Первые три точки последовательности, генерируемой методом деления пополам.

вательности интервалов неопределенности. При делении пополам эта последовательность сходится линейно, причем асимптотический параметр ошибки равен $^1/_2$.

4.1.1.2. Метод Ньютона. Слабое место алгоритма деления пополам состоит в том, что учитываются лишь знаки функции f в пробных точках, а модули ее значений в расчет не принимаются. Если же функция f *ведет себя достаточно хорошо*, модули тоже имеют информативную ценность, и представляется разумным использовать их при выборе очередного приближения искомого корня. Ориентируясь на значения, а не только на знаки f, можно строить *моделирующие* f функции $\tilde{f}$, точки нулей которых вычисляются просто. Соответственно поиск корня x^* можно организовать как итеративный процесс, в котором очередной оценкой x^* служит точка нуля очередной функции $\tilde{f}$.

Если f дифференцируема, первым претендентом на роль модельной функции $\tilde{f}$ для $(k+1)$-й итерации будет прямая, *касательная* к f в текущей точке x_k. Если производная $f'(x_k)$ не равна нулю, эта прямая пересечет ось абсцисс в точке

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \qquad (4.1)$$

Итеративную процедуру, заданную формулой (4.1), принято называть *методом Ньютона для поиска нуля*. Ее геометрическая интерпретация ясна из рис. 4b. В отличие от деления пополам, где результатом каждого шага является новый интервал неопределенности, итерация метода Ньютона дает новую оценку корня. Что же каса-

ется интервала неопределенности, то он в методе Ньютона никак не используется и вообще может оставаться невыявленным в течение всего процесса.

Некоторые популярные и на первый взгляд не имеющие ничего общего с методом Ньютона вычислительные процедуры в действительности оказываются его реализациями. Например, формула $x_{k+1}=\frac{1}{2}(x_k+a/x_k)$ правила «подели и сложи», с помощью которого школьников учат вычислять $\sqrt{a}$, есть не что иное, как преобразованная формула ньютоновской итерации для поиска нуля функции $f(x)=x^2-a$.

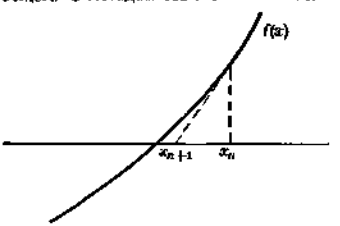

Рис. 4b. Итерация метода Ньютона для поиска нуля функции $f(x)$; очередная точка x_{n+1} определяется пересечением прямой, касательной к $f(x)$ в x_n, с осью абсцисс.

Метод Ньютона фактически исходит из предположения о том, что при оценивании корня функции f ее можно считать «похожей на линейную». Поэтому естественно ожидать, что, если такое «сходство» действительно имеет место, он будет работать хорошо. И в самом деле, когда производная $f'(x^*)$ не равна нулю и начальная точка x_0 «достаточно близка» к x^* (т. е. x_0 находится в той окрестности x^*, где f хорошо аппроксимируется линейной зависимостью), метод Ньютона сойдется к x^* *квадратично* (см. разд. 2.3.6).

Пример 4.1. При вычислении по методу Ньютона величины $\sqrt{4}$ с начальным приближением $x_0=2.5$ получаются ошибки $\{x_k^2-4\}$, $k=0, \ldots, 4$, равные 2.25, 2025, 2.439×10^{-3}, 3.717×10^{-7} и 8.0×10^{-15} (расчеты проводились с шестнадцатиразрядной точностью).

Основные трудности с методом Ньютона возникают из-за того, что его образцовая скорость сходимости *локальна*, т. е. реализуется лишь в малой окрестности искомого корня. Если же точка x_0 недостаточно близка к x^*, этот метод может безнадежно разойтись. Например, в изображенном на рис. 4c случае ньютоновское приближение оказывается абсолютно бессмысленным. Кроме того, поскольку ньютоновский шаг не определен, когда производная $f'(x_k)$ равна нулю, его реализация при «малых» $f'(x_k)$ всегда будет связана с численными трудностями. Таким образом, метод Ньютона следует применять с осторожностью: далеко не любую задачу на поиск нуля удается решить с его помощью.

4.1.1.3. Методы секущих и ложного положения. В числе недостатков метода Ньютона помимо прочего называют необходимость

вычисления производной f' на каждой итерации. В практических задачах это может оказаться затруднительным, а иногда и просто невозможным. В то же время линейные модели $\hat{f}$ функции f можно строить, не используя производные. К примеру, в качестве $\hat{f}$

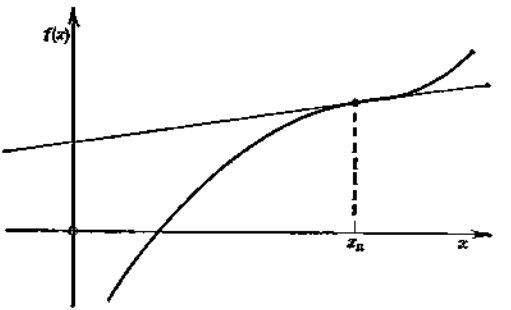

Рис. 4с. Случай расходимости метода Ньютона.

можно каждый раз брать прямую, проходящую через точки, полученные на двух предыдущих итерациях. По существу это означает замену в ньютоновской формуле производной $f'(x_k)$ ее конечно-разностной аппроксимацией $(f_k - f_{k-1})/(x_k - x_{k-1})$, где через f_k обозначено $f(x_k)$. Тем самым мы приходим к процедуре поиска нуля, которая задается формулой вида

$$x_{k+1} = x_k - \left(\frac{x_k - x_{k-1}}{f_k - f_{k-1}}\right) f_k.$$

Эту процедуру называют *методом секущих* или *методом линейной интерполяции*. Как и метод Ньютона, она порождает оценки искомого корня, а не интервалы неопределенности. Для запуска метода секущих нужны значения функции f в двух точках.

Если производная $f'(x^*)$ не равна нулю и точки x_0, x_1 достаточно близки к x^*, метод секущих сойдется сверхлинейно со скоростью $r \approx 1.6180$. Таким образом, он может работать весьма эффективно. Так, применение его в примере 4.1 с $x_0 = 1$, $x_1 = 2.5$ дает для $k = 2, \ldots, 7$ ошибки $\{x_k^* - 4\}$, равные -5.51×10^{-1}, -6.53×10^{-2}, 2.44×10^{-2}, -1.00×10^{-3}, -1.53×10^{-4} (расчеты проведены с шестнадцатиразрядной точностью).

К сожалению, метод секущих нередко расходится, и это возможно из-за того, что при совпадении знаков f_k и f_{k-1} его очередная точка попадает вне отрезка, соединяющего две предыдущих. Если бы

знаки f_k и f_{k-1} всегда различались, текущая оценка корня обязательно ложилась бы между предыдущими, и тогда сходимость была бы гарантирована. Данное условие можно обеспечить ценой несложной модификации метода — достаточно слегка изменить правило выбора из тройки x_{k+1}, x_k, x_{k-1} пары точек для построения следующей линейной аппроксимации. Вместо того чтобы из двух точек

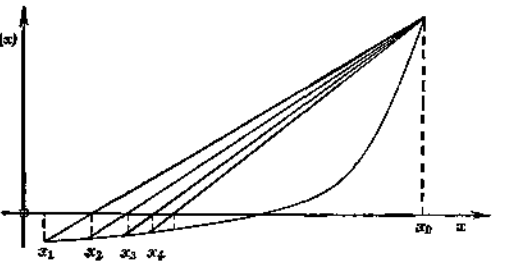

Рис. 4d. Пример медленной сходимости метода ложного положения.

x_k, x_{k-1} всегда оставлять первую, надо оставлять ту их пару, в которой знак функции f отличен от знака f_{k+1}. Соответствующую процедуру называют *методом ложного положения*. Этот метод начинает работать с точек x_0 и x_1, удовлетворяющих неравенству $f_0 f_1 < 0$. Для него ограниченность и сходимость последовательности оценок искомого корня обеспечены. Однако эти гарантии достигаются ценой потери быстродействия. Сходимость метода ложного положения всего лишь линейна, причем асимптотический параметр ошибки может быть сколь угодно близким к единице. Ситуация, в которой метод сходится очень медленно, изображена на рис. 4d (в этом примере $f_k < 0$ для всех $k \geq 1$ и потому далекая от корня точка x_0 никогда не будет отброшена). Как правило, деление пополам оказывается намного эффективнее метода ложного положения, хоть и получен он из значительно более быстродействующей схемы. Последнее говорит о том, что, модифицируя какой-либо алгоритм с целью расширить его область сходимости, никогда не лишне подумать о последствиях такой модификации в отношении скоростных качеств.

*** 4.1.1.4. Дробная интерполяция и методы высоких порядков.** Выше были представлены методы поиска нуля гладкой функции f, основанные на использовании линейных приближений. Можно применять и более изощренные способы аппроксимации f. Например, имея несколько значений f и ее производных, в качестве модельной функции f можно взять полином соответствующего порядка и искать

очередное приближение нуля f как корень этого полинома. Правда, здесь возникает неприятная проблема выбора корня, так как у нелинейного полинома их, как правило, несколько.

Вопрос о выборе корня не возникает, если интерполировать f дробной функцией вида

$$\tilde{f} = \frac{x-c}{d_0+d_1x+d_2x^2}.$$

Величины параметров c, d_0, d_1 и d_2 в данном случае подбираются так, чтобы значения функций $\tilde{f}$, f и их производных совпали в двух точках. Если же производные по каким-то причинам недоступны, можно воспользоваться дробной интерполяционной функцией без слагаемого с x^2 в знаменателе. Ее параметры однозначно определяются по трем значениям f.

4.1.1.5. Регуляризованные алгоритмы поиска нуля. Рассмотренную ранее процедуру деления пополам можно интерпретировать как один из способов реализации общей схемы поиска нуля путем генерирования последовательности вложенных интервалов $\{I_j\}$, каждый из которых содержит искомый корень x^*. Точнее говоря, общий подход состоит в том, чтобы при заданном интервале I_0, содержащем x^*, строить систему $\{I_j\}$, $j=1, \ldots$, удовлетворяющую условиям $I_j \subset I_{j-1}$ и $x^* \in I_j$. В частности, имея I_{j-1}, интервал I_j можно определять по знакам f на границах I_{j-1} и в какой-нибудь внутренней точке u этого интервала. Так работают многие методы и в том числе *метод деления пополам*.

В методе деления пополам в качестве u выбирается середина I_{j-1}. Однако никто не мешает определять u по методу Ньютона или, скажем, с помощью дробной аппроксимации. При этом не обязательно строить аппроксимирующую функцию по значениям f на границах интервала неопределенности. Можно, например, использовать для этого «лучшие» из точек, известных к началу очередной итерации. Так, в примере, изображенном на рис. 4d, линейную модель для второго шага эффективнее построить по точкам x_1 и x_2, хотя интервалом неопределенности будет $[x_2, x_0]$.

Приведенные соображения используются в так называемых *регуляризованных* методах поиска нуля, которые считаются лучшими среди существующих. Эти методы получаются комбинированием надежных, но не слишком быстрых схем (типа деления пополам) с быстродействующими, но не очень надежными (типа линейной или дробной аппроксимации). В результате получаются алгоритмы, которые в хороших случаях сходятся очень быстро, а в плохих работают немногим хуже обычных гарантированных процедур.

Рассмотрим конструкцию регуляризованного метода с линейной интерполяцией. Допустим, что на очередной итерации интервал неопределенности равен $[a, b]$. По двум «лучшим» из найденных на предыдущих итерациях точек построим линейную аппроксимирую-

щую функцию, которая примет нулевое значение в некоторой точке u. Если бы не надо было специально заботиться о сходимости, мы просто вычислили бы $f(u)$ и отбросили одну из старых точек, получив тем самым набор для построения следующего приближения. Однако теперь, прежде чем включать u в этот набор и вычислять $f(u)$, мы должны убедиться, что u — «разумная» точка.

Определение «разумности» u содержит несколько требований. Во-первых, точка u должна принадлежать отрезку $[a, b]$. Во-вторых, расстояние от u до лучшей из имеющихся точек должно быть меньше половины длины интервала неопределенности. Иначе шаг деления пополам скорее всего приведет к большему сокращению этого интервала (т. е. второе условие нужно для гарантии того, что в неблагоприятной ситуации метод не потребует значительного увеличения количества вычислений функции по сравнению с делением пополам).

Наконец, необходимо убедиться, что точка u «существенно отлична» от лучшей из предыдущих, так как если окажется, что они «численно неразличимы», то, сохранив u, именно по ним придется строить линейную аппроксимацию на следующем шаге, а такая аппроксимация из-за ошибок округления почти наверняка будет бессмысленной. Таким образом, мы просто потеряем этот шаг. Когда нарушается какое-нибудь из двух первых условий, u заменяют серединой интервала неопределенности. Если же нарушится третье, вместо u обычно берут точку, лежащую по ту же сторону от лучшей из прежних, что и u, но отстоящую от нее на фиксированное «малое» расстояние δ. При этом величина δ не должна превосходить половины допуска на длину окончательного интервала неопределенности. Указанное правило замены приводит к тому, что два последних шага алгоритма, как правило, оказываются «δ-шагами» и точка x^* локализуется на отрезке длины 2δ.

4.1.2. МЕТОДЫ ОДНОМЕРНОЙ МИНИМИЗАЦИИ

Методы безусловной минимизации функции одной переменной во многом аналогичны рассмотренным выше процедурам поиска нуля. Более того, точку минимума функции f в принципе можно искать как точку нуля производной f' с помощью любой из этих процедур. Например, деление пополам действительно привлекается иногда для этой цели. Однако, как правило, здесь разумнее обращаться к методам, специально предназначенным для задач минимизации.

Первые два из представленных ниже методов поиска минимума требуют вычислений только самой минимизируемой функции u не используют значений ее производных. Они работают по принципу последовательного сокращения интервала неопределенности и опираются на некоторое условие, позволяющее указывать отрезок локализации минимума функции, если она *унимодальна*. Есть несколько эквивалентных определений унимодальности. Одно из них звучит

так: функция $f(x)$ унимодальна на отрезке $[a, b]$, если существует единственная точка $x^* \in [a, b]$, такая, что для любых $x_1, x_2 \in [a, b]$ и $x_1 < x_2$

$$f(x_2) > f(x_1), \quad \text{если} \quad x_2 < x^*;$$
$$f(x_1) < f(x_2), \quad \text{если} \quad x_1 > x^*.$$

Когда известно, что f достигает безусловного минимума и унимодальна на отрезке $[a, b]$, положение точки минимума можно уточнить,

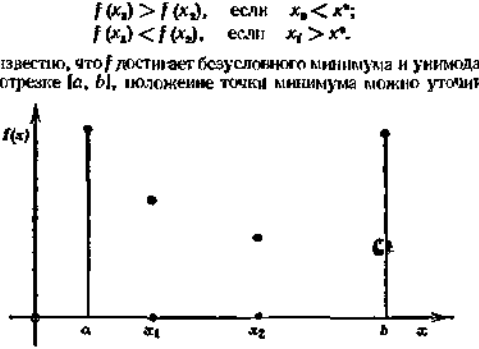

Рис. 4е. Сокращение интервала неопределенности для унимодальной функции

вычислив f в *двух* внутренних точках отрезка. Например, в ситуации, изображенной на рис. 4е, по значениям f в x_1 и x_2 можно утверждать, что минимум реализуется на отрезке $[x_1, b]$.

4.1.2.1. Поиск Фибоначчи. Для построения конкретного метода одномерной минимизации, работающего по принципу последовательного сокращения интервала неопределенности, надо задать правило выбора на каждом шаге двух внутренних точек. Конечно, желательно, чтобы одна из них всегда использовалась в качестве внутренней и для следующего интервала. Тогда частота вычислений функции сократится вдвое и одна итерация потребует расчета только одного нового значения f.

Стратегией, обеспечивающей максимальное гарантированное сокращение интервала неопределенности при заданном количестве вычислений функции и соответственно претендующей на «оптимальность», является *поиск Фибоначчи*. Эта стратегия опирается на числа Фибоначчи $\{F_i\}$, которые вырабатываются рекуррентной формулой вида

$$F_k = F_{k-1} + F_{k-2}, \qquad F_0 = F_1 = 1.$$

Начальными членами последовательности $\{F_i\}$ будут 1, 1, 2, 3, 5, 8, 13, ... При фиксированном количестве N обращений к процедуре расчета функции поиск Фибоначчи состоит в просмотре точек, дробящих интервалы неопределенности в отношениях, задаваемых

числами $F_{N-1}, F_{N-2}, \ldots$. Точнее говоря, если принять длину исходного интервала за F_N, то длиной k-го интервала будет F_{N-k}, а его оцениваемые внутренние точки будут отстоять от левой границы на F_{N-k-2} и на F_{N-k-1}, причем в одной из них значение f всегда известно из предыдущего шага. Проще всего эта процедура выглядит в случае, когда предполагаются только два вычисления функции.

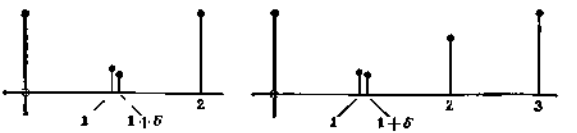

Рис. 41. Точки поиска Фибоначчи для $N=2$.

При исходном интервале $[0, F_3]$ ($F_3=2$) значения f надо подсчитать в двух точках, отвечающих F_0 и F_1, а точнее в $x_1=1$ и $x_2=1+\delta$, где δ «пренебрежимо мало». При этом независимо от соотношения между значениями $f(x_1)$ и $f(x_2)$ новый интервал неопределенности будет иметь длину, «почти равную» половине начальной (см. левую диаграмму на рис. 41).

На рис. 41 изображена также ситуация с тремя вычислениями функции. В этом примере длина начального интервала неопределенности равна $[0, F_3]$ ($F_3=3$) и в качестве двух первых внутренних точек берутся F_1 и F_2 (т. е. $x_1=1$, $x_2=2$). В соответствии с соотношением между значениями $f(x_1)$ и $f(x_2)$ новым интервалом неопределенности будет $[0, 2]$. В середине его значение f уже известно. Поэтому, чтобы получить окончательный интервал длины $(1+\delta)$, требуется только одно дополнительное вычисление функции.

Результатом поиска Фибоначчи с N вычислениями функции является интервал неопределенности, составляющий (с точностью до δ) $1/F_N$-ю часть длины исходного интервала.

4.1.2.2. Метод золотого сечения. Поиск Фибоначчи удобен далеко не всегда. В частности, он плохо подходит для решения одномерных подзадач, возникающих в схемах многомерной минимизации. Запросы к точности решений таких подзадач практически никогда не формулируются в терминах числа обращений к процедуре расчета функции. Значит, для них пришлось бы либо хранить избыточный набор чисел Фибоначчи, либо многократно генерировать эти числа по мере необходимости. К тому же поиск Фибоначчи нелегко приспособить к часто используемому критерию остановки, требующему, чтобы значения функции на окончательном интервале неопределенности разнились не более чем на заданную величину.

Из-за указанных недостатков поиска Фибоначчи вместо него часто используют другую, почти столь же эффективную теоретически,

но не требующую задания длины окончательного интервала неопределенности процедуру. Ее называют *методом золотого сечения*. Можно показать, что

$$\lim_{k\to\infty}\frac{F_{k-1}}{F_k}=\frac{2}{1+\sqrt{5}}=\tau\approx 0{,}6180,$$

где τ есть решение квадратного уравнения $\tau^2+\tau-1=0$. На этом числе и основан рассматриваемый метод. Когда он применяется к начальному интервалу $[0, 1]$, первыми пробными внутренними точками будут τ и $1-\tau$ (т. е. примерно 0.6180 и 0.3820). Затем левый или правый кусок интервала будет отброшен, но в том и другом случае одна из этих точек обязательно окажется внутренней для нового интервала, делящей его две части, длины которых снова относятся,

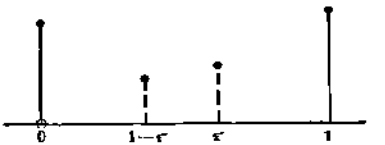

Рис. 4g. Расположение точек в методе золотого сечения.

как τ к $1-\tau$. Золотое сечение можно, таким образом, рассматривать как предельный случай поиска Фибоначчи. Расположение точек в этом методе показано на рис. 4g.

В процедуре золотого сечения длина интервала неопределенности на каждом шаге уменьшается в τ раз, т. е. сходится к нулю линейно.

4.1.2.3. Полиномиальная интерполяция. В методах золотого сечения и поиска Фибоначчи выбор очередного интервала неопределенности всегда ограничен только двумя альтернативами. При этом в правиле, на основании которого осуществляется этот выбор, учитываются лишь соотношения между значениями функции f, но не сами значения. Если же f – гладкая функция, используя ее значения, можно, как и в случае с поиском нуля, получать существенно более быстродействующие процедуры минимизации. Общий принцип их функционирования таков: на каждом шаге f аппроксимируется некоторой модельной функцией $\hat{f}$, точку минимума которой найти легко, и эта точка берется в качестве текущего приближения искомой. Поскольку линейная функция (если она не постоянна) конечного минимума не имеет, простейшей пригодной моделью функции f будет парабола вида $\hat{f}=^1/_2ax^2+bx+c$.

Если $a>0$, то $\hat{f}$ имеет минимум в точке x^*, удовлетворяющей равенству $ax^*+b=0$.

ритма. Он запускается в точке $(-1.2, 1.0)^T$. Видно, что для перехода в окрестность решения потребовалось довольно много вычислений функции. Хотя метод многогранника не предназначен для гладких задач, все же пример достаточно показателен. Надо только добавить, что в общем случае с негладкой функцией он будет работать несравненно хуже, чем в рассмотренном.

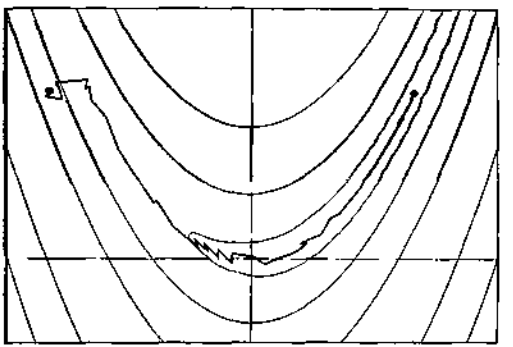

Рис. 41. Траектория поиска минимума функции Розенброка методом многогранника.

*4.2.3. СОСТАВНЫЕ НЕДИФФЕРЕНЦИРУЕМЫЕ ФУНКЦИИ

На практике нередко возникают задачи минимизации функций, которые сами по себе являются негладкими, но составлены из гладких. В таких случаях один из подходов к поиску решения может заключаться в том, чтобы заменить негладкую составную задачу безусловной минимизации эквивалентной ей *гладкой задачей с ограничениями*. Затем можно попытаться решить последнюю каким-нибудь методом оптимизации при ограничениях, учитывающим ее специфику. Соответствующие методы рассмотрены в разд. 6.8. Следует подчеркнуть, что в библиотечных программах решения негладких задач, о которых идет речь, указанное преобразование, как правило, реализуется *неявно*, так что пользователь может и не догадываться о нем.

Типы применяемых преобразований проиллюстрированы ниже на примере трех наиболее часто встречающихся составных задач.

Пример 4.3. Первой рассмотрим задачу l_∞ вида

$$\text{найти} \min_{x\in R^n} \max \{f_1(x), f_2(x), \ldots, f_m(x)\}. \qquad (4.3)$$

где $\{f_i(x)\}$ — набор гладких функций.

Эта негладкая задача преобразуется в гладкую введением новой переменной x_{n+1}, имеющей при данном x смысл верхней границы для всех значений $\{f_i(x)\}$. Получающаяся в результате эквивалентная (4.3) задача формулируется так:

$$\text{найти} \min_{x\in R^{n+1}} x_{n+1}$$
$$\text{при ограничениях } f_i(x)\leqslant x_{n+1},\ i=1,\ \ldots,\ m.$$

В ней есть n нелинейных ограничений-неравенств, т. е. она принадлежит классу задач, значительно более сложных по структуре, чем задачи безусловной минимизации. Таким образом, гладкость достигнута ценой усложнения структуры, и это принципиальный момент.

Пример 4.4. Следующей возьмем задачу l_1:

$$\text{найти} \min_{x\in R^n} \sum_{i=1}^{m} |f_i(x)|. \qquad (4.4)$$

В ее названии отражено, что минимизируемой функцией является p-норма вектора $(f_1(x), f_2(x), \ldots, f_m(x))$ с $p=1$ (см. разд. 2.2.4.2).

Построение гладкого аналога задачи (4.4) опирается на тот простой факт, что значение $f_i(x)$ всегда можно расщепить на положительную и отрицательную составляющие. Точнее говоря, всегда возможно представление $f_i(x)=r_i-s_i$, где величины r_i и s_i неотрицательны. При этом гарантировано неравенство $|f_i(x)|\leqslant r_i+s_i$, обращающееся в равенство, когда $s_i r_i=0$. Отсюда ясно, что задача (4.4) эквивалентна следующей:

$$\text{найти} \min_{r_i\in R^m:\ s_i\in R^m} \sum_{i=1}^{m} (r_i+s_i)$$
$$\text{при ограничениях } f_i(x)=r_i-s_i,\ i=1,\ 2,\ \ldots,\ m;$$
$$r_i\geqslant 0;\ s_i\geqslant 0,\ i=1,\ 2,\ \ldots,\ m.$$

Пример 4.5. В заключение рассмотрим задачу

$$\text{найти} \min_{x\in R^n} \sum_{i=1}^{m} \max\{f_i(x),\ 0\}.$$

Рассуждения, подобные использованным в предыдущем примере, здесь приводят к такому гладкому аналогу:

$$\text{найти} \min_{x\in R^n;\ r,\ s\in R^m} \sum_{i=1}^{m} r_i$$

$$\text{при ограничениях } f_i(x) - r_i \leqslant s_i, \qquad i = 1, 2, \ldots, m;$$

$$r_i \geqslant 0;\ s_i \geqslant 0, \qquad i = 1, 2, \ldots, m.$$

В этом и предыдущем случаях переход к гладкой задаче сопровождается существенным увеличением числа переменных. Это, однако, не может послужить серьезной причиной для отказа от предлагаемых преобразований, поскольку не сильно затруднит работу алгоритма, в котором простые ограничения на переменные учитываются специальным образом (см. разд. 5.5.1).

Задачи l_∞ и l_1 обычно возникают из потребностей согласования данных, и поэтому для них характерны «малые» оптимальные значения минимизируемых сумм. Если же минимум просто равен нулю, то соответствующую точку можно искать как решение гладкой задачи о наименьших квадратах:

$$\text{найти} \min_{x\in R^n} \sum_{i=1}^{m} f_i^2(x).$$

Когда минимум в задаче l_∞ или l_1 «мал», но все же отличен от нуля, его можно попытаться определить, решая серию задач вида

$$\text{найти} \min_{x\in R^n} \sum_{i=1}^{m} w_i^{(k)} f_i^2(x). \tag{4.5}$$

Для них разработаны специальные алгоритмы (см. разд. 4.7).

Обозначим через $\tilde{x}^{(k)}$ оптимальную точку в задаче (4.5), и пусть $w_i^{(0)} = 1/m$, $i = 1, \ldots, m$. Тогда формула построения последовательности весов в (4.5) для l_∞ будет выглядеть следующим образом:

$$w_i^{(k+1)} = \frac{1}{S} f_i^2(\tilde{x}^{(k)})\, w_i^{(k)}.$$

Здесь $S = \sum_{i=1}^{m} f_i^2(\tilde{x}^{(k)})\, w_i^{(k)}$ и соответственно $\sum_{i=1}^{m} w_i^{(k+1)} = 1$. Аналогичная формула для задачи l_1 такова:

$$w_i^{(k+1)} = \frac{S}{|f_i(\tilde{x}^{(k)})|}.$$

В ней $S = \sum_{i=1}^{m} 1/|f_i(\tilde{x}^{(k)})|$ (предполагается, что значения $f_i(\tilde{x}^{(k)})$ отличны от нуля при всех i). Чаще всего двух-трех итераций

по k оказывается вполне достаточным, чтобы получить приемлемые решения исходных задач.

И в случае с l_∞, и в случае с l_1 нет нужды очень точно определять промежуточные значения $\hat{x}^{(k)}$. Поэтому число итераций, необходимых для отыскания пригодной точки $\hat{x}^{(k)}$, обычно невелико. Время, затрачиваемое на решение задач (4.5), сокращается еще и потому, что $\hat{x}^{(k)}$, как правило, оказывается хорошим начальным приближением для поиска $\hat{x}^{(k+1)}$.

Предлагаемая процедура может не сработать для задачи l_1, если какое-то из значений $f_i(\hat{x}^{(k)})$ для точного $\hat{x}^{(k)}$ будет нулевым. В этом случае формула пересчета весов теряет смысл, и не очень ясно, чем ее заменить. Можно было бы предложить несколько способов преодоления указанной трудности, например переходить к новому начальному набору весов или отбрасывать обращающиеся в нуль компоненты f (предусмотрев возможность их возвращения на последующих итерациях). Однако гарантий удачи ни один из таких приемов не дает. Утешением служит то, что на практике рассматриваемая ситуация встречается крайне редко.

Эффективность применения описанных способов решения задач согласования данных очень сильно зависит от того, насколько правильно выбрана модель согласования. Если она хороша, то неплохой результат можно получить уже обычным методом наименьших квадратов (первый шаг в представленных схемах), и тогда для решения задач l_1 и l_∞ потребуется очень мало дополнительных итераций по k. Поскольку при этом промежуточные задачи (4.5) можно решать довольно грубо, общее время, затрачиваемое на реализацию процедуры, будет сравнимо с временем, нужным для решения одной задачи минимизации суммы квадратов с хорошей точностью.

Замечания и избранная библиография к разделу 4.2

Изложенный выше метод многогранника представляет собой модификацию алгоритма Спендли, Хекста и Химсворта (1962), сконструированную Нелдером и Мидом (1965). Иные модификации читатель найдет в работе Паркинсона и Хатчинсона (1972). Достаточно полный обзор прочих методов прямого поиска содержится в работе Свэна (1972).

Методы решения задач минимизации составных функций были предметом активных исследований последних лет. О них еще пойдет речь в разд. 6.8, где будут даны соответствующие ссылки. Схема решения задач l_1 и l_∞ с линейными f_i путем последовательного поиска минимумов взвешенных сумм квадратов предложена Лоусоном (см. Лоусон и Хансон (1974)).

4.3. МЕТОДЫ ДЛЯ ГЛАДКИХ ФУНКЦИЙ МНОГИХ ПЕРЕМЕННЫХ

4.3.1. МОДЕЛЬНАЯ СХЕМА МИНИМИЗАЦИИ ГЛАДКИХ ФУНКЦИЙ

В этом разделе рассматриваются методы безусловной минимизации дважды непрерывно дифференцируемой функции $F(x)$ векторного аргумента x.

Для разработки любого оптимизационного алгоритма принципиальное значение имеет выбор *меры прогресса* при переходе от одного приближения к другому. Понятно, что, конструируя итеративный метод, важно иметь некоторое разумное правило, по которому будет выноситься суждение о том, «лучше» новая точка предыдущей или нет, а такое правило предполагает наличие указанной меры. Эта мысль красной нитью пройдет через все последующее изложение. Для задач безусловной минимизации естественной мерой прогресса служит приращение целевой функции. Представляется разумным требовать, чтобы F убывала на каждой итерации, т. е. соблюдалось *условие спуска*, состоящее в том, что $F_{k+1}<F_k$ для всех $k\geqslant 0$. Методы, которые удовлетворяют этому требованию, называются *методами спуска*. Именно они будут предметом обсуждения в оставшейся части данной главы.

Все представленные ниже методы укладываются в рамки следующей модельной схемы:

Алгоритм U (*модельная схема решения n-мерных задач безусловной минимизации*). Имея текущее приближение x_k искомой точки x^*, на очередной итерации надо выполнить следующие действия:

U1. [Проверка соблюдения условий останова.] Проверить условия останова и, если они выполнены, вычисления прекратить и взять x_k в качестве искомого решения; в противном случае перейти к следующему шагу.

U2. [Расчет направления поиска.] Рассчитать ненулевой n-мерный вектор p_k, называемый направлением поиска.

U3. [Расчет длины шага.] Вычислить положительное число α_k (длину шага), обеспечивающее неравенство $F(x_k+\alpha_k p_k)<F(x_k)$.

U4. [Пересчет оценки решения.] Положить $x_{k+1}\leftarrow x_k+\alpha_k p_k$, $k\leftarrow k+1$ и вернуться к шагу U1.

Важно отметить, что шаг U3 в модельной схеме предполагает решение некоторой одномерной задачи (определения скаляра α_k).

Возможность соблюдения условия спуска при положительном α_k гарантирована лишь при наличии у p_k определенных свойств. Чаще всего в качестве такой гарантии используется требование, чтобы вектор p_k, вычисляемый на k-й итерации, был *направлением*

спуска в x_k, т. е. удовлетворяющее неравенству $g_k^T p_k < 0$. Здесь и в дальнейшем через g_k обозначается вектор $g(x_k)$. Точно так же через F_k, G_k впредь будут обозначаться $F(x_k)$ и $G(x_k)$ соответственно. Когда p_k — направление спуска, неравенство $F(x_k + \alpha p_k) < F(x_k)$ выполнится при любом достаточно малом положительном α (см. разд. 3.2.2).

4.3.2. СХОДИМОСТЬ МОДЕЛЬНОЙ СХЕМЫ

Прежде чем перейти к конкретным методам, рассмотрим условия, при которых последовательность, генерируемая по модельной схеме, сходится к точке x^*, в которой градиент $g(x^*)$ равен нулю. Строгий анализ сходимости выходит за рамки данной книги, так что мы ограничимся краткой сводкой соответствующих результатов. Подробные доказательства читатель найдет в литературе, ссылки на которую даны в конце раздела.

В модельной схеме значение минимизируемой функции F монотонно убывает. Однако неравенства $F_{k+1} < F_k$ сами по себе еще не обеспечивают сходимости последовательности $\{x_k\}$ к точке минимума F. Это видно из примера с функцией x^2 и последовательностью точек $1/2 + 2^{-k}$. Из $F_{k+1} < F_k$ и ограниченности $F(x)$ снизу вытекает только одно — существование предела последовательности $\{F_k\}$. Совпадет ли этот предел с $F(x^*)$ или нет, не ясно.

Строго монотонно убывающая последовательность $\{F_k\}$, генерируемая по модельной схеме, может не сойтись к минимуму по двум причинам. Во-первых, как бы ни выбиралось направление p_k, все можно испортить неудачным способом построения шагов α_k, назначая их так, что величины убывания F по итерациям будут слишком быстро стремиться к нулю. Во-вторых, решения не получить, если метод расчета направлений p_k выдает векторы, вдоль которых функция F почти постоянна. Последнее означало бы, что направление p_k почти касательно к линии уровня $F(x) = F_k$, т. е. g_k и p_k почти ортогональны (величина $-g_k^T p_k / (\|g_k\| \cdot \|p_k\|)$ близка к нулю). Значит, если мы хотим получить гарантированно сходящуюся процедуру, надо позаботиться о том, чтобы выбор α_k обеспечивал «существенное убывание» F на каждой итерации и чтобы угол между направлением p_k и градиентом g_k всегда отличался от прямого не менее, чем на некоторую фиксированную ненулевую величину.

В доказательстве сходимости модельной схемы кроме двух сформулированных выше условий оговаривается некоторое необременительное требование к функции F. Относится оно к ее множеству уровня. Соответствующее определение звучит так: для данной функции F и числа β *множеством уровня* $L(\beta)$ называется совокупность всех точек x, удовлетворяющих неравенству $F(x) \leqslant \beta$. Требование же, о котором идет речь, состоит в том, что множество $L(F(x_0))$ должно быть ограниченным и замкнутым. Везде в этой главе будет

предполагаться, что оно выполнено. Заметим, что тем самым из рассмотрения исключаются функции типа e^x, которые ограничены снизу, но строго убывают при стремлении $\|x\|$ в бесконечность.

Итак, если (i) функция F дважды непрерывно дифференцируема, (ii) множество уровня $L(F(x_0))$ замкнуто и ограничено, (iii) F «существенно убывает» на каждой итерации и (iv) при всех k угол между p_k и градиентом отличается от прямого не менее чем на фиксированную ненулевую величину, то алгоритм U генерирует последовательность точек, для которой

$$\lim_{k\to\infty} \|g_k\| = 0.$$

Сходимость такого типа принято называть *глобальной*, поскольку она не предполагает близости x_0 к стационарной точке.

4.3.2.1. Вычисление длины шага. Качество процедуры выбора шага в методе спуска оценивается тем, какое изменение функции он обеспечивает на каждой итерации. В частности, для сходимости нужно, чтобы шаг приводил к «существенному убыванию» F. Этому требованию можно удовлетворить, подчинив α_k специальным ограничениям. Известно несколько наборов таких ограничений. Например, существенное убывание F обеспечено, если для α_k выполнены условия Гольдштейна — Армийо:

$$0 < -\mu_1\alpha_k g_k^T p_k \leqslant F_k - F_{k+1} \leqslant -\mu_2\alpha_k g_k^T p_k. \qquad (4.6)$$

Здесь μ_1, μ_2 — числа, удовлетворяющие неравенствам $0<\mu_1\leqslant\mu_2<1$. Двусторонние ограничения на α_k в (4.6) гарантируют, что шаг будет и не «слишком малым» и не «слишком большим».

Вычисляя шаг на основании правила (4.6), обычно его представляют произведением некоторой начальной оценки $\alpha^{(0)}$ на степени положительного параметра ω. В результате $\alpha^{(k)}$ находится как первый из элементов последовательности $\{\omega^j\alpha^{(0)}\}$, $j=0, \ldots,$ для которого выполняется неравенство (4.6).

Эффективность описанной процедуры расчета шага в конкретных алгоритмах сильно зависит от выбора оценки $\alpha^{(0)}$. Типичны реализации с $\alpha^{(0)}=1$, причем это объясняется не тем, что именно такая оценка дает максимум вероятности удовлетворить (4.6) с первой попытки, а тем, что она лучше других согласуется с ньютоновскими и квазиньютоновскими способами задания направлений спуска (см. разд. 4.4 и 4.5). В общем получается, что процедура хороша лишь тогда, когда назначаемая априори величина $\alpha^{(0)}$, как правило, будет хорошим шагом и другие элементы последовательности $\{\omega^j\alpha^{(0)}\}$ часто просматривать не придется.

Мы хотим подчеркнуть, что само по себе условие (4.6) еще не служит гарантией качества шага, и, когда речь идет о выборе хорошего $\alpha^{(0)}$, имеется в виду нечто большее, чем выбор с единствен-

ной целью удовлетворить этому условию. Для большинства возникающих на практике задач значение $\alpha^{(0)}=10^{-2}$ удовлетворяло бы ему при разумных μ_1 и μ_2 на всех итерациях (т. е. подходящий шаг α_k устанавливался бы ценой вычисления лишь одного значения функции), и тем не менее при любом из используемых правил расчета направлений спуска задание такого $\alpha^{(0)}$ было бы исключительно *неэффективно*. Следовательно, работу процедуры выбора шага нельзя оценивать только по количеству вычислений функции, требуемых для достижения условий, гарантирующих сходимость. Обязательно нужно учитывать полное уменьшение функции на каждой итерации. К сожалению, излишне увлекаясь теоремами сходимости, об этом нередко забывают.

Каков бы ни был критерий приемлемости α_k (имеются в виду ограничения типа (4.6)), обычно он задает некий *диапазон* подходящих значений, одни из которых «лучше», а другие «хуже». Интуиция подсказывает, что «лучшими» следует считать те значения, которые приводят к большему изменению F. Сможет ли алгоритм расчета длины шага эффективно отыскивать «хорошие» в этом смысле α_k, зависит как от способа генерирования в нем пробных шагов, так и от ограничений, налагаемых на α_k из соображений сходимости.

Понятно, что повышение качества выбора шагов α_k на итерациях метода спуска всегда будет связано с какими-то дополнительными условиями, и эти усилия оправданны только до тех пор, пока они с лихвой окупаются увеличением эффективности итераций. В равновесии, к которому следует стремиться, ни упрощение, ни усложнение процедуры определения α_k не должны приводить к улучшению функционирования метода в целом. Точка такого равновесия меняется от метода к методу и даже от задачи к задаче. Поэтому нужны разные способы вычисления, и в частности разные критерии приемлемости α_k. Один из таких критериев (правило (4.6)) мы уже рассмотрели. Теперь перейдем к критериям иного сорта, согласованным с интерпретацией задачи выбора шага как задачи на поиск одномерного минимума. Иногда желательно или даже существенно, чтобы шаг α_k был хорошим приближением точки минимума функции F вдоль направления p_k. Во-первых, это обеспечивает «существенное» уменьшение F, а, во-вторых, высокая скорость сходимости многих методов спуска достигается только в том случае, когда шаг выбирается именно этим способом. Практичный критерий приемлемости шага α_k как приближенного решения задачи одномерной минимизации состоит в требовании, чтобы модуль производной функции F по направлению p_k в точке $x_k+\alpha_k p_k$ был существенно меньше, чем в x_k:

$$|g(x_k+\alpha_k p_k)^T p_k| \leqslant -\eta g_k^T p_k. \qquad (4.7)$$

Здесь η — число из диапазона $0\leqslant\eta<1$, определяющее точность, с которой α_k аппроксимирует стационарную точку F вдоль p_k. Варьируя значение η, мы можем тем самым управлять трудоем-

костью процедуры вычисления α_k. Заметим кстати, что неравенство (4.7) не допускает «слишком малых» шагов.

Вычисление α_k на основе критерия (4.7) с «малым» η называют «аккуратным одномерным поиском»; если же $\eta = 0$, то говорят о «точном одномерном поиске».

Неравенство (4.7) не содержит изменения функции, и поэтому оно, вообще говоря, не гарантирует существенного убывания F. Чтобы обеспечить такое убывание, (4.7) обычно дополняют условием вида

$$F(x_k) - F(x_k + \alpha p_k) \geqslant -\mu \alpha g_k^T p_k, \tag{4.8}$$

где $0 < \mu \leqslant 1/2$. Оно означает, что α_k будет выбираться из того диапазона, в котором график функции $F(x_k + \alpha p_k)$ лежит не выше прямой $l(\alpha) = F(x_k) + \mu \alpha g_k^T p_k$.

Множество значений α_k, удовлетворяющих (4.7) и (4.8), обозначим через $\Gamma(\eta, \mu)$. Можно показать, что при $\mu \leqslant \eta$ это множество не пусто.

Основное достоинство условия (4.7) как критерия приемлемости шага состоит в том, что оно хорошо согласуется с эффективными процедурами одномерной минимизации. В частности, его удобно использовать в комбинации с регуляризованными методами одномерного поиска на основе полиномиальной интерполяции. Для большинства функций соответствующие алгоритмы вычисления шага работают очень быстро. Надо отметить также, что при малых μ (скажем, $\mu \sim 10^{-4}$) любой шаг α, удовлетворяющий (4.7), будет, как правило, автоматически удовлетворять (4.8).

Среди разнообразных методов выбора α_k выделяется класс эффективных процедур, работающих по принципу генерирования последовательности $\{I_j\}$ вложенных интервалов, содержащих множество $\Gamma(\eta, \mu)$. Для построения таких последовательностей используются рассмотренные в разд. 4.1.2 регуляризованные схемы одномерного поиска с полиномиальной интерполяцией. При этом вычисления прерываются, как только точка минимума интерполирующего полинома попадает в множество $\Gamma(\eta, \mu)$. Ее и принимают в качестве искомого α_k.

Чтобы запустить процедуру генерирования вложенных интервалов, нужен начальный отрезок локализации искомого шага. Его задают неравенствами

$$0 \leqslant \delta \leqslant \|\alpha p_k\| \leqslant \Delta, \tag{4.9}$$

где δ есть минимальное разрешенное расстояние между x_{k+1} и x_k, а Δ — оценка сверху расстояния от x_k до точки минимума F вдоль p_k. Корректные значения параметров δ и Δ, вообще говоря, зависят от x_k и F. По смыслу δ аналогичен параметру разделения точек на итерациях регуляризованных процедур поиска нуля (см. разд. 4.1.1.5). Он должен отражать точность, с которой удается вычислять значения F (подробности см. в разд. 8.5). Что же касается

параметра Λ, то о его величине в общем случае ничего конкретного сказать нельзя. Библиотечные программы безусловной минимизации обычно предполагают, что он будет задан пользователем (см. разд. 8.1.3.4). Ситуация упрощается, если задача поиска безусловного минимума в действительности решается как подзадача в методе минимизации при линейных ограничениях (см. разд. 5.2). Тогда в качестве Δ можно брать максимальный шаг, не нарушающий ограничений.

Подобно алгоритмам на основе правила Гольдштейна — Армийо, регулярированные методы вычисления шага с полиномиальной интерполяцией требуют задания исходной оценки $\alpha^{(0)}$, причем хорошая оценка, как правило, сокращает объем вычислений. К сожалению, как показано в разд. 4.4.2.1, бывают случаи, в которых разумную оценку $\alpha^{(0)}$ дать *невозможно*. В этих случаях выбор в (4.9) верхней границы для $\|\alpha p_k\|$ приобретает особое значение, так как становится единственным средством избежать бессмысленных расчетов в процессе одномерной минимизации.

Выше неявно предполагалось, что работа процедуры выбора шага α_k сопровождается вычислениями градиента функции F в каждой пробной точке. Соответственно ни о каких трудностях контроля соблюдения неравенств (4.7) речи не было. Однако если расчет градиента $g(x)$ — относительно дорогое удовольствие, то имеет смысл воспользоваться другим критерием приемлемости шага и взять для одномерной минимизации алгоритм, которому не нужны значения производных. В частности, (4.7) можно заменить неравенством

$$\frac{|F(x_k+\alpha p_k)-F(x_k+\nu p_k)|}{\alpha-\nu} \leqslant -\eta g_k^T p_k, \qquad (4.10)$$

где ν — некоторое число из диапазона $0 \leqslant \nu \leqslant \alpha$. Это неравенство не допускает слишком малых шагов и вместе с (4.8) обеспечивает существенное убывание F на каждой итерации.

4.3.2.2. Вычисление направления поиска. Для конкретизации модельной схемы поиска безусловного минимума кроме алгоритма вычисления шага надо определить, как будет выбираться «хорошее» направление p_k. В отличие от одномерного случая, когда возможны только два движения — «вперед» и «назад», уже в двумерной задаче множество приемлемых направлений бесконечно. Таким образом, проблема выбора p_k действительно существует. Вся оставшаяся часть данной главы посвящена описанию различных способов (с указанием соответствующих мотивировок) ее решения. Надо сказать, что именно способ задания p_k «определяет лицо алгоритма» минимизации. Поэтому названия алгоритмам обычно дают по реализованным в них правилам вычисления направлений спуска, а к вычислению шага принято относиться как к некой отдельной процедуре.

Рассмотрим линейную аппроксимацию функции F, основанную на ее тейлоровском разложении в окрестности точки x_k:

$$F(x_k+p) \approx F_k + g_k^T p.$$

Если ориентироваться на эту аппроксимацию и предполагать, что шаг вдоль p будет единичным, то представляется разумным стремиться к такому направлению p, которому отвечает большая по модулю отрицательная величина $g_k^T p$. Это вроде бы сулит значительное уменьшение F. Ясно, что в данном случае необходима какая-то нормировка сравниваемых векторов. Иначе по любому p, такому, что $g_k^T p < 0$, можно было бы построить вектор p, доставляющий величине $g_k^T p$ любое отрицательное значение: достаточно было бы взять в качестве p произведение $\bar{p}$ на соответствующее положительное число. Тем самым сравнение потеряло бы всякий смысл. Итак, можно выбирать p_k из соображений минимизации величины $g_k^T p$ на множестве всех векторов p, нормализованных подходящим образом. Это означает, что p_k будет решением задачи вида

$$\text{найти } \min_{p \in \mathbb{R}^n} \frac{g_k^T p}{\|p\|}, \tag{4.11}$$

где $\|\cdot\|$ — некоторая норма.

Решение задачи (4.11) целиком определяется выбором нормы, фигурирующей в ее постановке. Если эта норма задается симметричной положительно определенной матрицей C, т. е. $\|p\| = (p^T C p)^{1/2}$, то решением (4.11) будет

$$p_k = -C^{-1} g_k. \tag{4.12}$$

В частности, если взять евклидову норму, т. е. $\|p\| = (p^T p)^{1/2}$, вектор p_k оказывается антиградиентом:

$$p_k = -g_k. \tag{4.13}$$

Поскольку он является решением задачи (4.11) с нормой типа обычной длины, его называют *направлением наискорейшего спуска*. Соответственно и реализацию модельной схемы с вычислением p_k по формуле (4.13) называют *методом наискорейшего спуска*.

Если градиент не равен нулю, направление наискорейшего спуска, безусловно, будет направлением убывания F (так как в этом случае $g_k^T p_k = -g_k^T g_k < 0$), причем угол между ним и градиентом всегда далек от прямого. Значит, комбинация любого из рассмотренных ранее правил выбора шага с вычислением p_k по формуле (4.13) дает глобально сходящийся алгоритм безусловной минимизации.

К сожалению, свойство глобальной сходимости некой процедуры минимизации еще не означает, что она эффективна. Это положение прекрасно иллюстрируется анализом *скорости сходимости* метода наискорейшего спуска. Подобный анализ обычно начинают

с изучения поведения оцениваемого метода на квадратичных функциях. Это упрощает выкладки и в то же время позволяет выявлять некоторые общие свойства метода, так как любая гладкая функция, если только рассматривать ее в малой окрестности, очень похожа на квадратичную (см. разд. 3.2.3).

Пусть $F(x)$—квадратичная форма вида $c^Tx+\frac{1}{2}x^TGx$, где вектор c и матрица G не зависят от x, причем G симметрична и положительно определена. Если для поиска минимума такой функции $F(x)$ воспользоваться методом наискорейшего спуска с вычислением шага путем точного одномерного поиска, то он сгенерирует последовательность приближений, *линейно* сходящуюся к решению. Обозначив через λ_{max} и λ_{min} максимальное и минимальное собственные числа матрицы G, можно показать, что при этом будет справедлива оценка

$$F(x_{k+1})-F(x^*) \approx \frac{(\lambda_{max}-\lambda_{min})^2}{(\lambda_{max}+\lambda_{min})^2}(F(x_k)-F(x^*)) =$$
$$=\frac{(\varkappa-1)^2}{(\varkappa+1)^2}(F(x_k)-F(x^*)),$$

где $\varkappa$ есть $\mathrm{cond}\,(G)$ — спектральное число обусловленности матрицы G.

Представленный результат обескураживает тем, что допускает значения параметра асимптотической ошибки, *сколь угодно близкие к единице*, т. е. процентное сокращение ошибки на одной итерации может быть сколь угодно малым. Например, уже при $\varkappa=100$ (что считается неплохой обусловленностью) оценка параметра ошибки оказывается равной $(99/101)^2\approx 0.96$, так что ожидаемый выигрыш за счет одного шага очень невелик. И в самом деле, даже для небольшого улучшения оценки решения по методу наискорейшего спуска обычно требуются сотни итераций. Это относится к квадратичным задачам и тем более к задачам с функциями общего вида.

На рис. 4j изображена «траектория движения» метода наискорейшего спуска в задаче минимизации функции Розенброка (пример 4.2). Гладкие кривые — это линии уровня функции, отрезки ломаной соответствуют шагам спуска на отдельных итерациях. Шаги определялись аккуратным одномерным поиском. Алгоритм начал работать с точки $(-1.2, 1.0)^T$ и мог бы надолго «застрять» в окрестности точки $(0.3, 0.1)^T$, если бы не помогла счастливая случайность — на одной из итераций процедура одномерного поиска неожиданно нашла *второй* минимум по направлению. После этого было выполнено еще несколько сотен итераций, но ощутимого изменения целевой функции это не дало. Счет был прерван после 1000 итераций вдали от искомого решения. Этот конкретный пример с умеренно плохо обусловленной функцией еще раз подтверждает, что существование доказательства формальной сходимости метода и отсутствие его реальной сходимости (т. е. сходимости за приемлемое число шагов) вполне совместимы.

Мы хотим подчеркнуть, что линейность сходимости алгоритма сама по себе еще не может служить основанием для вывода о его непригодности. Другое дело, что показатель пошагового прогресса

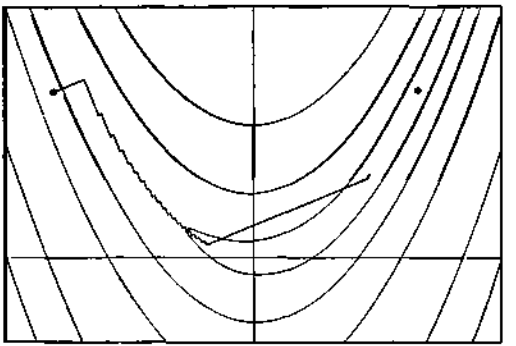

Рис. 4.1. Траектория поиска минимума функции Розенброка методом наискорейшего спуска.

для линейно сходящихся алгоритмов часто оказывается очень малым. Последующие разделы этой главы посвящены методам со *сверхлинейной* сходимостью.

Замечания и библиография к разделу 4.3

В качестве источника, содержащего строгое математическое толкование термина «существенное убывание функции» и многие фундаментальные теоремы сходимости, мы рекомендуем книгу Ортеги и Рейнболдта (1970). Подробное обсуждение процедур выбора шага на основе регуляризованных методов с полиномиальной интерполяцией читатель найдет в статье Гилла и Мюррея (1974е). Алгоритмы, основанные на применении условия (4.6), предлагались Гольдштейном и Прайсом (1967) и Флетчером (1970а). Вулф (1969) и Пауэлл (1976а) использовали вместо (4.7) неравенство

$$g(x_k+\alpha_k p_k)^T p_k \geqslant \gamma g_k^T p_k, \tag{4.14}$$

где $0<\gamma<1$. Оно не допускает слишком малых α_k, но в отличие от (4.7) всегда позволяет делать сколь угодно большие шаги. Метод одномерного поиска на основе (4.14) и (4.8) с $\eta \geqslant \mu$ описан в статье Шанно и Фуа (1976).

Для некоторых алгоритмов (таких, как метод сопряженных градиентов из разд. 4.8.3.3) теоретически важно, чтобы задача одномерной минимизации решалась очень точно, и поэтому при их реализации возникает желание взять η меньшим чем μ. Надо, однако, иметь в виду, что в данном случае удовлетворить парам условий типа (4.7) (или (4.10)) и (4.8) будет нелегко, а то и просто невозможно. Например, множество $\Gamma(\eta, \mu)$ может оказаться пустым. Правда, это бывает редко, но все же бывает, и в особенности на начальных итерациях работы алгоритма, когда норма $\|g_k\|$ велика или когда функция F плохо отмасштабирована (как, например, штрафные и барьерные функции, рассматриваемые в разд. 6.2.1). Исчерпывающее обсуждение общего случая, в котором η может принимать *любое* значение из $[0, 1]$, выходит за рамки данной книги. Отметим только, что всегда найдутся шаги, обеспечивающие существенное убывание $F(x)$ и удовлетворяющие неравенствам (4.7) или (4.10) при *каком-то* значении η из этого диапазона. За дальнейшими подробностями мы отсылаем читателя к работе Гилла, Мюррея, Сондерса и Райт (1979).

Идея использования направления наискорейшего спуска для построения алгоритма многомерной минимизации восходит к Коши (1847) и послужила предметом обширного исследования. Детальный анализ скорости сходимости метода наискорейшего спуска дан в книгах Канторовича и Акилова (1964) и Лейбергера (1973).

Существует довольно много методов безусловной минимизации, логика функционирования которых несколько отличается от изложенной в модельной схеме разд. 4.3.1. В частности, в них длина шага α_k всегда берется равной единице, так что новое приближение определяется по формуле $x_{k+1}=x_k+p_k$. В данном случае, чтобы найти подходящее направление p_k, обеспечивающее соблюдение условия спуска, может понадобиться перебрать несколько пробных векторов. Методы указанного типа часто называют *методами доверительной окрестности* (в противовес *методам с регулировкой шага*, о которых речь шла выше). Краткий обзор таких методов приводится в замечаниях к разд. 4.4. Кроме того, один из методов доверительной окрестности, предназначенный для решения нелинейных задач о наименьших квадратах, будет разобран в разд. 4.7.3.

4.4. МЕТОДЫ ВТОРОГО ПОРЯДКА

4.4.1. МЕТОД НЬЮТОНА

Алгоритм, который нам предстоит рассмотреть в этом разделе, исторически является первым из методов, основанных на *квадратичной аппроксимации* минимизируемой функции F. Такая аппроксимация, оставаясь достаточно простой, в то же время намного точнее, чем линейная, используемая в классическом методе наискорей-

шего спуска. Это позволяет строить на ее основе эффективные алгоритмы.

Располагая первыми и вторыми производными целевой функции F, в качестве ее квадратичной модели можно взять сумму первых трех членов тейлоровского разложения F в окрестности текущей точки x_k, т. е. воспользоваться приближенным равенством вида

$$F(x_k+p) \approx F_k + g_k^T p + \frac{1}{2} p^T G_k p. \qquad (4.15)$$

Минимум его правой части (если он существует) достигается на векторе p_k, доставляющем минимум квадратичной форме

$$\Phi(p) = g_k^T p + \frac{1}{2} p^T G_k p. \qquad (4.16)$$

Будучи стационарной точкой $\Phi(p)$, этот вектор должен (см. разд. 3.2.3) удовлетворять равенству

$$G_k p_k = -g_k. \qquad (4.17)$$

Алгоритм минимизации, в котором направление p_k определяется системой (4.17), называют *методом Ньютона*, а решение этой системы — *ньютоновским направлением*.

В задаче поиска минимума произвольной квадратичной функции с положительно определенной матрицей вторых производных метод Ньютона дает решение за одну итерацию, причем *независимо* от выбора начальной точки (заметим, что шаг α_k при этом надо взять единичным). Значит, мы вправе надеяться, что в случаях, когда матрицы G_k оказываются положительно определенными и модельные функции (4.15) хорошо приближают исходную, этот метод будет быстросходящимся. И в самом деле, если метод Ньютона применить к нелинейной функции F общего вида, то при определенных условиях он сойдется к искомой точке x^* квадратично. Эти условия состоят в следующем: матрица Гессе минимизируемой функции в точке x^* должна быть положительно определенной, начальное приближение x_0 должно лежать достаточно близко к x^*, а последовательность шагов $\{\alpha_k\}$ должна сходиться к единице.

Высокая скорость локальной сходимости метода Ньютона делает его чрезвычайно привлекательным алгоритмом безусловной минимизации. К тому же использование вторых производных позволяет организовать контроль соблюдения достаточных условий оптимальности (см. разд. 3.2.2). Короче говоря, метод Ньютона считается чем-то вроде эталона, с которым надо сравнивать другие алгоритмы, и немало усилий потрачено на разработку процедур, приближающихся к нему по быстродействию. Однако, говоря о достоинствах, не следует забывать и недостатки. Метод Ньютона может работать плохо или даже отказать, если используемые в нем квадратичные приближения функции F будут неверно описывать поведение F вне малых окрестностей опорных точек x_k.

Если матрица G_k положительно определена, то система (4.17) разрешима единственным образом и определяет единственный минимум модельной функции (4.15). В данном случае гарантировано, что решением (4.17) будет направление спуска, так как $g_k^T p_k = -g_k^T G_k^{-1} g_k < 0$. При этом косинус угла между градиентом g_k и вектором p_k отличается от нуля не менее, чем на величину, обратную числу обусловленности G_k. Значит, если известно, что все матрицы G_k будут положительно определенными, а их числа обусловленности — равномерно ограниченными сверху, то можно утверждать, что применение ньютоновских направлений в комбинации с каким-нибудь методом выбора шага α_k на основе рассмотренных в разд. 4.3.2.1 критериев дает глобально сходящийся алгоритм. (Раньше метод Ньютона с регулировкой шага называли «релаксационным», но сейчас этот термин практически не употребляется.) Регулировка шага обязательна, поскольку единичный шаг вдоль ньютоновского направления может привести к увеличению F несмотря на то, что он приводит в точку минимума модельной функции. Важно, однако, подчеркнуть, что квадратичная сходимость метода Ньютона достижима лишь при условии, что длины шагов будут достаточно быстро сходиться к своему «естественному» значению, равному единице. Отметим одно интересное обстоятельство: ньютоновское направление можно рассматривать как направление «наискорейшего спуска», если норму в (4.11) определить по формуле $\|p\| = (p^T G_k p)^{1/2}$.

Когда матрица G_k не является положительно определенной, квадратичная модельная функция может не иметь ни конечной точки минимума, ни какой-либо другой стационарной точки. Напомним (разд. 3.2.3), что $\Phi(p)$ наверняка будет неограниченной снизу, если у G_k есть отрицательные собственные числа, и что необходимым и достаточным условием наличия у $\Phi(p)$ единственной стационарной точки является невырожденность G_k. Если же G_k — вырожденная матрица, стационарные точки будут существовать только при условии, что градиент g_k принадлежит подпространству, натянутому на столбцы G_k.

В настоящее время нет общепринятого определения «метода Ньютона» для расчета направления спуска при знакоопределенной матрице G_k, поскольку среди специалистов нет согласия относительно того, как использовать локальную квадратичную аппроксимацию F в этом случае. Здесь мыслимы разные стратегии. Например, можно считать, что «лучшее» направление поиска следует искать среди направлений, вдоль которых квадратичная модельная функция неограничена. Можно рассуждать и по-другому. В общем есть много схем организации спуска, построенных на базе метода Ньютона и работоспособных в отсутствие положительной определенности G_k. Все эти схемы, по возможности использующие ньютоновские направления и уклоняющиеся от них только тогда, когда это

необходимо, принято называть *модифицированными методами Ньютона.*

Появление у G_k отрицательных собственных значений помимо всего прочего неизбежно усложняет проблему выбора масштабов при спуске. Использование расстояния *до* точки минимума модельной функции в качестве «естественного» масштаба, как это делается на обычных итерациях метода Ньютона, здесь невозможно, поскольку это расстояние становится *бесконечным*. И независимо от того, смотреть ли на эту проблему как на проблему нормировки направлений спуска или как на проблему выбора начальной оценки шага, универсального решения не видно. В то же время отсутствие разумной стратегии масштабирования приводит к тому, что поиск точки x_{k+1}, удовлетворяющей условию существенного убывания F, превращается в трудоемкий процесс. Поэтому каждая модификация метода Ньютона должна включать какой-то свой явный или неявный способ масштабирования шагов.

4.4.2. СТРАТЕГИИ ДЛЯ ЗНАКООПРЕДЕЛЕННОЙ МАТРИЦЫ ГЕССЕ

Общий принцип работы большинства модифицированных методов Ньютона состоит в следующем: на каждой итерации сначала строится некоторая «связанная» с G_k положительно определенная матрица $\bar{G}_k$, а затем p_k вычисляется как решение системы

$$\bar{G}_k p_k = -g_k. \tag{4.18}$$

Поскольку положительная определенность $\bar{G}_k$ гарантирована, такой вектор p_k наверняка будет направлением спуска. При этом процедуру построения $\bar{G}_k$ организуют так, чтобы $\bar{G}_k$ совпадала с исходной матрицей Гессе, если последняя сама является положительно определенной.

Формула (4.18) не применима, когда x_k — седловая точка. В самом деле, в ней градиент g_k равен нулю, и соответственно решение системы (4.18) тоже будет нулевым. Здесь в качестве p_k надо взять какое-нибудь *направление отрицательной кривизны* — вектор, удовлетворяющий неравенству $p_k^T G_k p_k < 0$. В седловой точке, где матрица G_k не является знакоопределенной, такие векторы существуют, и при движении вдоль любого из них функция F будет убывать. Идти по направлению отрицательной кривизны рекомендуется и в том случае, когда норма $\|g_k\|$ не равна нулю, но «очень мала».

Если числа обусловленности матриц $\bar{G}_k$ ограничены равномерно по k, то комбинация формулы (4.18) для расчета направления спуска и любого из критериев выбора шага, приведенных в разд. 4.3.2.1, дает глобально сходящуюся модификацию метода Ньютона.

Итак, в модифицированных ньютоновских методах направление спуска находится из решения линейной системы (4.18), причем матрица $\overline{G}_k$ должна совпадать с G_k, если G_k является положительно определенной. Последнее предполагает, что на каждой итерации каким-то образом выясняется, все ли собственные числа матрицы G_k положительны (а это, за исключением особых случаев, по виду самой G_k не определить). В обсуждаемых ниже модификациях метода Ньютона выяснение определенности G_k и построение $\overline{G}_k$ осуществляются параллельно в рамках одной процедуры. Это делается на основе некоторых *матричных разложений*, которые позволяют выявить знаки собственных чисел G_k и приспосабливаются для генерации $\overline{G}_k$.

4.4.2.1. Методы, основанные на спектральном разложении. Для построения связанной с G_k положительно определенной матрицы можно использовать собственные числа и векторы G_k (см. разд. 2.2.5.4). Основой служит спектральное разложение G_k вида

$$G_k = U\Lambda U^T = \sum_{i=1}^{n} \lambda_i u_i u_i^T, \tag{4.19}$$

где Λ—диагональная матрица, а U—ортонормальная матрица, составленные из собственных чисел и собственных векторов соответственно. Из этого разложения следует, что линейное преобразование G_k «делит» пространство своих аргументов на два взаимно ортогональных подпространства. Одно из них, $\mathcal{U}_+$, натянуто на собственные векторы $\{u_i\}$, которым отвечают «существенно положительные» собственные значения. Набранную из этих векторов матрицу обозначим через U_+. Второе, $\mathcal{U}_-$, натянуто на остальные собственные векторы, образующие матрицу U_-. Заметим, что любая положительная комбинация тех столбцов матрицы U_-, которым отвечают *отрицательные* собственные значения, будет направлением отрицательной кривизны.

При переходе от G_k к $\overline{G}_k$ желательно не затрагивать структуры преобразования G_k в отношении векторов из $\mathcal{U}_+$. Тогда для существенно положительно определенных G_k будет гарантировано равенство $\overline{G}_k = G_k$, т. е. модификация будет осуществляться только тогда, когда она действительно необходима. Данное условие обеспечено, если строить матрицу $\overline{G}_k$ в виде

$$\overline{G}_k = U\overline{\Lambda}U^T, \tag{4.20}$$

где $\overline{\Lambda}$—положительная диагональная матрица с $\overline{\lambda}_i$, равными λ_i при всех i, для которых λ_i существенно положительны. Что же касается тех диагональных элементов матрицы $\overline{\Lambda}$, которые отвечают столбцам U_-, то их можно выбирать разными способами

в зависимости от того, какие свойства требуются от $\bar{G}_k$. Заметим только, что подстановка вместо отрицательных λ_i положительных $\bar{\lambda}_i$ во всяком случае означает, что последствия применения преобразований G_k и $\bar{G}_k$ в соответствующей части $\mathcal{U}_-$ будут «противоположными».

Для иллюстрации двух способов замены отрицательных собственных чисел возьмем следующий пример с диагональной матрицей Гессе.

Пример 4.6. Пусть матрица вторых производных G_k и градиент g_k выглядят так:

$$G_k = \begin{pmatrix} 1 & 0 \\ 0 & -5 \end{pmatrix}, \quad g_k = \begin{pmatrix} 2 \\ 5 \end{pmatrix}.$$

У такой матрицы Гессе (очевидно) есть одно отрицательное и одно положительное собственные значения, а ее собственными векторами являются столбцы единичной матрицы. Обычное немонотоновское направление, рассчитываемое как решение системы (4.17), в данном случае равно $(-2, 1)^T$ и приводит в седловую точку модельной функции (4.15). Оно составляет острый угол с градиентом g_k, т. е. указывает в сторону возрастания функции.

Первый из предлагаемых способов построения $\bar{G}_k$ исходит из естественного стремления использовать в качестве $\bar{G}_k$ «ближайшую» к G_k положительно определенную матрицу. Этот способ состоит в том, что каждое собственное значение λ_i, меньшее, чем некоторое малое число $\delta > 0$, определяющее «порог существенной положительности», заменяется на δ. В примере 4.6 результатом будет

$$\bar{G}_k = \begin{pmatrix} 1 & 0 \\ 0 & \delta \end{pmatrix}.$$

При таком выборе $\bar{G}_k$ решением системы (4.18) является вектор $(-2, -5/\delta)^T$. Он явно тяготеет к подпространству $\mathcal{U}_-$ и имеет большую норму. И то и другое — характерные особенности рассматриваемого подхода. По сути дела, это своеобразное отражение того, что квадратичная модельная функция не ограничена снизу, и для ее «минимизации» надо сделать бесконечный шаг вдоль какого-нибудь направления отрицательной кривизны. Следует отметить, что большое значение нормы вектора p_k вычислительных трудностей не порождает: во-первых, p_k всегда можно перенормировать, а во-вторых, можно выставить подходящую верхнюю границу на длину шага в процедуре одномерного поиска.

Следующий метод определения $\bar{G}_k$ заключается в том, чтобы заменять собственные числа матрицы G_k их абсолютными значениями. При этом составляющей направления p_k в подпространстве $\mathcal{U}_-$ будет вектор, равный по длине и противоположный по

знаку к вектору, указывающему в точку максимума модельной функции в $\mathcal{U}_-$. В примере 4.6 рассматриваемый способ модификации дает матрицу

$$\overline{G}_k = \begin{pmatrix} 1 & 0 \\ 0 & 5 \end{pmatrix}.$$

Норма вектора p_k из (4.18) в данном случае равна норме обычного ньютоновского направления.

Матрицы $\overline{G}_k$, вычисляемые по правилу (4.20), положительно определены, и их числа обусловленности при ограниченных максимальных собственных значениях матриц G_k также будут ограниченными. Различие между незнакоопределенной матрицей G_k и отвечающей ей $\overline{G}_k$ при любом из двух представленных способов выбора $\overline{\lambda}_j$ характеризуется неравенством $|G_k - \overline{G}_k| \leqslant \leqslant 2|\lambda_{min}|$, где λ_{min} — минимальное собственное число G_k. Кривизна минимизируемой функции вдоль направления p_k, вычисленного по формулам (4.18) и (4.20), может быть и отрицательной, и положительной.

Расчет полной собственной системы G_k обычно требует от $2n^3$ до $4n^3$ арифметических операций. Таким образом, модификации G_k на основе спектрального разложения — довольно дорогая процедура. В настоящее время разработаны более эффективные способы построения $\overline{G}_k$ (как, например, метод, рассматриваемый в следующем разделе). В них подпространства $\mathcal{U}_+$ и $\mathcal{U}_-$ описываются приближенно, и это позволяет существенно сократить необходимый объем вычислений.

*4.4.2.2. Методы, основанные на факторизации Холесского. Любая симметричная положительно определенная матрица представима в виде произведения LDL^T, где L — единичная нижняя треугольная матрица, а D — положительная диагональная матрица (см. разд. 2.2.5.2). Это представление называют факторизацией (разложением) Холесского. Коль скоро столбцы матрицы L с номерами от 1-го по $(j-1)$-й известны, ее j-й столбец вычисляется по формулам

$$d_j = g_{jj} - \sum_{s=1}^{j-1} d_s l_{js}^2, \tag{4.21a}$$

$$l_{ij} = \frac{1}{d_j}\left(g_{ij} - \sum_{s=1}^{j-1} d_s l_{js} l_{is}\right). \tag{4.21b}$$

Аналогичные формулы существуют и для построчной организации вычислений.

По аналогии с рассмотренной выше модификацией G_k на основе спектрального разложения, казалось бы, естественно предложить и такую процедуру построения $\overline{G}_k$: формируется разло-

жение Холесского матрицы G_k и в качестве $\bar{G}_k$ берется произведение $L\bar{D}L^T$, где $\bar{d}_i = \max\{|d_i|, \delta\}$. Однако данный подход страдает двумя серьезными недостатками. Во-первых, для знакоопределенной матрицы G_k факторизации Холесского может просто не существовать (см. соответствующий пример в разд. 2.2.5.2). Во-вторых, даже если она существует, процедура расчета факторов, вообще говоря, будет численно неустойчивой, поскольку их элементы могут быть сколь угодно большими независимо от обусловленности G_k. Кроме того, матрица $\bar{G}_k$ в данном случае может очень сильно отличаться от G_k, даже если G_k всего лишь «слегка знакоопределена».

Пример 4.7. Возьмем матрицу

$$G_k = \begin{pmatrix} 1 & 1 & 2 \\ 1 & 1+10^{-20} & 3 \\ 2 & 3 & 1 \end{pmatrix}.$$

Ее собственные значения приблизительно равны 5.1131, —2.2019 и 0.0888 (слагаемое 10^{-20} введено, чтобы можно было построить разложение Холесского; при вычислении собственных чисел в рамках принятой точности оно не сказывается).

Точные факторы Холесского для матрицы из примера 4.7 выглядят так:

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 2 & 10^{20} & 1 \end{pmatrix}, \quad D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 10^{-20} & 0 \\ 0 & 0 & -(3+10^{20}) \end{pmatrix}.$$

Поэтому при рассматриваемом подходе к построению $\bar{G}_k$ норма $\|G_k - \bar{G}_k\|_E$ будет величиной порядка 10^{20}. В то же время второй, более радикальный из методов предыдущего раздела дает матрицу $\bar{G}_k$, для которой $\|G_k - \bar{G}_k\| \approx 4.4038$.

Итак, на основе обычной факторизации Холесского хорошего метода построения $\bar{G}_k$ не получить. Для этого нужно воспользоваться модифицированной факторизацией. Данный подход состоит в том, чтобы строить матрицы Холесского L и D, подчиняющиеся двум требованиям: все диагональные элементы D должны быть существенно положительными; модули всех элементов треугольного фактора L должны быть равномерно ограничены сверху. Точнее говоря, требуется, чтобы для всех $k = 1, 2, \ldots, n$ и некоторых заданных положительных δ и β выполнялись неравенства

$$d_k > \delta, \quad |r_{ik}| \leqslant \beta, \quad i > k, \tag{4.22}$$

где r_{ik} — введенные для удобства изложения вспомогательные величины, по определению равные $l_{ik}\sqrt{d_k}$ (выбор β обсуждается

шюке). При этом процедура расчета модифицированных L и D фактически представляет собой обычный алгоритм факторизации Холесского с попутным увеличением (по мере необходимости) диагонали исходной матрицы с целью добиться неравенств (4.22).

Рассмотрим j-й шаг предлагаемой процедуры. Пусть первые $j-1$ столбцов модифицированных факторов Холесского известны и для $k=1, \ldots, j-1$ условия (4.22) выполнены. Вычислим величину

$$\gamma_j = \left| \xi_j - \sum_{s=1}^{j-1} d_s l_{js}^2 \right|, \tag{4.23}$$

где через ξ_j обозначен диагональный элемент g_{jj} матрицы G_k. В качестве пробного значения $\bar{d}$ для j-го диагонального элемента D возьмем $\bar{d}=\max\{\gamma_j, \delta\}$, где δ—малое положительное число из (4.22). Если окажется, что при $d_j=\bar{d}$ значения r_{ij}, найденные в соответствии с формулой (4.21b), удовлетворяют (4.22), зафиксируем полученные d_j и j-й столбец L и перейдем к следующему шагу. Если же неравенства (4.22) при $d_j=\bar{d}$ нарушаются, т. е. какие-то из r_{ij} оказываются больше, чем β, окончательное значение d_j определим по формуле, получающейся из (4.23) заменой ξ_j на $g_{jj}+e_{jj}$, где число e_{jj} подбирается так, чтобы максимальная из величин r_{ij} совпала с β.

Матрицы L и D, полученные по окончании описанной процедуры, будут факторами Холесского для положительно определенной матрицы $\bar{G}_k$, связанной с G_k следующим образом:

$$\bar{G}_k = LDL^T = G_k + E.$$

Здесь E—неотрицательная диагональная матрица, j-й элемент которой равен e_{jj}. Таким образом, положительно определенная матрица $\bar{G}_k$ может отличаться от исходной матрицы Гессе G_k только диагональными элементами.

При заданной матрице G_k диагональная поправка E с очевидностью зависит от величины β. Желательно, чтобы эта величина была достаточно большой, так как заниженное значение β приведет к неоправданной модификации. Для положительно определенной матрицы G_k из (4.21a) следует, что при любом $i=1, \ldots, n$ и каждом $j\,(j<i)$ справедливо неравенство $l_{ij}^2 d_j \leqslant g_{ii}$. Значит, чтобы обеспечить равенство нулю поправки E, если матрица G_k существенно положительно определена, в качестве β надо взять величину, удовлетворяющую неравенству $\beta^2 \geqslant \gamma$, где γ — значение максимального из диагональных элементов G_k.

Когда у матрицы G_k есть отрицательные собственные значения, поправка E будет ненулевой независимо от выбора β. При этом для $n>1$ справедлива оценка

$$\|E(\beta)\|_\infty \leqslant \left(\frac{\xi}{\beta} + (n-1)\beta\right)^2 + 2(\gamma + (n-1)\beta^2) + \delta.$$

Здесь ξ — максимальный модуль недиагонального элемента G_k. Правая часть неравенства достигает минимума при $\beta^2=\xi\sqrt{n^2-1}$ и неограниченно возрастает с увеличением β. Это говорит о том, что в общем случае слишком большие значения β столь же нежелательны, как и слишком малые. Помимо того что завышенное значение β чревато потерей численной устойчивости процедуры построения $\bar{G}_k$, оно может еще и привести к неоправданно большому уклонению $\bar{G}_k$ от G_k; выбор бесконечного β, грубо говоря, означает, что диагональными элементами модифицированного фактора D будут модули диагональных элементов обычного (если обычное разложение возможно). Учитывая приведенные соображения, величину β вычисляют по формуле

$$\beta^2=\max\{\gamma,\ \xi/\sqrt{n^2-1},\ \varepsilon_M\}.$$

Здесь ε_M — машинная точность. Она вводится в формулу расчета β, чтобы обеспечить устойчивость вычислений, когда норма G_k очень мала.

Итак, что же такое представленная процедура модифицированной факторизации Холесского, если охарактеризовать ее двумя фразами? Это — численно устойчивый алгоритм, генерирующий положительно определенную матрицу, которая может отличаться от исходной только диагональными элементами. Это — оптимизированный алгоритм в том смысле, что параметр β подбирается в нем путем минимизации априорной оценки нормы поправки E при условии сохранения существенно положительно определенной матрицы неизменной. Следует отметить также, что реальная величина нормы E почти всегда оказывается значительно меньше априорной оценки.

Фактическое значение нормы E можно дополнительно уменьшить, если использовать симметричные перестановки столбцов и строк G_k. На очередном, j-м шаге факторизации в качестве j-й строки и j-го столбца надо брать ту из непропутых ранее пар, для которой величина γ_j в (4.23) максимальна. Такая стратегия приведет к разложению вида

$$P^TG_kP+E=LDL^T,$$

где P — некоторая перестановочная матрица. В методе с перестановками поправка E инвариантна относительно нумерации переменных.

Ниже приводится детальное описание всех операций, выполняемых по ходу построения модифицированного разложения Холесского с перестановками. Дана наиболее эффективная схема организации расчетов. Все фигурирующие в ней величины при реализации вычислений на ЭВМ могут размещаться в памяти, первоначально выделяемой для записи исходной матрицы G_k. При этом коэффициенты рассчитываемых факторов занимают места ее использованных элементов. В процессе вычисления j-го столбца матрицы L участвуют вспомогательные величины $c_{is}=l_{js}d_s$, $s=1, \ldots, j$; $i=j, \ldots, n$. Их

также естественно записать в массив, выделенный для G_k. Точнее говоря, позиции использованных элементов G_k сначала отводятся под запись чисел c_{ij}, а затем, по мере того как необходимость в каких-то c_{ij} отпадает, вместо них записываются соответствующие коэффициенты фактора L.

Алгоритм МС (*метод разложения модифицированного разложения Холесского*).

МС1. [Расчет порога для элементов.] Вычислить $\beta^2 \leftarrow \max\{\gamma, \xi/\nu, \varepsilon_M\}$, где $\nu = \max\{1, \sqrt{n^2-1}\}$, а γ и ξ суть максимальные значения модулей диагонального и недиагонального элементов G_k.

МС2. [Инициализация.] Присвоить индексу столбца j значение 1. Положить $c_{ii} \leftarrow g_{ii}$, $i = 1, \ldots, n$.

МС3. [Перестановка строк и столбцов.] Найти индекс q, такой, что $|c_{qq}| = \max_{j \le i \le n} |c_{ii}|$. Поменять местами все данные, отвечающие столбцам матрицы G_k с номерами q и j, а затем проделать то же самое с данными, отвечающими ее q-й и j-й строкам.

МС4. [Поиск максимальной по модулю величины $l_{jj}d_j$.] Вычислить $c_{ij} \leftarrow g_{ij} - \sum_{s=1}^{j-1} l_{js}c_{is}$ для $i = j+1, \ldots, n$ и найти $\theta_j = \max_{j+1 \le i \le n} |c_{ij}|$ (если $j = n$, взять $\theta_j = 0$).

МС5. [Расчет j-го диагонального элемента фактора D.] Вычислить $d_j \leftarrow \max\{\delta, |c_{jj}|, \theta_j^2/\beta^2\}$ и поправку $E_j \leftarrow d_j - c_{jj}$. Если $j = n$, вычисления прекратить.

МС6. [Расчет j-й строки L.] Вычислить $l_{is} = c_{js}/d_s$ для $s = 1, \ldots, j-1$. Для $i = j+1, \ldots, n$ пересчитать диагональные элементы c_{ii} по формуле $c_{ii} \leftarrow c_{ii} - l_{ij}c_{ij}$. Присвоить j значение $j+1$ и вернуться к шагу МС3.

Для построения модифицированного разложения Холесского нужно выполнить около $\frac{1}{6}n^3$ арифметических операций (примерно столько же, сколько требуется для построения обычного разложения для положительно определенной матрицы). Применительно к матрице из примера 4.7 факторы модифицированной факторизации Холесского и поправка E при $\beta^2 = 1.061$ (с точностью до четырех значащих цифр) таковы:

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 0.2652 & 1 & 0 \\ 0.5303 & 0.4295 & 1 \end{pmatrix}, \quad D = \begin{pmatrix} 3.771 & 0 & 0 \\ 0 & 5.750 & 0 \\ 0 & 0 & 1.121 \end{pmatrix} \quad \text{и} \quad E = \begin{pmatrix} 2.771 & 0 & 0 \\ 0 & 5.016 & 0 \\ 0 & 0 & 2.243 \end{pmatrix}$$

В данном случае $\|G_k - \bar{G}_k\|_F = \frac{1}{2}\|E\|_F \approx 6.154$.

Модифицированная факторизация Холесского помимо прочего позволяет определять и направления отрицательной кривизны. Пусть s — индекс, отвечающий минимальной из $n-j$ величин c_{ii}

на шаге МСЗ представленной выше процедуры. Если матрица G_k не является знакоопределенной, при каком-то j окажется, что величина c_{ss} отрицательна. Тогда вектор p, найденный как решение уравнения $L^T p = e_s$, будет направлением отрицательной

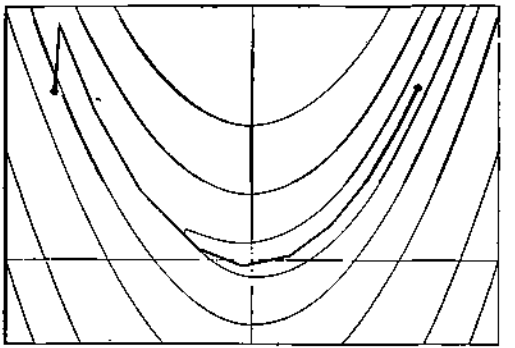

Рис. 4k. Траектория поиска минимума функции Розенброка модифицированным методом Ньютона.

кривизны. Так, для матрицы из примера 4.7 получим $s = 3$, и для соответствующего вектора p

$$\frac{p^T G_k p}{\|p\|_2^2} \approx -1.861.$$

На рис. 4k изображена траектория метода минимизации с вычислением вторых производных, основанного на применении модифицированного разложения Холесского, в задаче с функцией Розенброка (пример 4.2). Как видно из этого рисунка, после первой итерации метод четко отслеживает дно оврага, почти «идеально» аппроксимируя его кусочно-линейной кривой.

Замечания и избранная библиография к разделу 4.4

Известно много модификаций метода Ньютона. Большинство ранних численно неустойчивы (подробный разбор этих методов читатель найдет в работе Мюррея (1972a)). Подход с вычислением собственных векторов и значений был предложен Гриншадтом (1967). Метод с модифицированным разложением Холесского разработан Гиллом и Мюрреем (1974a).

Бытует мнение, что модифицированные ньютоновские алгоритмы работают хорошо лишь вблизи решения. Это породило массу «гибридных» схем, в которых на начальных итерациях, пока текущая точка далека от оптимальной, используется какой-нибудь относительно грубый алгоритм (обычно метод наискорейшего спуска), а алгоритм ньютоновского типа включается на завершающей стадии счета. Мы на основании своего вычислительного опыта беремся утверждать, что объективных мотивов для такой гибридизации нет. Хорошо продуманный и тщательно реализованный метод ньютоновского типа будет эффективен и вдали от искомого решения. В подтверждение этого тезиса можно еще раз сослаться на пример с функцией Розенброка. В то время как метод наискорейшего спуска с первых же итераций работает из рук вон плохо (см. рис. 4j), ньютоновский метод с модифицированной факторизацией Холесского уже со второй итерации считает вполне удовлетворительно (см. рис. 4k).

В основу модификации метода Ньютона можно положить *знаконеопределенное* симметричное разложение Банча и Парлетта (1971). Они показали, что для любой симметричной матрицы G_k существует представление

$$P^T G_k P = L B L^T,$$

где P — перестановочная, L — нижняя треугольная, а B — *блочно-диагональная* матрицы, причем блоки B имеют размерности, не превышающие двух. Возможность симметричных перестановок отражена в этой формуле потому, что такие перестановки необходимы для численной устойчивости процедуры расчета рассматриваемого разложения.

Знаконеопределенная симметричная факторизация обладает одним существенным достоинством: она позволяет выяснить *инерцию* матрицы G_k (инерция — это тройка целых чисел, задающих количества положительных, отрицательных и нулевых собственных значений). Дело в том, что инерции G_k и фактора B ее разложения совпадают (хотя сами наборы собственных значений у этих матриц, вообще говоря, различны). При этом двумерные блоки фактора B строятся так, что у каждого из них всегда есть одно отрицательное и одно положительное собственное значения. Следовательно, число положительных собственных значений G_k равно сумме числа положительных одномерных блоков матрицы B и числа ее двумерных блоков.

Идея использования знаконеопределенной симметричной факторизации для построения $\overline{G}_k$ принадлежит Мору и Соренсену (1979). Состоит она в том, чтобы, получив спектральное разложение для B ($B = U\Lambda U^T$) и составив матрицу $\overline{B}$ по правилу разд. 4.4.2.1, т. е. положив $\overline{B} = U\overline{\Lambda}U^T$, где $\overline{\lambda}_i = \max\{|\lambda_i|, \delta\}$,

вычислять $\bar{G}_k$ по формуле

$$P^T\bar{G}_kP=L\bar{U}\bar{U}^TL^T.$$

Ясно, что существенно положительно определенная матрица G_k останется неизменной. Расчет факторов знакоопределенного симметричного разложения можно организовать так, что он потребует около n^3 арифметических операций и $O(n^2)$ операций сравнения. Схожая факторизация, требующая $O(n^2)$ сравнений, предложена Флетчером (1976), а также Бунчем и Кауфманом (1977).

Модифицированный метод Ньютона можно строить на базе любой численно устойчивой процедуры факторизации знакоопределенной симметричной матрицы. Мы рассмотрели три из них. Другие читатель найдет, например, у Осена (1971), Кэнисла и Дэкса (1979).

Когда матрица G_k не является знакоопределенной и соответственно существуют направления отрицательной кривизны, учитывать их при выборе p_k можно по-разному. Ранее была сформулирована одна из простейших стратегий: брать в качестве p_k решение уравнения (4.18) во всех случаях, когда норма градиента достаточно велика по отношению к модулю значения целевой функции, а иначе — использовать в роли p_k какое-нибудь из направлений отрицательной кривизны. При этом направления отрицательной кривизны будут вычисляться только вблизи седловой точки. Сформулированное правило можно расценивать как конкретную реализацию более общего правила, заключающегося в том, чтобы определять p_k по формуле

$$p_k=\varphi_1p_1+\varphi_2p_2,$$

где p_1 — решение уравнения (4.18), p_2 — направление отрицательной кривизны, а φ_1 и φ_2 — неотрицательные числа. Подразумевается, что в случае положительной определенности матрицы G_k вес φ_2 равен нулю, а φ_1 — единице (тем самым из формулы убирается несуществующий вектор p_2). Если в упомянутой выше стратегии нулевой вес φ_1 является исключением, то Флетчер и Фриман (1977), например, считают, что брать φ_1 нулевым надо как можно чаще. Грэхем (1976) предлагает определять веса по норме поправки для матрицы Гессе G_k при переходе от G_k к $\bar{G}_k$. Более общие схемы, в которых φ_1 и φ_2 вычисляются как функции длины шага α_k, можно найти в работах Мак-Кормика (1977), Море и Соренсена (1979) и Гольдфарба (1980).

В замечаниях к разд. 4.3 было отмечено, что методы безусловной минимизации гладких функций распадаются на два обширных класса. Имеются в виду «методы с регулировкой шага» и «методы доверительной окрестности». Хотя в данной книге основное внимание уделяется первым, пришла пора поподробнее поговорить и о вторых. Исходной посылкой для построения методов доверительной окрестности служит суждение о том, что на точку минимума модельной квадратичной функции разумно ориентироваться лишь при усло-

вий, что она ищется в области, где эта функция хорошо описывает поведение F. Размеры такой области предлагается характеризовать ограничением на величину нормы вычисляемого вектора направления поиска.

Формализованное правило выбора очередной точки x_{k+1}, воплощающее высказанные идеи, звучит так: в качестве x_{k+1} надо брать сумму x_k+p_k, где p_k — решение при некотором Δ вспомогательной задачи вида

$$\text{найти } \min_{p\in R^n}\left(g_k^T p+\frac{1}{2}p^T G_k p\right) \tag{4.24}$$

$$\text{при ограничении } \|p\|_2\leqslant\Delta.$$

Как искать такие p_k, подсказывает следующее соображение: если λ — число, при котором матрица $G_k+\lambda I$ положительно определена, то решение p системы уравнений

$$(G_k+\lambda I)p=-g_k \tag{4.25}$$

будет решением задачи (4.24) с $\Delta\geqslant\|p\|_2$, когда $\lambda=0$, и с $\Delta=\|p\|_2$, когда $\lambda>0$. Соответственно p_k можно вычислять как решение (4.25). Из сказанного ясно, что при положительно определенной матрице G_k и достаточно большом Δ решение задачи (4.24) будет обычным ньютоновским направлением (т. е. решением системы (4.25)) с $\lambda=0$, а если при каком-то Δ оно окажется отличным от ньютоновского направления, то это является признаком существенности ограничения на норму вектора p и будет означать, что $\|p\|_2=\Delta$. В методе доверительной окрестности направление поиска обычно определяется просмотром решений задачи (4.24) при нескольких значениях параметра Δ. В результате выбирается вектор p, для которого величина $F(x_k+p)$ «существенно меньше», чем $F(x_k)$. Такой вектор всегда найдется, так как при достаточно малом Δ квадратичная аппроксимация хорошо описывает поведение F. Нетрудно показать, что при $\Delta\to 0$ вектор p, убывая по норме до нуля, стремится по направлению к антиградиенту.

Идея построения правила расчета направления поиска на основе понятия доверительной окрестности принадлежит Левенберту (1944) и Маркардту (1963). Они сформулировали ее для задач о наименьших квадратах (см. разд. 4.7.3). Применение данного подхода к функциям общего вида обсуждалось Гольдфельдом, Квандтом и Тротгером (1966).

В некоторых методах доверительной окрестности k-я итерация начинается с решения уравнения (4.25) при $\lambda=0$. Если по ходу вычислений выясняется, что матрица G_k не является знакоопределенной или норма $\|p\|$ найденного решения оказывается больше текущего порогового значения Δ, то включается процедура поиска приближенного решения нелинейного уравнения вида

$$\varphi(\lambda)\equiv\|(G_k+\lambda I)^{-1}g_k\|_2=\Delta. \tag{4.26}$$

Квадрат функции, стоящий в его левой части, можно представить так:

$$\varphi^2(\lambda) = \sum_{i=1}^{n} \left(\frac{\mu_i^T g_k}{\lambda + \lambda_i} \right)^2.$$

Здесь $\{\mu_i\}$, $\{\lambda_i\}$ — собственные векторы и числа матрицы G_k. Таким образом, $\varphi^2(\lambda)$ является рациональной функцией с полюсами в точках λ, совпадающих с собственными значениями матрицы $-G_k$.

Хэбден (1973) предложил применять для поиска корня уравнения (4.26) регулировавший метод с рациональной интерполяцией (см. разд. 4.1.1.4). Его процедура не требует построения спектрального разложения G_k и использует при вычислении значений $\varphi(\lambda)$ и $d\varphi/d\lambda$ разложение Холесского для матрицы $G_k+\lambda I$. Если при каком-то λ расчет факторов этого разложения выявляет знаконеопределенность $G_k+\lambda I$, то находится число μ, такое, что $G_k+\lambda I+\mu I$ будет положительно определенной матрицей, и в качестве очередной оценки решения (4.26) берется сумма $\lambda+\mu$. Поиск приближенного корня уравнения (4.26) методом Хэбдена обычно требует двух- или трехкратного построения разложения Холесского. Когда матрица G_k не является знаконеопределенной, независимо от установленной точности вычисления λ понадобится не менее двух разложений.

После того как удовлетворительное решение уравнения (4.26) и соответствующий вектор p найдены, выполняется проверка пригодности точки x_k+p на роль очередного приближения x_{k+1}. Она подойдет, если значение $F(x_k+p)$ окажется существенно меньшим, чем F_k (эталоном служит изменение, предсказываемое по квадратичной аппроксимации). В противном случай найденный вектор p отбрасывается, порог Δ уменьшается (например, Δ можно пересчитывать на основе кубической интерполяции с использованием значений функции F и ее производных в точках x_k и x_k+p) и снова вызывается процедура решения уравнения (4.26). Когда уменьшение F признано приемлемым, можно попытаться увеличить Δ. Детали различных схем пересчета Δ читатель найдет в работах Флетчера (1971a, 1980), Гэя (1979a), Хэбдена (1973), Море (1977) и Соренсена (1980a). Подробное обсуждение процедуры расчета ρ при фиксированном Δ приведено в работе Гэя (1979b).

Важно отметить, что у методов с регулировкой шага и у методов доверительной окрестности есть много общих черт. (1) И те и другие конструируются так, чтобы при приближении к точке min муча хорошей функции они становились эквивалентными классическому методу Ньютона. (2) И в тех и в других очередное приближение определяется некоторой скалярной величиной, значение которой подбирается из условия согласования истинного изменения функции F и его оценки, получающейся по квадратичной модели. В методах с регулировкой шага этим скаляром является длина шага α_k, а в методах доверительной окрестности — порог Δ. Правда, принципы выбора α_k и Δ различны. Ве-

личина α_k ищется как приближение «идеального» шага (шага в точку минимума вдоль направления p_k), а Δ подбирается в соответствии с «качеством» предыдущих значений F без ориентации на какой-то «идеал». (3) Если матрица G_k не является знакоопределенной и норма $\|g_k\|$ мала или равна нулю, методы обоих типов должны сделать шаг вдоль направления отрицательной кривизны, вычисляемого по разложению самой G_k или ее модифицированной версии. (4) Когда матрица G_k становится знакоопределенной при «большом» градиенте (что существенно усложняет задачу предварительной оценки нормы $\|x_{k+1}-x_k\|$), в методах обоих типов расчет x_{k+1} ориентируется на ее положительно определенную «составляющую». В методах с регулировкой шага это проявляется явно, в соответствующей модификации G_k, а в методах доверительной окрестности — неявно, через связь $\|x_{k+1}-x_k\|$ с размерами предыдущих шагов, выполненных при положительно определенных G_k.

Хорошая реализация метода доверительной окрестности всегда включает какие-то элементы, присущие методу с регулировкой шага, и наоборот. Например, в алгоритме с регулировкой шага обязательно должно быть предусмотрено ограничение на длину шага вида $\|x_{k+1}-x_k\| \leqslant \Delta$ (см. разд. 4.3.2.1). В свою очередь хороший метод доверительной окрестности будет использовать для настройки Δ какую-нибудь регулировочную процедуру с полиномиальной интерполяцией. Сходство между методами обоих типов наиболее отчетливо проявляется в следующей ситуации. Предположим, что они запускаются в точке x_k, где матрица G_k положительно определена. Пусть далее ньютоновское направление p в x_k таково, что $\|p\|_2 \leqslant \Delta$ и $F(x_k+p) > F_k$. Тогда алгоритм с регулировкой шага будет искать с помощью полиномиальной интерполяции длину шага $\hat{\alpha}$ ($\hat{\alpha} < 1$) вдоль p, обеспечивающую существенное убывание функции F. Аналогичная величина $\hat{\alpha}$ будет вычислена и в методе доверительной окрестности, только здесь она сыграет роль множителя при пересчете порога Δ. После этого пути алгоритмов разойдутся. В методе с регулировкой шага очередным приближением станет точка $x_k+\hat{\alpha}p$, лежащая в прежнем направлении, а в методе доверительной окрестности шаг будет сделан в новом направлении, найденном из решения нелинейного уравнения (4.26) с меньшим значением Δ.

Наряду со сходными чертами между методами с регулировкой шага и методами доверительной окрестности имеются и значительные различия. Наиболее важные связаны с характером использования информации о вторых производных. В методах с регулировкой шага матрицу G_k стараются модифицировать так, чтобы изменения не затрагивали подпространства, натянутого на ее собственные векторы с положительными собственными значениями. Это гарантирует сохранность существенно положительно определенных G_k. Если же

замена G_k осуществляется в методе доверительной окрестности, то она отражается на всех векторах, так как результатом замены G_k на $G_k+\lambda I$ будет сдвиг на λ всех собственных значений. При этом замены возможны независимо от определенности G_k, т. е., даже если G_k существенно положительно определена, направление поиска в методе доверительной окрестности может отличаться от ньютоновского.

В подзадаче (4.24) ограничение на длину вектора p ставится через евклидову норму. Выбор именно этой нормы объясняется относительной простотой поиска приближенного решения (4.24). Если же посчитать целесообразным тратить на определение пробных p больше усилий, то можно пользоваться и другими нормами. Флетчер (1972а), например, предлагал подчинять выбор p простым ограничениям вида $l_i \leqslant p_i \leqslant u_i$. В этом случае для построения p надо решить задачу поиска минимума квадратичной формы на параллелепипеде.

4.5. МЕТОДЫ ПЕРВОГО ПОРЯДКА

4.5.1. НЬЮТОНОВСКИЕ МЕТОДЫ С КОНЕЧНО-РАЗНОСТНОЙ АППРОКСИМАЦИЕЙ

Для того чтобы получить быстросходящийся алгоритм безусловной минимизации, не обязательно пользоваться точными значениями вторых производных. Метод, практически не уступающий в скорости сходимости методу Ньютона, можно построить на основе *аппроксимации* матрицы Гессе по конечным разностям градиентов. Во многих случаях такой метод окажется самым подходящим. Имеются в виду ситуации, когда вычисление аналитических значений вторых производных минимизируемой функции F затруднено или просто невозможно.

За оценку i-го столбца матрицы G_k обычно принимают его правую конечно-разностную аппроксимацию

$$y_i = \frac{1}{h}(g(x_k+he_i)-g(x_k)).$$

Здесь h — число, именуемое *конечно-разностным интервалом*, а e_i есть i-й единичный вектор, выполняющий функцию *конечно-разностного направления*. В данном разделе мы ради простоты изложения будем полагать, что все конечные разности вычисляются на одном и том же интервале h. На самом же деле приходится строить набор интервалов $\{h_i\}$, $i=1, \ldots, n$, и технически делается это так, как описано в разд. 8.6.

Матрица Y, составленная из столбцов y_i, вообще говоря, несимметрична. Поэтому для аппроксимации матрицы Гессе G_k берут полусумму $\hat{G}_k = \frac{1}{2}(Y+Y^T)$, которая симметрична по построению. Метод, получающийся из обычного ньютоновского заменой в уравнении (4.17) матрицы G_k на $\hat{G}_k$, будем называть

дискретным методом Ньютона. Подобно своему классическому прототипу, он порождает целый класс модифицированных методов; понятно, что техника модификаций, разработанная для обобщения метода Ньютона на случаи с анакопределенными G_k, применима и к его конечно-разностному аналогу.

Коль скоро предполагается использовать конечно-разностную аппроксимацию, то естественно встает вопрос о выборе величины интервала h. Здесь обнаруживаются определенные противоречия между теорией и практикой оптимизации. Если бы расчеты на машинах проводились с абсолютной точностью, для достижения квадратичной сходимости дискретного метода Ньютона надо было бы по мере убывания $\|g\|$ устремлять h к нулю. В действительности же очень малые величины h недопустимы, так как при таких h вычисляемые значения элементов матрицы $\tilde{G}_k$ из-за больших ошибок компенсации (см. разд. 2.1.5) не будут иметь никакого смысла. Значит, интервал h, с одной стороны, должен быть достаточно мал, чтобы обеспечивать удовлетворительную сходимость, а с другой — достаточно велик, чтобы гарантировать приемлемую точность вычисления конечно-разностной аппроксимации. Нужен компромисс, и, к счастью, найти его не так уж трудно. Дело в том, что для большинства практических задач эффективность метода не очень чувстви-

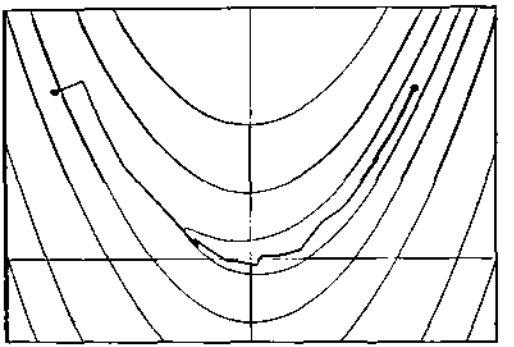

Рис. 41. Траектория поиска минимума функции Розенброка дискретным методом Ньютона.

тельна по отношению к выбору h (чего нельзя сказать о выборе интервала конечно-разностной аппроксимации градиента в методах без вычисления производных, рассматриваемых в разд. 4.6). Подробнее вопрос о назначении h будет разбираться в разд. 8.6.

Дискретный метод Ньютона обычно сходится «квадратично» вплоть до того момента, когда достигается предельная точность решения (т. е. получено приближение, которое из-за погрешностей машинной арифметики нельзя было бы улучшить и в том случае, если бы была известна точная матрица Гессе). Детальное обсуждение пределов точности читатель найдет в разд. 8.2.2.

Итерации дискретного ньютоновского метода в задаче минимизации функции Розенброка (пример 4.2) изображены на рис. 4l. Бросается в глаза практическое отсутствие разницы между этой картинкой и рис. 4k, изображающим итерации по ньютоновскому методу с вычислением значений вторых производных по формулам.

Итак, если говорить о достоинствах дискретных ньютоновских методов, то они те же, что и у обычных методов ньютоновского типа, – высокая скорость сходимости и способность «ощущать» приближение седловой точки и уходить от нее. К недостаткам же следует отнести необходимость вычисления на каждой итерации n значений градиента, которые требуются для построения n столбцов аппроксимации матрицы Гессе, когда в качестве конечно-разностных направлений используются столбцы единичной матрицы. Поэтому при $n \geqslant 10$ или около того дискретные ньютоновские методы нередко начинают проигрывать другим методам первого порядка, представленным ниже. Однако они могут сохранить первенство и при больших n, если матрица Гессе разрежена и известная структура расположения ее ненулей позволяет экономно организовать процедуру построения конечно-разностной аппроксимации (см. разд. 4.8.1).

4.5.2. КВАЗИНЬЮТОНОВСКИЕ МЕТОДЫ

4.5.2.1. Теория. Залогом эффективности методов ньютоновского типа, рассмотренных в разд. 4.4 и 4.5.1, является учет информации о кривизне минимизируемой функции, содержащейся в матрице Гессе и позволяющей строить локально точные квадратичные модели F. *Квазиньютоновские* методы тоже ориентированы на квадратичные модели, но данные относительно кривизны F накапливаются в них на основе наблюдения за изменением градиента g во время итераций спуска. Последнее принципиально отличает эти методы от ньютоновских, в которых сведения о свойствах F всегда относятся к одной точке. *Теория квазиньютоновских методов опирается на возможность аппроксимации кривизны нелинейной функции без явного формирования ее матрицы Гессе.*

Обозначим через s_k шаг из точки x_k и рассмотрим разложение градиента в ряд Тейлора в окрестности x_k «по степеням» s_k

$$g(x_k+s_k)=g_k+G_ks_k+\ldots$$

В силу этого разложения линейная оценка кривизны F вдоль s_k, т. е. произведение $s_k^TG_ks_k$, задается формулой вида

$$s_k^TG_ks_k \approx (g(x_k+s_k)-g_k)^Ts_k. \qquad (4.27)$$

Равенство станет тогда тем, если отнести его к квадратичной функции из (4.15). Формула (4.27) лежит в основе всех квазиньютоновских методов первого порядка.

К началу очередной k-й итерации квазиньютоновского поиска известна некоторая аппроксимация B_k матрицы Гессе минимизируемой функции. Матрица B_k служит средством отображения информации о кривизне, накопленной на предыдущих итерациях. Используя ее в качестве матрицы Гессе моделиной квадратичной формы, очередное направление p_k квазиньютоновского поиска определяют как решение аналогичной (4.17) системы уравнений:

$$B_k p_k = -g_k. \tag{4.28}$$

Исходное приближение B_0, если нет какой-либо дополнительной информации, обычно полагают равным единичной матрице. При таком выборе B_0 первая итерация квазиньютоновского поиска будет эквивалентна шагу наискорейшего спуска.

После того как определится новая точка x_{k+1}, приближение B_k матрицы Гессе обновляется с учетом вновь полученной информации о кривизне, т. е. совершается переход от матрицы B_k к матрице B_{k+1}. Он задается *формулой пересчета* типа

$$B_{k+1} = B_k + U_k, \tag{4.29}$$

где U_k — некоторая *поправочная* матрица. Обозначим через s_k вектор, равный изменению аргумента x на k-й итерации ($s_k = x_{k+1} - x_k = \alpha_k p_k$), а через y_k соответствующее приращение градиента ($y_k = g_{k+1} - g_k$). Тогда основное свойство всех квазиньютоновских правил пересчета (4.29) выразится равенством

$$B_{k+1} s_k = y_k. \tag{4.30}$$

В силу (4.27) оно означает, что B_{k+1} будет правильно отражать кривизну F вдоль s_k. Это равенство принято называть *квазиньютоновским условием*.

Один шаг даст информацию о кривизне F вдоль одного единственного направления, поэтому мы вправе надеяться, что в формуле (4.29) удастся обойтись поправкой U_k малого ранга. Так оно и есть: квазиньютоновскому условию можно удовлетворить и не одним способом, даже если искать U_k среди матриц ранга один, т. е. подбирать B_{k+1} в форме

$$B_{k+1} = B_k + uv^T, \tag{4.31}$$

где u и v — некоторые векторы. В данном случае условие (4.30) принимает вид равенства

$$B_{k+1} s_k = (B_k + uv^T) s_k = y_k,$$

откуда следует, что

$$u(v^T s_k) = y_k - B_k s_k.$$

Таким образом, независимо от выбора v вектор u должен быть параллелен разности $y_k - B_k s_k$, причем мы полагаем, что она не равна нулю (т. е. сама матрица B_k квазиньютоновскому условию не удовлетворяет). Когда некоторый не ортогональный s_k вектор v задан, в качестве u в соответствии с последним равенством надо взять $(1/v^T s_k)(y_k - B_k s_k)$, и при этом

$$B_{k+1} = B_k + \frac{1}{v^T s_k}(y_k - B_k s_k) v^T. \tag{4.32}$$

Даже поправки ранга один определяются квазиньютоновским условием с точностью до вектора v, т. е. неоднозначно. Тем более неоднозначно определены поправки высших рангов. Как они могут выглядеть, подсказывает следующее соображение. Для любого ортогонального s_k вектора w и любого z произведение матрицы ранга один zw^T на s_k равно нулю. Значит, выполнив условие (4.30) при некоторой B_{k+1}, мы не нарушим его, если добавим к B_{k+1} любое число матриц типа zw^T (хотя сама матрица B_{k+1}, конечно, изменится). Итак, в выборе U_k на основе квазиньютоновского условия (4.30) есть большая свобода. Эту свободу используют для того, чтобы обеспечить матрице B_{k+1} различные дополнительные свойства.

Симметрия. Поскольку сама матрица Гессе симметрична, представляется естественным позаботиться о том, чтобы такими были и ее квазиньютоновские приближения. Для этого надо строить формулы пересчета, которые гарантировали бы *сохранение симметрии*, т. е. обеспечивали бы симметричность B_{k+1} при симметричной B_k. В случае с поправками ранга один это требование определяет U_k однозначно. Чтобы пересчет по формуле (4.31) сохранял симметрию, вектор v должен быть коллинеарен u. При этом формула (4.32) преобразуется к виду

$$B_{k+1} = B_k + \frac{1}{(y_k - B_k s_k)^T s_k}(y_k - B_k s_k)(y_k - B_k s_k)^T,$$

где $y_k - B_k s_k$ и $(y_k - B_k s_k)^T s_k$ предполагаются ненулевыми. Эту формулу пересчета называют *симметричной формулой ранга один*.

Поскольку среди формул единичного ранга симметрию сохраняет только одна, для того, чтобы найти другие симметричные формулы, надо обратиться к поправкам ранга два. Конструировать их можно, например, так: имея симметричную матрицу B_k, положим $B^{(0)} = B_k$ и вычислим матрицу $B^{(1)}$ по формуле (4.32), т. е. возьмем

$$B^{(1)} = B^{(0)} + \frac{1}{v^T s_k}(y_k - B_k s_k) v^T, \quad v^T s_k \neq 0.$$

Полученная матрица удовлетворяет квазиньютоновскому условию, но, вообще говоря, несимметрична. Симметризуем ее, т. е. построим матрицу $B^{(2)}$ вида

$$B^{(2)} = \frac{1}{2}(B^{(1)} + B^{(1)T}).$$

Теперь мы получили симметричную матрицу, но она не удовлетворяет квазиньютоновскому условию. Снова воспользуемся пересчетом (4.32) и т. д. В результате определится последовательность матриц $\{B^{(j)}\}$, для которых при $j=0, 1, \ldots$

$$B^{(2j+1)} = B^{(2j)} + \frac{1}{v^T s_k}(y_k - B^{(2j)}s_k)\,v^T,$$

$$B^{(2j+2)} = \frac{1}{2}(B^{(2j+1)} + B^{(2j+1)T}).$$

Эта последовательность имеет предел B_{k+1}, вычисляемый по формуле

$$B_{k+1} = B_k + \frac{1}{v^T s_k}((y_k - B_k s_k)\,v^T + v\,(y_k - B_k s_k)^T) - \frac{(y_k - B_k s_k)^T s_k}{(v^T s_k)^2}\,vv^T. \tag{4.33}$$

Поправка к B_k в этой формуле, вообще говоря, имеет ранг два и определена для любого вектора v, не ортогонального s_k. В классе правил пересчета, получающихся из (4.33) при различных v, содержится, в частности, и симметричная формула ранга один. Она отвечает выбору в (4.33) вектора $v = y_k - B_k s_k$. Если же взять $v = s_k$, то получится хорошо известная симметричная формула Пауэлла — Бройдена (PSB-формула). Еще одна распространенная формула пересчета квазиньютоновских матриц получается из (4.33), если v положить равным y_k. Она называется формулой Дэвидона — Флетчера — Пауэлла (DFP) и выглядит следующим образом:

$$B_{k+1} = B_k - \frac{1}{s_k^T B_k s_k} B_k s_k s_k^T B_k + \frac{1}{y_k^T s_k} y_k y_k^T + (s_k^T B_k s_k)\,w_k w_k^T, \tag{4.34}$$

где

$$w_k = \frac{1}{y_k^T s_k}\,y_k - \frac{1}{s_k^T B_k s_k} B_k s_k.$$

Непосредственной подстановкой легко проверить, что вектор w_k ортогонален s_k. Значит, добавив к B_{k+1} матрицу ранга один, кратную $w_k w_k^T$, мы не нарушим ни квазиньютоновского условия, ни симметрии. Это соображение приводит к однопараметрическому семейству формул пересчета вида

$$B_{k+1}^{\varphi} = B_k^{\varphi} - \frac{1}{s_k^T B_k s_k} B_k^{\varphi} s_k s_k^T B_k^{\varphi} + \frac{1}{y_k^T s_k} y_k y_k^T + \varphi_k\,(s_k^T B_k^{\varphi} s_k)\,w_k w_k^T, \tag{4.35}$$

где скалярная величина φ_k может зависеть от y_k и $B_k^{\varphi} s_k$. Коль скоро φ_k можно выбирать, естественно возникает вопрос о том, как делать это наилучшим образом. Данному вопросу было посвящено немало исследований. Однако в большинстве своем они носили чисто теоретический характер, и приводить их обзора мы не будем. Ограничимся выводом: в настоящее время принята точка зрения, что самой эффективной в классе формул (4.35)

является простейшая из них, отвечающая выбору $\varphi_k \equiv 0$. Эта формула пересчета, известная под названием формулы Бройдена—Флетчера—Гольдфарба—Шанно (BFGS-формулы), выглядит так:

$$B_{k+1} = B_k - \frac{1}{s_k^T B_k s_k} B_k s_k s_k^T B_k + \frac{1}{y_k^T s_k} y_k y_k^T. \qquad (4.36)$$

Все представленные выше формулы пересчета выведены независимо от того, как определяется вектор s_k. Если же s_k в действительности получается в результате шага α_k вдоль направления p_k, найденного из решения уравнения (4.28), то формулы можно упростить, используя равенство $B_k s_k = -\alpha_k g_k$. Например, BFGS-формула будет выглядеть так:

$$B_{k+1} = B_k + \frac{1}{g_k^T p_k} g_k g_k^T + \frac{1}{\alpha_k g_k^T p_k} y_k y_k^T. \qquad (4.37)$$

Когда шаги α_k определяются в результате точной одномерной минимизации, между различными формулами из однопараметрического семейства (4.35) устанавливается очень специфическая связь. Пусть $F(x)$—дважды непрерывно дифференцируемая функция и зафиксированы некоторые x_0 и B. Пусть далее через $\{x_k\}$, $\{B_k\}$, $\{p_k\}$ и $\{\alpha_k\}$ обозначены последовательности, генерируемые квазиньютоновским поиском с BFGS-формулой, а через $\{x_k^\varphi\}$, $\{B_k^\varphi\}$, $\{p_k^\varphi\}$ и $\{\alpha_k^\varphi\}$—последовательности, соответствующие какой-нибудь другой формуле пересчета из семейства (4.35). Тогда оказывается, что если α_k и α_k^φ выбираются как точки одномерных локальных минимумов, ближайших к $\alpha = 0$, то, начиная с $k = 0$ и до тех пор, пока матрицы B_k, B_k^φ определены, будут выполнены соотношения

$$x_k^\varphi = x_k, \quad B_k = B_k^\varphi + \left(\frac{\varphi_{k-1}}{g_{k-1}^T p_{k-1}^\varphi}\right) g_k g_k^T. \qquad (4.38)$$

Таким образом, при оговоренных условиях *все методы семейства* (4.35) *генерируют одинаковые последовательности точек*. Второе равенство в (4.38), кроме того, показывает, что элементы квазиньютоновской аппроксимации матрицы Гессе вовсе не обязательно будут приближаться к соответствующим элементам оригинала. Как видно из (4.38), матрицы B_k и B_k^φ, вообще говоря, значительно отличаются друг от друга на всех итерациях, и если для одной из них отклонение от матрицы Гессе окажется «малым», то для другой оно наверняка будет «большим».

Теоретическая эквивалентность различных формул семейства (4.35) при точной одномерной минимизации еще не означает, что они эквивалентны с практической точки зрения. Хотя количества итераций поиска минимума одной и той же функции с одного и того же начального приближения будут одинаковыми для всех формул, количества вычислений функции на этапах

одномерного поиска *скорее всего окажутся существенно различными*. Это связано с тем, что величина первого пробного шага в процедуре поиска α_k для разных B_k^2 будет разной. В одном случае она окажется хорошей оценкой искомого α_k, в другом — плохой. Соответственно и вся процедура вычисления α_k в одном случае проработает быстрее, а в другом — медленнее.

Положительная определенность. Конечная цель квазиньютоновского поиска состоит в том, чтобы найти точку x^*, в которой реализуется локальный минимум функции F. В большинстве случаев этот минимум будет сильным, а матрица Гессе в x^* — положительно определенной. Соответственно хотелось бы, чтобы матрицы $\{B_k\}$ тоже были положительно определенными. Кроме того, что более существенно, при положительно определенной B_k локальная квадратичная модель имеет единственный минимум, и направление p_k, найденное из (4.28), оказывается направлением спуска. Короче говоря, принято требовать, чтобы формула пересчета квазиньютоновских матриц обеспечивала *сохранение положительной определенности*, т. е. гарантировала бы наличие этого свойства у B_{k+1}, если им обладает B_k.

Анализ условий сохранения положительной определенности для разных формул пересчета проводится по-разному; мы выведем такие условия для BFGS-формулы. Когда матрица B_k положительно определена, существует невырожденная матрица R, такая, что $B_k = R^T R$. Используя это разложение, (4.36) можно переписать в виде

$$B_{k+1} = R^T W R, \tag{4.39}$$

где W вычисляется по формуле

$$W = I - \frac{1}{\bar{s}^T \bar{s}} \bar{s}\bar{s}^T + \frac{1}{\bar{y}^T \bar{s}} \bar{y}\bar{y}^T, \tag{4.40}$$

в которой $\bar{s} = Rs_k$ и $\bar{y} = (R^T)^{-1} y_k$. Из (4.39) следует, что матрица B_{k+1} будет положительно определенной в том и только том случае, если положительно определена W. При этом, так как W отличается от единичной матрицы лишь добавкой ранга два, $n-2$ ее собственных значений (для $n \geqslant 3$) совпадают с единицей. Таким образом, все зависит от двух оставшихся собственных значений. Мы обозначим их через λ_1 и λ_2. Можно показать, что отвечающие им собственные векторы должны быть линейными комбинациями от $\bar{s}$ и $\bar{y}$. Имея это в виду, непосредственной подстановкой легко установить, что числа λ_1 и λ_2 связаны соотношениями вида

$$\lambda_1 + \lambda_2 = \frac{\bar{y}^T \bar{s} + \bar{y}^T \bar{y}}{\bar{y}^T \bar{s}}, \quad \lambda_1 \lambda_2 = \frac{\bar{y}^T \bar{s}}{\bar{s}^T \bar{s}},$$

Поскольку величина $\bar{s}^T \bar{s}$ положительна, из полученных равенств вытекает, что λ_1 и λ_2 будут больше нуля тогда и только

тогда, когда положительно скалярное произведение $y_k^T s_k$, а оно по построению равно $y_k^T s_k$. Значит, для того чтобы BFGS-формула сохранила положительную определенность, необходимо и достаточно выполнения неравенства

$$y_k^T s_k > 0. \tag{4.41}$$

Неравенство (4.41) будет выполнено, если выбирать шаги α_k на основе «достаточно аккуратного» одномерного поиска. В самом деле, по определению имеем

$$y_k^T s_k = \alpha_k (g_{k+1}^T p_k - g_k^T p_k).$$

Поскольку p_k (при положительно определенной B_k) является направлением спуска, величина $-g_k^T p_k$ и шаг α_k в правой части равенства положительны. Отрицательным может быть только скалярное произведение $g_{k+1}^T p_k$, но, повышая точность одномерной минимизации, его всегда можно сделать сколь угодно близким к нулю (так как $g_{k+1}^T p_k$ обращается в нуль, если α_k — точное решение задачи минимизации F вдоль p_k). Тогда сумма в скобках и величина $y_k^T s_k$ станут положительными, что и требовалось установить.

Конечная сходимость для квадратичных функций. Подчиняя очередное приближение B_{k+1} матрицы Гессе квазиньютоновскому условию $B_{k+1} s_k = y_k$, мы добиваемся, чтобы это приближение правильно отражало кривизну F вдоль определенного направления. Однако последующие модификации B_{k+1} могут приводить к утрате данного свойства. Для того чтобы приобретаемая информация о кривизне сохранялась в течение нескольких итераций, т. е. для некоторых j ($j \leqslant k$) было обеспечено равенство

$$B_{k+1} s_j = y_j, \tag{4.42}$$

надо принимать специальные меры. Когда квазиньютоновский поиск используется для минимизации квадратичной формы, соблюдение равенства (4.42) для n линейно независимых векторов $\{s_j\}$ гарантирует, что на $(n+1)$-м шаге будет найдено точное решение. Действительно, пусть G — матрица Гессе минимизируемой формы, а S — матрица, l-й столбец которой равен s_l. Тогда (4.42) преобразуется к виду

$$B_{n+1} S = GS,$$

где учтено, что $y_l = G s_l$. Поскольку S по предположению невырождена, отсюда следует, что $B_{n+1} = G$, а это значит, что $(n+1)$-й шаг будет ньютоновским, т. е. приведет в точку минимума.

Если применить для минимизации квадратичной функции какую-нибудь формулу из однопараметрического семейства (4.35) и на каждой итерации точно решать задачу одномерного поиска, то

для $j=0, \ldots, n-1$ гарантированы соотношения

$$B_k s_j = y_j, \quad j < k;$$
$$s_i^T G s_j = 0, \quad i \neq j.$$

Второе из них обеспечивает линейную независимость векторов $\{s_i\}$ и тем самым конечную сходимость метода. Таким образом, все квазиньютоновские формулы из (4.35) дают минимум квадратичной функции за конечное число шагов, и для всех последнее приближение матрицы G (если она невырождена) совпадает с ней, хотя промежуточные приближения B_k для разных формул из (4.35) будут разными.

4.5.2.2. Реализация. Направления поиска p_k и в ньютоновских, и в квазиньютоновских методах определяются в результате решения некоторых систем линейных уравнений. В ньютоновских методах это - системы с матрицами Гессе G_k. Последние вычисляются на разных итерациях совершенно независимо друг от друга, и поэтому расчет p_k всякий раз приходится начинать «с нуля». Точнее говоря, чтобы найти p_k, на каждой итерации приходится заново строить какое-то матричное разложение. Иначе дело обстоит с квазиньютоновскими методами. В них матрица очередной системы для построения p_k отличается от предыдущей лишь поправкой малого ранга. В данном случае каждый раз заново строить ее разложение неэкономно (см. разд. 2.2.5.7).

Проблему эффективной организации расчета направлений квазиньютоновского поиска можно решать по-разному. Авторы первых квазиньютоновских алгоритмов делали это просто: они отказывались от системы (4.28), конструируя такие процедуры, которые оперируют приближениями не самой матрицы Гессе, а *обратной к ней матрицы*. Соответствующие формулы пересчета выводились независимо от рассмотренных в предыдущем разделе, но могут быть получены и из них как формулы пересчета матриц B_k^{-1}. Когда метод минимизации строится на квазиньютоновских приближениях H_k обращенной матрицы Гессе, направления поиска p_k вычисляются по формуле

$$p_k = -H_k g_k, \tag{4.43}$$

а квазиньютоновское условие принимает вид равенства

$$H_{k+1} y_k = s_k. \tag{4.44}$$

На первый взгляд может показаться, что переход к аппроксимации обращенной матрицы Гессе сулит значительное упрощение расчета направления поиска, поскольку решение системы (4.28) «с нуля» требует порядка n^3 операций, в то время как число операций, необходимых для вычисления матрично-векторного произведения в (4.43), пропорционально n^2. Однако приведенная оценка трудоемкости построения p_k в методе с аппроксимацией прямой матрицы Гессе относится только к самому примитивному способу организации вычислений. Эта оценка

будет совсем иной, если вместо явной формы задания квазиньютоновских матриц использовать их разложения Холесского. Коль скоро LDL^T-разложение матрицы B_k известно, число операций, необходимых для решения системы (4.28), будет кратно n^2, причем трудоемкость расчета факторов L_{k+1} и D_{k+1} матрицы B_{k+1}, получающейся из B_k квазиньютоновской модификацией малого ранга, оказывается примерно такой же, как и трудоемкость расчета очередной матрицы H_{k+1} (см. разд. 2.2.5.7).

Имея разложения Холесского матриц B_k, можно получать полезные оценки их чисел обусловленности. Например, нетрудно показать, что если максимальный и минимальный диагональные элементы фактора D_k равны $d_{\max}$ и $d_{\min}$, то справедливо неравенство $\operatorname{cond}(B_k) \geqslant d_{\max}/d_{\min}$. В момент завершения процесса минимизации оценка числа обусловленности позволит дать заключение о пригодности полученного приближенного решения (см. разд. 8.3.3). На промежуточных итерациях такие оценки нужны для выявления ситуаций, когда матрица B_k обусловлена столь плохо, что в численном решении p_k системы (4.28) скорее всего не окажется ни одной правильной цифры. При этом факторы разложения легко подправить так, что обусловленность результирующей матрицы станет приемлемой. Подобные корректировки являются одновременно и средством достижения глобальной сходимости (см. разд. 4.5.2.3).

Подключение факторизации Холесского позволяет избежать еще одной неприятной проблемы, которая часто оказывается камнем преткновения для обычных квазиньютоновских методов. Имеется в виду проблема *потери (из-за ошибок округления) положительной определенности квазиньютоновских матриц*. В теоретических дискуссиях она часто замалчивается, но для практиков очень важна, и в особенности, если говорить о задачах минимизации функций сложной структуры. С точки зрения теории гарантировать положительную определенность нетрудно. Например, для BFGS-формулы (4.37) достаточно соблюдать неравенство (4.41). Однако на самом деле нередко случается, что это неравенство выполнено, а вычисленная квазиньютоновская матрица из-за погрешностей машинной арифметики оказывается знаконеопределенной.

Пример 4.8. Рассмотрим двумерный случай с матрицей

$$B_k = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

и пусть $s_k = (1,\ 10^{-3})^T$, $y_k = (0,\ 1)^T$. Неравенство $y_k^T s_k > 0$ соблюдено, т. е. пересчет B_k по BFGS-формуле должен дать положительно определенную матрицу. Тем не менее, если вычисления провести на машине с пятиразрядной десятичной точностью, результатом этого пересчета будет матрица с нулем в (1,1)-й позиции, а это исключает положительную определенность. При

абсолютно точных вычислениях в данной позиции стояло бы малое положительное число, но при расчетах с ошибками округления оно теряется.

Утрата положительной определенности нежелательна не только из-за того, что вектор p_k в этом случае может указывать в сторону возрастания минимизируемой функции, но и потому, что она чревата внесением в квазиньютоновскую аппроксимацию бессмысленных поправок, уничтожающих накопленную информацию о кривизне F. При этом по данным функционирования обычного (без факторизации) квазиньютоновского алгоритма уловить момент, когда матрица B_k перестала быть положительно определенной, иногда просто невозможно: индикаторами здесь служат векторы p_k, а они могут оставаться направлениями спуска и в отсутствие положительной определенности B_k. Если же в таком алгоритме потеря положительной определенности выявлена, то, как правило, приходится возвращаться к единичной матрице, стирая тем самым полезную информацию о кривизне F. Короче говоря, при реализации квазиньютоновского подхода очень важно позаботиться о *численной положительной определенности приближений матрицы Гессе* (или обратной к ней матрицы).

Коль скоро квазиньютоновский поиск организуется с привлечением факторизации Холесского, избежать потери положительной определенности нетрудно. Здесь все зависит от значений диагональных элементов фактора D_k: пока они больше нуля, положительная определенность соответствующей квазиньютоновской матрицы гарантирована. Контролируя эти значения при пересчете диагонального фактора, мы всегда выявим опасность потери положительной определенности в тот самый момент, когда она возникает, и сможем принять своевременные меры.

На рис. 4m изображена траектория квазиньютоновского поиска при минимизации функции Розенброка (пример 4.2). Для аппроксимации матрицы Гессе использовалась BFGS-формула, а шаги α_k определялись как аккуратные приближения точек минимумов по направлениям спуска. Отметим, что, как и ньютоновский метод, алгоритм обеспечивает быстрый прогресс вдали от решения. Затруднения возникают только в окрестности начала координат, где кривизна минимизируемой функции меняется наиболее быстро.

***4.5.2.3. Сходимость; экстремальность квазиньютоновских поправок.** В данном разделе приводится перечень некоторых интересных свойств квазиньютоновских методов. Более полные сведения читатель найдет по ссылкам, данным в замечаниях.

Глобальная сходимость квазиньютоновского поиска к стационарной точке может быть доказана для дважды непрерывно дифференцируемой функции F в следующих предположениях: (i) множество $L(F(x_0))$ ограничено; (ii) матрицы B_k положительно определены и их числа обусловленности равномерно ограничены; (iii) выбор шагов

α_k осуществляется на основе какого-нибудь из критериев разд. 4.3.2.1. Требование к матрицам B_k гарантирует, что угол между направлением поиска и антиградиентом всегда будет отличаться от прямого не менее чем на некоторую ненулевую константу. Если $\{B_k\}$ генерируются процедурой с использованием разложений Холесского (см. разд. 4.5.2.2), удовлетворить этому требованию

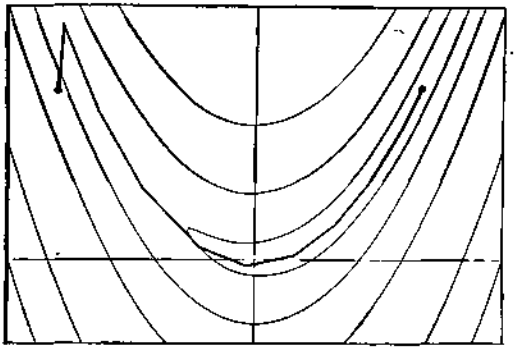

Рис. 4п. Траектория поиска минимума функции Розенброка квазиньютоновским методом с BFGS-формулой.

нетрудно, так как в правилах пересчета факторов можно предусмотреть контроль числа обусловленности и (выполняемые по мере необходимости) корректировки, непосредственно ограничивающие его значение.

Если же матрицы B_k всегда вычисляются по какой-то обычной квазиньютоновской формуле и никакого явного средства ограничения их чисел обусловленности не используется, то для доказательства сходимости придется усилить предположения о способе выбора шагов α_k и о свойствах минимизируемой функции. Так, например, сходимость квазиньютоновского процесса с применением какой-нибудь из формул однопараметрического семейства (4.35) может быть доказана в предположении, что одномерный поиск осуществляется точно и собственные значения матрицы Гессе минимизируемой функции равномерно по x ограничены сверху и снизу положительными константами. Сходимость при выборе α_k на основе приближенной одномерной минимизации пока удалось доказать только для BFGS-формулы. Точнее говоря, установлено, что соответствующий метод сойдется, если вычислять α_k согласно критериям (4.7) и (4.8), причем довольно неестественное предположение относительно соб-

ственных чисел матрицы Гессе пришлось сохранить. (Читатель, наверное, догадывается о том, что требования, предъявляемые к минимизируемой функции, обусловлены в основном сложностью доказательств и не отражают реальных возможностей квазиньютоновских методов. На самом деле класс функций, для которых они сходятся, значительно шире оговариваемого в формулировках теорем сходимости.)

Наряду с работами, посвященными вопросам существования сходимости квазиньютоновских методов, немало усилий было затрачено и на исследование скоростей их сходимости. Мы снова ограничимся краткой сводкой результатов, а более полные сведения можно найти в рекомендуемой ниже литературе.

Некоторые свойства сходимости квазиньютоновских методов удается получить анализом общей итерационной схемы, в которой

$$x_{k+1}=x_k+p_k, \tag{4.45}$$

где p_k удовлетворяет равенству $B_k p_k=-g_k$. Заметьте, что в этой схеме длина шага α_k вдоль p_k равна единице. При необременительных предположениях относительно матриц B_k (выполненных, в частности, для BFGS-, PSB- и DFP-формул) и минимизируемой функции F можно доказать следующее: если в стационарной точке x^* матрица $G(x^*)$ положительно определена и x_0, B_0 «достаточно близки» к x^*, $G(x^*)$ соответственно, то последовательность (4.45) линейно сойдется к x^*. Свойство, которое требуется от матриц B_k при доказательстве этой локальной сходимости, называют *ограниченной изначальностью*; применительно к перечисленным выше формулам оно проявляется в виде неравенств

$$\|B_{k+1}-G(x^*)\|\leqslant\gamma_1\|B_k-G(x^*)\|+(1+\gamma_2)\max\{\|x_k-x^*\|,\|x_{k+1}-x^*\|\}.$$

Здесь γ_1 и γ_2 — некоторые неотрицательные константы, а выбор нормы не конкретизирован, поскольку разным формулам пересчета отвечают разные нормы.

В определенных условиях квазиньютоновские методы сходятся сверхлинейно. При выводе таких условий обычно учитывают, что для сходящейся сверхлинейно последовательности $\{x_k\}$ должно выполняться соотношение

$$\lim_{k\to\infty}\frac{\|x_{k+1}-x_k\|}{\|x_k-x^*\|}=1. \tag{4.46}$$

В этом можно убедиться на примере с квадратично сходящейся к нулю последовательности чисел 10^{-4}, 10^{-8}, 10^{-16} и т. д.

Если известно, что последовательность точек, генерируемая по схеме (4.45), сходится с линейной скоростью, то для того, чтобы сходимость на самом деле была *сверхлинейной*, необходимо и достаточно, чтобы матрицы B_k удовлетворяли условию

$$\lim_{k\to\infty}\frac{\|(B_k-G(x^*))(x_{k+1}-x_k)\|}{\|x_{k+1}-x_k\|}=0. \tag{4.47}$$

Одно из наиболее интересных следствий этого результата состоит в том, что из него вытекает возможность получить сверхлинейную сходимость квазиньютоновского поиска даже тогда, когда сходимости B_k и $G(x^*)$ нет. Оказывается, условию (4.47) можно удовлетворить и в ее отсутствие. При некоторых предположениях это условие будет, в частности, выполнено для BFGS-, PSB- и DFP-формул.

Предыдущие рассуждения относились к случаю, когда шаги α_k берутся единичными. Если же квазиньютоновский метод предполагает регулировку шага, то на сверхлинейную сходимость можно рассчитывать лишь при условии, что последовательность $\{\alpha_k\}$ будет достаточно быстро стремиться к единице. По этой причине процедуры одномерного поиска в квазиньютоновских методах обычно рекомендуется запускать с единичной начальной оценки шага.

Интересно отметить, что поправочные матрицы U_k в некоторых квазиньютоновских формулах представляют собой решения специфических оптимизационных задач типа: найти для матрицы B_k поправку U_k минимальной нормы, обеспечивающую матрице B_k+U_k заданные свойства (такие, как подчинение квазиньютоновскому условию, симметричность, положительная определенность и т. д.). Так, например, PSB-формуле отвечает задача вида

$$\begin{gathered} \text{найти } \min \|U\|_F^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} U_{ij}^2 \\ \text{при условиях } Us_k = y_k - B_k s_k, \\ U = U^T. \end{gathered} \tag{4.48}$$

Здесь через $\|U\|_F$ обозначена норма Фробениуса матрицы U. Аналогичные задачи, но с иными нормами можно поставить в соответствие и другим квазиньютоновским формулам.

Замечания и избранная библиография к разделу 4.5

Первый квазиньютоновский метод был разработан Дэвидоном (1959). Автор назвал его методом «переменной метрики». В этом названии отражена интерпретация вектора p_k из уравнения (4.28) как шага в точку минимума квадратичной модельной функции. Будучи положительно определенной, ее матрица Гессе G задает «метрику» по правилу, приведенному в разд. 4.3.2.2. Вектор p_k, указывающий в точку минимума этой функции, есть направление наискорейшего спуска в данной метрике. Ну а поскольку матрица G от итерации к итерации меняется, можно говорить о наискорейшем спуске с пошаговым изменением метрики. Отсюда и название — «метод переменной метрики».

Метод Дэвидона был разрекламирован и усовершенствован Флетчером и Пауэллом (1963) (что объясняет термин «DFP-формула»). Они первыми показали, что для квадратичных функций при точной одномерной минимизации он даст решение за конечное число ша-

гов. Формулу BFGS независимо предложили Бройден (1970), Флетчер (1970a), Гольдфарб (1970) и Шанно (1970).

Начиная с 1963 г. интерес к квазиньютоновским методам непрерывно возрастал. За время, прошедшее с тех пор, накопилась обширнейшая литература, в которой проанализированы всевозможные аспекты методов этого сорта. Хорошие обзоры с дополнительными ссылками читатель найдет в работах Авриеля (1976), Бродли (1977a), Денниса и Море (1977) и Флетчера (1980).

Квазиньютоновские методы очень близки к методам сопряженных градиентов (см. Назарет (1979) и разд. 4.8.3).

Идея применения факторизации Холесского в качестве средства, позволяющего получать численно устойчивые реализации квазиньютоновского поиска, принадлежит Гиллу и Мюррею (1972) (см. также работы Флетчера и Пауэлла (1974), Гилла, Мюррея и Сондерса (1975) и Бродли (1977b)). Следующим шагом по повышению устойчивости было бы устранение неоправданных потерь точности в ситуациях, когда норма квазиньютоновской поправки оказывается много меньше суммы норм составляющих ее простейших матриц. Например, любую модификацию ранга два можно было бы осуществлять по формуле вида

$$B_{k+1} = B_k + uu^T - vv^T,$$

где $u^Tv = 0$ (см. Гилл и Мюррей (1978c)).

Основные заслуги в исследовании сходимости квазиньютоновских методов принадлежат Бройдену, Деннису, Море, Пауэллу и Стоеру (см. Бройден (1967, 1970), Бройден, Деннис и Море (1973), Пауэлл (1971, 1975, 1976c) и Стоер (1975, 1977)). Критерий (4.46) достижения сверхлинейной скорости квазиньютоновского поиска получен Деннисом и Море (1974). Обзор результатов по сходимости с хорошей библиографией содержится в работах Денниса и Море (1977) и Бродли (1977b). Замечательный результат (4.38) о теоретической эквивалентности методом однопараметрического семейства (4.35) получен Диксоном (1972a, b). Экстремальные свойства квазиньютоновских поправок первым заметил Гринштадт (1970). Читателя, желающего ознакомиться с этим аспектом квазиньютоновских формул поподробнее, мы отсылаем к работам Денниса и Море (1977), Денниса и Шнабеля (1979, 1980).

После того как квазиньютоновская техника прочно вошла в практику вычислений и накопленный опыт позволил ранжировать разные методы по эффективности, лучшие из них стали предметом интенсивных теоретических исследований. Целью было выделить те их особенности, которые обусловливают высокую реальную скорость сходимости, а затем формализовать и решить задачу о выборе оптимального значения параметра в (4.35). К примеру, одно из самых неожиданных свойств квазиньютоновских методов, выявленное на основе экспериментов, состоит в том, что более эффективными (в смысле числа обращений к процедуре расчета минимизируемой

функции) оказываются реализации без точного одномерного поиска. Попытки истолковать это явление предпринимались несколькими авторами, в том числе и Флетчером (1970а). Заметив, что для любой формулы однопараметрического семейства (4.35) справедливо равенство

$$B_k^{\varphi} \cdot (1-\varphi) B_k^{c} + \varphi B_k^{1},$$

где B_k^{c} и B_k^{1}—квазиньютоновские приближения матрицы Гессе, отвечающие BFGS- и DFP-формулам соответственно, он ввел выпуклый подкласс формул из (4.35), определенный условием $\varphi \in [0, 1]$, и показал, что любая формула из этого подкласса для квадратичной функции F гарантирует монотонную сходимость собственных чисел матрицы

$$R_k = G^{1/2} B_k^{-1} G^{1/2} \tag{4.49}$$

к единице *независимо от того, как выбираются шаги* α_k.

Наряду с попытками формализации задачи о поиске оптимального метода велись и работы иного сорта, в которых неоднозначность выбора формулы пересчета квазиньютоновских приближений снималась на основе некоторых «естественных» дополнительных требований. Например, Дэвидон (1975), Орен и Спедикато (1976) предложили подбирать очередную квазиньютоновскую матрицу так, чтобы оценка сверху числа ее обусловленности была минимальной. Тем самым определились и формулы пересчета. Орен и Лейбергер (1974) тоже ориентировались на число обусловленности, но не самой квазиньютоновской матрицы, а произведения (4.49). Целой введения дополнительного параметра они построили метод, который для квадратичной функции при точном одномерном поиске гарантирует, что это число будет монотонно убывать (см. также Орен (1974а, b)). В последнее время стали появляться также попытки конструировать формулы, обеспечивающие конечную сходимость квазиньютоновского поиска для функций более общих, чем квадратичные (см. Дэвидон (1979), Соренсен (1980b)). Дэвидон (1975) предложил класс методов, которые для квадратичных форм сходятся за конечное число итераций в отсутствие точного одномерного поиска. Спедикато (1975) выделил семейство формул, инвариантных относительно некоторого нелинейного масштабирования.

4.6. МЕТОДЫ МИНИМИЗАЦИИ ГЛАДКИХ ФУНКЦИЙ БЕЗ ВЫЧИСЛЕНИЙ ПРОИЗВОДНЫХ

На практике нередко приходится сталкиваться с задачами минимизации функций, которые хотя и являются гладкими, но настолько сложны, что оказывается трудным или невозможным организовать вычисление аналитических значений даже первых производных. К примеру, это бывает, когда функция определена как результат сложной цепи расчетов по какой-нибудь математической

модели, например имитационной. Бывает и так, что сначала вообще не ясно, является ли функция гладкой. Тогда нужно выяснить это, и здесь усилий жалеть не стоит. Конечно, в рассматриваемой ситуации все равно придется обратиться к методам без вычисления производных, но к каким именно методам, зависит от гладкости. Если производных не существует, прибегают к методам типа алгоритма многогранника из разд. 4.2.1, специально разработанным для негладких задач. Если же производные есть, но просто не поддаются вычислению, эти методы применять не следует — слишком уж осторожно используется в них информация о значениях функции. Коль скоро известно, что функция гладкая, этой информацией можно оперировать и посмелее. Пример методов предыдущего раздела, использующих значения только первых производных и тем не менее правильно аппроксимирующих кривизну минимизируемой функции, а потому и весьма эффективных, показывает, что в этом есть смысл.

4.6.1. КОНЕЧНО-РАЗНОСТНАЯ АППРОКСИМАЦИЯ ПЕРВЫХ ПРОИЗВОДНЫХ

Когда поставлена задача минимизации гладкой функции, точные значения производных которой недоступны, естественно возникает желание воспользоваться каким-нибудь из методов первого порядка, подменив истинный градиент $g(x)$ его *конечно-разностной аппроксимацией*. Часто это и в самом деле оказывается выходом из положения. Однако подобная адаптация методов первого порядка — проблема, к сожалению, нетривиальная, и это связано с их чувствительностью по отношению к ошибкам конечно-разностных приближений. Природа этих ошибок рассматривается в следующем разделе.

4.6.1.1. Ошибки при правой конечно-разностной аппроксимации. Чтобы не загромождать изложения, мы ограничимся анализом ошибки, возникающей при численном оценивании первой производной дважды непрерывно дифференцируемой функции *скалярного* аргумента x. Чаще всего оценку вычисляют по *правой конечно-разностной формуле* (см. разд. 2.3.5). В этом случае аналитическим приближением производной $f'(x)$ служит величина

$$\varphi_F(f,\ h) = \frac{f(x+h)-f(x)}{h}. \tag{4.50}$$

Численная аппроксимация значений $f'(x)$ по формуле (4.50) связана с ошибками трех типов.

Ошибка отбрасывания. Эта ошибка равна остаточному члену тейлоровского разложения, отброшенному при выводе формулы аппроксимации, т. е. представляет собой разность

$$\varphi_F(f,\ h) - f'(x) = \frac{h}{2} f'(\xi) = T_F(h), \tag{4.51}$$

где ξ — некоторая точка отрезка $[x,\ x+h]$ (см. разд. 2.3.4.).

Ошибка условий. При реализации формулы (4.50) на машине функция f будет вычисляться с ошибками. Это значит, что вместо $f(x)$ и $f(x+h)$ в правую часть (4.50) войдут величины $\tilde{f}(x)$ и $\tilde{f}(x+h)$, связанные с истинными значениями f следующим образом:

$$\tilde{f}(x)=f(x)+\sigma, \quad \tilde{f}(x+h)=f(x+h)+\sigma_h.$$

Здесь $|\sigma|$, $|\sigma_h|$ — абсолютные ошибки вычисления f в точках x и $x+h$ (см. разд. 2.1.6). Соответственно численной оценкой производной в отсутствие прочих ошибок будет величина, равная

$$\varphi_F(\tilde{f}, h)=\frac{\tilde{f}(x+h)-\tilde{f}(x)}{h}.$$

Ее выражение через аналитическую оценку выглядит так:

$$\varphi_F(\tilde{f}, h)=\frac{f(x+h)-f(x)}{h}+\frac{\sigma_h-\sigma}{h}=\varphi_F(f, h)+C(\varphi_F, h).$$

Отклонение $C(\varphi_F, h)$ численной оценки от аналитической называют *ошибкой условий* (по классификации разд. 2.1.5 она относится к *ошибкам компенсации*). Справедливо равенство

$$C(\varphi_F, h)=\frac{2\psi_F\max\{|\sigma|, |\sigma_h|\}}{h}, \tag{4.52}$$

где $|\psi_F|\leqslant 1$.

Если модуль $|f|$ не слишком мал, то σ и σ_h можно выразить через относительные ошибки вычисления f, т. е. воспользоваться представлениями вида

$$\sigma=\varepsilon f(x), \quad \sigma_h=\varepsilon_h f(x+h),$$

где $|\varepsilon|\leqslant\varepsilon_F$, $|\varepsilon_h|\leqslant\varepsilon_K$. Тогда выражение для ошибки условий преобразуется следующим образом:

$$C(\varphi_F, h)=\frac{2}{h}\theta_F M_F\varepsilon_K. \tag{4.53}$$

Здесь $|\theta_F|\leqslant 1$ и $M_F=\max\{|f(x)|, |f(x+h)|\}$.

Ошибки округления. Вычисление оценки φ_F по заданным $\tilde{f}(x)$ и $\tilde{f}(x+h)$ будет сопровождаться ошибками округления, неизбежными при машинном вычитании и делении. Однако эти ошибки, как правило, *пренебрежимо малы* в сравнении с рассмотренными выше. Поэтому в дальнейшем они не упоминаются.

4.6.1.2. Выбор конечно-разностного интервала. Ошибка численного оценивания производной $f'(x)$ величиной $\varphi_F(\tilde{f}, h)$ складывается из ошибок отбрасывания (4.51) и условий (4.52) (или (4.53)). Первая из них пропорциональна конечно-разностному интервалу h, а вторая — обратной ему величине. Таким образом, последствия варьирования h для них противоположны.

Пример 4.9. Рассмотрим результаты численного оценивания при разных h производной функции

$$f(x)=(e^x-1)^6+\left(\frac{1}{\sqrt{1+x^3}}-1\right)^6 \tag{4.54}$$

в точке $x=1$. Они получены на машине IBM 370 с использованием одинарной точности вычислений. Минимальная добавка, которая приведет к изменению единицы в операции сложения с плавающей запятой, равна машинной точности ε_M. Для IBM 370 этим порогом является $\varepsilon_M=16^{-5}\approx 0{,}95\times 10^{-6}$. Он и был взят в качестве первого значения h. Кроме того, расчеты выполнены еще для четырех значений. Каждое последующее было больше предыдущего в десять раз. Данные расчетов приведены в табл. 4а. Ее первые три колонки содержат пробные значения h и вычисляемые в точках $x=1$ и $x+h$ значения функции (4.54). В четвертую колонку записаны модули $|T(h)|$ ошибок отбрасывания. Они найдены при разных h для аналитических оценок $\varphi_F(h)$, вычисленных по точным (подсчитанным с двойной точностью) значениям функции. В пятой колонке представлены модули ошибок условий, которые рассчитывались с использованием аналитических оценок φ_F. Наконец, шестая колонка содержит численные оценки $\varphi_F(\hat{f}, h)$. Истинная величина производной $f'(x)$ (округленная до шести значащих цифр) равна $0{,}954866\times 10^1$.

В идеале конечно-разностный интервал h следовало бы выбрать так, чтобы уклонение оценки $\varphi_F(\hat{f}, h)$ от настоящей производной было минимально, т. е. из решения задачи минимизации суммы ошибок условий и отбрасывания. В частности, для примера 4.9 из табл. 4а видно, что оптимальное значение h лежит между $100\ \varepsilon_M$ и $1000\ \varepsilon_M$. Однако в реальных процедурах оценивания производных таких таблиц не составляют, и величины, фигурирующие в (4.51) и (4.53), остаются неизвестными (так как если бы знать истинное значение производной и иметь возможность вычислений с повышенной точностью, как это было при построении табл. 4а, то занимался подбором h было бы незачем). Поэтому отыскать значение h, доставляющее минимум содержащейся в φ_F ошибке, не удается.

Таблица 4а. Ошибка условий и ошибка отбрасывания в φ_F при $\varepsilon_M=0{,}953674\times 10^{-6}$

h	$f(x)$	$\hat{f}(x+h)$	$\lvert T(h)\rvert$	$\lvert C(\varphi_F, h)\rvert$	$\varphi_F(\hat{f}, h)$
ε_M	0.303828×10^1	0.303828×10^1	0.115897×10^{-4}	0.254867×10^0	0.700000×10^1
$10\varepsilon_M$	0.303828×10^1	0.303837×10^1	0.115711×10^{-3}	0.487710×10^{-1}	0.950000×10^1
$10^2\varepsilon_M$	0.303828×10^1	0.303919×10^1	0.115718×10^{-2}	0.298130×10^{-1}	0.952000×10^1
$10^3\varepsilon_M$	0.303828×10^1	0.304739×10^1	0.115790×10^{-1}	0.223475×10^{-2}	0.955800×10^1
$10^4\varepsilon_M$	0.303828×10^1	0.313045×10^1	0.116520×10^0	0.275015×10^{-2}	0.966490×10^1

На практике конечно-разностный интервал подбирают из соображений минимизации *вычислимой оценки суммарной ошибки.*

В качестве такой оценки берут

$$\frac{h}{2}|\Phi|+\frac{2}{h}\overline{C},$$

где Φ — оценка второй производной $f''(\xi)$, а $\overline{C}$ — верхняя граница неизвестного множителя в выражении для ошибки условий. (Метод расчета величин Φ и $\overline{C}$, использующий только значения функции, представлен в разд. 8.6.) В предположении, что величина Φ не равна нулю, искомый минимум достигается при

$$\hat{h}=\sqrt{\frac{4\overline{C}}{|\Phi|}}. \tag{4.55}$$

Интересно посмотреть, каким будет интервал $\hat{h}$ для функции (4.54) из примера 4.5, если в роли Φ использовать точное значение второй производной $f''(x)$, а $\overline{C}$ определить по точному значению ошибки условий. Оказывается, что при $x=1$ величина $\hat{h}$ равна 4.497×10^{-4} (и попадает в диапазон, предсказанный на основе табл. 4в), а для $x=50$ получим $\hat{h}=1.436\times10^{-3}$.

4.6.1.3. Построение набора конечно-разностных интервалов. Результаты предыдущего раздела без изменений переносятся на многомерный случай, когда в процессе минимизации некоторой функции F ее градиент $g(x)$ аппроксимируется в точке x_k вектором $\hat{g}(x_k)$, j-я компонента которого вычисляется по формуле

$$\hat{g}_j(x_k)=\frac{1}{h_j}(F(x_k+h_je_j)-F(x_k)).$$

Здесь нужен набор конечно-разностных интервалов $\{h_j\}$. Строятся они независимо друг от друга на основе формулы (4.55), но, так как реализация этой формулы (определение фигурирующих в ней величин $\overline{C}$ и Φ), описанная в разд. 8.6, предполагает вычисление функции по меньшей мере в двух дополнительных точках, обычно считают нецелесообразным прибегать к ней на каждом шаге основного процесса. Процедуры типа представленной в разд. 8.6, как правило, используются только один раз — в начальной точке x_0.

Обозначим через $\hat{h}_j$ оценку оптимального конечно-разностного интервала для j-й переменной, вычисленную в x_0. Величина $\hat{h}_j$ представляет собой *абсолютный* интервал. Вводится и понятие *относительного* интервала. Набор относительных интервалов $\{\delta_j\}$ определяют так:

$$\delta_j=\frac{\hat{h}_j}{1+\sigma_j|(x_0)_j|}.$$

Здесь σ_j некий параметр, удовлетворяющий неравенству $0\leqslant\sigma_j\leqslant1$, причем обычно σ_j полагают равным единице. Опыт показывает, что оптимальные значения относительных интерва-

лов устойчивее оптимальных значений абсолютных. Следовательно, если отказаться от расчета на каждом шаге оценок (4.55), то имеет смысл, определив набор $\{\delta_j\}$ по $\{\bar{h}_j\}$ в точке x_0, на последующих итерациях вычислять h_j по формуле

$$h_j = \delta_j (1 + \sigma_j |(x_k)_j|). \qquad (4.56)$$

Важно отметить, что фиксированного набора относительных интервалов, подходящего для всех точек, где в процессе поиска минимума придется аппроксимировать градиент, может не существовать. Поэтому на те редкие случаи, когда интервалы h_j из (4.5.6) окажутся неприголными и в алгоритме минимизации из-за этого возникнут трудности, предусматривают возможность повторных обращений к формуле (4.55).

4.6.1.4. Выбор конечно-разностных формул. Как уже было сказано ранее, наиболее ходовым средством оценивания первых производных является правая конечно-разностная формула. Она требует вычисления лишь одного дополнительного значения функции на одну компоненту вектора градиента и там, где норма $\|g(x)\|$ достаточно велика, обычно обеспечивает приемлемое качество оценок. Однако по мере приближения к искомой точке безусловного минимума величина $\|g(x)\|$ стремится к нулю. Значит, даже при хорошо отмасштабированной функции и правильно выбираемых конечно-разностных интервалах *правая аппроксимация рано или поздно теряет надежность*. К сожалению, это обычно происходит *задолго до того, как достигнута предельная точность решения задачи*. Поэтому для завершающих итераций поиска минимума правая конечно-разностная формула может не подойти.

Иногда от правой формулы приходится отказываться и на ранних итерациях, где от x_k до оптимума еще далеко. Ее следует заменять всякий раз, когда приращение функции при сдвиге j-й компоненты вектора x_k на h_j мало относительно h_j: в данном случае возникающая при ее реализации ошибка условий обесценивает результат вычисления. Более точная аппроксимация желательна и тогда, когда шаг спуска оказывается малым по сравнению с приращениями, используемыми для оценивания градиента.

Если точности правой конечно-разностной аппроксимации не хватает, вместо нее можно использовать *центральную аппроксимацию* $\bar{g}(x_k)$, j-я компонента которой определяется по формуле

$$\bar{g}_j(x_k) = \frac{1}{2h_j}(F(x_k + h_j e_j) - F(x_k - h_j e_j)). \qquad (4.57)$$

Для нее ошибка отбрасывания является величиной порядка h_j^2 (см. разд. 2.3.5). Что же касается ошибки условий, то она останется пропорциональной $1/h_j$.

При переключении с правой конечно-разностной аппроксимации на центральную, вероятнее всего, потребуется *увеличить*

величины интервалов h_j. Когда функция F хорошо отмасштабирована (в смысле определений разд. 8.7), разумной оценкой $\bar{h}_j$ оптимального интервала центральной аппроксимации будет $\bar{h}_j = (\bar{h}_j)^{2/3}$.

Для центральной аппроксимации, как и для правой, целесообразно ввести набор относительных интервалов $\{\bar{\delta}_j\}$. Тогда, подобрав однажды значения $\{\bar{h}_j\}$ путем минимизации оценки суммарной ошибки и зафиксировав соответствующие величины $\{\bar{\delta}_j\}$, в дальнейшем можно вычислять h_j для (4.57) по формулам, аналогичным (4.56).

Когда градиент аппроксимируется по центральной конечно-разностной формуле, оценивание одной его компоненты требует двух дополнительных вычислений функции. Поэтому не стоит пользоваться центральной аппроксимацией там, где пригодна правая, и не стоит окончательно переключаться на нее до того, как поиск зайдет в малую окрестность решения и уровень ошибок правой аппроксимации станет неприемлемым. Если же из-за плохих локальных свойств F к симметричной формуле придется прибегнуть в точке x_k, далекой от решения, то впоследствии рекомендуется вернуться к правой формуле.

4.6.2. КВАЗИНЬЮТОНОВСКИЕ МЕТОДЫ БЕЗ ВЫЧИСЛЕНИЯ ПРОИЗВОДНЫХ

Каждую схему из разд. 4.5 можно переделать в алгоритм минимизации без вычисления производных, заменив в ней настоящие градиенты их конечно-разностными оценками. Лучше всего для этой цели подходят квазиньютоновские схемы, рассмотренные в разд. 4.5.2. Будучи правильно реализованными, конечно-разностные квазиньютоновские алгоритмы оказываются весьма эффективными и проявляют те же свойства устойчивости и быстродействия, что и их прототипы с точными производными. Однако *было бы большим заблуждением думать, что хорошую процедуру можно получить простой комбинацией любого квазиньютоновского метода и какой-нибудь формулы аппроксимации градиента с фиксированными конечно-разностными интервалами* (подразумевается замена всех значений градиентов, фигурирующих в квазиньютоновском методе, их конечно-разностными оценками без какой-либо модификации его логики вычислений).

Во-первых, ошибки приближений производных могут существенно влиять на ход процесса минимизации, и, чтобы уменьшить их эффект, при построении алгоритма надо использовать рекомендации разд. 4.6.1. В частности, следует предусмотреть возможности смены формул аппроксимации и подбора значений конечно-разностных интервалов в зависимости от наблюдаемых характеристик минимизируемой функции.

Во-вторых, для построения каждой численной оценки градиента потребуется n (или $2n$) дополнительных вычислений функций, и, стало быть, при переходе от обычного квазиньютоновского метода к конечно-разностному необходимо позаботиться о сокращении частоты вычисления этих оценок. Это прежде всего означает, что процедуру выбора длины шага, предполагающую оценивание градиентов, нужно заменить другой, которой оно не требуется. Кроме того, как правило, придется повысить точность одномерного поиска: это ведет к сокращению числа итераций и тем самым к сокращению требуемого числа оценок градиента.

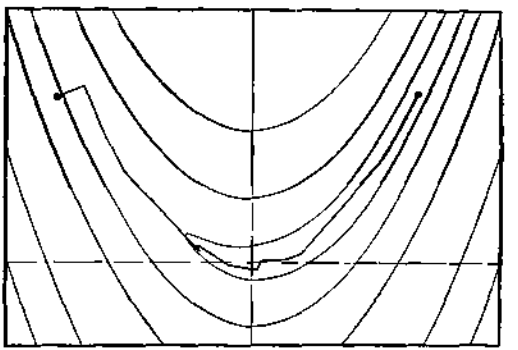

Рис. 4п. Траектория поиска минимума функции Розенброка BFGS-методом с конечно-разностной аппроксимацией градиентов.

На рис. 4п изображена траектория движения конечно-разностного квазиньютоновского алгоритма при минимизации функции Розенброка (см. пример 4.2). Этот алгоритм был построен на основе BFGS-формулы пересчета квазиньютоновских матриц. Шаги по направлениям спуска определялись аккуратным одномерным поиском. Вдали от решения алгоритм ведет себя практически так же, как обычный BFGS-метод (ср. рис. 4т), но по мере приближения к нему начинает работать сравнительно хуже. Это объясняется тем, что здесь становятся существенными ошибки использованной в нем правой конечно-разностной формулы.

Замечания и избранная библиография к разделу 4.6

Среди известных на сегодняшний день алгоритмов минимизации гладких функций без вычисления производных есть много таких, которым не нужны численные оценки градиентов; их можно найти

в работах Пауэлла (1964), Гринштадта (1972) и Брента (1973а). Однако в настоящее время считается, что конечно-разностные квазиньютоновские методы эффективнее.

В числе первых публикаций по конечно-разностным квазиньютоновским методам следует назвать работу Стюарта (1967). Он предложил процедуру, в которой интервалы аппроксимации заново рассчитываются *для каждого* шага по формуле (4.55), причем оценка $|\Phi|$ строится по диагональным элементам DFP-приближения матрицы Гессе (см. (4.34)). Данный способ выбора $|\Phi|$ основан на предположении, что диагональные элементы квазиньютоновской матрицы хорошо оценивают порядки величин точных вторых производных. Метод построения интервала для центральной конечно-разностной формулы читатель найдет в работе Кертиса и Рейда (1974). Вопросы использования конечно-разностных оценок в рамках оптимизационных процедур рассматриваются в статье Гилла и Мюррея (1972).

Общую проблему определения приближенных значений производных с применением конечных разностей называют проблемой *численного дифференцирования*. Исследования по ней уже давно выделены в самостоятельную область вычислительной математики. Современный обзор методов численного дифференцирования (включающий достаточно полное обсуждение роли ошибок условий при реализации конечно-разностных формул на машине) содержится в работах Лайнесса (1977). По этой теме можно порекомендовать также работы Андерсена и Бломфилда (1974), Дальквиста и Бьёрка (1974), Лайнесса и Молера (1967), Лайнесса и Сэнда (1971), Оливера и Рафхеда (1975) и Оливера (1980). В настоящее время существует много стандартных программ численного дифференцирования. Они всегда нацелены на поиск конечно-разностного интервала, обеспечивающего минимальную суммарную ошибку вычисляемого приближения. Соответствующие алгоритмы выбора интервала для центральной конечно-разностной формулы описаны, например, у Дюмонте и Виньеса (1977), Сиплмена и Винарски (1979). Однако для оптимизационных приложений подобные программы не подходят. Обеспечивая избыточную точность вычисления оценок производных, они обычно требуют слишком большого числа обращений к процедуре расчета значений функции.

Замена настоящих градиентов их конечно-разностными оценками лучше всего проходит в квазиньютоновских методах. Для методов ньютоновского типа результаты получаются менее удовлетворительными. Теоретически построить ньютоновский алгоритм без вычисления производных нетрудно — достаточно в обычном алгоритме заменить и градиент, и матрицу Гессе их конечно-разностными приближениями (см. Миффлин (1975)). Однако с практической точки зрения такой подход в общем случае бесперспективен, поскольку на каждой итерации требует вычисления $O(n^2)$ дополнительных значений функции. К тому же здесь нужно особенно

тщательно подбирать конечно-разностные интервалы — ведь они должны обеспечивать приемлемую точность сразу двух оценок. Тем не менее, когда матрица Гесса разрежена, ньютоновские методы без вычисления производных иногда оказываются конкурентоспособными (см. разд. 4.8.1).

4.7. МЕТОДЫ РЕШЕНИЯ ЗАДАЧ О НАИМЕНЬШИХ КВАДРАТАХ

4.7.1. ПРОИСХОЖДЕНИЕ ЗАДАЧ О НАИМЕНЬШИХ КВАДРАТАХ; ОСНОВАНИЯ ДЛЯ ИСПОЛЬЗОВАНИЯ СПЕЦИАЛЬНЫХ МЕТОДОВ

Среди задач на поиск безусловного минимума особое место занимают задачи минимизации функций $F(x)$ вида

$$F(x)=\frac{1}{2}\sum_{i=1}^{m} f_i(x)^2=\frac{1}{2}\|f(x)\|_2^2. \qquad (4.58)$$

В этой записи $f(x)$ — нелинейная векторная функция с m компонентами $f_i(x)$. Ее евклидову норму $\|f(x)\|_2$ называют *невязкой* в точке x. (Множитель $1/2$ введен в (4.58), чтобы скомпенсировать двойку, возникающую при дифференцировании.)

Задачи рассматриваемого типа возникают, например, при настройке математических моделей на реальные данные. Под математической моделью мы подразумеваем здесь функцию $\varphi(x, t)$ с независимым аргументом t и векторным параметром x. Предположим, что она сконструирована для предсказания значений некоторой величины y в зависимости от t, и пусть m фактических значений y при определенных t измерены. Результаты этих измерений, вообще говоря сопровождающихся ошибками, обозначим через y_{t_i}, а соответствующие значения переменной t — через t_i. Тогда согласование модели с реальностью будет состоять в том, чтобы подобрать параметр x, при котором разности $f_i(x)=\varphi(x, t_i)-y_{t_i}$ «близки» к нулю. Один из способов формализации данной задачи — ставить ее как задачу поиска минимума суммы квадратов от $f_i(x)$. Если модель составлена правильно, можно ожидать, что величина $\|f(x^*)\|$ окажется «малой», хотя количество наблюдений m обычно немного превосходит размерность n параметра x (при небольшом числе наблюдений хорошее согласование можно получить для любой модели).

Функции (4.58) возникают и в несколько иной ситуации в случае, когда речь идет о настройке свободных параметров объекта, поведение которого описывается в непрерывном времени и от которого надо добиться траектории, близкой к заданной. Здесь нет измерений, сопровождающихся ошибками, и в качестве «ориентира» обычно дается непрерывная кривая. Естественная постановка

задачи выбора параметров теперь будет выглядеть так:

$$\text{найти } \min_{x \in \mathbb{R}^n} \int_{t_0}^{t_k} (\varphi(x, t) - \mathcal{F}(t))^2 dt,$$

где $\mathcal{F}(t)$ — «целевая траектория». Заменив интеграл суммой с помощью какой-нибудь подходящей квадратурной формулы, мы снова приходим к задаче о наименьших квадратах:

$$\text{найти } \min_{x \in \mathbb{R}^n} \sum_{i=1}^{m} (\bar{\varphi}(x, t_i) - \bar{\mathcal{F}}(t_i))^2,$$

где через $\bar{\varphi}$ и $\bar{\mathcal{F}}$ обозначены произведения φ и $\mathcal{F}$ на веса принятой квадратурной схемы. В данном случае оптимальное значение критерия будет малым, если не требовать от объекта «невозможного».

Хотя для минимизации функций (4.58) можно использовать универсальные методы, как правило, этого не делают, а обращаются к специальным алгоритмам, разработанным именно для задач о наименьших квадратах. Последние достаточно специфичны, и естественно попытаться учесть это. В частности имеется в виду особая структура градиента функции (4.58) и ее матрицы Гессе. Обозначим через $J(x)$ $m \times n$-матрицу Якоби для $f(x)$, и пусть $G_i(x)$ — матрица Гессе для $f_i(x)$. Тогда выражения для градиента $g(x)$ и матрицы Гессе $G(x)$ функции (4.58) будут выглядеть так:

$$g(x) = J(x)^T f(x); \tag{4.59a}$$

$$G(x) = J(x)^T J(x) + Q(x). \tag{4.59b}$$

Здесь через $Q(x)$ обозначена сумма $\sum_{i=1}^{m} f_i(x) G_i(x)$. Из (4.59b) видно, что элементы $G(x)$ образованы характерными комбинациями первых и вторых производных функций f_i.

Большинство алгоритмов решения задач о наименьших квадратах опирается на предположение о том, что слагаемое $J(x)^T J(x)$ в (4.59b) рано или поздно станет доминирующим. Это предположение не соблюдается, если минимальные невязки велики, т. е., грубо говоря, если норма $\|f(x^*)\|$ сравнима с максимальным собственным значением матрицы $J(x^*)^T J(x)$. В данном случае преимущества специальных методов над универсальными теряются. Однако чаще встречаются ситуации, когда невязки в решении достаточно малы и применение специальной техники оправданно.

4.7.2. Метод Гаусса — Ньютона

Обозначим через x_k текущую оценку решения задачи минимизации функции (4.58) и условимся, что нижний индекс k у любой величины будет означать ее принадлежность x_k. Тогда ньютонов-

ская система (4.17) в силу (4.59) примет вид

$$(J_k^T J_k + Q_k)\, p_k = -J_k^T f_k. \tag{4.60}$$

Ее решением будет некоторый вектор p_N (ньютоновское направление).

Если при стремлении a_k к оптимуму норма $\|f_k\|$ приближается к нулю, то матрица Q_k тоже будет приближаться к нулевой. Значит, ньютоновское направление можно *аппроксимировать* решением системы

$$J_k^T J_k p_k = -J_k^T f_k. \tag{4.61}$$

Заметим, что эта система *заведомо разрешима* и включает только первые производные от f.

Решение (4.61) представляет собой оптимальный вектор в *линейной задаче о наименьших квадратах* вида

$$\text{найти } \min_{p \in \mathbb{R}^n} \frac{1}{2} \|J_k p + f_k\|^2. \tag{4.62}$$

Если матрица J_k имеет полный столбцовый ранг, этот вектор определен однозначно. Мы будем обозначать его через p_{GN}. Метод, в котором p_{GN} используется в качестве направления поиска, известен под названием *метод Гаусса—Ньютона*, а сам вектор p_{GN} принято называть *направлением Гаусса—Ньютона*.

Когда норма $\|Q(x_k)\|$ близка к нулю и матрица J_k *имеет полный столбцовый ранг*, направление Гаусса—Ньютона мало отличается от ньютоновского. Точнее говоря, если при достаточно малом положительном ε выполнено равенство $\|Q(x_k)\| = \varepsilon$, то тем самым обеспечено соотношение

$$\frac{\|p_N - p_{GN}\|}{\|p_N\|} = O(\varepsilon).$$

Следовательно, если норма $\|f(x^*)\|$ равна нулю и столбцы матрицы $J(x^*)$ линейно независимы, *метод Гаусса—Ньютона может достигать квадратичной скорости сходимости*, хотя при расчете p_{GN} учитываются только первые производные.

В ранних реализациях метода Гаусса—Ньютона для решения системы (4.61), как правило, использовали какую-нибудь универсальную процедуру, предварительно формируя матрицу $J_k^T J_k$ явным образом. Этот подход обладает одним серьезным недостатком: поскольку число обусловленности произведения $J_k^T J_k$ равно квадрату числа обусловленности матрицы J_k, оно может оказаться очень большим даже при неплохо обусловленной J_k, а это означает, что расчет направления поиска будет сопровождаться большими ошибками, которых можно было бы избежать. Например, работая на машине с двенадцатиразрядной точностью при числе обусловленности матрицы J_k, равном 10^6, мы вправе рассчитывать на шестиразрядную точность вычисления p_{GN}, однако

при решении системы (4.61) обычным способом в рассматриваемой ситуации можно получить ответ, не содержащий *ни одной* правильной цифры.

Угроза неоправданной потери точности при «лобовой» реализации метода Гаусса — Ньютона вполне реальна, так как для прикладных нелинейных задач о наименьших квадратах плохая обусловленность является скорее правилом, чем исключением. Это относится прежде всего к задачам, возникающим из потребностей оценивания параметров математических моделей, и связано с тем, что такие модели часто бывают некорректно определенными. Чтобы не утяжелять последствий плохой обусловленности, вектор p_{GN} надо искать как решение задачи (4.62) с использованием полной ортогональной факторизации (см. разд. 2.2.5.3) или разложения по особым числам (см. разд. 2.2.5.5).

Особенно тщательно метод Гаусса — Ньютона должен быть реализован в случаях, когда нет гарантии линейной независимости столбцов J_k. Сразу отметим, что если J_k имеет дефект ранга, то у задачи (4.62) будет целое многообразие решений. Следовательно, требуется правило выбора конкретного вектора из этого многообразия. Лучше всего брать в качестве p_{GN} решение задачи (4.62) с минимальной евклидовой нормой. Используя полную ортогональную факторизацию или разложение по особым числам, построить его нетрудно.

Каждая реализация метода Гаусса — Ньютона, в которой при линейной зависимости столбцов J_k направление p_{GN} определяется как решение (4.62) минимальной нормы, должна включать некую стратегию оценивания ранга J_k. Это означает, что в процедуре факторизации, используемой для расчета p_{GN}, надо будет предусмотреть какие-то дополнительные действия, которых нет в стандартных процедурах, предполагающих, что ранг r факторизуемой матрицы известен заранее и ее первые r столбцов линейно независимы.

К сожалению, определение понятия «ранг» в контексте приближенной арифметики с плавающей запятой теряет универсальность и становится проблемно-зависимым. Вопрос о ранге матрицы в конкретном случае остается открытым до тех пор, пока не введено явное масштабирование, т. е. не установлено, какие величины следует считать «пренебрежимо малыми».

Пример 4.10. Суть дела легко понять на примере матрицы

$$J = \begin{pmatrix} 1 & 1 \\ 0 & \varepsilon \end{pmatrix}.$$

где ε — ненулевое число, малое по модулю относительно единицы. С формальной точки зрения столбцы матрицы J линейно независимы, и ее ранг равен двум. Однако в действительности второй столбец может быть машинной версией первого, и тогда их следует признать «численно эквивалентными»; в этом случае «ранг» должен

считаться равным единице. Значит, вывод о значении ранга матрицы J будет определяться тем, следует ли пренебречь числом ε в задаче, *откуда они возникли*.

Если бы не погрешности машинной арифметики, выявить линейную зависимость столбцов матрицы во время построения ее полного ортогонального разложения было бы просто: когда факторизация осуществляется последовательными преобразованиями Хаусхолдера (см. разд. 2.2.5.3), зависимость очередного, $(k+1)$-го столбца от k предыдущих проявилась бы в равенстве нулю всех его преобразованных компонент с номерами от $k+1$ до m. Таким образом, при точных вычислениях необходимым условием точной линейной зависимости является обращение в нуль некоторых величин. Естественно возникает вопрос: если вычисления осуществляются приближенно, то не будет ли «близость» этих величин к нулю необходимым условием «приближенной» линейной зависимости? Оказывается, не будет, и это легко доказать от противного. Иначе была бы справедлива теорема о том, что плохо обусловленная треугольная матрица должна иметь хотя бы один малый диагональный элемент, а известно, что можно построить треугольную матрицу, обусловленную сколь угодно плохо и не имеющую «малых» элементов на диагонали. Итак, исходя при оценивании ранга матрицы во время ее QR-факторизации из «малости» элементов необнуленного субдиагонального блока, мы рискуем дать завышенную оценку.

Несколько проще оценивать ранг на основе разложения по особым числам: близость вырождения обязательно проявится в малости этих чисел. Однако и тут могут возникнуть определенные трудности. Возьмем, к примеру, разложение с особыми числами 1, 10^{-1}, 10^{-2}, . . ., 10^{-k}, 10^{-11}, . . ., 10^{-20}. Для него выбор порога «малости» существенно повлияет на оценку ранга, и, поскольку этот выбор в значительной степени произволен, здесь легко ошибиться.

Проблема построения оценок ранга для метода Гаусса — Ньютона важна потому, что от них очень сильно зависит его функционирование. Это видно из следующего примера.

Пример 4.11. Пусть f — двумерная функция, компоненты которой f_1 и f_2 являются в некоторой точке величинами порядка единицы, а ее матрица Якоби в этой точке равна

$$J=\begin{pmatrix}1 & 0\\ 0 & \varepsilon\end{pmatrix},$$

где ε — число, малое относительно единицы. Тогда, считая, что ранг J равен двум, в качестве направления Гаусса — Ньютона мы получим вектор

$$-\begin{pmatrix}f_1\\ f_2/\varepsilon\end{pmatrix}. \qquad (4.63)$$

Если же пренебречь величиной ε и принять единичную оценку ранга J, направлением поиска будет

$$-\begin{pmatrix} f_1 \\ 0 \end{pmatrix}. \tag{4.64}$$

Угол между векторами (4.63) и (4.64) близок к прямому, причем первый из них почти ортогонален градиенту. Таким образом, изменение оценки ранга на единицу повлекло за собой разворот направления поиска почти на девяносто градусов.

К сожалению, приходится признать, что такой стратегии оценивания ранга в методе Гаусса — Ньютона, которую можно было бы порекомендовать на все случаи жизни, не существует. Например, когда F похожа на плохо обусловленную квадратичную форму, лучше всего оценивать ранг по максимуму. Это следует из соображений близости к методу Ньютона. Однако в других случаях данная стратегия может оказаться неподходящей, так как есть доводы и в пользу занижения оценок. В частности, при почти вырожденной матрице J_k завышенная оценка ранга может привести к очень большому по норме решению (4.62), в то время как меньшая оценка часто дает вектор более разумной длины, практически не изменив минимальной невязки в (4.62). Кроме того, анализ возмущений для задачи (4.62) показывает, что относительное изменение ее точного решения пропорционально относительному изменению f_k с коэффициентом (cond (J_k)) ^{2}. Соответственно, занижая оценку ранга и уменьшая тем самым величину cond (J_k), мы можем избежать необходимости решения при расчете p_{GN} плохо обусловленной задачи.

4.7.3. МЕТОД ЛЕВЕНБЕРГА — МАРКАРДТА

Распространенной альтернативой методу Гаусса — Ньютона является *метод Левенберга — Маркардта*. В нем направление поиска определяется как решение системы уравнений вида

$$(J_k^T J_k + \lambda_k I) p_k = -J_k^T f_k, \tag{4.65}$$

где λ_k — некоторое неотрицательное число. В этом методе шаг вдоль p_k всегда полагается единичным, т. е. очередной точкой x_{k+1} будет $x_k + p_k$. Можно показать, что p_k представляет собой решение задачи на условный минимум вида

$$\text{найти } \min_{p \in R^n} \frac{1}{2} \|J_k p + f_k\|_2^2$$

$$\text{при ограничении } \|p\|_2 \leqslant \Delta,$$

где Δ — параметр, связанный с λ_k. Таким образом, метод Левенберга—Маркардта относится к классу методов доверительной окрестности, обсуждавшихся в замечаниях к разд. 4.4. Монотонное убывание минимизируемой функции достигается в нем за счет подбора «хороших» значений λ_k. При λ_k, равном нулю, p_k

будет направлением Гаусса—Ньютона, когда λ_k стремится в бесконечность, норма $\|p_k\|$ стремится к нулю и вектор p_k в пределе становится параллельным антиградиенту. Следовательно, неравенство $F(x_k+p_k)<F_k$ всегда можно обеспечить, выбрав λ_k достаточно большим.

Обозначим через $p_{LM}(\lambda_k)$ решение системы (4.65) при каких-то x_k и положительном λ_k. Оказывается, что если матрица J_k имеет дефект ранга, то в общем случае независимо от величин $\|Q(x_k)\|$ и λ_k будет справедливо соотношение

$$\frac{\|p_N - p_{LM}(\lambda_k)\|}{\|p_N\|} = O(1).$$

*4.7.4. КВАЗИНЬЮТОНОВСКИЕ МЕТОДЫ

Методы Гаусса—Ньютона и Левенберга—Марквардта опираются на предположение о доминирующей роли слагаемого $J_k^T J_k$ в правой части (4.59b); исходя из него, матрицей Q_k предлагается пренебречь. Это предположение не вполне оправданно для так называемых *задач с большой невязкой*, в которых норму $\|f(x^*)\|$ нельзя считать «малой», но которые еще не окончательно утрачивают специфики задач о наименьших квадратах. Если нормы $\|G_i(x)\|$ для $i=1, \ldots, m$ являются величинами порядка единицы, то $Q(x)$ будет существенной составляющей матрицы Гессе функции (4.58), когда $\|f(x^*)\|$ превышает *минимальное* собственное значение $J(x^*)^T J(x^*)$. При этом *максимальное* собственное число последней может существенно превосходить $\|f(x^*)\|$, и именно такие задачи мы назовем *задачами с большой невязкой*. Для них и предназначены описанные ниже специальные методы. Что же касается задач с очень большими значениями $\|f(x^*)\|$, то никакой выгоды из особой структуры минимизируемых функций для них извлечь не удается, и здесь следует обращаться к универсальным методам.

Отказавшись от вычисления вторых производных, метод решения задач с большой невязкой можно строить на основе квазиньютоновских приближений M_k неизвестной составляющей $Q(x)$ матрицы Гессе. Надо отметить, что использование точного значения другой составляющей несколько усложняет формулы по сравнению со случаем, когда квазиньютоновская аппроксимация применяется для оценки матрицы Гессе в целом. Система уравнений для расчета направления поиска при рассматриваемой организации вычислений выглядит так:

$$(J_k^T J_k + M_k)\, p_k = -J_k^T f_k,$$

а аналогичным выражению (4.30) условием, которому должно подчиняться очередное квазиньютоновское приближение M_{k+1},

будет

$$(J_{k+1}^T J_{k+1} + M_{k+1}) s_k = -g_k.$$

Здесь $s_k = x_{k+1} - x_k$ и $g_k = J_{k+1}^T f_{k+1} - J_k^T f_k$. Заметим, что матрица M_{k+1} зависит от J_{k+1}.

Процедуру вычисления матриц M_k можно строить на основе любой из формул пересчета, рассмотренных в разд. 4.5.2. В частности, если исходить из BFGS-формулы (4.36), получим

$$M_{k+1} = M_k - \frac{1}{s_k^T W_k s_k} W_k s_k s_k^T W_k + \frac{1}{y_k^T s_k} y_k y_k^T, \qquad (4.66)$$

где $W_k = J_{k+1}^T J_{k+1} + M_k$.

Если матрица $J_{k+1}^T J_{k+1} + M_k$ положительно определена, то при вычислении M_{k+1} по формуле (4.66) матрица $J_{k+1}^T J_{k+1} + M_{k+1}$ тоже будет положительно определенной. Это свойство (4.66) полезно в асимптотическом плане. Оно означает, что на поздних итерациях метода, когда J_{k+1} и J_k мало отличаются друг от друга, сохранность положительной определенности матрицы системы для расчета направления поиска обеспечена. Однако в начале счета, пока до решения далеко, матрица $J_k^T J_k + M_k$ может получаться знаконеопределенной, и соответственно должны быть предусмотрены какие-то корректировки, которые гарантировали бы, что всегда будет найдено направление спуска.

Некоторые из методов с квазиньютоновской аппроксимацией матрицы $Q(x)$, как и их универсальные прототипы, теоретически сходятся сверхлинейно. Однако свойствами сохранения положительной определенности и n-шаговой сходимости в квадратичном случае эти методы не обладают. Причина — использование квазиньютоновской техники *только для одной* составляющей матрицы Гессе. Трудности увязки точной и приближенной информации о кривизне объясняют также, почему алгоритмы с квазиньютоновской аппроксимацией $Q(x)$ на практике почти никогда не сходятся так быстро, как их универсальные аналоги.

*4.7.5. СКОРРЕКТИРОВАННЫЙ МЕТОД ГАУССА — НЬЮТОНА

Когда метод Гаусса — Ньютона, описанный в разд. 4.7.2, сходится, он сходится с хорошей скоростью и может оказаться даже эффективнее (в терминах числа обращений к процедуре расчета функций), чем обычный метод Ньютона. Однако он может и не сойтись. В этом разделе будет описан алгоритм, который следует расценивать как модификацию метода Гаусса — Ньютона с целью обеспечить сходимость для задач с большой невязкой и с дефектом ранга.

Рассмотрим разложение матрицы J_k по особым числам:

$$J_k = U \begin{pmatrix} S \\ 0 \end{pmatrix} V^T.$$

Здесь $S=\operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$—матрица, диагональные элементы которой упорядочены таким образом, что $\sigma_i \geqslant \sigma_{i+1}$ $(1 \leqslant i \leqslant n-1)$; U—ортонормальная $m \times m$-матрица; V—ортонормальная $n \times n$-матрица. Подставив это выражение для J_k в ньютоновскую систему (4.60) и сократив полученное равенство на невырожденную матрицу V, мы придем к системе вида

$$(S^2+V^TQ_kV)V^Tp_N=-S\bar{f}, \tag{4.67}$$

где через $\bar{f}$ обозначен вектор, составленный из первых n компонент m-мерного вектора U^Tf_k. Уравнение Гаусса—Ньютона (4.61) получится, если пренебречь в левой части (4.67) матрицей V^TQ_kV. Таким образом, вектор p_{GN} будет удовлетворять равенству

$$S^2V^Tp_{GN}=-S\bar{f}.$$

Если матрица S невырождена, отсюда следует, что

$$p_{GN}=-VS^{-1}\bar{f}.$$

Трудности с методом Гаусса—Ньютона возникают тогда, когда матрица Q_k «существенна», например когда матрица J_k имеет дефект ранга (т. е. S вырождена) или в задачах с большой невязкой. Чтобы избежать их, не надо пренебрегать матрицей Q_k в тех скалярных уравнениях ньютоновской системы, где составляющая $J_k^TJ_k$ не доминирует. На этом принципе и строится *скорректированный метод Гаусса—Ньютона*. В нем на каждой итерации определяется целочисленный параметр r $(0 \leqslant r \leqslant n)$, который, грубо говоря, равен количеству скалярных уравнений из (4.67) с «доминирующими» особыми числами. Такими, естественно, считаются первые r из них (поскольку особые числа упорядочены по убыванию), т. е. система (4.67) разбивается на первые r и последние $n-r$ уравнений. В соответствии с данным разбиением каждый из объектов в (4.67) тоже распадается на две части: матрица V—на матрицу V_1 (состоящую из первых r столбцов V) и на матрицу V_2 (состоящую из последних $n-r$ столбцов); матрица S—на диагональную матрицу S_1, равную $\operatorname{diag}(\sigma_1, \ldots, \sigma_r)$ (где $\sigma_r > 0$), и на матрицу $S_2=\operatorname{diag}(\sigma_{r+1}, \ldots, \sigma_n)$; вектор $\bar{f}$—на $\bar{f}_1$ и $\bar{f}_2$, состоящие из первых r и последних $n-r$ компонент $\bar{f}$.

Каждый n-мерный вектор p можно записать в виде линейной комбинации столбцов V. В частности, для ньютоновского направления p_N будет

$$p_N=V_1p_1+V_2p_2. \tag{4.68}$$

Подставив это выражение для p_N в первые r скалярных уравнений из (4.64), получим

$$(S_1^2+V_1^TQ_kV_1)p_1+V_1^TQ_kV_2p_2=-S_1\bar{f}_1.$$

Если предположить, что все коэффициенты данной системы, связанные с Q_k, имеют порядок $O(\varepsilon)$, где ε — малое число, то, пренебрегая ими, мы придем к системе, определяющей приближение вектора p_k с точностью до $O(\varepsilon)$. Ее решением является

$$\bar{p}_1 = -S_1^{-1}\bar{f}_1.$$

Отвечающий ему вектор $V_1\bar{p}_1$ называют *направлением Гаусса — Ньютона в подпространстве, порождаемом* V_1.

Подставим теперь выражение (4.68) в последние $n-r$ скалярных уравнений из (4.67). Результатом будет система вида

$$V_2^TQ_kV_1p_x + (S_2^T + V_2^TQ_kV_2)\,p_y = -S_2\bar{f}_2.$$

Замена в ней p_1 на $\bar{p}_1$ приводит к векторному уравнению для расчета приближения p_2, аппроксимирующего p_2 с точностью до $O(\varepsilon)$. Это уравнение выглядит так:

$$(S_2^T + V_2^TQ_kV_2)\,\bar{p}_2 = -S_2\bar{f}_2 - V_2^TQ_kV_1\bar{p}_1. \qquad (4.69)$$

Его решение $\bar{p}_2$ вместе с $\bar{p}_1$ определяет вектор

$$p_c = V_1\bar{p}_1 + V_2\bar{p}_2,$$

именуемый *скорректированным направлением Гаусса — Ньютона*. При $r=0$ этот вектор есть не что иное, как обычное ньютоновское направление (4.68), а при $r=n$ он будет обычным направлением Гаусса — Ньютона. Стало быть, в общем случае p_c *есть нечто промежуточное между ньютоновским направлением и направлением Гаусса — Ньютона.*

В скорректированном методе Гаусса — Ньютона параметр r обновляется на каждой итерации. Выбор его очередного значения связывается с тем, удалось ли на предыдущем шаге спуска получить удовлетворительное уменьшение функции. Правила пересчета могут быть разными, но в основу всегда закладывается тенденция к увеличению r. Выбор r существенно определяет p_c и в то же время достаточно произволен, т. е., казалось бы, здесь возникают те же проблемы, что и с оценкой ранга в методе Гаусса — Ньютона. Аналогия действительно есть, но благодаря реальной возможности подстройки значений r по результатам спуска вопрос о назначении r в скорректированном методе Гаусса — Ньютона стоит далеко не так остро, как вопрос об оценке ранга матрицы J_k в обычном методе.

Возможны три разновидности скорректированного метода Гаусса — Ньютона. Во-первых, если доступны аналитические значения вторых производных, можно использовать в расчетах сами матрицы Q_k; во-вторых, можно отказаться от вычисления аналитических вторых производных и применять конечно-разностные приближения матриц Q_kV_2, конструируемые по приращениям градиента вдоль столбцов V_2; в-третьих, можно вместо матриц Q_k брать их квазиньютоновские приближения. В любом случае решение сис-

темы (4.69) (или (4.67), если $r=0$) следует искать каким-нибудь алгоритмом типа модифицированной факторизации Холесского (см. разд. 4.4.2.2), который позволял бы при необходимости «подправлять» его, с тем чтобы в результате получалось направление спуска.

4.7.6. НЕЛИНЕЙНЫЕ УРАВНЕНИЯ

Задача о наименьших квадратах (как и любая задача безусловной минимизации) очень близка к задаче поиска корня системы нелинейных уравнений, т. е. точки x^*, в которой некоторая n-мерная векторная функция $f(x)$ обращается в нуль:

$$f(x^*)=0. \tag{4.70}$$

Близость задач определяется тем, что локальный безусловный минимум гладкой функции реализуется в точке нуля ее градиента.

Решение произвольной нелинейной системы (4.70) можно искать методом Ньютона, который для векторного случая строится в точности по той же схеме, что и для скалярного (см. разд. 4.1.1.2). Из тейлоровского разложения функции f в окрестности текущего приближения x_k мы получаем ее линейную аппроксимацию. Последняя удовлетворяет в искомой точке x^* приближенному равенству

$$f(x^*)\approx f(x_k)+J(x_k)(x^*-x_k). \tag{4.71}$$

Отсюда и выводим правило расчета ньютоновского шага p_N. Он представляет собой оценку разности x^*-x_k, получающуюся приравниванием правой части (4.71) к нулю. Таким образом, p_N будет решением аналогичной (4.1) и (4.17) системы уравнений вида

$$J(x_k)p_N=-f(x_k), \tag{4.72}$$

а очередным приближением в методе Ньютона для решения (4.70) будет $x_{k+1}=x_k+p_N$. При обычных предположениях относительно невырожденности матрицы $J(x^*)$ и близости начального приближения x_0 и x^* последовательность $\{x_k\}$, генерируемая описанным способом, квадратично сойдется к x^*.

Идея линеаризации в окрестности текущего приближения применима и в том случае, когда уравнений в системе (4.70) меньше, чем неизвестных (при этом матрица $J(x_k)$ в (4.72) является прямоугольной). Тогда векторное уравнение (4.72) будет иметь целое многообразие решений.

Задача поиска решения системы нелинейных уравнений — это классическая задача вычислительной математики, а для нее разработано большое число специальных алгоритмов. Многие среди них опираются на ньютоновское уравнение (4.72). Однако для решения (4.70) можно использовать и методы минимизации суммы квадратов — ведь ясно, что x^* из (4.70) является точкой минимума

функции (4.58). Правда, если матрица $J(x)$ может вырождаться, не исключено, что оптимизационный метод сойдется в точку x локального минимума функции (4.58), где ее градиент $J(x)^T f(x)$ равен нулю, но значение $f(x)$ окажется ненулевым. Поэтому без нужды обращаться к такому методу не следует. И все же в ряде случаев оптимизационный подход предпочтительнее, например, когда есть риск, что определение нелинейной системы (4.70) содержит погрешности, из-за которых у нее, возможно, не будет точного решения, хотя минимальная невязка заведомо мала. Что же касается выбора среди оптимизационных методов, то здесь надо учитывать два фактора — эффективность метода при $m=n$ и наличие в нем средств, позволяющих избежать трудностей, связанных с «квадратичным» ухудшением обусловленности при переходе от (4.72) к (4.61).

Замечания и избранная библиография к разделу 4.7

Общий обзор методов решения нелинейных задач о наименьших квадратах читатель найдет в работах Ленниса (1977), Рэмзина и Велина (1977).

Метод Гаусса — Ньютона подробно рассматривался многими авторами, в том числе Бен-Израэлем (1967), Флетчером (1968) и Велином (1974). О его недостатках писали Пауэлл (1972), Гилл и Мюррей (1976а). Идея применения QR-факторизации для решения линейных задач о наименьших квадратах принадлежит Бузингеру и Голубу (1965). Связь между задачей о наименьших квадратах и псевдообратными матрицами обсуждается в статье Петерса и Уилкинсона (1970). Основные сведения относительно разложений матриц по особым значениям можно почерпнуть из работ Голуба и Райнша (1971), Лоусона и Хансона (1974). Возможность применения методов сопряженных градиентов (см. разд. 4.8.5) для учета составляющей матрицы Гессе от (4.58) со вторыми производными функций f_i выше не рассматривалась; впервые на нее указал Рухе (1979), который предложил воспользоваться ею как средством «ускорения» метода Гаусса — Ньютона.

Алгоритм Левенберга — Маркардта был независимо сформулирован Левенбергом (1944) и Маркардтом (1963). Деталями его реализации занимались многие. Флетчер (1971а), например, построил процедуру подбора значения параметра λ_k в (4.65) в зависимости от отношений между фактическими изменениями функции в результате пробных шагов и ожидаемыми величинами этих изменений. Есть и другой способ реализации схемы Левенберга — Маркардта, когда подбирается Λ, а λ_k при заданном Λ вычисляется версией метода Хебдена, специально приспособленной для задач о наименьших квадратах (см. Хебден (1973), Мор (1977) и замечания к разд. 4.4).

Описания квазиньютоновских методов минимизации суммы квадратов приводятся в работах Денниса (1973), Бетса (1976), Денниса, Гэя и Уэлша (1977), Гилла и Мюррея (1978а). Авторами скорректированного метода Гаусса — Ньютона являются Гилл и Мюррей (1978а).

Среди прикладных задач о наименьших квадратах нередко встречаются задачи с функциями $\{f_i\}$ специального вида. Например, может оказаться, что все $\{f_i\}$ линейны по части переменных, либо структура матрицы Якоби будет такой, что задачу удастся преобразовать в несколько независимых подзадач меньшей размерности. В таких случаях имеет смысл обращаться к специальным методам. В их числе можно упомянуть методы решения сепарабельных задач о наименьших квадратах, впервые предложенные Голубом и Перейрой (1973). Другие ссылки на методы этого типа можно найти в работах Кауфмана и Перейры (1978), Рухе и Ведина (1980).

По методам решения систем нелинейных уравнений существует обширная литература; см., например, Бройден (1965), Пауэлл (1970b), Болт (1975), Брент (1973b), Гэй и Шнабель (1978), Море (1977).

4.8. МЕТОДЫ РЕШЕНИЯ ЗАДАЧ БОЛЬШОЙ РАЗМЕРНОСТИ

Когда размерность n аргумента минимизируемой функции становится очень большой, при обращении к обычным реализациям универсальных схем предыдущих разделов могут возникнуть две трудности: во-первых, время вычислений может вырасти настолько, что браться за них станет бессмысленным; во-вторых, что более критично, ни в оперативной, ни во вспомогательной памяти машины может не найтись места, достаточного для хранения матриц, используемых при расчете направлений поиска. Это не означает, однако, что к большим задачам быстросходящиеся схемы минимизации не применимы.

К счастью, природа многих прикладных задач большой размерности позволяет строить для них специальные эффективные реализации таких схем. Типичной чертой матриц Гессе встречающихся на практике функций большого числа переменных является очень низкий процент ненулевых элементов. Вообще отношение числа ненулей в матрице к общему числу ее элементов принято называть *заполненностью*. Матрица со значительной долей ненулей называется *плотной* или *сильно заполненной*; если же процент ненулей мал, то говорят, что матрица *заполнена слабо* или *разрежена*. Для реальных задач безусловной минимизации заполненность матриц Гессе имеет тенденцию падать с увеличением n, причем обычно абсолютное количество ненулей возрастает с n лишь линейно.

Наряду с разреженностью матрица Гессе в большой задаче почти всегда характеризуется фиксированной *структурой*. Это значит, что ее нулевые и ненулевые элементы не разбросаны как попало, а всегда (в любой точке) занимают постоянные позиции,

которые известны заранее и определяются природой минимизируемой функции.

В данном разделе будет показано, каким образом разреженность и структурированность матриц Гессе удается использовать для сокращения потребностей в памяти и вычислениях. На чем можно сэкономить, понятно: не надо выполнять бессмысленных умножений на нуль и не надо выделять память для хранения нулевых блоков матриц.

4.8.1. ДИСКРЕТНЫЕ МЕТОДЫ НЬЮТОНА ДЛЯ ФУНКЦИЙ С РАЗРЕЖЕННЫМИ МАТРИЦАМИ ГЕССЕ

Среди всевозможных методов минимизации с применением самой матрицы Гессе или ее приближений дискретные методы Ньютона лучше всего приспосабливаются к большим задачам с разреженными и структурированными матрицами.

При построении конечно-разностной аппроксимации плотной матрицы Гессе в качестве *конечно-разностных векторов* используются столбцы единичной матрицы, и расчет каждого столбца искомого приближения требует вычисления градиента в одной дополнительной точке (см. разд. 4.5.1). Если же матрица Гессе $G(x)$ разрежена и структура ее заполнения известна, то, подобрав конечно-разностные векторы специальным образом, полную аппроксимацию для $G(x)$ можно построить ценой вычисления градиентов в значительно меньшем чем n числе точек.

Пусть, например, $G(x)$ является трехдиагональной матрицей вида

$$G = \begin{pmatrix} \times\times & & & & \\ \times\times\times & & & & \\ & \times\times\times & & & \\ & & \times\times\times & & \\ & & & \ddots & \\ & & & \times\times & \\ & & & & \times\times\times \\ & & & & \times\times \end{pmatrix}$$

и подсчитаны векторы

$$y_i = \frac{1}{h\|z_i\|_2}\left(g(x_k + hz_i) - g(x_k)\right), \quad i = 1, 2,$$

где $z_1 = (1, 0, 1, 0, \ldots)^T$, $z_2 = (0, 1, 0, 1, \ldots)^T$, а h — некоторый подходящий конечно-разностный интервал. Легко проверить, что y_1 будет приближением суммы нечетных столбцов $G(x)$, а y_2 — суммы четных. Таким образом, обозначив через $y_{i,j}$ компоненту вектора y_i с номером j и введя аналогичные обозначения для компонент z_i, получим

$$y_{1,1} \approx \frac{\partial^2 F}{\partial x_1^2}; \quad y_{2,1} \approx \frac{\partial^2 F}{\partial x_1 \partial x_2}; \quad y_{1,2} \approx \frac{\partial^2 F}{\partial x_2 \partial x_3} + \frac{\partial^2 F}{\partial x_2 \partial x_4}$$

и т. д. Отсюда, в частности, видно, что

$$y_{1,2} - y_{2,1} \approx \frac{\partial^2 F}{\partial x_2 \partial x_3}.$$

Подобные же формулы определят и оценки остальных производных. Таким образом, для аппроксимации всех элементов трехдиагональной матрицы G требуется только два дополнительных вычисления градиента *независимо от значения n*.

Возможности применения рассматриваемой техники не ограничиваются ленточными матрицами. Чтобы убедиться в этом, возьмем в качестве еще одного примера матрицу «стреловидной» структуры:

$$G = \begin{pmatrix} \times & & & & \times \\ & \times & & & \times \\ & & \times & & \times \\ & & & \ddots & \vdots \\ & & & \times & \times \\ \times & \times & \times & \cdots & \times \end{pmatrix}.$$

В силу симметрии для построения ее конечно-разностной аппроксимации $\hat{G}$ также достаточно двух дополнительных вычислений градиента. Здесь в роли конечно-разностных векторов надо использовать $z_1 = (1, 1, \ldots, 1, 0)^T$ и $z_2 = (0, 0, \ldots, 0, 1)^T$. Приращение градиента вдоль z_1 даст первые $n-1$ диагональных элементов матрицы $\hat{G}$; оставшиеся ненули $\hat{G}$ определятся по конечным разностям вдоль z_2.

Процедуры решения задач минимизации функций большого числа переменных с разреженными матрицами Гессе, построенные на основе дискретного метода Ньютона, обычно работают в два этапа: сначала запускается алгоритм предварительной обработки, который, перенумеровывая переменные задачи, преобразует исходную структуру заполнения матрицы Гессе к виду, позволяющему уменьшить количество вычислений градиента при оценке вторых производных; затем начинается собственно минимизация. Поскольку сокращение количества вычислений градиента для расчета $\hat{G}_k$ проявится столько раз, сколько будет выполнено итераций спуска, экономить на предварительной обработке задачи и поиска подходящего набора конечно-разностных векторов не следует.

Научиться эффективно вычислять конечно-разностную аппроксимацию $\hat{G}_k$ разреженной матрицы Гессе,— значит, сделать первый шаг построения дискретного ньютоновского алгоритма решения большой задачи. Нужен и второй — научиться решать большую систему линейных уравнений вида

$$\hat{G}_k p_k = -g_k \qquad (4.73)$$

с разреженной матрицей $\hat{G}_k$, причем, как и в случае с плотной матрицей, алгоритм должен быть устроен так, чтобы при необходи-

мости автоматически осуществлялась некая модификация, гарантирующая, что будет найдено удовлетворительное направление спуска. Здесь возникает одна серьезная трудность. Дело в том, что решение линейной системы обычно ищут путем преобразования ее матрицы какой-нибудь процедурой факторизации. При этом те позиции, где в исходной матрице стояли нули, у факторов генерируемого разложения могут оказаться ненулевыми. Происходит так называемое *заполнение* — рождение новых ненулевых элементов. Когда матрица системы записана в машине компактно (без нулей), для этих элементов придется выделять дополнительную память, и, если их будет слишком много, ресурсов может не хватить.

Итак, разреженность системы (4.73) еще не гарантирует возможности ее эффективного решения. Например, в модифицированной факторизации Холесского (см. разд. 4.4.2.2) столбцы треугольного фактора L_k будут линейными комбинациями столбцов $\hat{G}_k$, и, следовательно, матрица L_k может оказаться довольно плотной даже при очень разреженной $\hat{G}_k$. В данном случае воспользоваться факторизацией Холесского не удастся. Чтобы заполнение в процессе какой-то факторизации матрицы $\hat{G}_k$ было приемлемым, она должна иметь определенную структуру.

Когда расположение ненулей в матрице $\hat{G}_k$ таково, что чрезмерное заполнение в процессе факторизации Холесского исключается, использующий ее дискретный модифицированный метод Ньютона оказывается весьма эффективной процедурой. Его скорость сходимости, как правило, квадратична (при оговорке относительно предельной точности, сделанной в разд. 4.5.1), и сходится он обычно к точке сильного локального минимума.

*4.8.2. КВАЗИНЬЮТОНОВСКИЕ МЕТОДЫ ДЛЯ ФУНКЦИЙ С РАЗРЕЖЕННЫМИ МАТРИЦАМИ ГЕССЕ

В разд. 4.5.2.3. говорилось о том, что некоторые квазиньютоновские формулы для расчета приближенной матрицы Гессе могут быть получены из решения задачи минимизации нормы уклонения последующего приближения от предыдущего при определенных ограничениях, среди которых всегда содержится квазиньютоновское условие (4.30). В частности, была представлена задача (4.48), решение которой дает PSB-формулу. В этом разделе мы обсудим возможности применения оптимизационного подхода для вывода формул квазиньютоновской аппроксимации, обеспечивающих приближениям матрицы Гессе ту же структуру заполненности, что и у нее самой.

Когда матрица Гессе минимизируемой функции разрежена и структура ее заполненности известна, формулы расчета квазиньютоновских приближений, сохраняющих эту структуру, можно получить, введя в задачу (4.48) требование равенства нулю соответ-

стоящих элементов поправочной матрицы. Если обозначить через $\mathcal{N}$ набор индексов $\{(i, j)|G_{ij}=0\}$, то обобщением (4.48) в указанном смысле будет задача вида

$$\text{найти } \min\|U\|=\sum_{i=1}^{n}\sum_{j=1}^{n}U_{ij}^2 \tag{4.74}$$

$$\text{при ограничениях } Us_k=y_k-B_ks_k,$$
$$U=U^T,$$
$$U_{ij}=0, \quad (i, j)\in\mathcal{N}.$$

Ее решением является матрица U_k, гарантирующая наличие у B_{k+1} той же структуры заполненности, что и у матрицы Гесса, если только этой структурой обладает B_k. Явные формулы расчета U удобно записываются через векторы $\{\sigma^{(j)}\}$, которые определены следующим правилом: чтобы получить вектор $\sigma^{(j)}$, надо взять вектор $s^{(j)}$ и обнулить все его компоненты, соответствующие нулевым позициям j-го столбца матрицы G_k. Довольно длинные и сложные выкладки показывают, что

$$U_k=\sum_{j=1}^{n}\lambda_j(e_j\sigma^{(j)T}+\sigma^{(j)}e_j^T). \tag{4.75}$$

Здесь e_j есть j-й единичный вектор, а числа λ_j представляют собой множители Лагранжа, связанные с оптимумом задачи (4.74). Составленный из них вектор λ является решением линейной системы уравнений

$$Q\lambda=y_k-B_ks_k, \tag{4.76}$$

где

$$Q=\sum_{j=1}^{n}(\sigma_j^{(j)}\sigma^{(j)}+[\sigma^{(j)}]_j e_j)e_j^T.$$

Матрица Q симметрична и имеет ту же структуру заполненности, что и B_k; она положительно определена в том и только том случае, когда $[\sigma^{(j)}]_j>0$ для всех j. Заметим, что ранг поправочной матрицы U_k (4.75), вообще говоря, равен n и что линейную систему (4.76) придется на каждом шаге решать «с нуля». Следует также иметь в виду, что сохранение положительной определенности квазиньютоновских приближений при использовании поправок (4.75) не гарантировано.

На каждой итерации рассматриваемой квазиньютоновской схемы кроме системы (4.76) придется решать систему (4.28) и тоже «с нуля», так как поправка (4.75) имеет высокий ранг (в отличие от поправок в обычных квазиньютоновских методах). Таким образом, итерация схемы требует решения двух больших систем линейных уравнений с разреженными матрицами, причем для записи матрицы Q вспомогательной системы (4.76) в памяти машины придется выделить специальное место.

Исходя из полученного явного вида решения задачи (4.74), можно предложить *универсальный* прием модификации квазиньютоновских методов, позволяющий любой из них преобразовать в метод с сохранением структуры заполненности. Для того чтобы построить разреженную поправочную матрицу, ориентируясь на какую-нибудь обычную квазиньютоновскую формулу, надо (i) вычислить матрицу U, структура заполненности которой та же, что и у матрицы Гессе, а ненулевые элементы определены по обычной формуле; (ii) решить линейную систему уравнений

$$Q\lambda = y_k - B_k s_k - U s_k;$$

(iii) вычислить для найденного λ матрицу (4.75) и сложить ее с U (результат и есть искомая разреженная поправка). В данном случае (4.75) будет решением задачи минимизации нормы корректирующей добавки к обычной квазиньютоновской матрице U_k, обнуляющей те элементы последней, которым отвечают нули матрицы Гессе.

Исследования по квазиньютоновским методам с сохранением структуры заполненности пока находятся в ранней стадии, и здесь еще многое предстоит сделать. Что же касается методов этого типа, именующихся на сегодня, то эксперименты показывают, что по числу обращений к процедуре расчета функции, требуемых для решения задачи, они проигрывают дискретным ньютоновским алгоритмам. Кроме того, они лишены трех важных достоинств обычных квазиньютоновских методов: они не сохраняют положительную определенность, не позволяют пересчитывать матрицы B_k в обратимой форме и не являются более экономными, чем методы ньютоновского типа, в отношении затрат на расчет направления поиска.

4.8.3. Методы сопряженных градиентов

Методы сопряженных градиентов относятся к классу алгоритмов минимизации, в которых вычисление направлений поиска не предполагает решения каких-либо систем линейных уравнений. Это принципиально отличает их от всех методов, обсуждавшихся до сих пор. Они нужны для задач, к которым последние неприменимы из-за того, что фигурирующие в них матрицы оказываются слишком большими или слишком сильно заполненными.

4.8.3.1. Квадратичные функции. Рассмотрим задачу поиска минимума квадратичной функции:

$$\Phi(x) = c^T x + \frac{1}{2} x^T G x, \qquad (4.77)$$

где G — симметричная положительно определенная матрица. Пусть x_k — текущее приближение и известны $k+1$ линейно независимых векторов $p_0, p_1, \ldots, p_k$, порождающих некоторое подпространство $\mathcal{P}_k$. Найдем выражение для точки минимума Φ на многообразии $x_k + \mathcal{P}_k$. Для этого введем матрицу P_k, столбцами которой явля-

ются $\{p_i\}$. Это позволит представить задачу минимизации Φ на многообразии $x_k+\mathcal{P}_k$ в виде

$$\text{найти } \min_{w\in R^{n-k}} \Phi(x_k+P_k w).$$

В соответствии с определением Φ оптимальное значение аргумента w будет точкой минимума квадратичной функции

$$w^T P_k^T g_k + \frac{1}{2} w^T P_k^T G P_k w, \tag{4.78}$$

где $g_k=\nabla\Phi(x_k)=c+Gx_k$. Эта точка вычисляется по формуле

$$w=-(P_k^T G P_k)^{-1} P_k^T g_k,$$

и, следовательно, минимум Φ на многообразии $x_k+\mathcal{P}_k$ будет достигаться в

$$x_{k+1}=x_k-P_k(P_k^T G P_k)^{-1} P_k^T g_k. \tag{4.79}$$

На равенство (4.79) можно смотреть как на формулу задания пошаговой процедуры минимизации Φ. Эта процедура имеет несколько интересных свойств. Прежде всего легко убедиться, что градиент g_{k+1} функции Φ в точке x_{k+1} будет ортогонален векторам $\{p_i\}$ (столбцам матрицы P_k). В самом деле,

$$P_k^T g_{k+1}=P_k^T(c+Gx_{k+1})=P_k^T g_k-P_k^T G P_k(P_k^T G P_k)^{-1} P_k^T g_k=0,$$

т. е. $g_{k+1}^T p_i=0$, $i=0, \ldots, k$. Поскольку мы считаем, что точки x_j, $j=1, \ldots, k$, тоже определены минимизацией Φ на соответствующих многообразиях, полученные равенства позволяют утверждать, что

$$g_l^T p_i=0, \quad j>l, \tag{4.80}$$

для всех $j=1, \ldots, k$, $k+1$. Отсюда в свою очередь следует, что (4.79) можно переписать так:

$$x_{k+1}=x_k+\gamma P_k(P_k^T G P_k)^{-1} e_k, \tag{4.81}$$

где e_k есть k-й столбец единичной матрицы, а $\gamma=-g_k^T p_k$.

Формула (4.81) сильно упростится, если матрица $P_k^T G P_k$ диагональна. Это будет, если векторы $\{p_i\}$, $i=0, \ldots, k$, *взаимно сопряжены* относительно G, т. е. если

$$p_i^T G p_j=0, \quad i\neq j, \quad i, j=0, 1, \ldots, k. \tag{4.82}$$

В данном случае (4.81) преобразуется к виду

$$x_{k+1}=x_k+\alpha_k p_k, \tag{4.83}$$

где $\alpha_k=-g_k^T p_k/p_k^T G p_k$. Легко проверить, что α_k есть не что иное, как длина шага в точку минимума Φ вдоль p_k. Таким образом, процедура последовательной минимизации Φ на многообразиях, определяемых сопряженными векторами, принимает стандартную форму метода спуска.

По определению Φ для любых точек x_{i+1} и x_i, связанных равенством

$$x_{i+1}=x_i+\alpha_i p_i,$$

имеем

$$g_{i+1}-g_i=G(x_{i+1}-x_i)=\alpha_i Gp_i. \tag{4.84}$$

Обозначив разность $g_{i+1}-g_i$ через y_i, получим, что *условие сопряженности* $p_i^TGp_j=0$ эквивалентно *условию ортогональности* $y_i^Tp_j=0$. Это понадобится нам в дальнейшем.

Мы установили, что среди методов минимизации квадратичных функций (4.77), укладывающихся в общую схему спуска из разд. 4.3.1, есть метод, k-я итерация которого приводит в точку минимума Φ на многообразии $x_{k-1}+\mathcal{P}_{k-1}$. Теоретически этот метод конечен: при точных вычислениях он должен найти решение не более чем за n шагов, поскольку многообразие $x_{n-1}+\mathcal{P}_{n-1}$ совпадает со всем пространством значений аргумента x, и, следовательно, если оптимальная точка не была найдена ранее, то она во всяком случае будет найдена на n-м шаге. Чтобы окончательно определить метод, надо задать правило построения взаимно сопряженных направлений $\{p_i\}$. Мы покажем сейчас, что направление p_k для очередной итерации можно вычислять как линейную комбинацию градиента в текущей точке x_k и предыдущего направления p_{k-1}. Доказательство проведем по индукции.

Пусть p_0 совпадает с антиградиентом $-g_0$ в начальной точке x_0 и k шагов метода (4.83) спуска по взаимно сопряженным направлениям $p_0, p_1, \ldots, p_{k-1}$ уже выполнены, причем каждый из векторов p_i был суммой антиградиента $-g_i$ и линейной комбинации от $p_0, p_1, \ldots, p_{i-1}$. Направление p_k для очередного шага тоже будем искать в виде

$$p_k=-g_k+\sum_{i=1}^{k-1}\beta_{ki}p_i. \tag{4.85}$$

Из этой формулы следует, что g_k есть линейная комбинация направлений $p_0, p_1, \ldots, p_k$. Аналогично и g_i для $i<k$ будет линейной комбинацией от $p_0, p_1, \ldots, p_i$. Поэтому, учитывая (4.80), можно утверждать, что

$$g_k^Tg_i=0, \quad i<k. \tag{4.86}$$

Чтобы выяснить, какие коэффициенты β_{kj} обеспечивают сопряженность p_k всем предыдущим направлениям p_i, умножим (4.85) на p_i^TG:

$$\begin{aligned} p_i^TGp_k&=-p_i^TGg_k+\sum_{j=0}^{k-1}\beta_{kj}p_i^TGp_j=\\ &=-\frac{1}{\alpha_i}(g_{i+1}-g_i)^Tg_k+\beta_{ki}p_i^TGp_i. \end{aligned} \tag{4.87}$$

Здесь использованы равенство (4.84) и предположение о взаимной сопряженности направлений $p_0, \ldots, p_{k-1}$. В силу (4.86) при $i<k-1$ первое слагаемое правой части (4.87) обращается в нуль, и поэтому, чтобы обеспечить взаимную сопряженность p_k и p_i при $i<k-1$, надо взять соответствующие β_{ki} нулевыми. Таким образом, ненулевым среди множителей β_{kj} в (4.85) может быть только $\beta_{k,k-1}$. Впредь будем обозначать его через β_{k-1}. Множитель β_{k-1} надо подобрать так, чтобы вектор p_k был сопряжен к p_{k-1}. Для этого достаточно удовлетворить условию ортогональности $y_{k-1}^T p_k = 0$. Соответственно, умножив (4.85) на y_{k-1}^T и приравняв левую часть полученного равенства нулю, находим, что искомый множитель должен быть решением уравнения

$$0 = -y_{k-1}^T g_k + \beta_{k-1} y_{k-1}^T p_{k-1},$$

т. е.

$$\beta_{k-1} = \frac{y_{k-1}^T g_k}{y_{k-1}^T p_{k-1}}. \tag{4.88}$$

Итак, в качестве векторов p_k для (4.83) следует брать

$$p_k = -g_k + \beta_{k-1} p_{k-1}, \tag{4.89}$$

где β_{k-1} вычисляются по формуле (4.88). Взаимная ортогональность градиентов и определение p_{k-1} позволяют предложить еще две внешне отличные от (4.88), но теоретически эквивалентные ей формулы расчета β_{k-1}:

$$\beta_{k-1} = \frac{y_{k-1}^T g_k}{\|g_{k-1}\|^2}, \quad \beta_{k-1} = \frac{\|g_k\|^2}{\|g_{k-1}\|^2}.$$

4.8.3.2. Линейный метод сопряженных градиентов. На описанный в предыдущем разделе метод минимизации квадратичных функций можно смотреть и как на метод решения систем линейных уравнений с положительно определенными матрицами—ведь, отыскав минимум для $c^T x + \frac{1}{2} x^T C x$, мы тем самым найдем решение системы

$$Cx = -c. \tag{4.90}$$

Именно в таком качестве он был опубликован впервые. Когда о методе сопряженных градиентов говорят как о процедуре решения линейных систем, к его названию добавляют эпитет «линейный». Замечательная особенность данного метода состоит в том, что для поиска решения системы ему требуются только произведения ее матрицы на некоторые векторы и не нужна сама матрица. Это означает, например, что если C представлена произведением $R^T R$, то при обращении к методу сопряженных градиентов явно формировать элементы этого произведения не понадобится.

В описаниях линейного метода сопряженных градиентов суммы $c+Cx_j$ (градиенты функции (4.77)) принято обозначать через r_j

и называть невязками. На k-й итерации последовательно вычисляются

$$\begin{aligned} p_k &= -r_k + \beta_{k+1} p_{k-1}; \\ \alpha_k &= \frac{\|r_k\|^2}{p_k^T G p_k}; \\ x_{k+1} &= x_k + \alpha_k p_k; \\ r_{k+1} &= r_k + \alpha_k G p_k; \\ \beta_k &= \frac{\|r_{k+1}\|^2}{\|r_k\|^2}. \end{aligned} \tag{4.91}$$

Чтобы запустить метод, надо задать начальное приближение x_0 и отвечающую ему невязку $r_0 = c + Gx_0$. Величины β_{-1} и p_{-1}, фигурирующие в (4.91) при $k=0$, полагаются нулевыми.

Теоретически линейный метод сопряженных градиентов должен найти точное решение системы (4.90) за число итераций, не превышающее размерности вектора неизвестных. Точнее говоря, при вычислениях без погрешностей решение будет получено за $m \leqslant n$ шагов, где m — число различных собственных значений матрицы G. Таким образом, метод можно считать *конечным*. Однако если исходить из практических соображений, то его скорее следует отнести к классу *итерационных* методов, поскольку при реализации на машине он редко дает удовлетворительное решение за n шагов. Это связано с тем, что из-за неточностей машинной арифметики сопряженность вычисляемых направлений p_k быстро теряется. Когда собственные значения матрицы G_k распадаются на группы почти одинаковых чисел, линейный метод сопряженных градиентов может сойтись очень быстро; если же подобной структуры собственных значений нет, для определения хорошего численного приближения ему может потребоваться значительно больше, чем n итераций.

4.8.3.3. *Нелинейные функции общего вида.* Метод сопряженных градиентов из разд. 4.8.3.1 легко обобщается на задачи минимизации нелинейных функций общего вида. Для этого надо заменить используемую в нем формулу расчета α_k какой-нибудь процедурой одномерного поиска и уточнить, будет ли направление p_k всегда вычисляться по формуле (4.89) или допускаются периодические отступления. Последние принято называть *рестартами* или *восстановлениями*. Можно, например, принимая во внимание n-шаговую теоретическую сходимость в квадратичном случае, через каждые n шагов отказываться от направления (4.89) и брать в качестве очередного p_k антиградиент $-g_k$. Соответствующий метод называют *традиционным*.

Траектория традиционного метода сопряженных градиентов в задаче минимизации функции Розенброка (пример 4.2) изображена на рис. 4о. Хотя этот метод не предназначен для решения задач столь малой размерности, приведенная иллюстрация все же полезна,

поскольку показывает его циклический характер. Отметим, что он работает значительно лучше, чем наискорейший спуск, и хуже, чем квазиньютоновский метод с BFGS-формулой (см. рис. 4j и 4m).

Определение точного (в машинном смысле) минимума $F(x)$ вдоль p_k в общем случае требует больших усилий. Поэтому многие реализации традиционного метода сопряженных градиентов допускают

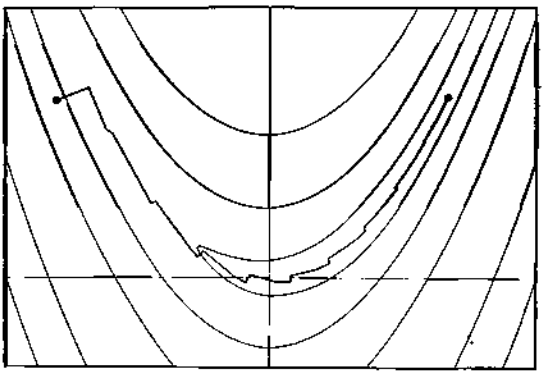

Рис. 40. Траектории поиска минимума функции Розенброка методом сопряженных градиентов.

приближенный одномерный поиск. Здесь, однако, приходится специально позаботиться о том, чтобы вектор p_k всегда был направлением спуска. При точном одномерном поиске это условие соблюдается автоматически — оно оказывается следствием вытекающего из (4.89) равенства

$$g_k^T p_{k-1} = -g_k^T g_k + \beta_{k-1} g_k^T p_{k-1}.$$

Если минимум вдоль p_{k-1} найден точно, скалярное произведение $g_k^T p_{k-1}$ равно нулю, и поэтому величина $g_k^T p_k$, совпадающая с $-g_k^T g_k$, будет отрицательной. Если же одномерный поиск ведется приближенно, то произведение $\beta_{k-1} g_k^T p_{k-1}$ может оказаться положительным и превосходящим $g_k^T g_k$. Тогда p_k будет указывать в сторону возрастания $F(x)$. Чаще всего на этот случай предусматривают возможность внеочередного рестарта, заменяя неудачное направление антиградиентом. Надо только иметь в виду, что, если такие замены из-за слишком примитивной организации одномерного поиска станут очень частыми, метод практически превратится в алгоритм наискорейшего спуска, т. е. станет совершенно неэффективным.

Проблему обеспечения правильной ориентации векторов p_k при неточном одномерном поиске можно решать и по-другому, вводя в алгоритм выбора шага дополнительное условие. Формулируется оно следующим образом. Обозначив через $\bar{g}_{k+1}$, p_{k+1} и $\bar{\beta}_k$ значения величин g_{k+1}, p_{k+1} и β_k в точке $x_k + \alpha^l p_k$, где α^l — пробный шаг одномерного поиска, будем считать, что α^l может претендовать на роль α_k только в том случае, если при некотором малом положительном σ выполнено неравенство

$$\bar{g}_{k+1}^T \bar{p}_{k+1} \geqslant \sigma \|\bar{g}_{k+1}\|_2^2 \|\bar{p}_{k+1}\|_2. \qquad (4.92)$$

Если же оно не выполняется ни при одном пробном α^l, то одномерный поиск будем вести до тех пор, пока не будет найден точный минимум $F(x)$ вдоль p_k. При таком ограничении на выбор α_k правильная ориентация p_{k+1} гарантирована.

Поскольку направление p_{k+1} в методе сопряженных градиентов определяется просто, проверка неравенства (4.92) лишь незначительно усложнит процедуру одномерного поиска. Заметим кстати, что формировать p_{k+1} явным образом не обязательно, так как величины $\bar{g}_{k+1}^T \bar{p}_{k+1}$ и $\|p_{k+1}\|$ легко вычисляются по $\bar{g}_{k+1}^T \bar{g}_{k+1}$, β_k и $\bar{g}_{k+1}^T p_k$. Ну а если критерию приемлемости шага удается удовлетворить с первой попытки, то о дополнительных затратах на проверку (4.92) вообще говорить не приходится: в данном случае все связанные с ней величины все равно пришлось бы вычислять для других целей.

4.8.3.4. Методы сопряженных градиентов с рестартами. Когда реализация схемы сопряженных градиентов предполагает периодические возвраты (рестарты) к направлениям наискорейшего спуска, то выигрыш на итерации рестарта почти всякий раз оказывается меньше того, который был бы получен, если бы рестарта не было. Это связано с тем, что антиградиент, как правило, является неудачным направлением спуска. Соответственно возникает желание переключаться не на него, а на какое-то другое направление, например начинать очередной цикл из n итераций с последнего направления предыдущего цикла. Однако здесь нужна некоторая осторожность.

Если в нелинейном методе сопряженных градиентов взять в качестве начального направления p_0 произвольный вектор, то взаимной сопряженности между генерируемыми им направлениями, вообще говоря, не будет, так как g_0 не будет их линейной комбинацией. Чтобы сохранить сопряженность, вместо (4.89) надо использовать формулу вида

$$p_k = -g_k + \beta_{k-1} p_{k-1} + \gamma_k p_0, \qquad (4.93)$$

где $\gamma_k = g_k^T g_k / g_0^T p_0$, а β_{k-1} по-прежнему вычисляется по формуле (4.88). Обобщение данного подхода для задач с неквадратичными

функциями состоит в том, чтобы на очередном цикле из n итераций строить направления p_k, $k=0, \ldots, n-1$, по правилу

$$p_k = -g_k + \beta_{k-1} p_{k-1} + \gamma_k p_t \qquad (4.94)$$

где $\gamma_k = y_t^T g_k / y_t^T p_t$, а p_t — последнее направление предшествующего цикла. Вектор p_t принято называть *направлением рестарта*.

Следует отметить, что p_k из (4.94) может указывать в сторону возрастания $F(x)$ даже в том случае, если подзадача одномерной минимизации вдоль p_{k-1} решена точно. Значит, обращаясь к (4.94), надо предусмотреть возможность аварийных рестартов — замены неприемлемого p_k из (4.94) вектором, вычисленным по обычной формуле. При этом «хорошим» направлением p_k естественно считать такое, вдоль которого минимизируемая функция убывает «достаточно быстро». Например, можно ввести аналогичное (4.92) условие вида

$$-g_k^T p_k \geqslant \rho \|p_k\|_2 \|g_k\|_2,$$

где ρ — некоторое положительное число, и каждый раз, когда для p_k из (4.94) оно нарушится, начинать новый цикл итераций с p_{k-1} в качестве направления рестарта и с p_k, пересчитанным в соответствии с формулой (4.89).

Когда одномерный поиск осуществляется точно, правильная ориентация p_k при описанном способе аварийного рестарта гарантирована. Если же используется приближенный поиск, то на итерациях рестарта (как аварийного, так и обычного) точка минимума вдоль p_t должна уточняться на основе критерия (4.92).

4.8.3.5. Сходимость. Традиционный метод сопряженных градиентов сходится в тех же предположениях, что и метод наискорейшего спуска. Для достаточно широкого класса функций теоретическая скорость его сходимости при точной одномерной минимизации оказывается *n-шаговой сверхлинейной*. Это означает, что

$$\lim_{j \to \infty} \frac{\|x_{nj+n} - x^*\|}{\|x_{nj} - x^*\|} = 0.$$

Доказательство n-шаговой сверхлинейной сходимости традиционного метода существенно использует наличие рестартов. Алгоритмы без рестартов для большинства функций сходятся линейно. Однако это — теория, а на практике все выглядит несколько иначе. Наш опыт показывает, что из-за ошибок округления реальная скорость сходимости метода сопряженных градиентов почти всегда линейна независимо от того, применяются рестарты или нет. Исключения бывают только в очень специальных случаях, например когда минимизируемая функция $F(x)$ представляет собой квадратичную форму с хорошо обусловленной матрицей. Впрочем, это и не суть важно, поскольку при больших n свойство «n-шаговой сверхлинейной сходимости» особой ценности не имеет. Оно не исклю-

чает, что для определения оптимума с приемлемой точностью понадобится число итераций, кратное n с солидным множителем, а для большой задачи иногда и $2n$ итераций уже недопустимо много. На наш взгляд, применение метода сопряженных градиентов следует считать успешным, когда удовлетворительное решение удалось получить менее чем за $5n$ шагов.

Если собственные значения матрицы Гессе функции F в точке ее минимума распадаются на небольшое число групп, в каждой из которых их разброс невелик, метод сопряженных градиентов может найти хорошее решение значительно быстрее, чем за n итераций. Так бывает для некоторых штрафных функций, используемых при решении задач с нелинейными ограничениями (см. разд. 6.2.1). Что же касается функций, не обладающих указанным свойством, то для их минимизации обычно требуется от n до $5n$ итераций, причем чаще всего хватает $2n$ итераций. В заключение следует сказать, что, хотя схема сопряженных градиентов далека от идеала, на сегодня она является *единственным* разумным универсальным средством решения задач безусловной минимизации с очень большим числом переменных.

*4.6.4. КВАЗИНЬЮТОНОВСКИЕ МЕТОДЫ С ОГРАНИЧЕННОЙ ПАМЯТЬЮ

В этом разделе мы рассмотрим методы, в которых правильная ориентация направлений поиска обеспечивается при значительно менее жестких требованиях к выбору шага, чем в нелинейном методе сопряженных градиентов. Они называются *квазиньютоновскими с ограниченной памятью*. Объединяет их идея использования в качестве p_k произведения $-Mg_k$, где M — положительно определенная матрица, полученная из единичной ограниченным числом квазиньютоновских корректировок по какой-нибудь формуле для аппроксимации *обращенной матрицы Гессе*. При этом M никогда не формируется явно. В памяти машины хранятся только векторы, определяющие квазиньютоновские поправки, и этого вполне достаточно для вычисления произведений M на g_k.

Чтобы задать конкретный метод рассматриваемого типа, надо выбрать конкретную квазиньютоновскую формулу и зафиксировать число запоминаемых поправочных векторов. Мы приведем в качестве примера «одношаговый метод», построенный на основе BFGS-формулы. В нем матрица M для очередной итерации получается из единичной однократным пересчетом в соответствии с результатами предыдущего шага. Вектор p_k в этом методе вычисляется так:

$$p_k = -g_k + \frac{1}{y_{k-1}^T s_{k-1}} (s_{k-1}^T g_k y_{k-1} + y_{k-1}^T g_k s_{k-1}) - \frac{s_{k-1}^T g_k}{y_{k-1}^T s_{k-1}} \left(1 + \frac{y_{k-1}^T y_{k-1}}{y_{k-1}^T s_{k-1}}\right) s_{k-1}. \qquad (4.95)$$

Нетрудно проверить, что при точной одномерной минимизации направления p_k из (4.95) оказываются теми же, что и в методе сопряженных градиентов: когда $s_{k-1}^T g_k = 0$, формула (4.95) переходит в (4.89). Таким образом, квазиньютоновские методы с ограниченной памятью могут генерировать сопряженные направления, причем точный одномерный поиск здесь по существу. Чтобы убедиться в последнем, заметим, что независимо от выбора квазиньютоновской формулы матрица M будет обеспечивать равенство $s_{k-1} = M y_{k-1}$ (см. разд. 4.5.2.2). Значит, с учетом определения вектора p_k имеем

$$y_{k-1}^T p_k = -y_{k-1}^T M g_k = -s_{k-1}^T g_k.$$

Отсюда видно, что y_{k-1} и p_k ортогональны (а это — необходимое условие сопряженности) только тогда, когда величина $g_k^T s_{k-1}$ равна нулю, т. е. когда минимум вдоль p_{k-1} найден точно.

Важное достоинство квазиньютоновских методов с ограниченной памятью состоит в том, что в них легко добиться правильной ориентации векторов p_k: гарантировано, что p_k будут направлениями спуска, если скалярные произведения $y_j^T s_j$ положительны для всех пар y_j и s_j из формулы пересчета. Это в свою очередь обеспечено, если для выбора шага применяется алгоритм, который приводит к уменьшению модуля производной вдоль направления поиска (см. разд. 4.3.2.1).

4.8.5. Методы сопряженных градиентов с улучшенной обусловленностью

4.8.5.1. Квадратичные функции. Допустим, что требуется решить систему $Gx = -c$, где G — положительно определенная матрица, и предполагается использовать для этого линейный метод сопряженных градиентов. В разд. 4.8.3.2 было указано, что при абсолютно точных вычислениях он найдет решение за число шагов, равное числу различных собственных значений матрицы G. Поэтому реальная скорость сходимости метода существенно повысится, если исходную систему заменить эквивалентной, но с матрицей, имеющей много единичных собственных значений. Идея *предварительного улучшения обусловленности* и состоит в том, чтобы осуществить преобразование G такого рода.

Соображения, лежащие в основе реализации предлагаемого подхода, заключаются в следующем. Пусть W — положительно определенная матрица. Тогда вектор x, удовлетворяющий равенству $Gx = -c$, можно найти, решив систему вида

$$W^{-1/2} G W^{-1/2} y = -W^{-1/2} c \qquad (4.96)$$

и положив $x = W^{-1/2} y$. Обозначим матрицу этой системы $W^{-1/2} G W^{-1/2}$ через R. Поскольку $W^{-1/2} R W^{1/2} = W^{-1} G$, у R будут те же собственные числа, что и у матрицы $W^{-1}G$ (см. разд. 2.2.3.4).

Значит, чтобы «облегчить работу» методу сопряженных градиентов, надо постараться подобрать W так, чтобы как можно больше собственных значений $W^{-1}G$ лежало вблизи единицы, и заменить исходную систему на (4.96). Не претендуя на строгость, можно сказать, что W подбирается здесь с целью минимизации числа обусловленности матрицы $W^{-1}G$.

Вычислительную процедуру поиска решения системы $Gx=-c$ применением линейного метода сопряженных градиентов к (4.96) организуют таким образом, что в расчетах используется только матрица W, а $W^{1/2}$ не нужна. Чтобы понять, как это делается, надо выписать формулы, подобные (4.91) для системы (4.96), обозначив вектор переменных через u_k, невязки через v_k, а направления через w_k. Затем надо умножить уравнения для пересчета переменных и направлений на $W^{-1/2}$, а уравнение для пересчета невязок на $W^{1/2}$ и заменить $W^{-1/2}u_k$ на x_k, $W^{-1/2}w_k$ на p_k, а $W^{1/2}v_k$ на r_k. Результатом будет следующий алгоритм: задав x_0, вычислив $r_0=c+Gx_0$ и положив $\beta_{-1}=0$, $p_{-1}=0$, для $k=0, 1, \ldots$ найти

$$\begin{aligned} p_k &= -W^{-1}r_k+\beta_{k-1}p_{k-1};\\ \alpha_k &= \frac{r_k^TW^{-1}r_k}{p_k^TGp_k};\\ x_{k+1} &= x_k+\alpha_kp_k;\\ r_{k+1} &= r_k+\alpha_kGp_k;\\ \beta_k &= \frac{r_{k+1}^TW^{-1}r_{k+1}}{r_k^TW^{-1}r_k}. \end{aligned} \tag{4.97}$$

Определяя алгоритм решения системы $Gx=-c$, формулы (4.97) одновременно задают алгоритм поиска минимума квадратичной функции $c^Tx+\frac{1}{2}x^TGx$. При такой интерпретации эти формулы лучше переписать иначе, с тем чтобы их можно было применять и для минимизации нелинейных функций общего вида. Соответствующее правило расчета направления p_k будет звучать так: для вычисления p_k надо решить систему

$$Wz_k=-g_k,$$

после чего взять

$$p_k=z_k+\beta_{k-1}p_{k-1},$$

где $\beta_{k-1}=-y_{k-1}^Tz_k/y_{k-1}^Tp_{k-1}$. При переходе к нему от (4.97) учтено, что $r_k=g_k$, что направления p_k в (4.97) взаимно сопряжены относительно G и что α_k в (4.97) есть шаг в точку минимума вдоль p_k.

Матрицу W, «улучшающую обусловленность», можно определять по-разному. Например, можно сразу строить обратную к ней матрицу W^{-1}, используя какой-нибудь квазиньютоновский метод с ограниченной памятью. В результате r ($r\ll n$) итераций такого метода

мы получим матрицу M, удовлетворяющую квазиньютоновскому условию для r пар векторов $\{s_j, y_j\}$, т. е. каждая из этих пар будет связана равенством $s_j = My_j$. При этом для матрицы Гессе минимизируемой квадратичной формы имеем $Gs_j = y_j$, и, следовательно,

$$s_j = MGs_j.$$

Таким образом, матрице MG обеспечено по меньшей мере r единичных собственных значений, что и позволяет рекомендовать M на роль W^{-1}.

4.8.5.2. Нелинейные функции общего вида. Изложенный выше метод сопряженных градиентов с улучшением обусловленности непосредственно обобщается на задачи минимизации неквадратичных функций. Формулы расчета направлений останутся прежними, только матрица W будет меняться от шага к шагу. Точнее говоря, вместо W в системе для расчета z_k будет фигурировать W_k. Как и в квадратичном случае, можно сразу строить W_k^{-1}, используя какой-нибудь квазиньютоновский метод с ограниченной памятью. Однако это — довольно сложный прием. Более простой и успешно применяемый способ построения W_k опирается на идею диагонального масштабирования.

В данном разделе речь идет о задачах, к которым обычные квазиньютоновские методы не применимы: мы считаем, что для записи квазиньютоновских приближений B_k в памяти машины не хватает места. Однако можно хранить и пересчитывать от шага к шагу только диагональные элементы B_k. Сами по себе они не дают возможности строить хорошие направления поиска, но составленная у них матрица вполне пригодна на роль W_k. Обозначим через γ_j и ψ_j j-е элементы векторов g_k и y_k соответственно, а через $\Delta_k = \operatorname{diag}\{\delta_1, \ldots, \delta_n\}$ и $\Delta_{k-1} = \operatorname{diag}\{\bar{\delta}_1, \ldots, \bar{\delta}_n\}$ приближения диагонали матрицы Гессе на k-й и $k-1$-й итерациях. Тогда переход от Δ_{k-1} к Δ_k мог бы выглядеть так:

$$\bar{\delta}_j = \delta_j + \frac{1}{\alpha_{k-1} g_{k-1}^T p_{k-1}} \gamma_j^2 + \frac{1}{y_{k-1}^T p_{k-1}} \psi_j^2.$$

Это соотношение получено из BFGS-формулы (4.37) и совместно с уравнением

$$\Delta_k z_k = -g_k$$

для расчета z_k задает один из методов сопряженных градиентов с диагональным масштабированием. При его реализации следует предусмотреть контроль положительности диагональных элементов Δ_k и «замораживать» их значения, когда пересчет дает отрицательные величины.

Мы предложили для вычисления Δ_k именно BFGS-формулу не случайно. Она ближе других методу сопряженных градиентов, так как в квадратичном случае при точном одномерном поиске они

порождают одинаковые векторы p_k. Что же касается подхода в целом, то он способствует правильному масштабированию направлений спуска и тем самым облегчает выбор хорошего начального шага в процедуре одномерного поиска.

Диагональное масштабирование применимо и в квазиньютоновских методах с ограниченной памятью. Здесь матрицы Λ_k могут использоваться вместо единичной в качестве начальных матриц циклов квазиньютоновских корректировок. Например, одношаговый BFGS-метод с диагональным масштабированием определяется формулой вида

$$p_k = -\bar{g}_k + \frac{1}{y_{k-1}^T s_{k-1}}(s_{k-1}^T \bar{g}_k y_{k-1} + y_{k-1}^T \bar{g}_k s_{k-1}) - \\ - \frac{s_{k-1}^T \bar{g}_k}{y_{k-1}^T s_{k-1}}\left(1 + \frac{y_{k-1}^T y_{k-1}}{y_{k-1}^T s_{k-1}}\right) s_{k-1},$$

где $\bar{g}_k = \Lambda_k^{-1} g_k$.

4.8.6. РЕШЕНИЕ НЬЮТОНОВСКИХ УРАВНЕНИЙ ЛИНЕЙНЫМ МЕТОДОМ СОПРЯЖЕННЫХ ГРАДИЕНТОВ

Встречаются ситуации, когда вычислить большую разреженную матрицу G_k нетрудно и разместить ее в памяти машины можно, но применить для решения ньютоновской системы какой-нибудь метод с матричной факторизацией нельзя (см. разд. 4.8.1). Бывает и так, что явно формировать G_k нецелесообразно, но организовать эффективный расчет произведения G_k на любой вектор легко; к примеру, если G_k представляет собой произведение нескольких разреженных матриц, то сама она может оказаться плотной и не поместиться в память, хотя для хранения сомножителей места хватит; трудностей с вычислением произведений G_k на векторы при этом не будет. И в том и в другом случае окончательно отказываться от ньютоновской схемы минимизации не обязательно.

Если расчет *относительно небольшого* числа матрично-векторных произведений с G_k неоправдителен, то для минимизации соответствующей функции можно использовать в качестве направлений поиска приближенные решения ньютоновских систем

$$G_k p_k = -g_k, \tag{4.98}$$

генерируемые какой-нибудь из версий линейного метода сопряженных градиентов (см. разд. 4.8.3.2). В частности, можно каждый раз обрывать работу этого алгоритма после *фиксированного числа* итераций и брать в качестве p_k результат последней из них. Такой способ выбора p_k называют *усеченным методом Ньютона*. В названии отражено, что последовательность шагов метода сопряженных градиентов прерывается раньше, чем выполнится полный цикл, теоретически достаточный для построения точного решения. Если

для расчета направления поиска осуществляется только один шаг метода (4.91), вектор p_k будет равен антиградиенту $-g_k$; если же проделать n шагов, то p_k (в отсутствие ошибок округления) будет точным решением системы (4.98). Значит, направление поиска в усеченном методе Ньютона, использующем процедуру (4.91), лежит «между» антиградиентом и обычным ньютоновским направлением. Когда матрица G_k положительно определена и начальным приближением в линейном методе сопряженных градиентов служит $-g_k$, все последующие приближения будут направлениями спуска.

Усеченный метод Ньютона эффективен только тогда, когда малого количества итераций метода сопряженных градиентов хватает для определения хорошего направления поиска. Поэтому здесь особое значение приобретает описанная в разд. 1.8.5.1 техника предварительного улучшения обусловленности, и она важна не только как средство повышения скорости сходимости метода сопряженных градиентов. Применение этой техники по сути дела означает, что при построении усеченного метода Ньютона мы будем «отталкиваться» от алгоритма с расчетом направлений спуска по формуле $p_k = -Mg_k$, где M — некоторая положительно определенная матрица. Точный смысл сказанного заключается в том, что первым приближением в линейном методе сопряженных градиентов будет вектор $-Mg_k$, и если, например, матрица M определяется каким-нибудь квазиньютоновским методом с ограниченной памятью, этот вектор скорее всего окажется значительно лучшим направлением спуска, чем антиградиент. Поэтому и последующие приближения будут намного лучше, чем при обращении к обычному линейному методу сопряженных градиентов, причем независимо от того, насколько сильно введение M повлияет на скорость сходимости этих приближений. В данном случае усеченный метод Ньютона будет представлять собой нечто среднее между классическим методом Ньютона и квазиньютоновским алгоритмом с ограниченной памятью, применяемым для построения матриц M.

Замечания и избранная библиография к разделу 4.8

Идея адаптации квазиньютоновской схемы к задачам большой размерности принадлежит Шуберту (1970), который первым предложил модификации обычных квазиньютоновских формул, позволяющие строить разреженные поправочные матрицы. Иной вывод аналогичных формул дали Авила и Конкус (1979). Задача (4.74) для определения симметричных разреженных поправок была поставлена Пауэллом (1976а) и независимо решена Тойнтом (1977) и Маринлом (1978). Способ вывода «разреженных» аналогов квазиньютоновских формул однопараметрического семейства (4.35) получил Шанно (1980); Тала (1979) существенно упростил построения Шанно и обобщил их для более широкого класса формул. Дальнейшие сведения

из теории и практики применения квазиньютоновской схемы к большим задачам читатель найдет в работах Тойнта (1978, 1979), Денниса и Шнабеля (1979), Тапы (1980), Пауэлла (1981).

Метод численного дифференцирования вдоль специально подбираемых конечно-разностных векторов впервые был использован Керписом, Пауэллом и Райдом (1974). Они предложили его как средство эффективной организации поиска решения больших систем нелинейных уравнений. Применение этого метода в задачах минимизации рассматривалось Гиллом и Мюрреем (1973а). Различные формы метода, эксплуатирующие симметрию, представлены в работе Пауэлла и Тойнта (1979). Обсуждение способов перестановки строк и столбцов разреженной матрицы производных, позволяющих получать желаемые структуры заполненности, содержится в работе Колемана и Море (1980).

Когда матрица Гессе разрежена, но имеет плотные факторы Холесского, можно прибегнуть к ее приближенному (частичному) разложению, генерируемому по обычным формулам. Оно пригодно для расчета направления поиска (см. Тапа (1980)) и может быть использовано для улучшения обусловленности матрицы системы в усеченном методе Ньютона.

Авторами линейного метода сопряженных градиентов являются Хестенс и Штифель (1952). Для задач минимизации нелинейных функций он обобщен Флетчером и Ривсом (1964). Теоретическая эквивалентность метода сопряженных градиентов и BFGS-метода для квадратичных функций доказана Назаретом (1979). Современный обзор различных версий метода сопряженных градиентов можно найти в работах Гилла и Мюррея (1979а), Флетчера (1980) и Хестенса (1980а).

Диксон (1975) и Назарет (1977) предложили методы, которые в квадратичном случае генерируют сопряженные направления независимо от того, точно или приближенно решаются подзадачи одномерной минимизации. Однако их методы обладают серьезным недостатком: для произвольной нелинейной функции они могут порождать направления, вдоль которых она возрастает. Здесь уместно отметить, что точный одномерный поиск практически не требуется и в обычном методе сопряженных градиентов. Хотя предположение о точной одномерной минимизации существенно используется для их теоретического обоснования, на практике хорошо реализованный приближенный поиск дает сравнимые, а иногда и лучшие результаты (см. Гилл и Мюррей (1979а)).

Формула (4.93) метода сопряженных градиентов с произвольным начальным направлением была предложена Билом (1972) и популяризована Пауэллом (1977а). Авторами квазиньютоновских методов с ограниченной памятью являются Перри (1977) и Шанно (1978). Бакли (1978), Назарет (1979) и Носедал (1980) использовали для построения алгоритмов такого сорта связь между BFGS-методом и традиционным методом сопряженных градиентов.

Одной из первых публикаций по методам улучшения обусловленности линейных систем уравнений является работа Аксельсона (1974). Подробности различных методов, полученных из несколько иных соображений, приводятся у Конкуса, Голуба и О'Лири (1976). О том, как использовать квазиньютоновские формулы для улучшения обусловленности в традиционном методе сопряженных градиентов, можно прочесть в работе Назарета и Нокедала (1978). Идея диагонального масштабирования с квазиньютоновской аппроксимацией принадлежит Гиллу и Мюррею (1979а).

Применение линейного метода сопряженных градиентов для приближенного решения ньютоновской системы обсуждалось в предыдущем разделе в предположении, что матрица Гессе положительно определена. О'Лири (1980а), Гилл, Мюррей и Нэш (1981) разработали модификации этого метода, которые подойдут для случаев, когда это предположение не выполнено. Авторами усеченного метода Ньютона, использующего в качестве направлений спуска промежуточные векторы линейного метода сопряженных градиентов, являются Дембо и Штайхаут (1980). Они предложили и схему автоматического выбора числа итераций поиска решения ньютоновской системы, теоретически гарантирующую сверхлинейную сходимость основного алгоритма. Целесообразность применения в усеченном методе Ньютона техники улучшения обусловленности впервые была отмечена Гиллом, Мюрреем и Нэшем (1981).

Когда аналитические значения производных функций недоступны, для их минимизации можно применить аналог метода сопряженных градиентов, в котором вместо точных производных используются их конечно-разностные приближения. Однако в общем случае для поиска хорошего решения по такой схеме потребуется очень много вычислений функции.

ГЛАВА 5

ЗАДАЧИ С ЛИНЕЙНЫМИ ОГРАНИЧЕНИЯМИ

Забросить ключ от дома и уйти
Не как изгнанник, впавший в бредни без конца,
Но выбрав свой маршрут разумно,
Менял скорость, если силы спали.
У. Х. Оден, «The Journey» (1928)

В данной главе рассматривается техника решения задач оптимизации при ограничениях типа линейных равенств или неравенств. Речь пойдет только о задачах с гладкими целевыми функциями. Будут приведены методы, в полной мере использующие специфику линейных ограничений и получающиеся комбинированием приемов безусловной минимизации и вычислительной линейной алгебры. Существенная роль отводится в них множителям Лагранжа (см. разд. 3.3). Что же касается негладких задач с линейными ограничениями, то их можно решать, например, по схеме, изложенной в разд. 6.2.2.2.

Как и прежде, основное внимание мы уделим методам, в эффективности которых убедились на собственном опыте. Иные методы даны лишь постольку, поскольку это способствует пониманию каких-то вещей либо служит основой для последующих построений. Ссылки на литературу вынесены в замечания, которыми завершается каждый раздел. Там же указаны имена авторов представленных схем и результатов.

5.1. МЕТОДЫ ПОИСКА МИНИМУМА ПРИ ОГРАНИЧЕНИЯХ-РАВЕНСТВАХ

В этом разделе мы займемся задачей минимизации гладкой функции на множестве, заданном набором линейных *равенств*:

$$\text{LEP} \qquad \text{найти} \min_{x \in R^n} F(x)$$

$$\text{при ограничениях } \hat{A}x = \hat{b}.$$

Здесь $\hat{A}$ есть $l \times n$-матрица, i-я строка которой составлена коэффициентами i-го ограничения. В дальнейшем считается, что l не превосходит n и что ранг $\hat{A}$ равен l, т. е. ее строки линейно независимы. С практической точки зрения это предположение вполне естественно: линейная зависимость ограничений LEP означала бы, что либо они не совместны, либо среди них есть лишние, которые можно отбросить, не изменив решения. (Хоро-

ная программа оптимизации при линейных ограничениях должна начинать с проверки независимости равенств и исключения лишних, если таковые найдутся.) Относительно целевой функции F будем предполагать, что она дважды непрерывно дифференцируема, ограничена снизу на допустимом множестве и что ее минимум на нем достигается в конечной точке.

Необходимые условия оптимальности для LEP были получены в разд. 3.3.1. Они формируются так: если x^* — решение LEP, то найдется l-мерный вектор множителей Лагранжа λ^*, такой, что

$$g(x^*) = \hat{A}^T\lambda^*. \tag{5.1}$$

Введя матрицу Z, составленную из векторов базиса подпространства, ортогонального строкам $\hat{A}$, это равенство можно заменить эквивалентным

$$Z^T g(x^*) = 0. \tag{5.2}$$

Кроме того, из оптимальности x^* следует положительная полуопределенность матрицы $Z^T G(x^*) Z$. Напомним, что вектор $Z^T g(x)$ принято называть *спроектированным градиентом* функции F в x, а матрицу $Z^T G(x) Z$ — ее *спроектированной матрицей Гессе*.

5.1.1. ПРИНЦИП ОРГАНИЗАЦИИ АЛГОРИТМОВ

5.1.1.1. Влияние линейных ограничений-равенств. Как уже было отмечено в разд. 3.3.1, результатом подчинения n-мерного вектора переменных l линейно независимым ограничениям-равенствам будет сокращение размерности допустимого множества до $n-l$. Чтобы убедиться в этом, рассмотрим подпространство, натянутое на строки $\hat{A}$, и его ортогональное дополнение. Через Y обозначим матрицу, столбцы которой образуют базис первого (например, можно взять $Y = \hat{A}^T$), а через Z — матрицу из базисных векторов второго. Тогда любой n-мерный вектор x можно однозначно представить линейной комбинацией столбцов Y и Z, т. е. для любого x однозначно определятся l-мерный x_Y и $(n-l)$-мерный x_Z, такие, что $x = Yx_Y + Zx_Z$.

Пусть теперь x — допустимый вектор задачи LEP. Для него имеем

$$\hat{A}x = \hat{A}(Yx_Y + Zx_Z) = \hat{b},$$

а так как $\hat{A}Z = 0$, отсюда получим

$$\hat{A}Yx_Y = \hat{b}. \tag{5.3}$$

Поскольку матрица $\hat{A}Y$ невырождена, это равенство фиксирует x_Y, т. е. из всех допустимых x составляющая x_Y будет одной и той же. Отличаться же они будут только $(n-l)$-мерными составляющими x_Z, значения которых ничем не ограничены. Стало

быть, размерность допустимого множества задачи LEP действительно равна $n-l$, причем, решив систему (5.3), построив матрицу Z и заменив аргумент x его разложением по столбцам Y и Z с x_Y, найденным из (5.3), мы можем свести LEP к $(n-l)$-мерной задаче *безусловной минимизации* по x_Z.

Пример 5.1. Рассмотрим множество, заданное единственным ограничением

$$x_1+x_2+x_3=3.$$

В данном случае $\hat{A}=(1\ 1\ 1)$. Матрицу Y возьмем равной $\hat{A}^T$, а одной из возможных Z будет

$$Z=\begin{pmatrix} -\frac{\sqrt{3}}{3} & -\frac{\sqrt{3}}{3} \\ \frac{3+\sqrt{3}}{6} & -\frac{3-\sqrt{3}}{6} \\ -\frac{3-\sqrt{3}}{6} & \frac{3+\sqrt{3}}{6} \end{pmatrix}. \tag{5.4}$$

Подставив $\hat{A}$ и выбранную Y в (5.3), получим $3x_Y=3$, т. е. $x_Y=1$. Соответственно множество всех допустимых точек определится формулой

$$x=\begin{pmatrix}1\\1\\1\end{pmatrix}+\begin{pmatrix} -\frac{\sqrt{3}}{3} & -\frac{\sqrt{3}}{3} \\ \frac{3+\sqrt{3}}{6} & -\frac{3-\sqrt{3}}{6} \\ -\frac{3-\sqrt{3}}{6} & \frac{3+\sqrt{3}}{6} \end{pmatrix} x_Z,$$

где x_Z — двумерный вектор.

5.1.1.2. **Модельная схема.** Наиболее эффективные алгоритмы решения задачи LEP работают по принципу генерирования последовательности допустимых точек с монотонно убывающими значениями целевой функции. Их практичность обусловлена простотой описания для LEP возможных перемещений. Чтобы шаг p из какой-нибудь допустимой точки этой задачи не нарушил ее ограничений, необходимо и достаточно ортогональности p строкам $\hat{A}$. Значит, имея текущее допустимое приближение x_k, направление поиска следует искать среди векторов p_k, удовлетворяющих условию

$$\hat{A}p_k=0. \tag{5.5}$$

Если Z — матрица, составленная из векторов базиса подпространства, ортогонального строкам $\hat{A}$, то можно утверждать, что (5.5) выполнится тогда и только тогда, когда p_k будет линейной комбинацией столбцов Z, т. е. справедливо соотношение эквивалентности

вида

$$Ap_k = 0 \Leftrightarrow p_k = Zp_Z, \tag{5.6}$$

где p_Z есть $(n-l)$-мерный вектор. Равенство справа и служит конструктивным описанием всех возможных для LEP направлений в любой допустимой точке.

Используя (5.6), легко сформулировать аналог модельной схемы безусловной минимизации из разд. 4.3.1, предназначенный для LEP:

Алгоритм LE (*модельная схема решения* LEP). Имея допустимую начальную точку x_0 и сделав присвоение $k \leftarrow 0$, надо выполнить следующие действия:

LE1. [Проверка соблюдения условий останова.] Проверить, выполняются ли в x_k условия останова. Если выполняются, вычисления прекратить и взять x_k в качестве искомого решения.

LE2. [Расчет допустимого направления поиска.] Вычислить ненулевой $(n-l)$-мерный вектор p_Z и направление поиска

$$p_k = Zp_Z. \tag{5.7}$$

LE3. [Расчет длины шага.] Вычислить положительное число α_k (длину шага), обеспечивающее неравенство

$$F(x_k + \alpha_k p_k) < F(x_k).$$

LE4. [Пересчет приближения.] Положить $x_{k+1} \leftarrow x_k + \alpha_k p_k$, $k \leftarrow k+1$ и вернуться к шагу LE1.

Поскольку допустимость p_k из (5.7) гарантирована при любом p_Z, выбор p_Z, позволяющего удовлетворить условию спуска на шаге LE3, осуществляется из соображений, аналогичных используемым в процедурах безусловной минимизации. В частности, при ненулевом $Z^T g_k$ вектор p_Z подбирают так, чтобы было $g_k^T Z p_Z < 0$, т. е. чтобы формула (5.7) давала направление спуска. Что же касается способов определения α_k, то они просто те же, что и для случая без ограничений: так как допустимым будет любой шаг вдоль p_k, никаких модификаций здесь не нужно.

Ниже мы займемся вопросом, как получать «хорошие» направления поиска для задачи LEP, вычисляя p_Z на основе методов гл. 4. Адаптация этих методов к LEP затруднений не вызывает, и побеспокоиться надо только об одном: формулы не должны опираться на те свойства задач безусловной минимизации, которых нет у LEP. К примеру, конструируя метод поиска безусловного минимума, разумно предполагать, что матрица Гессе целевой функции F положительно определена — ведь в большинстве случаев в окрестности решения так оно и будет. При решении задачи LEP это предположение было бы неоправданным, поскольку матрица Гессе ее целевой функции, вообще говоря, знаконеопределена даже в точке оптимума.

Имея в виду эквивалентность (5.6), мы будем далее работать то с p_k, то с p_Z. Важно, однако, помнить, что направление поиска p_k — это n-мерный вектор из подпространства в $\mathcal{R}^n$ размерности $n-l$, в то время как p_Z по определению является $(n-l)$-мерным вектором. Сначала будут рассмотрены способы вычисления p_Z при заданной матрице Z, а два метода построения Z описаны позже в разд. 5.1.3.

5.1.2. РАСЧЕТ НАПРАВЛЕНИЯ ПОИСКА

5.1.2.1. Методы наискорейшего спуска. Разбор методов решения задачи LEP мы начнем с простейших, являющихся обобщениями метода наискорейшего спуска для задач безусловной минимизации (см. разд. 4.3.2.2). Выводятся они по той же схеме, что и последний, но только теперь направление наискорейшего спуска надо искать среди представимых в виде (5.7).

Рассмотрим разложение F в ряд Тейлора в окрестности x_k вдоль произвольного допустимого направления Zp_Z:

$$F(x_k+Zp_Z)=F_k+g_k^TZp_Z+\frac{1}{2}p_Z^TZ^TG_kZp_Z+\ldots. \qquad (5.8)$$

Здесь F_k, g_k и G_k — значения функции F, ее градиента и матрицы Гессе в точке x_k.

Пренебрегая в правой части (5.8) всеми слагаемыми порядка выше первого, можно построить два аналога задачи (4.11) «о наискорейшем спуске». Первый получается, если нормировать сами направления p_k; выглядит он следующим образом:

$$\text{найти} \min_{p_Z\in\mathcal{R}^{n-l}} \frac{g_k^TZp_Z}{\|Zp_Z\|_2}.$$

Решение этой задачи дает формула (4.12), в которую вместо g_k^T надо подставить g_k^TZ, а вместо C — произведение Z^TZ. Соответствующим направлением поиска будет

$$p_k=-Z(Z^TZ)^{-1}Zg_k. \qquad (5.9)$$

Если же нормировать p_Z, задача «о наискорейшем спуске» принимает такой вид:

$$\text{найти} \min_{p_Z\in\mathcal{R}^{n-l}} \frac{g_k^TZp_Z}{\|p_Z\|_2}.$$

Связанное с ней направление p_k задается формулой

$$p_k=-ZZ^Tg_k. \qquad (5.10)$$

Заметим, что при ортонормальной Z векторы (5.9) и (5.10) совпадают.

Обе приведенные формулы расчета p_k дают направления спуска, причем углы между этими направлениями и градиентом целевой

функции строго отделены от прямого. Поэтому комбинирование (5.9) или (5.10) с подходящей процедурой выбора длины шага дает глобально сходящийся алгоритм. Это доказывается в точности по той же схеме, что и в случае без ограничений. Аналогия с методом наискорейшего спуска для безусловной минимизации имеется и в отношении быстродействия. Обе формулы (5.9), (5.10) гарантируют лишь линейную сходимость (см. разд. 4.3.2.2), причем относительный пошаговый прогресс может быть сколь угодно малым. Чтобы добиться более высокой скорости сходимости, при выборе направления поиска надо принимать во внимание кривизну целевой функции.

5.1.2.2. Методы вторых производных. Для тех задач LEP, целевые функции которых таковы, что вычисление аналитических значений их вторых производных затруднений не вызывает, можно предложить модифицированные ньютоновские алгоритмы, подобные рассмотренным в разд. 4.4. Они базируются на понятии *ньютоновского направления* для LEP. Последнее представляет собой решение задачи вида

$$\text{найти} \min_{p \in \mathbb{R}^n} F_k + g_k^T p + \frac{1}{2} p^T G_k p$$
$$\text{при ограничениях } \bar{A} p = 0. \tag{5.11}$$

В силу эквивалентности (5.6) вектор p_Z, отвечающий этому решению, доставляет минимум квадратичной функции, которая получится из правой части (5.8), если отбросить все слагаемые порядка выше второго. Эта функция является локальной аппроксимацией F в подпространстве, натянутом на столбцы Z.

Обозначим через G_Z спроектированную матрицу Гессе $Z^T G_k Z$, а через g_k спроектированный градиент $Z^T g_k$. Тогда решением задачи (5.11) будет вектор Zp_Z, где p_Z — решение уравнения

$$G_Z p_Z = -g_Z. \tag{5.12}$$

Оно аналогично уравнению (4.17), определяющему ньютоновское направление в случае без ограничений (и совпадает с ним при $Z = I$).

Пример 5.2. Возьмем задачу вида

$$\text{найти} \min_{x \in \mathbb{R}^n} x_1^2 x_2^2 + 4x_1^2 x_3^2 + x_2^4 x_3^2 + 3x_1 x_2 + 4x_2 x_3 + 5x_1 x_3 + x_1 x_3$$

при ограничении $x_1 + x_2 + x_3 = 3$.

В допустимой точке $x_0 = (-1, 5, -1)^T$ решением (5.12) (с Z из (5.4)) будет

$$p_Z = \begin{pmatrix} 0.72450 \\ -0.32483 \end{pmatrix},$$

и, следовательно,

$$p_0 = \begin{pmatrix} 0.23075 \\ -0.64004 \\ 0.40929 \end{pmatrix}.$$

(Все величины подсчитаны с пятиразрядной точностью.) Заметьте, что точка $x_k + \gamma p_0$ допустима при любом γ.

Если матрица G_Z положительно определена, то система (5.12) разрешима единственным образом и вдоль ньютоновского направления Zp_Z, построенного по ее решению, функция F убывает. В самом деле, при положительно определенной G_Z таковой будет и G_Z^{-1}, откуда следует, что

$$g_k^T Z p_Z = -g_k^T G_Z^{-1} g_Z < 0.$$

В отсутствие положительной определенности спроектированной матрицы Гессе ситуация осложняется. Тогда квадратичная модель функции F в пространстве векторов p_Z, полученная из разложения (5.8), либо не ограничена снизу, либо имеет целое многообразие точек минимума (см. разд. 3.2.3). Как выбрать на ее основе направление поиска p_k, не ясно. Значит, надо ее заменить другой, а проще говоря, надо как-то модифицировать систему (5.12) для расчета p_Z. Здесь подойдет любой из приемов, описанных в разд. 4.4.2 применительно к задачам безусловной минимизации. В частности, «скорректированное» ньютоновское направление для LEP можно строить, вычисляя p_Z как решение системы

$$\bar{G}_Z p_Z = -g_Z$$

с положительно определенной матрицей $\bar{G}_Z$, генерируемой по G_Z процедурой модифицированной факторизации Холесского (см. разд. 4.4.2.2).

При условиях, аналогичных выдвигаемым в случае без ограничений (т. е. при достаточной близости начального приближения к точке минимума и при положительной определенности в ней спроектированной матрицы Гессе), ньютоновский алгоритм, в котором направления поиска задаются формулами (5.7) и (5.12), а шаги вдоль p_k равны единице, оказывается квадратично сходящимся. Если же осуществляется регулировка длин шагов, то на квадратичную сходимость можно рассчитывать только тогда, когда последовательность этих длин $\{\alpha_k\}$ будет достаточно быстро стремиться к единице.

5.1.2.3. Дискретные ньютоновские методы. Имея в распоряжении лишь первые производные от F, для решения задачи LEP можно применить аналог дискретного метода Ньютона из разд. 4.5.1. При этом, так как уравнения расчета ньютоновского направления для LEP содержат только $(n-t)$-мерную спроектированную матрицу Гессе G_Z и не содержат матрицы G, конечно-разностная аппрокси-

мация последней сама по себе не нужна. Без нее можно обойтись и при построении приближения матрицы G_Z. Поскольку справедливы соотношения

$$g(x_k + h_i z_i) \approx g_k + h_i G_k z_i,$$

матрицу G_Z можно оценивать непосредственно, используя в качестве конечно-разностных векторов столбцы Z. В самом деле, введем векторы $\{w_i\}$, $i = 1, \ldots, n-t$, вида

$$w_i = \frac{1}{h_i}(g(x_k + h_i z_i) - g_k),$$

где h_i — подходящие конечно-разностные интервалы. Эти w_i приближают произведения $G_k z_i$ с точностью до $O(h_i)$. Составим из них $n \times (n-t)$-матрицу W. Тогда в качестве симметричной конечно-разностной аппроксимации для G_Z можно будет взять

$$\tilde{G}_Z \equiv \frac{1}{2}(Z^T W + W^T Z).$$

Отметим, что для построения $\tilde{G}_Z$ требуется только $(n-t)$ вычислений градиента. Таким образом, если ограничений в LEP много, расчет приближения спроектированной матрицы Гессе оказывается очень ведорогой процедурой.

При знаконеопределенной $\tilde{G}_Z$ система уравнений для вычисления p_Z должна быть модифицирована, и здесь, как и в случае с обычным методом Ньютона, можно использовать технику, описанную в предыдущей главе. Сходится дискретный метод Ньютона практически так же, как обыкновенный (см. разд. 4.5.1).

5.1.2.4. Квазиньютоновские методы. К задаче LEP можно приспособить и квазиньютоновскую схему оптимизации из разд. 4.5.2. Известны разные способы обобщения этой схемы на случай с линейными ограничениями-равенствами. Первым был предложен подход, который состоит в том, чтобы использовать для определения направлений поиска n-мерные *вырожденные* квазиньютоновские приближения обращенной матрицы Гессе, генерируемые по стандартным формулам. Если столбцы исходной квазиньютоновской матрицы ортогональны строкам $\hat{A}$, эти формулы теоретически гарантируют сохранение данного свойства у всех последующих квазиньютоновских матриц. Тем самым обеспечивается автоматическая допустимость направлений поиска, вычисляемых по обычной формуле (4.4.3). Однако на практике дела обстоят не столь гладко. Из-за ошибок округления в процессе пересчета квазиньютоновских приближений упомянутое свойство ортогональности быстро теряется, и вычисляемые направления поиска становятся непригодными.

Более успешный прием обобщения квазиньютоновской схемы безусловной минимизации на задачи LEP состоит в применении фиксированной матрицы Z (способы ее задания будут рассмотрены в разд. 5.1.3) и использовании квазиньютоновских формул для ап-

проксимации $(n-l)$-мерной спроектированной матрицы Гессе. В данном случае направление поиска будет определяться по решению аналогичной (4.28) системы вида

$$B_Z p_Z = -g_Z, \tag{5.13}$$

в которой B_Z – очередное квазиньютоновское приближение матрицы G_Z.

Для построения B_Z можно взять любую из квазиньютоновских формул разд. 4.5.2.1, заменив в ней обычные векторы и матрицы спроектированными. В качестве примера приведем результат такого преобразования над BFGS-формулой (4.37). Обозначим через B_Z квазиньютоновскую аппроксимацию для G_Z в точке x_k, и пусть $s_Z = Z^T s_k$, $y_Z = Z^T y_k$. Тогда приближением $\bar{G}_Z$ в x_{k+1}, вычисляемым по BFGS-формуле, будет

$$\bar{B}_Z = B_Z + \frac{1}{p_Z^T s_Z} g_Z g_Z^T + \frac{1}{a_k y_Z^T p_Z} y_Z y_Z^T. \tag{5.14}$$

Аналогично преобразуются и другие квазиньютоновские формулы.

Представленный подход привлекателен тем, что позволяет естественно перенести на случай с ограничениями полезное свойство сохранения положительной определенности. Наличие этого свойства у приближений спроектированной матрицы Гессе G_Z согласуется с необходимыми условиями оптимальности, в силу которых G_Z в точке минимума LEP по меньшей мере положительно полуопределена. В то же время сама матрица Гессе в этой точке, вообще говоря, будет знаконеопределенной, и поэтому любой метод решения LEP с квазиньютоновской аппроксимацией G положительно определенными матрицами выглядел бы искусственно. В сравнении с первым способом обобщения квазиньютоновского поиска в LEP, сводящимся к применению вырожденных приближений обращенной матрицы Гессе, рассматриваемая схема существенно выигрывает в том, что гарантирует постоянную (зависящую от способа задания матрицы Z) точность соблюдения равенства (5.5) для вычисляемых направлений поиска.

Как и в случае без ограничений, для повышения быстродействия и численной устойчивости процедуры решения LEP с использованием квазиньютоновских приближений матрицы G_Z можно применить факторизацию Холесского (см. разд. 4.5.2.2). Храня и пересчитывая от шага к шагу не саму B_Z, а ее факторы Холесского, легко предусмотреть корректировки, которые обеспечат положительную определенность B_Z и тем самым правильную ориентацию направлений поиска независимо от неточностей машинной арифметики.

К задаче LEP применимы также квазиньютоновские методы без вычисления производных (см. разд. 4.6.2). Поскольку в формулах расчета p_Z и обновления B_Z фигурирует только *спроектированный* градиент $Z^T g_k$ и отсутствует обычный, квазиньюто-

новский поиск решения LEP с конечно-разностной аппроксимацией производных (проблема выбора конечно-разностных интервалов обсуждается в разд. 8.6) можно организовать так, что на каждой итерации потребуется лишь $n-l$ дополнительных вычислений функции F. Для этого надо непосредственно оценивать произведение $Z^T g_k$ по разностям F вдоль $n-l$ столбцов Z. Как и дискретные методы Ньютона, данный подход будет тем эффективнее, чем больше число ограничений в задаче.

5.1.2.5. Методы с использованием схемы сопряженных градиентов. Среди рассмотренных в разд. 4.8 приемов безусловной минимизации функций большого числа переменных для LEP лучше всего приспосабливается техника сопряженных градиентов. Что же касается способов эксплуатации слабой заполненности, описанных в разд. 4.8.1 и 4.8.2, то к LEP их применяют редко, так как спроектированная матрица Гессе в общем случае сильно заполнена даже при разреженных A и G_k (исключение составляют задачи с ограничениями очень специальной структуры), а строить процедуру решения LEP на основе аппроксимации самой матрицы Гессе, как правило, неэффективно. Одна из возможностей приложения схемы сопряженных градиентов к задаче LEP состоит в том, чтобы взять ее в качестве алгоритма решения системы (5.12) (имеется в виду линейный метод сопряженных градиентов из разд. 4.8.6). При этом явно формировать матрицу G_Z не нужно: если в памяти поместятся разреженная G_k и разреженное представление Z, вычисление произведений вида $Z^T G_k Z v$ экономнее реализовать в три этапа, последовательно определяя $v_1 = Zv$, $v_2 = G_k v_1$ и $v_3 = Z^T v_2$. Если вторые производные недоступны, вектор v_2 можно заменить его конечно-разностной аппроксимацией. Последняя строится по приращению градиента целевой функции, отвечающему малому шагу из x_k вдоль v_1.

Рассматриваемую технику решения LEP, представляющую собой обобщение усеченного метода Ньютона (см. разд. 4.8.6), нередко комбинируют с приемами улучшения обусловленности, описанными в разд. 4.8.5. Делают это разными способами в зависимости от того, какая информация доступна и какое значение придается ускорению сходимости линейного метода сопряженных градиентов. Наибольшего эффекта удается достичь, когда памяти машины хватает для хранения $(n-l)\times(n-l)$ чисел. В этом случае в качестве матрицы, улучшающей обусловленность, используют квазиньютоновское приближение B_Z спроектированной матрицы Гессе.

5.3.1. ПРЕДСТАВЛЕНИЕ НУЛЬ-ПРОСТРАНСТВА ОГРАНИЧЕНИЙ

В данном разделе описаны два способа построения матрицы Z_k, фигурирующей в приведенных выше формулировках методов решения LEP. Отметим, что эту матрицу вовсе не обязательно форми-

ровать явно. Поэтому под «способами построения Z» мы, по существу, подразумеваем те алгоритмические приемы, которые применяются в процедурах решения LEP для обеспечения допустимости генерируемых направлений поиска (и определения допустимого начального приближения).

6.1.3.1. LQ-факторизация. Первый метод построения Z сводится к LQ-факторизации матрицы $\hat{A}$ (см. разд. 2.2.5.3). Допустим, что известна ортонормальная $n\times n$-матрица Q, такая, что

$$AQ = (L\ 0), \tag{5.15}$$

где L есть невырожденная нижняя треугольная $t\times t$-матрица. Тогда в качестве Y можно взять матрицу, составленную из первых t столбцов Q, а Z набрать из ее последних $n-t$ столбцов. При этом будет справедливо равенство $Z^TZ = I_{n-t}$. Иллюстрацией применения данного подхода служит матрица Z из примера 5.1.

Для методов решения задачи LEP как таковой нумерация столбцов в Z роли не играет, и поэтому для них расстановка соответствующих столбцов Q не существенна. Однако, если LEP решается как подзадача в методе поиска минимума при ограничениях-неравенствах, столбцы Q полезно рассматривать в определенном порядке (см. разд. 5.2.4.1 и замечания к разд. 5.2).

Если матрица Y задается предлагаемым способом, для нее справедливо равенство $\hat{A}Y = L$, и, следовательно, вектор x_Y^* будет решением системы

$$Lx_Y^* = b. \tag{5.16}$$

Последняя хороша тем, что число обусловленности ее матрицы не больше числа обусловленности $\hat{A}$. Определив x_Y^* из (5.16), в качестве начального приближения для LEP можно взять Yx_Y^*.

Рассматриваемый подход к построению матрицы Z *не связан с каким-либо ухудшением обусловленности в процессе решения LEP*, и в этом его существенное достоинство. Поясним сказанное на примере шестопокоординатного поиска. Для числа обусловленности $\mathrm{cond}\,(G_Z)$ фигурирующей в нем матрицы G_Z справедливо соотношение

$$\mathrm{cond}\,(G_Z) \leqslant \mathrm{cond}\,(G_h)\,(\mathrm{cond}\,(Z))^2.$$

Из него видно, что при большом $\mathrm{cond}\,(Z)$ матрица G_Z может оказаться очень плохо обусловленной даже когда G_h обусловлена хорошо. Если же Z определяется посредством LQ-факторизации, то обеспечено равенство $\mathrm{cond}\,(Z) = 1$, и соответственно

$$\mathrm{cond}\,(G_Z) \leqslant \mathrm{cond}\,(G_h).$$

Это означает, что *обусловленность исходной задачи не ухудшается в процессе ее решения.*

Определяя Z методом LQ-разложения, $n \times n$-матрицу Q можно формировать явно как произведение ортогональных преобразований, используемых для триангуляризации $\bar{A}$. Однако, когда число ограничений в LEP относительно мало, это скорее всего нежелательно с точки зрения как требуемой памяти, так и трудоемкости последующих вычислений. Многие алгоритмы поиска решения LEP используют только матрично-векторные произведения с Z, Z^T и Y, а их можно вычислять и не имея явных представлений Z, Y: достаточно хранить компактные записи ортогональных преобразований, используемых в процессе LQ-факторизации, и тогда искомые матрично-векторные произведения определятся последовательным применением этих преобразований к соответствующим векторам. Будет ли этот подход эффективнее, чем метод с явным формированием Q, зависит от отношения между n и t.

5.1.3.2. Метод исключения переменных. Второй способ формирования Z исходит из расчленения матрицы ограничений $\bar{A}$ на две составляющих:

$$\bar{A} = (V \; U), \tag{5.17}$$

где V есть $t \times t$-невырожденная матрица. (Для простоты мы предположили, что V образована t первыми столбцами $\bar{A}$; в общем же случае V может включать любой подходящий набор столбцов $\bar{A}$.)

Представляя в соответствии с (5.17) вектор x в виде $(x_V \; x_U)^T$, ограничения LEP можно записать так:

$$(V \; U)\begin{pmatrix} x_V \\ x_U \end{pmatrix} = \bar{b}$$

или, что то же самое,

$$Vx_V + Ux_U = \bar{b}.$$

Поскольку V по определению невырождена, отсюда следует равенство

$$x_V = V^{-1}(\bar{b} - Ux_U). \tag{5.18}$$

Оно позволяет расценивать t-мерный и $(n-t)$-мерный векторы x_V и x_U как «зависимый» и «независимый» соответственно. Суть в том, что ограничения LEP будут автоматически соблюдены при любом x_U, если только x_V вычисляется по формуле (5.18). Такой способ учета ограничений типа линейных равенств называют *методом исключения переменных*. Начальная допустимая точка в данном случае определяется следующим образом: $x_U = 0$, $x_V = V^{-1}\bar{b}$.

Методу исключения переменных отвечает ортогональная к $\hat{A}$ матрица Z вида

$$Z=\begin{pmatrix} -V^{-1}U \\ I \end{pmatrix}. \tag{5.19}$$

Для примера 5.1 эта формулы дает

$$Z=\begin{pmatrix} -1 & -1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Определяя матрицу Z по формуле (5.19), ее *практически никогда не формируют явно*. Величины, необходимые для расчета направлений поиска,—это матрично-векторные произведения с Z и Z^T. Они всегда могут быть получены из решения уравнений с V и V^T. Значит, для реализации схемы исключения переменных достаточно построить подходящее разложение V, и тогда Z не понадобится. Представление (5.19) в основном применяется при решении задач большой размерности (см. разд. 5.6).

5.1.4. Специальные формы целевой функции

5.1.4.1. Линейная целевая функция. Если функция $F(x)$ линейна (скажем, $F(x)=c^Tx$, где c—некоторый фиксированный вектор), то условие оптимальности (5.1) выполнимо лишь в том случае, когда c является линейной комбинацией строк $\hat{A}$. При произвольном c это предполагает невырожденность $\hat{A}$, т. е. наличие n линейно независимых ограничений. Последние полностью определят решение задачи, которое будет ее единственной допустимой точкой. Совпадение допустимого и оптимального множеств разрешимой задачи LEP с линейной F имеет место и при меньшем числе ограничений. Этот факт используется при построении алгоритмов решения линейных задач с ограничениями-неравенствами.

5.1.4.2. Квадратичная целевая функция. Задачи с линейными ограничениями и квадратичной целевой функцией называют задачами *квадратичного программирования*. В этом разделе мы рассмотрим одну из них:

$$\text{найти } \min_{x\in\mathbb{R}^N} c^Tx+\frac{1}{2}x^TGx \tag{5.20}$$

$$\text{при ограничениях } \hat{A}x=\hat{b}.$$

Здесь G и c—постоянные матрица и вектор.

Когда матрица Z^TGZ положительно определена, задача (5.20) имеет конечное решение, причем это решение единственно. В данном случае шаг $\bar{p}$ из произвольной допустимой точки $\bar{x}$ (с градиентом $\bar{g}=G\bar{x}+c$) в оптимальную x^* будет решением

задачи вида

$$\text{найти } \min_{p \in \mathbb{R}^n} \bar{g}^T p + \frac{1}{2} p^T G p \tag{5.21}$$

$$\text{при ограничениях } \bar{A} p = 0.$$

Последнее, как показано в разд. 5.1.1.2, строится по $(n-t)$-мерному вектору $\bar{p}_Z$ — решению следующей задачи безусловной минимизации:

$$\text{найти } \min_{p_Z \in \mathbb{R}^{n-t}} \bar{g}^T Z^T p_Z + \frac{1}{2} p_Z^T Z^T G Z p_Z. \tag{5.22}$$

Этот вектор в свою очередь является решением линейной системы

$$Z^T G Z \bar{p}_Z = -Z^T \bar{g}.$$

Соответственно шаг $\bar{p}$ в точку минимума (5.20) определяется так:

$$\bar{p} = -Z (Z^T G Z)^{-1} Z^T \bar{g}. \tag{5.23}$$

В разд. 3.3.1 показано, что необходимым условием оптимальности x^* в задаче (5.20) служит равенство

$$g(x^*) = \bar{g} + G\bar{p} = \bar{A}^T \lambda^*, \tag{5.24}$$

где λ^* есть t-мерный вектор множителей Лагранжа для задачи (5.21). *Система* (5.24) *будет разрешимой* относительно λ^* независимо от ранга $\bar{A}^T$.

Случаи, когда спроектированная матрица Гессе не является положительно определенной, разбираются применением к (5.22) рассуждений разд. 3.2.3. В частности, при знаконеопределенной $Z^T G Z$ конечного решения задачи (5.20) не *будет*.

5.1.5. ОЦЕНКИ МНОЖИТЕЛЕЙ ЛАГРАНЖА

Если x^* — решение задачи LEP, то должно выполняться равенство

$$g(x^*) = \bar{A}^T \lambda^*, \tag{5.25}$$

где λ^* — некоторый t-мерный вектор множителей Лагранжа. По определению аналогичные равенства будут иметь место и во всех других условно стационарных точках. Что же касается остальных допустимых точек, то в них система (5.25) (с неизвестным λ^*) несовместна, и поэтому для них множители Лагранжа не определены. Тем не менее, как будет показано в разд. 5.2.3, важно иметь некие средства оценивания этих множителей в точках, где (5.25) не выполнено. Ниже кратко рассмотрены возможные способы построения соответствующих оценок λ_k для произвольного x_k. При этом нас будут интересовать только *состоятельные* оценки, т. е. оценки λ_k,

обладающие тем свойством, что

$$\text{из } x_k \to x^* \text{ следует } \lambda_k \to \lambda^*. \tag{5.26}$$

Кроме того, важно, чтобы в методах оценивания множителей Лагранжа использовались те же матричные разложения, на которые опираются способы представления Z.

5.1.5.1. Линейные оценки множителей. Если матрица Z строится LQ-факторизацией $\hat{A}$, в качестве оценки для λ^* удобно использовать вектор λ_L — решение задачи

$$\text{найти } \min_{\lambda \in \mathcal{R}^l} \| \hat{A}^T\lambda - g_k \|_2. \tag{5.27}$$

В аналитических выкладках его определяют формулой

$$\lambda_L = (A\hat{A}^T)^{-1}\hat{A}g_k.$$

Однако в *алгоритмах* данную формулу лучше не применять: поскольку число обусловленности матрицы $\hat{A}\hat{A}^T$ равно квадрату числа обусловленности $\hat{A}$, это чревато большими ошибками вычислений. Значительно надежнее строить λ_L так, как описано в разд. 2.2.5.3. Тогда ошибка вычисления λ_L в окрестности стационарной точки не превысит величины, пропорциональной числу обусловленности A.

Если для представления Z используется метод исключения переменных, схему построения оценок множителей Лагранжа естественно ориентировать на доступное разложение матрицы V. В предположении, что V состоит из первых l столбцов A, подходящей оценкой будет решение первых l уравнений из (5.25):

$$V^T\lambda_l = g_V.$$

Здесь g_V — вектор, образованный первыми l компонентами градиента g_k. Заметим, что при $x_k = x^*$ и λ_L, и λ_V совпадут с точным вектором множителей Лагранжа.

Оба описанных способа расчета приближений λ^* дают *оценки первого порядка*. Последнее означает, что при малых $\| x_k - x^* \|$ уклонения λ_L и λ_V от λ^* будут величинами порядка $\| x_k - x^* \|$. Таким образом, если в x^* все множители Лагранжа отличны от нуля, в малой окрестности x^* как λ_L, так и λ_V оценит λ^* с хорошей относительной точностью. Размер этой окрестности, очевидно, зависит от кривизны функции F. Она определяет величину нормы $\| g_k - g(x^*) \|$. Кроме того, не претендуя на абсолютную строгость, в отношении оценки λ_L можно утверждать, что окрестность x^*, в которой она хороша, тем меньше, чем больше число обусловленности матрицы $\hat{A}$; когда строки $\hat{A}$ почти линейно зависимы, эта окрестность будет исчезающе малой. Что же касается оценки λ_V, то для нее размер окрестности

обратно пропорционален числу обусловленности матрицы V, которое может быть сколь угодно большим даже при хорошо обусловленной $\hat{A}$. В произвольной точке x_k норма невязки $\hat{A}^T\lambda_k - g_k$ ограничена сверху величиной $\|g_k - g(x^*)\|$. Аналогичная невязка $\hat{A}^T\lambda_* - g_k$ для оценки λ_* имеет первые t компонент равными нулю, а норма вектора, составленного из остальных компонент, зависит от числа обусловленности V. Мы хотим подчеркнуть, что указанное выше относилось к *точным* значениям оценок λ_k и λ_*. На деле все будет обстоять хуже. В частности, хотя теоретически λ_1, $\lambda_{1'}$ в x^* должны совпасть с λ^*, из-за ошибок округления их численно найденные значения скорее всего будут иными. При этом ошибки вычисления λ_1 и $\lambda_{1'}$ пропорциональны числам обусловленности $\hat{A}$ и V соответственно.

Представленный ниже пример иллюстрирует некоторые из высказанных положений.

Пример 5.3. Рассмотрим задачу, в которой

$$\hat{A}^T = \begin{pmatrix} 1 & 1 \\ 1+\varepsilon & 1 \\ 0 & -1 \end{pmatrix}, \quad g(x^*) = \begin{pmatrix} 2 \\ 2+\varepsilon \\ -1 \end{pmatrix}, \quad g_k = \begin{pmatrix} 2+\varepsilon \\ 2 \\ -1+\varepsilon \end{pmatrix}.$$

Заметим, что число $\mathrm{cond}\,(\hat{A})$ мало и что мала также норма $\|g(x^*) - g_k\|$. Точный вектор множителей Лагранжа равен $\lambda^* = (1, 1)^T$. Посмотрим, чему равны оценки λ_1 и $\lambda_{1'}$, вычисленные в x_k, причем V^T составим из двух первых строк $\hat{A}^T$. Для $\varepsilon = 10^{-5}$ получается, что $\lambda_1 = (1.0000, 0.99990)^T$ и $\lambda_{1'} = (-1.0, 3.0000)^T$ (обе оценки округлены до шести значащих цифр). Таким образом, малое уклонение от x^* из-за плохой обусловленности V привело к совершенно неудовлетворительной оценке $\lambda_{1'}$.

5.1.5.2. **Квадратичные оценки множителей.** При определенных условиях множители Лагранжа можно оценивать и с ошибкой второго порядка малости. Для этого нужно использовать информацию о вторых производных F в x_k. Обозначим через q шаг из x_k в x^*. По определению вектор точных множителей Лагранжа λ^* является решением системы

$$g(x^*) = g(x_k + q) = \hat{A}^T\lambda^*.$$

Разлагая $g(x)$ в ряд Тейлора в окрестности x_k, отсюда получим

$$\hat{A}^T\lambda^* = g_k + G_k q + O(\|q\|^2). \tag{5.28}$$

Эта формула и лежит в основе методов построения квадратичных оценок для λ^*.

Поскольку вектор q не известен, непосредственно использовать (5.28) для вычисления оценок λ^* нельзя. Однако, располагая его

приближением p, мы можем перейти от (5.28) к равенству

$$\bar{A}^T\lambda^* = g_k + G_k p + O(\|q\|^2 + \|p-q\|). \tag{5.29}$$

Оно позволяет, в частности, предложить в качестве оценки для λ^* вектор η_L — решение задачи минимизации суммы квадратов невязок переопределенной системы уравнений

$$\bar{A}^T\eta_L = g_k + G_k p. \tag{5.30}$$

Если при достаточно малых ε будет $\|q\| = O(\varepsilon)$ и $\|p-q\| = O(\varepsilon^2)$, то на основании (5.29) можно утверждать, что η_L окажется *оценкой второго порядка*, т. е. $\|\eta_L - \lambda^*\| = O(\varepsilon^2)$.

Важно отметить, что качество оценки η_L существенно определяется выбором p. Правая часть (5.30) при любом p является линейным приближением величины $g(x_k+p)$, и это приближение хорошо оценит $g(x^*)$ только тогда, когда p достаточно мало отличается от x^*-x_k.

В примере 5.2 решением будет $x^* = (-0.14916,\ 3.20864,\ -0.059\,18)^T$, а соответствующий вектор множителей Лагранжа равен $\lambda^* = 0.46927$ (все числа даны с пятиразрядной точностью). В точке $x_0 = (2.0,\ -1.0,\ 2.0)^T$, где $F(x_0) = 74.000$, оценки для λ^* получились равными $\lambda_L = 51.000$ и $\eta_L = 33.093$; обе имеют правильный знак, но сильно отличаются от λ^* по модулю. Интересно, что в *случайной точке* $x_1 = (0.18790,\ 2.66681,\ 0.14529)^T$ (где $F(x_1) = 5.2632$) были получены такие оценки: $\lambda_L = 15\,910$ $\eta_L = -323.14$. Не точна ни та, ни другая, но при этом оценка второго порядка имеет еще и неверный знак (хотя спроектированная матрица Гессе положительно определена). Только совсем вблизи от решения, в точке $x_3 = (-1.5000,\ 3.20998,\ -0.059\,98)^T$, квадратичная оценка η_L оказалась лучше, чем линейная λ_L ($\lambda_L = 0.40869$, $\eta_L = 0.46922$).

Когда в качестве p берется решение подзадачи (5.11) по схеме пьяповского поиска, система (5.30) будет совместной по определению. В данном случае, чтобы получить η_L, достаточно решить какие-нибудь t линейно независимых уравнений из (5.30). В частности, если Z строится методом исключения переменных, квадратичную оценку для λ^* дает решение системы вида

$$V^T\eta_L = \beta,$$

где через β обозначен вектор, состоящий из первых t компонент суммы $g_k + G_k p$.

Замечания и избранная библиография к разделу 5.1

Идея представления решения задач с линейными ограничениями-равенствами в терминах двух подпространств явно или неявно присутствует практически во всех посвященных им публикациях. Полное изложение различных способов задания матрицы Z читатель найдет в работе Гилла и Мюррея (1974c) (см. также Флетчер (1972b)).

Метод спроектированного наискорейшего спуска был предложен и исчерпывающе проанализирован Розеном (1960). Авторство на метод исключения переменных принадлежит Вулфу (1962) (см. также Мак-Кормик (1970a, b)). Вулф (1967) предложил также использовать в роли Z матрицу, составленную из последних $n-l$ столбцов результата обращения:

$$T = \begin{pmatrix} \hat{A} \\ V \end{pmatrix}, \tag{5.31}$$

где V – некоторая $(n-l)\times n$-матрица, обеспечивающая невырожденность T. При решении задач с неравенствами в качестве строк V естественно брать векторы коэффициентов части неактивных ограничений. Построение Z на основе ортогональной факторизации $\hat{A}$ эквивалентно применению схемы Вулфа, если V в (5.31) составить из последних $n-l$ столбцов Q (см. Гилл и Мюррей (1974c)).

Если размерность задачи с линейными равенствами позволяет применить для построения Z LQ-факторизацию, мы рекомендуем пользоваться именно таким способом: он является наиболее численно устойчивым. Применительно к задачам с ограничениями-неравенствами эффективнее использовать другое (хотя и тесно связанное с LQ-факторизацией) разложение (см. замечание к разд. 5.2). Однако, поскольку LQ-факторизация, по-видимому, лучше знакома читателю, мы будем обращаться к ней и при описании методов решения задач с неравенствами.

Идея обобщения квазиньютоновских методов на задачи с ограничениями-равенствами путем применения вырожденных приближений обращенной матрицы Гессе впервые была высказана в статье Дэвидона (1959). Для задач с неравенствами этот подход был детально проанализирован Гольдфарбом (1969). Схема с пересчетом спроектированной матрицы Гессе разработана Гиллом и Мюрреем (1973b, 1974d, 1977a). Более полные сведения относительно оценок первого и второго порядков для множителей Лагранжа читатель найдет в статье Гилла и Мюррея (1979b).

5.2. МЕТОДЫ АКТИВНОГО НАБОРА ДЛЯ ЗАДАЧ С ОГРАНИЧЕНИЯМИ ТИПА ЛИНЕЙНЫХ НЕРАВЕНСТВ

В этом разделе рассматривается задача минимизации функции F на множестве точек, выделенном набором линейных неравенств:

LIP найти $\min_{x\in\mathbb{R}^n} F(x)$

при ограничениях $Ax \geqslant b$.

Здесь A есть $m\times n$-матрица. Ради краткости изложения мы не будем касаться задач, среди ограничений которых есть и равенства, и неравенства: алгоритмы их решения получаются прямым обобщением

представленных ниже алгоритмов. Условия оптимальности для LIP разобраны в разд. 3.3.2. Напомним, что в них значительную роль играют только *активные* в оптимальной точке x^* ограничения. Если их число равно l и через $\bar{A}$ обозначить матрицу, i-я строка которой состоит из коэффициентов i-го активного в x^* ограничения, то необходимые условия первого порядка можно записать так:

$$g(x^*) = \bar{A}^T\lambda^* \tag{5.32a}$$

и

$$\lambda^* \geqslant 0. \tag{5.32b}$$

От аналогичных условий для задач с равенствами их существенно отличает ограничение (5.32b) на знак множителей Лагранжа. Как показано в разд. 3.3.2, появление отрицательного множителя означало бы, что существует допустимое направление спуска, и тем самым противоречило бы оптимальности x^*.

Задача LIP по природе своей сложнее задач с ограничениями-равенствами, так как заранее не известно, какие из ее условий в решении обратятся в равенства. Алгоритмы для LIP, которые нам предстоит обсудить, относятся к классу *методов активного набора*. Логика их организации опирается на следующие соображения. Если бы правильный список активных ограничений был известен априори, оптимум для LIP можно было бы получить из решения задачи

$$\text{найти} \min_{x \in \mathbb{R}^n} f(x)$$
$$\text{при ограничениях } \bar{A}x = \bar{b}, \tag{5.33}$$

для которой существует полный ряд эффективных алгоритмов. Они были рассмотрены в предыдущем разделе, где было показано, что условия типа равенств в каком-то смысле даже упрощают поиск, так как понижают размерность допустимого множества. Соответственно возникает желание использовать для LIP схему минимизации при ограничениях равенствах. Для этого надо ввести так называемый рабочий список — перечень ограничений LIP, интерпретируемых в процессе поиска именно как равенства. В него всегда будет входить только часть исходных ограничений, и, конечно, лучше всего было бы сразу записать туда те из них, которые активны в решении. Однако последние заранее не известны, и поэтому придется *идентифицировать* их состав. Ну а поскольку прогноз может оказаться ошибочным, надо предусмотреть выполняемые по ходу вычислений проверки правильности рабочего списка и задать способы его корректировки в тех случаях, когда она представляется сомнительной. Так устроены все методы активного набора. В качестве еще одной существенной характеристики этих методов следует упомянуть допустимость генерируемых с их помощью точек.

Рабочий список всегда включает только те ограничения, которые в текущей точке обращаются в равенства; однако не все обращаю-

ционе в равенства ограничения обязательно попадут в него. Мы подчеркиваем это потому, что некоторые авторы не делают различия между терминами «активный набор» и «рабочий список». По нашему же мнению, имеет смысл выделять те из активных ограничений, которые будут учтены при расчете направления поиска. Ради этого и введен специальный термин «рабочий список». Тем не менее, отдавая дань традиции, мы сохраняем название «методы активного набора».

Представленная общая характеристика методов активного набора подразумевает, что задача LIP будет решаться в два этапа. На первом определяется начальная допустимая точка, где некоторые из ограничений $Ax \geqslant b$ обратятся в равенства. На втором будет сгенерирована последовательность допустимых точек, сходящихся к решению.

5.2.1. МОДЕЛЬНАЯ СХЕМА

Введем обозначения, используемые ниже при изложении общей схемы решения LIP с применением техники рабочего списка. Как всегда, k будет номером очередной итерации, а x_k — текущим допустимым приближением. Рабочий список в x_k задают рядом атрибутов. Через t_k обозначим число содержащихся в нем ограничений, через I_k — множество их индексов, через A_k — матрицу, составленную из их коэффициентов (правые части этих ограничений соберем в вектор b_k), и, наконец, через Z_k — матрицу, набранную из векторов базиса подпространства, ортогонального строкам A_k.

Методы активного набора имеют разветвленную логику, причем правила выбора ветвей для разных методов могут быть существенно различными. Поэтому в предлагаемой ниже универсальной схеме эти правила не конкретизируются.

Пусть задана некая начальная допустимая точка x_0; способы ее отыскания рассматриваются в разд. 5.7. Для того чтобы запустить метод активного набора, надо положить $k := 0$ и определить t_0, I_0, A_0, b_0 и Z_0. Последующая процедура поиска решения будет выглядеть так:

Алгоритм L1 (*схема активного набора для решения* LIP).

L11. [Проверка соблюдения условий останова.] Если в текущей точке x_k условия останова выполнены, вычисления прекращаются и x_k выдается в качестве решения.

L12. [Выбор логической ветви.] Проверяется, стоит ли продолжать минимизацию в текущем подпространстве или имеет смысл удалить какое-нибудь ограничение из рабочего списка. Если принято решение сохранить рабочий список, осуществляется переход к шагу L13; в противном случае — к шагу L16.

L13. [Вычисление направления поиска.] Вычисляется ненулевой $(n-l_k)$-мерный вектор p_Z и направление поиска

$$p_k = Z_k p_Z. \tag{5.34}$$

L14. [Расчет длины шага.] Вычисляется $\bar{\alpha}$ — максимальный допустимый шаг из x_k вдоль p_k. Определяется положительный шаг α_k, такой, что $F(x_k+\alpha_k p_k) < F(x_k)$ и $\alpha_k \leqslant \bar{\alpha}$. Если $\alpha_k < \bar{\alpha}$, осуществляется переход к шагу L17; в противном случае — к шагу L15.

L15. [Введение ограничения в рабочий список.] Если α_k — шаг, при котором становится равенством r-е ограничение, то индекс r добавляется к множеству I_k. Остальные атрибуты рабочего списка модифицируются соответствующим образом и осуществляется переход к шагу L17.

L16. [Вывод ограничения из рабочего списка.] Выбирается ограничение (скажем, с номером s), которое надо вывести из рабочего списка. Индекс s исключается из I_k. Остальные атрибуты рабочего списка пересчитываются соответствующим образом и осуществляется возврат к шагу L11.

L17. [Пересчет приближения.] Выполняется присвоение $x_{k+1} \leftarrow x_k + \alpha_k p_k$, $k \leftarrow k+1$ и осуществляется возврат к шагу L11.

5.2.2. РАСЧЕТ НАПРАВЛЕНИЯ ПОИСКА И ДЛИНЫ ШАГА

В методах активного набора направления поиска выбираются из подпространства, выделенного рабочим списком. Точнее говоря, очередное направление должно удовлетворять равенству $\hat{A}_k p_k = 0$, и формула (5.34) есть не что иное, как удобное средство выражения данной связи. Заметим, что никакое перемещение вдоль p_k не изменит левой части ни одного из ограничений, содержащихся в рабочем списке. Для вычисления $(n-l_k)$-мерного вектора p_Z на шаге L13 можно воспользоваться любым из методов, описанных в разд. 5.1.2 применительно к задачам с ограничениями-равенствами.

В методах для задач с равенствами, обсуждавшихся в разд. 5.1, длина шага α_k выбирается без оглядки на ограничения; там от α_k требуется только одно — обеспечить «существенное убывание» F. Иначе дело обстоит с методами для задач с неравенствами. Чтобы очередное приближение было допустимым, шаг, в результате которого оно получается, не должен нарушать ограничений, не содержащихся в рабочем списке. Следовательно, существует оценка сверху для α_k — шаг вдоль p_k до первой точки пересечения границы допустимого множества какого-нибудь из этих ограничений.

Для удобства записи мы временно опустим индекс k у величин, связанных с очередной итерацией; таким образом, текущая точка будет обозначаться через x, направление поиска — через p,

множество индексов ограничений рабочего списка — через I и т. д. Пусть i — индекс некоторого ограничения, не содержащегося в рабочем списке в точке x, т. е. $i \notin I$. Если $a_i^T p \geqslant 0$, любой положительный шаг вдоль p не нарушит этого ограничения. Соответственно, если величины $a_i^T p$ окажутся неотрицательными при всех $i \notin I$, никакой верхней границы для длины шага не будет. В противном случае, когда $a_i^T p < 0$, существует предельный шаг γ_i, приводящий в точку, где i-е ограничение становится «удерживающим», т. е. реализующий равенство $a_i^T (x + \gamma_i p) = b_i$. Значение γ_i определяется по формуле

$$\gamma_i = \left\{ \frac{b_i - a_i^T x}{a_i^T p} \;\middle|\; i \notin I \;\text{ и }\; a_i^T p < 0 \right\}.$$

Введем величину $\bar{\alpha}$

$$\bar{\alpha} = \begin{cases} \min\{\gamma_i\}, & \text{если } a_i^T p < 0 \text{ для некоторого } i \notin I; \\ +\infty, & \text{если } a_i^T p \geqslant 0, \text{ для всех } i \notin I. \end{cases}$$

Это — максимальный неотрицательный допустимый шаг вдоль p. Он служит верхней оценкой окончательного шага α. Процедуры одномерного поиска в методах активного набора работают по следующей схеме. Сначала делается попытка отыскать шаг α_t, обеспечивающий существенное убывание F в соответствии с каким-нибудь из критериев разд. 4.3.2.1 и *удовлетворяющий условию* $\alpha \leqslant \bar{\alpha}$. Если такой шаг α_t найден, он и используется в качестве α. Однако может случиться, что указанного α_t не существует; тогда α полагают равным $\bar{\alpha}$, даже если $\bar{\alpha}$ не удовлетворяет выбранному критерию существенного убывания.

Мы хотели бы предостеречь читателя от применения естественной на первый взгляд схемы выбора шага, когда сначала вызывается какая-нибудь процедура одномерного поиска, предназначенная для алгоритмов безусловной минимизации, а затем в качестве α берется минимум из $\bar{\alpha}$ и шага α_t, найденного этой процедурой. Во-первых, если функция F не ограничена снизу вне допустимой области, шага α_t, удовлетворяющего критерию останова процедуры одномерной безусловной минимизации, может не существовать. Во-вторых, вычисление значений целевой функции для шагов, превосходящих $\bar{\alpha}$, есть бессмысленная работа, так как они все равно не реализуются; желательно ограничивать одномерный поиск просмотром только допустимых точек. Разумные процедуры одномерной минимизации при ограничении сверху на длину шага могут быть получены адаптацией регуляризованных схем разд. 4.3.2.1.

Если одномерный поиск дал шаг $\alpha < \bar{\alpha}$, текущий рабочий список передается на следующую итерацию неизменным. Если же

$\alpha = \bar{\alpha}$, его нужно дополнить, отразив тем самым тот факт, что еще одно ограничение обратилось в равенство. Когда появилось сразу несколько новых равенств, в рабочий список включают только одно из них (другие иногда приходится включать на последующих итерациях). Последствия расширения рабочего списка зависят от способа представления матрицы Z в ОЛ-метода, используемого для расчета p_k (см. разд. 5.2.4).

Логика шага 1.15 подразумевает, что ограничение вводится в рабочий список сразу же, как только «встретится» (помогает увеличить шаг). В дальнейшем, оставаясь в рабочем списке, оно будет сохранено как равенство, т. е. не нарушится. Таким образом, шаг 1.15 гарантирует допустимость встречаемых точек. Однако его можно обеспечить и по-другому. Например, ограничение с индексом i, обратившееся в равенство на k-й итерации, можно вводить в рабочий список $(k+1)$-й итерации только при условии, если окажется, что пренебрежение им приведет к неравенству $a_i^T p_{k+1} < 0$.

5.2.3. ИНТЕРПРЕТАЦИЯ ОЦЕНОК МНОЖИТЕЛЕЙ ЛАГРАНЖА

В этом разделе мы вкратце обсудим проблемы, связанные с организацией вывода ограничений из рабочего списка, причем основной акцент будет сделан на необходимость аккуратной интерпретации оценок множителей Лагранжа.

В методах активного набора поиск оптимума для LIP осуществляется последовательными циклами итераций решения промежуточных задач вида

$$\text{найти } \min_{x \in R^n} F(x)$$
$$\text{при ограничениях } A_k x = b_k. \tag{5.35}$$

Чередование этих циклов (этих задач) определяется рассмотренными выше правилами расширения рабочего списка и правилами его сокращения, о которых сейчас пойдет речь. Пусть x_k — решение текущей задачи (5.35). Тогда найдется вектор множителей Лагранжа λ — решение *совместной* системы уравнений

$$g_k = A_k^T \lambda. \tag{5.36}$$

Если какая-нибудь компонента λ окажется отрицательной, значение $F(x)$ можно будет уменьшить, сделав шаг из x_k вдоль допустимого для LIP направления, сохраняющего равенства во всех ограничениях (5.35), кроме того, которому отвечает отрицательный множитель. Это значит, что указанное ограничение целесообразно вывести из рабочего списка. Таким образом, правила сокращения рабочего списка можно ориентировать на знаки множителей Лагранжа, отвечающих решению (5.35).

Любая практичная стратегия вывода ограничений, основанная на знаках множителей Лагранжа, есть результат компромисса. С одной стороны, представляется неразумным решать промежуточные задачи (5.35) с высокой степенью точности, так как сам по себе оптимум в (5.35) интереса не представляет. И уж если рабочий список приходится сокращать (т. е. он не верен), то желательно сделать это, не тратя лишних усилий на поиск точного решения этой задачи, в соответствии с *принципом не множителей Лагранжа* (см. разд. 5.1.5). С другой стороны, существующие методы оценивания множителей не обеспечивают точности или хотя бы правильности знаков оценок, если точка x_t не будет «достаточно близкой» к решению задачи (5.35) (см. пример 5.3). При этом излишняя поспешность в сокращении рабочего списка чревата серьезной потерей эффективности поиска в целом. Во-первых, она может привести к хорошо известному явлению зигзага, когда какое-то ограничение то выводится из рабочего списка, то включается в него вновь, и это влечет за собой падение скорости сходимости к решению или, еще хуже, сходимость в точку, далекую от оптимальной. Во-вторых, так как вывод ограничения связан со значительными затратами на пересчет данных представления матрицы Z, при неоправданно частых сокращениях рабочего списка средняя трудоемкость одной итерации сильно возрастает. Из сказанного следует, что решение об отбрасывании ограничения должно приниматься лишь при определенной уверенности в знаке соответствующего множителя.

Поскольку приемов, гарантирующих правильность знаков оценок множителей, не существует, было разработано сложные стратегии одновременного измерения надежности оценки и выигрыша, который сулит вывод ограничения из рабочего списка. К счастью, хотя в точке x_t, далекой от решения (5.35), оценки множителей Лагранжа могут сильно отличаться от принципов, их знаки все же обычно оказываются верными, и это позволяет правильно выбирать кандидатов на вывод из рабочего списка.

Метод оценивания множителей Лагранжа и метод расчета направления поиска должны быть согласованы. Иначе после вывода из рабочего списка ограничения с отрицательной оценкой можно получить направление, не допустимое относительно отброшенного ограничения. Например, допустимость пьютоновского направления в рассматриваемой ситуации гарантирована только в том случае, если множители оцениваются методом второго порядка.

Пример 5.4. Рассмотрим задачу

$$\text{найти } \min_{x \in \mathbb{R}^2} x_1^2 + 2x_2^2$$

при ограничении $-x_1 - x_2 \geqslant 1$.

Ее единственное ограничение «отрезает» точку безусловного минимума F, и решением оказывается $x^* = (-2/3, -1/3)^T$. Соответствующий ему множитель Лагранжа равен $4/3$. В точке $x = (-3, 2)^T$, где

ограничение выполнено как равенство, оценка первого порядка для множителя Лагранжа, вычисленная по схеме наименьших квадратов, равна $\lambda_k^+ = -1$. Однако если в этой точке исключить ограничение из рабочего списка и после этого вычислить направление поиска методом Ньютона, то получится вектор, указывающий в точку безусловного минимума F и соответственно в недопустимую область ограничения. Если же оценку множителя вычислить методом второго порядка, то окажется, что ограничение выводить из рабочего списка нельзя, так как эта оценка будет положительной – в силу квадратичности F она просто-напросто совпадет с точным значением множителя.

Располагая вторыми производными, надежность принимаемых оценок множителей Лагранжа можно повысить, требуя определенного согласования между оценками первого и второго порядков. Когда они существенно различны, ни те, ни другие надежными считать нельзя. При этом любая наперед заданная степень соответствия достижима, так как разброс между оценками разных типов по мере приближения к x^* стремится к нулю.

Допустим, что Z строится методом LQ-факторизации, а множители Лагранжа оцениваются вектором η_k, определяемым в результате решения согласованной (5.11) и совместной системы (5.30). Тогда объем дополнительных вычислений, необходимых для построения оценки первого порядка λ_k, будет незначительным. Обозначим через δ_i разность между i-ми компонентами λ_k и η_k, т. е. $\delta_i = (\eta_k)_i - (\lambda_k)_i$. В качестве одного из тестов на соответствие между оценками $(\lambda_k)_i$ и $(\eta_k)_i$ можно использовать неравенство

$$2|\delta_i| < \pi \min\{|(\eta_k)_i|,\ |(\lambda_k)_i|\}.$$

Заметим, что оно выполнимо только при совпадении знаков $(\lambda_k)_i$ и $(\eta_k)_i$.

Выявление *возможности* вывода ограничения из рабочего списка вовсе не означает, что этой возможностью надо тут же воспользоваться. Здравый смысл подсказывает, что принимать решение о сокращении рабочего списка надо лишь в том случае, если это сулит больший выигрыш по критерию на очередной итерации. Последнее означает, что при выводе ограничения помимо отрицательности оценки его множителя Лагранжа надо проверять и другие условия. Например, если модуль множителя мал относительно нормы спроектированного на текущее подпространство градиента, то скорее всего большой прогресс будет достигнут, если сохранить рабочий список неизменным.

5.2.4. ВЫЧИСЛЕНИЯ ПРИ ИЗМЕНЕНИИ РАБОЧЕГО СПИСКА

Когда какое-то ограничение вводится в рабочий список, к текущей матрице A_k приписывается новая строка. (Для простоты обычно предполагают, что она становится последней.) При выводе

ограничения одна из строк $\hat{A}_k$ изымается. В обоих случаях неэффективно строить новую матрицу Z_+, начиная ее с нуля; значительно экономнее определять ее пересчетом старой. Как пересчитывают Z_k и другие матрицы при единичных изменениях рабочего списка, будет показано в оставшейся части данного раздела. Для упрощения записей индекс текущей итерации k в представленных ниже формулах опущен. Через $\hat{A}$ будет обозначаться матрица, отвечающая исходному рабочему списку, а через $\overline{A}$ — модифицированному. Буквой l будем обозначать число ограничений в рабочем списке до модификации.

5.2.4.1. Пересчет матрицы Z. Если матрица Z определяется по LQ-разложению $\hat{A}$, ее пересчет выполняют с помощью обычных ортогональных преобразований. При дополнении рабочего списка новым ограничением модифицированные факторы LQ-разложения можно вычислять по схеме, описанной в разд. 2.2.5.7 применительно к QR-факторизации. Обозначим через a^T строку, которая дописывается к $\hat{A}$ и становится $(l+1)$-й в матрице A. Тогда

$$\overline{A}=\begin{pmatrix}\hat{A}\\ a^T\end{pmatrix}=\begin{pmatrix}L & & 0\\ & & \\ a^TQ & & \end{pmatrix}Q^T=(\overline{L}\ \ 0)\,\overline{Q}^T. \tag{5.37}$$

Отсюда видно, что новый ортогональный фактор $\overline{Q}$ будет произведением Q на преобразование Хаусхолдера, обнуляющее компоненты вектора a^TQ с $(l+2)$-й по n-ю. Первые l компонент оно сохранит неизменными. Таким образом,

$$\overline{Q}=QH, \tag{5.38}$$

где H — матрица Хаусхолдера, причем первые l столбцов $\overline{Q}$ и Q идентичны, а столбцы $\overline{Q}$ с номерами от $l+1$ до n суть линейные комбинации соответствующих столбцов Q. Вместо однократного преобразования H можно использовать эквивалентную ему последовательность плоских поворотов (см. разд. 2.2.5.3).

Когда s-е ограничение выводится из рабочего списка, мы получаем

$$\hat{A}Q=(M\ \ 0),$$

где строки M с 1-й по $(s-1)$-ю образуют нижнюю треугольную матрицу, а остальные строки имеют по одному лишнему наддиагональному элементу. Чтобы привести M к нижнему треугольному виду, т. е. преобразовать ее в искомую матрицу $\overline{L}$, надо применить к Q справа последовательность плоских поворотов, которые не повлияют на последние $n-l$ столбцов Q. Соответственно матрица $\overline{Z}$ получится присоединением к Z одного

нового столбца:

$$\bar{Z} = (Z \quad \hat{z}). \tag{5.39}$$

Этот столбец будет линейной комбинацией первых t столбцов Q.

Генерируя Z методом исключения переменных, матрицу A разбивают на две составляющие: $A = (V \quad U)$. При этом для невырожденной составляющей V обычно строится LU-разложение. Обозначим через $\bar{V}$ результат модификации V (связанной с переходом от A к $\bar{A}$). Если к A дописывается новая строка, то $\bar{V}$ будет $(t+1)$-мерной невырожденной матрицей, отличающейся от V только последними строкой и столбцом, причем $(t+1)$-й столбец $\bar{V}$ должен выбираться из $(n-t)$ последних столбцов $\bar{A}$. Если же $\bar{A}$ получается из A отбрасыванием строки, то преобразование V в $\bar{V}$ состоит в изъятии соответствующих строки и столбца. В обоих случаях модифицированные факторы $\bar{L}$ и $\bar{U}$ получаются из исходных элементарными преобразованиями того же типа, которые применяются для построения LU-разложения «с нуля».

***5.2.4.2. Модификации других матриц.** Помимо Z при изменениях рабочего списка приходится пересчитывать другие матрицы. Какие именно — зависит от используемого метода построения вектора p_Z. Мы рассмотрим последствия изменений рабочего списка в случае, когда Z определяется по LQ-разложению $\hat{A}$; приемы пересчета, аналогичные представленным ниже, применяются и в схеме исключения переменных.

В методах ньютоновского типа введение нового ограничения в рабочий список требует пересчета только матрицы $\hat{Z}$. Что же касается спроектированной матрицы Гессе, то ее корректировать не приходится, поскольку ограничение вводится на заключительной стадии итерации, где она уже не нужна, а в начале следующей итерации ее все равно надо строить заново.

Когда в процессе ньютоновского поиска ограничение выводится из рабочего списка, в точке, где это происходит, спроектированная матрица Гессе G_Z, отвечающая исходному списку, уже построена, и здесь нужен пересчет. При этом новая матрица $\hat{Z}$ связана со старой Z формулой (5.39). Следовательно, новая спроектированная матрица Гессе будет

$$\begin{pmatrix} G_Z & Z^T G\hat{z} \\ \hat{z}^T G Z & \hat{z}^T G \hat{z} \end{pmatrix}. \tag{5.40}$$

Вектор $G\hat{z}$ можно сформировать обычным образом, но можно и заменить оценкой, вычисляемой по конечным разностям компонент градиента вдоль z. Для пересчета факторов модифицированного разложения Холесского матрицы G_Z при замене ее на (5.40) пока-

добится лишь один дополнительный шаг процедуры построковой факторизации, описанной в разд. 2.2.5.2.

В квазиньютоновских методах изменение рабочего списка вводит в необходимости пересчета матрицы B_Z, отражающей кривизну F в подпространстве с базисом Z. Когда в рабочий список вводится новое ограничение, факторы Холесского для новой (меньшей по размерности) аппроксимации спроектированной матрицы Гессе могут быть получены из факторов представления текущей матрицы B_Z с использованием формулы (5.38). Соответствующие правила достаточно сложны и здесь обсуждаться не будут (читатель найдет их по приведенным ниже ссылкам).

Вывод ограничения из рабочего списка означает увеличение подпространства, в котором совершается оптимизация; при этом никакой информации о кривизне F вдоль нового вектора z в текущей матрице B_Z не содержится. Поэтому в качестве новой аппроксимации спроектированной матрицы Гессе естественно взять

$$\begin{pmatrix} B_Z & 0 \\ 0 & \gamma \end{pmatrix}.$$

Величину γ обычно полагают равной единице. Однако, если есть какая-то дополнительная информация относительно масштабов вторых производных, γ можно определить и по-другому. Еще один способ заполнить последние строку и столбец новой квазиньютоновской матрицы – строить ее по образцу (5.46), используя конечно-разностное приближение для ∂^2. В данном случае пересчет факторов разложения B_Z по Холесскому осуществляется так же, как в методах ньютоновского типа.

Замечания и избранная библиография к разделу 5.2

Некоторые из известных алгоритмов решения задач специального вида не вполне вписываются в представленную выше схему «активного набора». Например, Кочи (1976) предложил ортофасный метод линейного программирования (см. разд. 5.3.1), в котором каждое приближение удовлетворяет части ограничений как равенствам, но при этом может быть неопустимым. В этом методе спуск осуществляется по отношению к *подходящей непрерывной функции* (см. разд. 6.2.2), в то время как значение исходного линейного критерия могут изменяться немонотонно. Аналогичный метод, использующий LU-факторизацию, разработан Бартельсом (1980).

Уклонения от общей схемы разд. 5.2.1 бывают и в способах расчета направлений поиска. В некоторых методах активного набора эти направления определяются в результате решения довольно сложных подзадач. В частности, существует класс методов, в которых в качестве таковых используются задачи квадратичного про-

граммирования с *ограничениями-неравенствами* вида

$$\text{найти } \min_{p \in \mathbb{R}^n} F_k + g_k^T p + \frac{1}{2} p^T B_k p$$

$$\text{при ограничениях } Ap \geqslant 0. \tag{5.41}$$

Здесь B_k — некоторая аппроксимация матрицы Гессе функции F на очередном шаге (ср. с (5.11)). Выбор p_k на основании задачи (5.41) связан с определенными трудностями. Во-первых, он предполагает хранение либо самой матрицы B_k, либо какого-нибудь из ее разложений (разбор методов решения (5.41) приводится в разд. 5.3.2). На i-й итерации решения (5.41) нужна только матрица $Z_i^T B_k Z_i$, но из-за того, что Z_i от шага к шагу может меняться, без хранения B_k целиком не обойтись. Во-вторых, матрица B_k не обязательно будет положительно определенной. (В частности, матрица Гессе функции F часто знаконеопределена даже в x^*.) Если же B_k знаконеопределена, у задачи (5.41) может не существовать конечного решения, а если оно и существует, то не обязательно будет направлением спуска для F, т. е. не исключено, что окажется $g_k^T p_k > 0$. Правильная ориентация решения задачи (5.41) с знаконеопределенной B_k не обеспечена даже в том случае, когда оно реализует сильный локальный минимум и на каждой итерации его поиска матрица $Z_i^T B_k Z_i$ положительно определена. (Более полное обсуждение квадратичных подзадач с неравенствами читатель найдет в работе Мюррея и Райта (1980).)

Известно немало различных путей преодоления описанных выше трудностей. Так, например, для задач с двусторонними ограничениями на переменные Брайтон и Каллам (1977, 1979) предложили метод, в котором первое пробное направление поиска определяется из решения упрощенной квадратичной подзадачи. Если оно оказывается неудовлетворительным, решается более сложная подзадача; если и она не дает хорошего направления, вызывается другая процедура расчета p_k, не связанная с квадратичным программированием. В экспериментах, проведенных Брайтоном и Каллам, в большинстве случаев первое пробное направление оказывалось удовлетворительным. Однако при знаконеопределенной B_k ни первая, ни вторая квадратичные подзадачи могут не дать подходящего p_k.

Флетчер (1972а) предложил алгоритм, в котором квадратичная подзадача для расчета направления поиска содержит все исходные ограничения плюс дополнительные двусторонние границы для каждой компоненты искомого вектора p_k. Этот алгоритм аналогичен методу Гриффита и Стюарта (1961), где роль промежуточной играет задача линейного программирования. Двусторонние ограничения на переменные подзадачи Флетчер вводит для того, чтобы ее решение не выходило из области, где квадратичная аппроксимация исходной целевой функции F достаточно точна. Значения границ

меняются от шага к шагу, адаптируясь в соответствии с наблюдаемыми отклонениями между значениями F и ее квадратичных приближений. В алгоритме Флетчера реализована идея «доверительной окрестности», используемая и в других областях оптимизации (см. замечания к разд. 4.4).

Достаточно полный обзор конкретных методов решения задач с линейными ограничениями читатель найдет в работах Гилла и Мюррея (1974c, 1977a). Метод ньютоновского типа с представлением Z_k по схеме исключения переменных описан Мак-Кормиком (1970b). Первый квазиньютоновский метод для задач с неравенствами был разработан Гольдфарбом (1969). Он опирается на принадлежащую Дэвидону (1959) идею вырожденной квазиньютоновской аппроксимации обращенной матрицы Гессе. Метод Гольдфарба относится к классу алгоритмов, который будет рассмотрен в разд. 5.4. С деталями методов, оперирующих разложениями спроектированной матрицы Гессе, можно познакомиться по трудам Гилла и Мюррея (1973b, 1974d, 1977a). Анализ ошибок методов пересчета ортогонального разложения матрицы ограничений из рабочего списка проведен Пэйдж (1980).

В практических вычислениях вместо обычного LQ-разложения матрицы $\hat{A}$ удобно использовать так называемое *TQ-разложение*. Оно определяется формулой

$$\hat{A}_k Q_k = (0 \quad T_k),$$

где T_k есть «отраженная» треугольная матрица, характерная тем, что $t_{ij}=0$ при $i+j \leqslant n$. (см. Гилл и др. (1980)). Ее можно интерпретировать как результат расстановки в обратном порядке столбцов нижней треугольной матрицы. Свойства устойчивости у LQ- и TQ-разложений одни и те же. Преимуществом TQ-разложения является то, что в нем матрица Z_k образуется *первыми* $n-l_k$ *столбцами* Q_k. Это удобно, так как при включении или изъятии ограничения из рабочего списка новый столбец Z_k или Y_k возникает в Q_k на нужной позиции. Например, описанное в разд. 5.2.4.1 правило модификации спроектированной матрицы Гессе при выводе ограничения из рабочего списка требует, чтобы новый столбец Z_k стал последним. Когда Z_k определяется TQ-разложением, данное требование выполняется автоматически.

Скорость сходимости метода активного набора существенно зависит от стратегии вывода ограничений из рабочего списка. Отбрасывая ограничение, мы всегда неявно предполагаем, что это приведет к увеличению выигрыша на очередной итерации. В некоторых методах (в частности, в методах из разд. 5.4) соответствующие изменения F могут быть оценены с квадратичной точностью для каждого из ограничений с отрицательным множителем Лагранжа. Вычисляя такие оценки, решение о выводе ограничения из рабочего списка принимают только тогда, когда это сулит значительную выгоду. При этом среди многих кандидатов на вывод выбирают то

ограничение, для которого оценка выигрыша максимальна. Когда вторые производные не доступны, вместо упомянутых оценок используют их приближения, вычисляемые с помощью *подстановочных тестов*. Один из первых методов с предсказанием изменений F на основе подстановочных тестов был предложен Розеном (1961). Исчерпывающее описание техники подстановочных тестов и некоторые рекомендации по их применению даны в работе Гилла и Мюррея (1974d).

Подробное обсуждение способов контроля точности оценок множителей Лагранжа и условий, при которых обеспечена допустимость ньютоновских и квазиньютоновских направлений спуска, приводится в статье Гилла и Мюррея (1979b). Детали различных приемов борьбы с зигзагами в методах нелинейного программирования можно найти у Вулфа (1966), Мак-Кормика (1969) и Зойтендейка (1970).

5.3. ЗАДАЧИ СПЕЦИАЛЬНЫХ ТИПОВ

5.3.1. ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ

Одним из частных случаев общей задачи LIP является *задача линейного программирования*, в которой целевая функция линейна:

$$\text{LP} \qquad \text{найти } \min_{x \in \mathbb{R}^n} c^T x \quad \text{при ограничениях } Ax \geqslant b. \tag{5.42}$$

В приложении к ней процедуры схемы активного набора, рассмотренные в разд. 5.2.1, приобретают ряд особых свойств. Линейность целевой функции позволяет внести в них разнообразные усовершенствования. Как было отмечено в разд. 5.1.4.1, множество оптимальных точек разрешимой линейной задачи с ограничениями-равенствами совпадает с ее допустимым множеством. Это позволяет доказать, что среди решений задачи (5.42) с матрицей A полного столбцового ранга найдется хотя бы одно, в котором будут активными n линейно независимых ограничений. Последнее в свою очередь дает возможность организовать движение к оптимуму (5.42) так, что число ограничений в рабочем списке на каждой итерации будет совпадать с числом переменных.

Мы временно опустим индекс k, помечающий объекты, связанные с текущим шагом. Пусть x — очередное приближение, причем матрица A ограничений из рабочего списка невырождена. В терминологии линейного программирования такую точку x называют *вершиной* допустимого множества. Она является решением соответствующей задачи с ограничениями-равенствами, и потому по схеме активного набора в ней должны быть просмотрены знаки компонент вектора множителей Лагранжа λ. Последний представляет собой

решение невырожденной системы линейных уравнений

$$\hat{A}^T\lambda = c. \tag{5.43}$$

Если все компоненты λ больше нуля либо равны ему, x есть решение LP. Если же среди них найдется отрицательная (скажем, $\lambda_s<0$), то целевую функцию можно уменьшить, двигаясь по направлению, вдоль которого левая часть s-го ограничения возрастает, а остальные ограничения из рабочего списка остаются равенствами. Формально это направление p определяется следующим образом:

$$\hat{a}_s^T p = \gamma, \quad \gamma > 0, \tag{5.44}$$

$$\hat{a}_j^T p = 0, \quad j \neq s \tag{5.45}$$

Поскольку модуль вектора p значения не имеет, мы можем взять в (5.44) $\gamma = 1$. Тогда пара условий (5.44), (5.45) будет означать, что p есть единственное решение линейной системы вида

$$\hat{A}p = e_s, \tag{5.46}$$

где через e_s, как обычно, обозначен s-й столбец единичной матрицы. Отсюда следует, что p есть s-й столбец матрицы $\hat{A}^{-1}$. Таким образом, после того как в вершине выделено ограничение, которое надо исключить из рабочего списка, направление поиска определяется однозначно.

Если множитель λ_s отрицателен, при движении вдоль направления p из (5.46) линейная целевая функция неограниченно убывает. Поэтому естественно сделать вдоль p максимально допустимый шаг. Он приведет на границу допустимого множества какого-то ограничения, которого не было в рабочем списке при расчете p, и его надо включить туда. Соответствующая точка снова будет вершиной. Таким образом, процедура выбора шага при решении LP предельно проста. Упрощаются и другие вычисления. Например, оценка первого порядка для выигрыша, который сулит вывод ограничения из рабочего списка, становится точной, и, следовательно, нет нужды прибегать к более сложным способам оценивания. Поскольку матрица $\hat{A}$ всегда невырождена, метод решения LP с использованием LQ-разложения $\hat{A}$ можно организовать так, что не придется хранить Q. Можно применить также LU-разложение $\hat{A}$. Проблем с перестоем факторов не будет, так как изменения в рабочем списке выражаются в модификациях $\hat{A}$ ранга один.

Описанный способ воплощения общей схемы активного набора из разд. 5.2.1 применительно к задачам линейного программирования носит название *симплекс-метод*. Он был разработан независимо от этой схемы на основе несколько иных соображений и является одним из самых известных численных методов оптимизации. Особенно интересное свойство этого метода — существование конечной

теоретической верхней границы числа итераций, необходимых для отыскания точного решения. И хотя с увеличением размерности задачи эта граница катастрофически возрастает, на практике симплекс-метод работает замечательно, и требуемое число итераций обычно растет с размерностью лишь линейно.

Решение LP можно искать и «несимплексным» способом, допуская приближения, которые не являются вершинами. Тогда, если в рабочем списке окажется менее n ограничений, направление поиска надо определить так же, как это делается в общем случае. Например, в качестве такового можно взять направление «наискорейшего спуска» $p = -ZZ^T c$ (ср. с (5.10)). Заметим, что если x не вершина, уменьшить значение критерия наверняка удастся (в отсутствие вырождения) без сокращения рабочего списка.

Наиболее распространенная форма постановки задачи LP отличается от (5.42). Она будет рассмотрена в разд. 5.6.1.

5.3.2. КВАДРАТИЧНОЕ ПРОГРАММИРОВАНИЕ

Запись общей задачи квадратичного программирования с ограничениями-неравенствами выглядит так:

$$\text{QP} \qquad \text{найти } \min_{x \in \mathbb{R}^n} c^T x + \frac{1}{2} x^T G x$$
$$\text{при ограничениях } Ax \geqslant b.$$

Здесь матрица G и вектор c фиксированы.

Почти все распространенные алгоритмы квадратичного программирования относятся к классу методов активного набора. Мы подчеркиваем это, поскольку даже очень близкие по существу алгоритмы порой описывают в совершенно разных терминах, и это может сбить с толку. К примеру, некоторые вычислительные схемы для QP основаны на применении «таблиц», в которой объединены матрица ограничений и матрица G, другие существенно используют понятие «ведущий элемент» и т. д. Обсуждая технику решения QP, мы сохраним обозначения, введенные в разд. 5.2.1. По-прежнему $\hat{A}_k$ будет матрицей ограничений из рабочего списка в точке x_k, а Z_k – матрицей, чьи столбцы формируют базис в подпространстве векторов, ортогональных строкам $\hat{A}_k$.

5.3.2.1. Задачи квадратичного программирования с положительно определенными спроектированными матрицами Гессе. Мы начнем с задач QP, про которые *заранее* известно, что спроектированная матрица Гессе $Z_k^T G Z_k$ будет положительно определенной на всех итерациях. За исключением особых случаев, соответствующие гарантии можно дать лишь при положительно определенной G.

Специфика рассматриваемых задач обеспечивает наличие конечного единственного минимума целевой функции F на подпростран-

стве, выделенном столбцами Z_k, и правильную ориентацию направления, указывающего в точку этого минимума. Чтобы построить такое направление, надо решить ньютоновскую систему

$$Z_k^T G Z_k p_Z = -Z_k^T g_k \tag{5.47}$$

и положить $p_k = Z_k p_Z$. В силу квадратичности F для значения шага вдоль p_k есть только две разумные альтернативы. Единичный шаг приведет в точку минимума задачи минимизации F на нуль-пространстве матрицы $\hat{A}_k$. Если он допустим, его и надо сделать. При этом очередное приближение будет условно стационарной точкой по отношению к ограничениям из рабочего списка. Для нее можно вычислить точные значения множителей Лагранжа, и тогда станет ясно, является ли она решением исходной задачи или критерий можно уменьшить, сократив рабочий список. Если же единичный шаг вдоль p_k недопустим, надо сделать максимальный допустимый шаг, после чего ввести сдерживающее ограничение в рабочий список.

Как и линейная, квадратичная целевая функция сулит целый ряд вычислительных преимуществ, позволяя существенно сократить усилия, затрачиваемые на расчет используемых в процессе поиска величин. В частности, так как матрица G постоянна, вычислять спроектированную матрицу Гессе и строить ее разложение заново на каждой итерации ни к чему: для корректировки факторов при изменениях в рабочем списке можно применить эффективные методы пересчета. Если, например, какое-то ограничение исключается из рабочего списка, то в спроектированной матрице Гессе появляются дополнительные столбец и строка (см. (5.40)). При этом для пересчета ее разложения по Холесскому пришлось вычислить только один новый столбец в верхнем треугольном факторе R. Квадратичность критерия облегчает и другие блоки схемы активного набора. Рассмотрим, например, расчет направления поиска в условно стационарной, но не оптимальной точке x_k, причем будем считать, что Z_k строится методом LQ-факторизации. По определению в x_k выполнено равенство

$$Z_k^T g_k = 0. \tag{5.48}$$

В соответствии с общей схемой здесь надо вычислить множители Лагранжа и вывести одно из ограничений с отрицательным множителем из рабочего списка. Затем следует пересчитать матрицу Z_k, и результатом этого будет $\bar{Z}_k$ вида (5.39). Таким образом, вектор g_k окажется ортогональным всем столбцам $\bar{Z}_k$, за исключением последнего, т. е.

$$\bar{Z}_k^T g_k = (\bar{z}^T g_k)\, e_{n-t+1}. \tag{5.49}$$

Направление поиска в x_k определяется по решению линейной системы (5.47), куда вместо Z_k надо подставить $\bar{Z}_k$. Полагая, что для

представления спроектированной матрицы Гессе используется факторизация Холесского, и учитывая (5.49), эту систему можно записать так:

$$R^T R p_Z = -\gamma c_{r-t+1}, \tag{5.50}$$

где $\gamma = s^T g_k$. Первый этап процедуры решения (5.50) состоит в расчете горизонтального уравнения с нижней треугольной матрицей R^T и той же правой частью, что у (5.50). В силу особой структуры этого уравнения искомым корнем будет вектор, равный e_{r-t+1}. Значит, p_Z будет решением треугольной системы вида

$$R p_Z = \beta e_{r-t+1}, \tag{5.51}$$

где β — некоторое число.

В действительности векторы p_Z *всегда (а не только в условно стационарных точках) будут решениями в ряде треугольных систем вида* (5.51). Последние проще систем, которые приходится решать в общем случае. Таким образом, для QP расчет p_Z можно организовать экономнее, чем в ньютоновских методах для неквадратичных задач, и это — хороший пример использования специфики QP для упрощения вычислений в схеме активного набора.

5.3.2.2. Задачи квадратичного программирования с знаконеопределенными спроектированными матрицами Гессе. В данном разделе речь пойдет о задачах QP, в которых для каких-то из возможных Z_k матрицы $Z_k^T G Z_k$ оказываются знаконеопределенными. С алгоритмической точки зрения они сложнее задач, обсуждавшихся в предыдущем разделе. В них встречаются условно стационарные точки, где нет минимума критерия на соответствующем нуль-пространстве, и потому системы (5.47) могут давать направления возрастания целевой функции.

Для задач рассматриваемого типа нужен ньютоновский метод, обеспечивающий правильную ориентацию направлений поиска даже в тех точках, где матрица G_Z не является знакоопределенной, причем желательно обойтись минимальными модификациями G_Z. Последнее существенно, так как иначе пропадет возможность эффективной организации вычислений с использованием квадратичности целевой функции. Таким образом, стандартные приемы корректировки знаконеопределенной матрицы (см. разд. 4.4.2) здесь не подходят. В то же время надо помнить, что, если не принять специальных мер, операции пересчета симметричных разложений знаконеопределенных матриц будут численно неустойчивы.

Общие соображения, лежащие в основе предлагаемого ниже метода, таковы. Коль скоро в очередной точке поиска спроектированная матрица Гессе стала знаконеопределенной, можно найти указывающий в сторону убывания F вектор p_Z (направление отрицательной кривизны), для которого $p_Z^T Z_k^T G Z_k p_Z < 0$. Если бы никаких ограничений, кроме содержащихся в текущем рабочем списке,

не было, большие шаги вдоль $p_k = Z_1 p_2$ привели бы в допустимые точки со сколь угодно большими по модулю отрицательными значениями критерия. Мы же предполагаем, что задача имеет *конечное* решение. Значит, при движении вдоль p_k обязательно обратится в равенство какое-то из ограничений, которых не было в рабочем списке k-й итерации. Сделав соответствующий шаг, его надо ввести туда, и после этого спроектированная матрица Гессе, возможно, станет положительно определенной. Тогда очередное направление спуска будем строить обычным образом. Если же этого не произойдет, надо найти новое направление отрицательной кривизны и т. д.

Для поиска решения произвольной задачи QP можно воспользоваться слегка усложненной версией метода, описанного в разд. 5.3.2.1. При этом требуется, чтобы начальная допустимая точка была либо вершиной, либо условно стационарной точкой с положительно определенной спроектированной матрицей Гессе. В данном случае эта матрица может стать знаконеопределенной только в момент вывода какого-то ограничения из рабочего списка на первой или одной из последующих итераций. Здесь в факторе R ее разложения по Холесскому должен появиться новый столбец, причем новый (последний) диагональный элемент R окажется квадратным корнем из отрицательного числа. Последнее означает, что обычным путем пересчет R не провести. Однако система (5.51) для вычисления p_Z такова, *что последний диагональный элемент R влияет только на длину* этого вектора и не влияет на его направление. Следовательно, в операции взятия квадратного корня при вычислении последнего диагонального элемента R мы можем, не изменив ориентации p_Z, использовать вместо аргумента его модуль. Тогда в рассматриваемой ситуации пересчет R пройдет нормально. Результатом будет разложение, верное с точностью до последнего диагонального элемента, а решение системы (5.51) даст вектор p_Z, определяющий направление отрицательной кривизны.

Спуск по найденному после сокращения рабочего списка направлению отрицательной кривизны приведет в точку, где в этот список будет введено новое ограничение. Можно показать, что число отрицательных собственных значений спроектированной матрицы Гессе при этом не увеличится и что пересчитанное после такого обмена ограничений разложение Холесского по-прежнему будет верным с точностью до последнего диагонального элемента. Значит, если обмен приведет к положительно определенной спроектированной матрице Гессе, для восстановления истинного разложения потребуется вычислить заново только последний диагональный элемент R.

Появление мнимого корня при пересчете фактора R означает, что обычные ограничения на модули элементов R (см. разд. 2.2.5.2) теряют силу. Соответственно появляется риск потери точности разложений из-за дефектов машинной арифметики. Однако угроза численной неустойчивости в действительности невелика, так как

относится лишь к *последнему* столбцу R. В конце концов, когда спроектированная матрица Гессе станет (в результате очередного расширения рабочего списка) положительно определенной, этот столбец можно будет построить заново.

Итак, допуская проявление знаконеопределенности в очень ограниченной форме, мы получаем достаточно надежный алгоритм, принцип которого состоит в том, чтобы сохранять спроектированную матрицу Гессе «настолько положительно определенной, насколько возможно». Формальным воплощением этой установки служит принятая стратегия формирования рабочего списка, в соответствии с которой при *знаконеопределенной* G_2 *его можно только дополнять*.

Требование, чтобы начальная точка была вершиной или условно оптимальной при ограничениях-равенствах, не сужает возможностей применения описанного метода. На самом деле, его можно запустить с любой допустимой точки (отыскав ее, если она заранее не известна, процедурой, описанной в разд. 5.7). Коль скоро имеющаяся допустимая точка x не удовлетворяет указанному требованию, дело легко поправить ценой введения в рабочий список нескольких временных искусственных ограничений вида $y_i \geqslant x_i$ или $y_i \leqslant x_i$. Параметры y_i в этих ограничениях полагаются равными значениям соответствующих компонент x, а их состав подбирается так, чтобы x стала вершиной. Искусственные ограничения «помечаются» и выводятся из рабочего списка в первую очередь, после чего забываются. Возможность исключения в условно стационарной точке любого из искусственных ограничений обеспечивается свободой выбора в них знаков неравенств. Отметим, что реализация предлагаемого приема не требует выделения дополнительной памяти. Его суть удобно пояснить на примере, когда в начальной допустимой точке все t ($t<n$) ограничений задачи обращаются в равенства и матрица Z^TGZ положительно определена. В этом случае основным содержанием первых $n-t$ итераций поиска, на каждой из которых снимется одно из искусственных ограничений, будет $n-t$ шагов процедуры столбцовой факторизации Холесского для матрицы Z^TGZ.

Описанный подход позволяет решать задачи с знаконеопределенными матрицами Гессе алгоритмом, который очень близок к применяемому для задач с положительно определенными матрицами. В отношении последних эти алгоритмы просто идентичны.

5.3.3. ЛИНЕЙНАЯ ЗАДАЧА О НАИМЕНЬШИХ КВАДРАТАХ С ОГРАНИЧЕНИЯМИ

К задачам квадратичного программирования, рассмотренным в разд. 5.3.2.1, очень близки задачи минимизации евклидовой нормы вектора невязок системы линейных уравнений при ограничениях типа линейных неравенств. Их называют *условными линейными за-*

дачами о наименьших квадратах. Общая постановка такова:

$$\text{LLS} \qquad \text{найти } \min_{x\in R^n} \frac{1}{2}\|Hx-d\|_2^2$$
$$\text{при ограничениях } Ax \geqslant b.$$

Здесь H есть $s\times n$-матрица. Типична ситуация с $s \geqslant n$; более того, как правило, $s \gg n$.

С формальной точки зрения задача LLS есть частный случай задачи QP из разд. 5.3.2 с $G = H^TH$ и $c = H^Td$. Соответственно, сформировав эти G и c явным образом, для ее решения можно воспользоваться любым из универсальных методов квадратичного программирования. Однако этого не делают, а применяют к LLS специальные методы, в которых нет операций с матрицей, чье число обусловленности равно квадрату числа обусловленности H.

Имея допустимую для LLS точку x_k, в которой линейно независимые ограничения с матрицей $\hat{A}_k$ обращаются в равенства, для определения шага p_k из x_k в точку минимума невязки при условии сохранения этих равенств надо решить аналогичную (5.47) систему

$$Z_k^T H^T H Z_k p_Z = -Z_k^T H^T d_k,$$

где $d_k = Hx_k - d$. Это — обычная ньютоновская система для безусловной задачи о наименьших квадратах вида

$$\text{найти } \min_{p_Z\in R^{n-t}} \frac{1}{2}\|HZ_k p_Z - d_k\|_2^2.$$

Значит, p_Z можно искать как оптимальный для этой задачи вектор. Данное соображение и приводит к эффективному способу вычисления p_Z. Каким образом — показано ниже, причем ради упрощения выкладок мы предположим, что у $\hat{A}_k$ достаточно строк, чтобы матрица HZ_k имела полный столбцовый ранг (это эквивалентно предположению о единственности решения сформулированной задачи безусловной минимизации).

Какова бы ни была ортонормальная матрица P_k размера $s\times s$, справедливо равенство

$$\|HZ_k p_Z - d_k\|_2^2 = \|P_k(HZ_k p_k - d_k)\|_2^2. \tag{5.52}$$

В частности, в качестве P_k можно взять матрицу, преобразующую HZ_k к верхнему треугольному виду (см. разд. 2.2.5.3), т. е. подобрать P_k так, чтобы выполнялось равенство

$$P_k HZ_k = \begin{pmatrix} R_k \\ 0 \end{pmatrix},$$

где R_k — невырожденная верхняя треугольная $(n-t_k)\times(n-t_k)$-матрица. Тогда получим

$$\|P_k(HZ_k p_Z - d_k)\|_2^2 = \left\|\begin{pmatrix} R_k \\ 0 \end{pmatrix} p_Z - P_k d_k\right\|_2^2.$$

Отсюда видно, что невязка вспомогательной задачи минимальна, когда компоненты вектора $R_k p_2$ равны первым $n-t_k$ компонентам $P_k d_k$, т. е. искомый p_2 будет единственным решением невырожденной линейной системы $R_k p_2 = \bar{d}_k$. Через $\bar{d}_k$ обозначен вектор, образованный первыми $n-t_k$ компонентами $P_k d_k$ (см. разд. 2.2.5.3).

Представленный способ определения p_2 не требует запоминания матрицы P_k, поскольку, применяя образующие P_k элементарные ортогональные преобразования параллельно к HZ_k и d_k, их можно тут же забывать. При $s \gg n$ такой способ организации вычислений сулит значительную экономию памяти.

Если матрица Z_k гипертрофируется ортогональной факторизацией $\bar{A}_k$, построение новых R_k и $\bar{d}_k$ при изменениях рабочего списка нетрудно реализовать по принципу пересчета. Когда рабочий список пополняется каким-то ограничением, Z_{k+1} получается последовательным умножением Z_k справа на несколько матриц плоского поворота (см. разд. 5.2.4.1). Эти повороты приведут к появлению под диагональю R_k ненулевых элементов. Чтобы обнулить последние, надо применить к R_k слева другой набор плоских поворотов (тем самым неявно определится новая матрица P_{k+1}). При сокращении рабочего списка матрица Z_{k+1} будет получена из Z_k присоединением к ней нового последнего столбца $z_{n-t_{k+1}}$. Здесь к R_k тоже надо дописать новый столбец, формируемый по вектору $Hz_{n-t_{k+1}}$. Для организации вычислений без хранения H удобнее оперировать разложением вида $P_k H \tilde{Q}_k = U_k$, где $\tilde{Q}_k = (Z_k \; Y_k)$ (перестановка ортогональной матрицы из LQ-разложения $\bar{A}_k$), а U_k — верхняя треугольная $n \times n$-матрица. В этом случае R_k будет левым верхним блоком U_k.

С практической точки зрения требование полноты ранга матрицы HZ_k на всех итерациях является довольно жестким. Однако предлагаемый алгоритм легко обобщить на случаи, когда это требование не выполняется. При этом в основу расчетов ложится полное ортогональное разложение HZ_k:

$$P_k H Z_k V_k = \begin{pmatrix} R_k & 0 \\ 0 & 0 \end{pmatrix}.$$

Здесь R_k — верхняя треугольная $r \times r$-матрица, где r — ранг HZ_k (см. разд. 2.2.5.3).

Замечания и избранная библиография к разделу 5.3

Симплекс-метод решения задач линейного программирования появился на свет в 1947 г. Его автором является Данциг (см. Данциг (1963)). Современные реализации симплекс-метода настолько совершенны, что без особых затруднений удается решать задачи,

ограничения которых исчисляются тысячами. Техника линейного программирования, рассчитанная на большое число ограничений и переменных, будет обсуждаться в разд. 5.6.1.

Линейность целевой функции делает целесообразным применение более тонких, чем в общем случае, правил вывода ограничений из рабочего списка. Обозначим через $p^{(r)}$ направление поиска, получающееся после отбрасывания ограничения с отрицательным множителем λ_r. Когда целевая функция F нелинейна, λ_r служит оценкой первого порядка для изменения F при единичном шаге вдоль $p^{(r)}$. При линейной F эта оценка становится точной, т. е. $c^T p^{(r)} = \lambda_r$.

Одна из стратегий сокращения рабочего списка состоит в том, чтобы среди кандидатов на вывод выбирать ограничение, которому отвечает наилучшая линейная оценка выигрыша за счет единичного шага вдоль соответствующего направления, т. е. минимальная величина λ_r. Можно действовать и по-другому, выводя то ограничение, для которого максимальна оценка выигрыша при единичном изменении по x. Для линейных задач эта стратегия оказывается особенно эффективной (см. Гринберг и Кэлан (1975), Гольдфарб и Райд (1977)). В данном случае между собой сравниваются величины

$$c^T p^{(r)} / \| p^{(r)} \|_k = \lambda_r / \| p^{(r)} \|_k,$$

причем нормы $\| p^{(r)} \|_k$ можно пересчитывать от шага к шагу по рекуррентным формулам. Симплекс-метод со второй стратегией сокращения рабочего списка принято называть методом отвесного ребра. Харрис (1973) предложил способ аппроксимации норм векторов $p^{(r)}$.

Некоторые из ранних методов квадратичного программирования были получены как модификации схем решения линейных задач. Наиболее удачными среди них оказались метод Била (1959, 1967a) и применительно к случаям с положительно определенной матрицей Гессе метод Данцига и Вулфа (см. Вулф (1959), Данциг (1963)). Авторами первых алгоритмов квадратичного программирования, работающих по схеме «активного набора», являются Мюррей (1971a) и Флетчер (1971b) (алгоритм Флетчера обсуждается в замечаниях к разд. 5.4).

Гилл и Мюррей (1978b) предложили метод решения задачи QP с разложением по Холесскому матрицы $Z_k^T G Z_k$, предполагающий, что фактор Q_k из LQ-разложения матрицы ограничений рабочего списка будет целиком храниться в памяти машины. Можно конструировать методы, не требующие запоминать всю матрицу Q_k. Однако такая экономия всегда достигается ценой снижения численной устойчивости. Мюррей (1971a), например, разработал метод решения задач QP с знаконеопределенными матрицами Гессе, требующий хранения только Y_k. При этом Z_k выбирается так, чтобы матрица $Z_k^T G Z_k$ была диагональной. Банч и Кауфман (1980) предложили схожий алгоритм с блочно-диаго-

пальными $Z_k^T G Z_k$ (в отношении задач с положительно определенными матрицами Гессе методы Мюррея и Батча—Кауфмана идентичны). Алгоритм решения QP с положительно определенной матрицей G, не требующий запоминания никаких блоков Q_k, сконструирован Пауэллом (1980). В нем Z_k строится по L_k и Λ_k. Однофазный алгоритм квадратичного программирования, который может двигаться по недопустимым точкам, читатель найдет у Конна и Синклера (1975). Хорошее сопоставление разных методов решения QP приведено у Джанга (1980).

Описанный в разд. 5.3.3 метод решения условных задач о наименьших квадратах в основном разработан Стоером (1971) (см. также Шиттковски и Стоер (1979)) и включает некоторые модификации, автором которых является Сондерс (1980). С другими методами, предназначенными для этих задач, можно познакомиться по книге Лоусона и Хэнсона (1974). Методы решения условной задачи минимизации l_1-нормы невязки линейных уравнений, использующие обсуждавшуюся технику ортогональных разложений, предлагались Бартелсом и Конном (1980).

*5.4. ЗАДАЧИ С МАЛЫМ ЧИСЛОМ ОГРАНИЧЕНИЙ ОБЩЕГО ВИДА

Методы организации вычисления направлений поиска, описанные в предыдущих разделах (впредь мы будем называть их *методами пуль-пространства*), вообще говоря, тем эффективнее, чем больше отношение числа ограничений задачи к числу ее переменных. Когда это отношение мало, предпочтительными могут оказаться схемы иного рода, использующие матрицы, размерность которых не увеличивается, а падает с уменьшением числа ограничений в рабочем списке. Это — так называемые *методы ранг-пространства*. Они рассматриваются ниже применительно к задачам с критериями двух типов, а именно с положительно определенными квадратичными формами и с нелинейными функциями общего вида.

*5.4.1. КВАДРАТИЧНЫЕ ЗАДАЧИ С ПОЛОЖИТЕЛЬНО ОПРЕДЕЛЕННЫМИ МАТРИЦАМИ ГЕССЕ

Возьмем задачу квадратичного программирования с положительно определенной матрицей Гессе G. Пусть x_k — ее допустимая точка, причем t_k ($t_k \ll n$) ограничений с матрицей A_k выполнены в x_k как равенства. Направление p_k, обеспечивающее допустимость $x_k + p_k$ относительно этих ограничений и доставляющее минимум целевой функции на соответствующем подпространстве, есть решение квадратичной подзадачи вида

$$\text{найти } \min_{p \in \mathbb{R}^n} g_k^T p + \frac{1}{2} p^T G p$$

$$\text{при ограничениях } A_k p = 0, \tag{5.53}$$

где $g_k = Gx_k + c$. Обозначим через λ_k отвечающий ему вектор множителей Лагранжа. Тогда необходимые условия оптимальности p_k можно будет записать в виде следующего $(n+t_k)$-мерного векторного равенства:

$$\begin{pmatrix} G & -\hat{A}_k^T \\ \hat{A}_k & 0 \end{pmatrix}\begin{pmatrix} p_k \\ \lambda_k \end{pmatrix} = \begin{pmatrix} -g_k \\ 0 \end{pmatrix}. \tag{5.54}$$

В силу положительной определенности G при $\hat{A}_k$, имеющей полный ранг, матрица

$$\begin{pmatrix} G & -\hat{A}_k^T \\ \hat{A}_k & 0 \end{pmatrix} \tag{5.55}$$

невырождена. В этом случае система (5.54) однозначно разрешима относительно пары p_k, λ_k, причем обратная к (5.55) матрица имеет вид

$$\begin{pmatrix} G^{-1} - G^{-1}\hat{A}_k^T(\hat{A}_k G^{-1}\hat{A}_k^T)^{-1}\hat{A}_k G^{-1} & G^{-1}\hat{A}_k^T(\hat{A}_k G^{-1}\hat{A}_k^T)^{-1} \\ -(\hat{A}_k G^{-1}\hat{A}_k^T)^{-1}\hat{A}_k G^{-1} & -(\hat{A}_k G^{-1}\hat{A}_k^T)^{-1} \end{pmatrix}.$$

Таким образом, для задачи квадратичного программирования с положительно определенной матрицей Гессе можно предложить две (эквивалентные в смысле результата) формы организации расчета направлений поиска и множителей Лагранжа:

Метод пуск-пространства	*Метод ранг-пространства*
$p_k = Z_k (Z_k^T G Z_k)^{-1} Z_k^T q_k$	$\lambda_k = (\hat{A}_k G^{-1}\hat{A}_k^T)^{-1}\hat{A}_k G^{-1} g_k$
$\lambda_k = (\hat{A}_k \hat{A}_k^T)^{-1}\hat{A}_k (Gp_k + g_k)$	$p_k = G^{-1}(\hat{A}_k^T\lambda_k - g_k)$

Вновь полученные выражения для p_k, λ_k сами по себе для практических вычислений непригодны. Как обычно, нужны некоторые преобразования. Обозначим через L_k нижний треугольный фактор LQ-разложения $\hat{A}_k$ (см. (5.15)), и пусть Y_k — матрица, составленная из первых t_k столбцов ортогонального фактора Q_k. Тогда подстановкой $\hat{A}_k = L_k Y_k^T$ в уравнения метода ранг пространства получим

$$p_k = G^{-1}(Y_k\theta - g_k), \tag{5.56}$$

где

$$\theta = (Y_k^T G^{-1} Y_k)^{-1} Y_k^T G^{-1} g_k. \tag{5.57}$$

Это и есть формулы, используемые на практике.

Поскольку Y_k представляет собой блок ортогональной матрицы, число обусловленности у $Y_k^T G^{-1} Y_k$ будет хуже, чем у G. Расчет p_k по формулам (5.56) и (5.57) можно организовать, храня факторы Холесского для G (их придется вычислить один раз) и пересчитывая при вводе и выводе ограничений из рабочего списка матрицы L_k, Y_k и факторы Холесского для $Y_k^T G^{-1} Y_k$. Если принять, что запись Y_k и факторов G требует примерно

столько же памяти, сколько запись несимметричной матрицы Q_k, то для сопоставления запросов на память от двух методов расчета p_k надо сравнивать размерности матриц $Z_k^T G Z_k$ и $Y_k^T G^{-1} Y_k$. Когда ожидаемые значения t_k близки к n, метод нуль-пространства будет экономнее. При малых ожидаемых t_k экономнее n, следовательно, предпочтительнее в условиях дефицита памяти оказывается метод ранг-пространства. В диапазоне $t_k \approx {}^1/_2 n$ выбор делают из иных соображений. В частности, следует иметь в виду, что метод ранг-пространства менее надежен с вычислительной точки зрения, так как при расчете p_k по формуле (5.56) могут возникать существенные ошибки компенсации. Это обстоятельство нередко становится решающим. Правда, если матрица G имеет специальную структуру заполнения, позволяющую очень экономно хранить ее факторы Холесского, им обычно пренебрегают.

Во многих случаях максимальные размеры матриц $Z_k^T G Z_k$ и $Y_k^T G^{-1} Y_k$ удается хорошо оценить заранее. К примеру, в задаче с пятью ограничениями общего вида размеры $Y_k^T G^{-1} Y_k$ никогда не превысят 5×5. Допустим далее, что матрица Гессе имеет вид

$$G = \begin{pmatrix} G_{11} & 0 \\ 0 & 0 \end{pmatrix},$$

где G_{11} есть невырожденная $r \times r$-матрица. Тогда, если известно, что подзадача (5.53) всегда будет однозначно разрешимой, т. е. $Z_k^T G Z_k$ всегда будет положительно определенной, можно утверждать, что число ограничений в рабочем списке не станет меньше $n-r$ и соответственно число столбцов Z_k не превзойдет r.

5.4.2. МЕТОДЫ ВТОРЫХ ПРОИЗВОДНЫХ ДЛЯ РЕШЕНИЯ ЗАДАЧ ОБЩЕГО ВИДА

В этом разделе мы рассмотрим два метода расчета направлений поиска, предназначенных для задач с произвольными гладкими целевыми функциями и малым числом линейных ограничений. Они предполагают доступность вторых производных целевой функции.

5.4.2.1. Метод, основанный на квадратичной подзадаче. Рассмотренный ранее ньютоновский метод нуль-пространства в обычной ситуации дает в качестве направления поиска решение задачи QP вида (5.53) с $G = G_k$. Здесь не важно, будет матрица G_k положительно определенной или нет. Чтобы в отсутствие положительной определенности G_k получить то же направление методом ранг-пространства, его формулы надо несколько усложнить.

Решение задачи (5.53) не изменится, если заменить G_k матрицей

$$G_\nu \equiv G_k + \nu \bar{A}_k^T \bar{A}_k. \tag{5.58}$$

Данное утверждение следует из равенства $Z_k^T G_\nu Z_k = Z_k^T G_k Z_k$. При этом, когда $Z_k^T G_k Z_k$ положительно определена, существует число $\bar{\nu}$,

такое, что G_ν будет положительно определенной при всех $\nu > \bar{\nu}$. Таким образом, чтобы применить метод ранг-пространства при знаконеопределенной G_k, надо подобрать подходящий параметр ν и использовать в (5.56) и (5.57) вместо G_k матрицу G_ν, факторизуя ее по модифицированной схеме Холесского (см. разд. 4.4.2.2). Когда F хорошо отмасштабирована (см. разд. 8.7) и ограничения отнормированы так, что $\|\hat{A}_k\|_\infty$ есть число порядка единицы, как правило, в качестве ν подойдет

$$\nu = \frac{1}{\delta}\left(\|G_k\|_\infty + \varepsilon_M\right), \tag{5.59}$$

где ε_M — относительная машинная точность и $\delta \in [\varepsilon_M^{1/2}, \varepsilon_M^{1/3}]$.

5.4.2.2. Метод, основанный на аппроксимации спроектированной матрицы Гессе. Второй способ вычисления p_k является эффективным, когда размерность Z_k близка к n, и основан на использовании следующего предельного соотношения:

$$\lim_{\nu \to \infty} G_\nu^{-1} = Z_k \left(Z_k^T G_k Z_k\right)^{-1} Z_k^T. \tag{5.60}$$

При этом

$$\|G_\nu^{-1} - Z_k \left(Z_k^T G_k Z_k\right)^{-1} Z_k^T\| = O\left(\frac{1}{\nu}\right).$$

Здесь G_ν — матрица (5.58).

Указанное свойство матрицы G_ν^{-1} позволяет рекомендовать в качестве p_k направление, рассчитываемое при конечном ν по решению p_ν системы уравнений

$$G_\nu p_\nu = -g_k. \tag{5.61}$$

Переход от p_ν к p_k осуществляется так:

$$p_k = (I - Y_k Y_k^T)\, p_\nu \; (\equiv Z_k Z_k^T p_\nu).$$

Получающееся направление представимо в виде (5.34) и соответственно является допустимым. Если матрица $Z_k^T G_k Z_k$ положительно определена и число ν достаточно велико, это p_k будет направлением спуска. Если же среди собственных значений $Z_k^T G_k Z_k$ есть отрицательные, параметра ν, обеспечивающего положительную определенность G_ν, не существует, и тогда p_ν из (5.61) может задать направление возрастания F. Однако, когда решение системы (5.61) ищется методом модифицированной факторизации Холесского, в этом случае выполнится необходимая корректировка G_ν, и правильная ориентация p_k будет гарантирована.

В описанном методе ньютоновское направление $-Z_k(Z_k^T G_k Z_k)^{-1} \times Z_k^T g_k$ аппроксимируется вектором $-Z_k Z_k^T G_\nu^{-1} g_k$, и ошибка этой аппроксимации имеет порядок $1/\nu$. При выборе ν по правилу (5.59) для хорошо отмасштабированной задачи она сравнима с ошибкой конечно-разностного приближения спроектированной матрицы Гессе $Z_k^T G_k Z_k$ (см. разд. 5.1.2.3). Когда ν увеличивается,

число обусловленности G_k возрастает. Однако отклонение вычисляемого p_k от истинного обычно почти целиком лежит в подпространстве ошибки оценивания ньютоновского направления и потому критического значения не имеет.

Замечания и избранная библиография к разделу 5.4

Схема ранг-пространства лежит в основе всех ранних реализаций квазиньютоновского поиска, предназначенных для решения задач с линейными ограничениями. Точнее говоря, все они опираются на возможности представления обратной к (5.55) матрицы в виде

$$\begin{pmatrix} G_k & -\bar{A}_k^T \\ \bar{A}_k & 0 \end{pmatrix}^{-1} = \begin{pmatrix} H^* & C^{*T} \\ -C^* & K^* \end{pmatrix} \tag{5.62}$$

и аппроксимации ее блоков по значениям градиентов. В (5.62) H^*, C^* и K^* есть матрицы размеров $n \times n$, $t_k \times n$ и $t_k \times t_k$ соответственно. В частности, в методе Гольдфарба (1969) квазиньютоновская техника применяется для построения приближений H^*, а линейные оценки множителей Лагранжа вычисляются по схеме наименьших квадратов с использованием обратной к $\bar{A}_k \bar{A}_k^T$ матрицы. Муртаф и Сарджент (1969) предложили алгоритм, основанный на прямой квазиньютоновской аппроксимации матриц G_k^{-1} и $(\bar{A}_k G_k^{-1} \bar{A}_k^T)^{-1}$. Дальнейшие комментарии по поводу этих методов можно найти в работе Гилла и Мюррея (1974d).

Методы ранг-пространства для задач с произвольными гладкими и знаконеопределенными квадратичными целевыми функциями пока еще служат предметом исследований. Надо отметить, что схемы этого сорта, рассчитанные на задачи квадратичного программирования с знаконеопределенными матрицами Гессе, более громоздки, чем соответствующие методы нуль-пространства. Причина — отсутствие возможности выяснить по разложению G и $\bar{A}_k G^{-1} \bar{A}_k^T$ определенность матрицы $Z_k^T G Z_k$. Из-за этого трудно отличать условно седловые точки от условно экстремальных. Однако когда x_k — точка условного минимума и какое-то ограничение выводится в ней из рабочего списка, то гарантировано, что $x_k + p_k$ тоже будет точкой условного минимума (на полупространстве, заданном новым рабочим списком), если только p_k не окажется направлением отрицательной кривизны. Значит, имея в качестве начального приближения x_0 вершину допустимого множества и отслеживая кривизну F вдоль используемых направлений, мы можем держать под наблюдением экстремальность последующих приближений. Коль скоро кривизна F вдоль текущего направления поиска оказалась отрицательной, с дальнейшими сокращениями рабочего списка надо будет повременить до тех пор, пока в него не войдет какое-нибудь новое ограничение (см. разд. 5.3.2.2). Данная стратегия модификаций рабочего списка лежит в основе метода квадратичного программирования,

предложенного Флетчером (1971b). Этот метод предполагает хранение и пересчет матриц H^* и C^* из (5.62) в явном виде. Описанный выше метод ранг-пространства для квадратичных задач с положительно определенными матрицами Гессе аналогичен квазиньютоновскому методу, представленному в работе Гилла и Мюррея (1974d).

Методы ранг-пространства для задач квадратичного программирования с положительно определенным критерием можно строить на основе непосредственной факторизации или обращения расширенной матрицы (5.55) (см., например, Бартелс, Голуб и Сондерс (1970)). Многие из методов такого сорта, разработанные для задач большой размерности, базируются на симплексной технике «следующего элемента». Томлин (1976) реализовал версию алгоритма Лемке (см. Лемке (1965)), в которой используется LU-разложение матрицы, аналогичной (5.55).

Подробное обсуждение методов ранг-пространства для задач с произвольными гладкими целевыми функциями приведено в диссертации Гилла (1975).

Более полные сведения относительно правил модификации рабочего списка в методах ранг-пространства читатель может найти в статье Ленарда (1979).

5.5. ЗАДАЧИ С ОГРАНИЧЕНИЯМИ СПЕЦИАЛЬНОГО ВИДА

На практике нередко встречаются задачи со специфичными ограничениями, и, учитывая эту специфику в алгоритмах, можно существенно повышать их эффективность. Мы рассмотрим два наиболее распространенных типа задач такого рода: задачи, где все условия суть простые ограничения на переменные, и задачи, совокупность условий которых представляет собой смесь простых ограничений на переменные и ограничений общего вида.

5.5.1. МИНИМИЗАЦИЯ ПРИ ПРОСТЫХ ОГРАНИЧЕНИЯХ НА ПЕРЕМЕННЫЕ

Важным в прикладном отношении частным случаем общей постановки LIP является задача, включающая только простые ограничения на переменные:

$$\text{найти } \min_{x \in E^n} F(x) \tag{5.63}$$

$$\text{при ограничениях } l \leqslant x \leqslant u.$$

С формальной точки зрения это — задача с $2n$ условиями типа линейных неравенств. Ясно, однако, что специфика последних, состоящая в том, что в любой точке в каждой паре двусторонних ограничений равенством может стать не более одного (если l_i не совпадает с u_i) и что матрица произвольного набора активных ограничений имеет полный ранг, сулит значительные алгоритмические преимущества.

Ограничение из (5.63) становится активным, когда соответствующая переменная принимает граничное значение. Задать для (5.63) рабочий список — значит разбить вектор переменных на «свободную» и «фиксированную» составляющие x_{FR} и x_{FX}. Согласно терминологии разд. 5.2.1, число компонент x_{FX} (фиксируемых переменных) на очередной итерации будем обозначать через t_k. Отвечающая им матрица $\hat{A}_k$ состоит из строк единичной матрицы, взятых с нужными знаками; в качестве столбцов Z_k можно использовать остальные строки единичной матрицы.

Ниже рассмотрена типовая итерация поиска решения задачи (5.63) по схеме активного набора. Как обычно, чтобы не загромождать формул, индекс итерации k опущен. Компоненты направления поиска с номерами фиксируемых переменных (их число равно t) должны быть нулевыми. Таким образом, определить надо только $n-t$ его «свободных» компонент. Индексом FR будут помечаться отвечающие им $(n-t)$-мерные векторы и матрицы; через g_{FR} будем обозначать $Z_k^T g$ и т. д.

В приложении к задаче (5.63) формулы всех алгоритмов из разд. 5.1.2 становятся особенно простыми. Например, ньютоновская система принимает такой вид:

$$G_{FR}p_{FR} = -g_{FR}. \tag{5.64}$$

При решении (5.63) дискретным ньютоновским методом конечными разностями придется аппроксимировать только элементы матрицы G_{FR}. Подобно ньютоновской схеме для (5.63), сильно упрощается и квазиньютоновская. Пусть B_{FR} есть $(n-t)$-мерное приближение матрицы Гессе по свободным переменным. Тогда системой, определяющей вектор p_{FR}, будет

$$B_{FR}p_{FR} = -g_{FR}.$$

BFGS-формула пересчета матрицы B_{FR} (при сохранении рабочего списка неизменным) выглядит следующим образом:

$$\overline{B}_{FR} = B_{FR} + \frac{1}{\bar{g}_{FR}^T p_{FR}} \bar{g}_{FR}\bar{g}_{FR}^T + \frac{1}{\alpha y_{FR}^T p_{FR}} y_{FR} y_{FR}^T.$$

Специфика матрицы $\hat{A}$ ограничений рабочего списка задачи (5.63) сильно упрощает расчет оценок множителей Лагранжа. Допустим, что j-й фиксируемой переменной является x_i. Тогда любой из рассмотренных в разд. 5.1.5.1 методов вычисления оценок первого порядка даст для j-го множителя одно и то же значение — взятую с соответствующим знаком i-ю компоненту градиента целевой функции. Если $x_i = l_i$, эта оценка равна g_i, а при $x_i = u_i$ упомянутой оценкой j-го множителя Лагранжа будет $-g_i$. Оценки второго порядка для (5.63) вычисляются по формуле вида

$$\eta_L = \lambda + Gp_{FR}.$$

Здесь λ — вектор линейных оценок, p_{t+R} — ньютоновское направление (т. е. решение системы (5.64)), а G — матрица, состоящая из элементов G, которые принадлежат строкам, отвечающим фиксированным переменным, и столбцам, соответствующим свободным. Следовательно, как линейное, так и квадратичное оценивание множителей Лагранжа для задачи (5.63) не требует решения никаких систем уравнений.

Простота вычисления оценок множителей может послужить основанием для применения к (5.63) методов, в которых допускается одновременный отход из рабочего списка сразу нескольких ограничений. Таким образом, если стремиться учесть специфику (5.63) в полной мере, то можно и отказаться от общей схемы поиска оптимума, изложенной в разд. 5.2.1.

Применяя для решения задачи (5.63) алгоритмы с конечно-разностной аппроксимацией производных, вычисления легко организовать так, чтобы обращения к процедуре расчета значений целевой функции в недопустимых точках были исключены. Последнее важно, когда эта функция определена только в допустимой области. Соответствующие модификации касаются лишь формул, по которым строятся оценки первых и вторых производных от F. Предположим, к примеру, что на очередной итерации j-я переменная x_j имеет одно из своих граничных значений. Чтобы решить, следует ли зафиксировать ее на этом значении, нужна оценка множителя Лагранжа. Коль скоро используемый метод оперирует конечно-разностными приближениями градиента, в качестве такой оценки можно взять приближение его j-й компоненты с подходящим знаком. Если значение x_j равно верхней границе u_j, для вычисления конечно-разностной аппроксимации величины g_j подойдет формула

$$\bar{g}_j = \frac{1}{h_j}(F(x) - F(x - h_j e_j)).$$

При $u_j - l_j \geqslant h_j$ в ней фигурируют только допустимые точки; в отсутствие этого неравенства точка $x - h_j e_j$ вышла бы за нижнюю границу по j-й компоненте. Чтобы не выйти при оценивании производных за пределы допустимого множества, специфицируя конечно-разностные формулы надо применять и для свободных переменных, если их значения близки к граничным. Шаги в таких случаях делают в сторону от ближайшей границы. Например, центральную конечно-разностную аппроксимацию производной вблизи нижней границы следует вычислять по формуле

$$\bar{g}_j = \frac{1}{2h_j}(4F(x + h_j e_j) - 3F(x) - F(x + 2h_j)),$$

причем должно быть $u_j - l_j \geqslant 2h_j$.

Выбор направлений спуска в процедурах поиска оптимума при простых ограничениях фактически осуществляется теми же способами, что и в методах безусловной минимизации. Поэтому последние

ближе к методам решения (5.63), чем методы предыдущих разделов, рассчитанные на задачи с линейными ограничениями общего вида. Стоит также отметить, что задачи на *безусловный минимум* нередко предпочтительнее решать методами оптимизации при простых ограничениях, вводя искусственные двусторонние границы. Здесь надо указать на следующее. Во-первых, для большинства прикладных задач нетрудно предсказывать разумные диапазоны изменения переменных, заведомо содержащих искомую точку, и если введение соответствующих границ неожиданно повлияет на решение, то это послужит признаком дефекта формулировки критерия. Во-вторых, наличие границ, возможно, предотвратит вычисления функции в бессмысленных точках и тем самым благоприятно отразится на ходе решения. При этом, если метод оптимизации при простых ограничениях реализован качественно, работать с ним будет не труднее, чем с аналогичным методом поиска оптимума без ограничений.

*5.5.2. ЗАДАЧА СО СМЕСЬЮ ПРОСТЫХ ОГРАНИЧЕНИЙ И ОГРАНИЧЕНИЙ ОБЩЕГО ВИДА

Выше, когда речь шла о методах минимизации при линейных ограничениях общего вида, мы не выделяли задач, в которых среди этих ограничений присутствуют простые. Специальные приемы воплощения схемы активного набора применительно к задачам такого сорта описаны в этом разделе.

На типичной итерации решения задачи со «смешанными ограничениями» рабочий список будет содержать и простые ограничения, и ограничения общего вида. Допустим, что число первых равно r, а вторых $l-r$; все векторы и матрицы, относящиеся к свободным и фиксируемым переменным, будем помечать индексами FR и FX соответственно. Тогда в предположении (которое делается ради наглядности выкладок), что фиксируются последние переменные, матрицу $\hat{A}_k$ можно представить следующим образом:

$$\hat{A}_k = \begin{pmatrix} \hat{A}_{FR} & \hat{A}_{FX} \\ 0 & I_r \end{pmatrix}. \tag{5.65}$$

Здесь $\hat{A}_{FX}$ есть $(l-r)\times(n-r)$-матрица. Специальная структура (5.65) позволяет организовать расчет направления поиска и оценок множителей Лагранжа экономнее, чем в общем случае.

Начнем с техники вычисления направлений поиска. В силу специфики $\hat{A}_k$ любая $n\times(n-l)$ матрица Z_k векторов базиса нуль-пространства $\hat{A}_k$ должна выглядеть так:

$$Z_k = \begin{pmatrix} Z_{FR} \\ 0 \end{pmatrix}.$$

Здесь Z_{FI} есть некоторая $(n-r)\times(n-l)$-матрица, составленная из векторов базиса нуль-пространства $\bar{A}_{IR}$. Поэтому, например, обычная система для вычисления вектора p_Z, по которому строится ньютоновское направление $p=(p_{ZR}, 0)$, преобразуется к виду

$$Z_{IR}^T G_{FI} Z_{FI} p_Z = -Z_{IR}^T g_{FR}$$

(ср. с (5.12)). По существу, здесь можно говорить о понижении размерности, равном количеству зафиксированных переменных. Расчет оценок множителей Лагранжа тоже можно ориентировать на специфику $\bar{A}_k$, снизив тем самым его трудоемкость. Представим l-мерный вектор $\bar{\lambda}$ объединением $(l-r)$-мерной составляющей $\bar{\lambda}_G$ (множители ограничений общего вида) и r-мерной $\bar{\lambda}_B$ (множители, отвечающие фиксируемым переменным). Тогда уравнение для $\bar{\lambda}$ можно записать так:

$$\bar{A}^T\bar{\lambda}\equiv\begin{pmatrix}\bar{A}_{FR}^T & 0\\ \bar{A}_F^T & I_r\end{pmatrix}\begin{pmatrix}\bar{\lambda}_G\\ \bar{\lambda}_B\end{pmatrix}=\begin{pmatrix}\bar{A}_{FR}^T\bar{\lambda}_G\\ \bar{A}_{FX}^T\bar{\lambda}_G+\bar{\lambda}_B\end{pmatrix}-\begin{pmatrix}g_{FR}\\ g_{FX}\end{pmatrix}. \quad (5.66)$$

Отсюда видно, что для подсчета линейных оценок множителей достаточно найти решение в смысле наименьших квадратов для первых $l-r$ уравнений из (5.66)

$$\bar{A}_{FR}^T\bar{\lambda}_G\approx g_{FR}$$

и взять

$$\bar{\lambda}_B=g_{FX}-\bar{A}_{FX}^T\bar{\lambda}_G.$$

Изменения блоков матрицы $\bar{A}_k$ при вариациях рабочего списка зависят от типа вводимого или выводимого ограничения. Если это — простое ограничение, поменяются столбцовые размерности матриц $\bar{A}_{FR}$ и $\bar{A}_{FX}$; в противном случае изменятся их строковые размерности.

Эффективное воплощение представленной техники вычислений возможно только на базе методов пересчета разложений матрицы $\bar{A}_{FR}$ (при изменении состава ее строк и столбцов), и такие методы существуют. Тем не менее надо сказать, что используют эту технику нечасто. Объясняется это трудностью ее программной реализации. Не желая писать сложных программ, часто отказываются от учета специфики простых ограничений и обрабатывают их так же, как ограничения общего вида. Подобная практика может быть оправдана, когда повышение трудоемкости итерации, с которым она связана, незначительно по отношению к трудоемкости вычисления целевой функции. Для малых задач с целевыми функциями достаточно сложной структуры такое соотношение характерно. Однако если говорить о задачах линейного или квадратичного программирования, то эффективность методов их решения существенно зависит от качества организации вспомогательных вычислений. В частности,

если есть основания полагать, что на каждой итерации процесса решения квадратичной задачи в рабочем списке будет существенная доля простых ограничений, то скорее всего вычисления стоит организовать так, как показано выше.

Замечания и избранная библиография к разделу 5.5

Алгоритм решения задач минимизации нелинейной функции при простых ограничениях, представленный в разд. 5.5.1, разработан Гиллом и Мюрреем (1976b). Есть и другой подход к решению таких задач. Он заключается в том, чтобы снимать их ограничения, используя специальные замены переменных, т. е. сводить их к задачам безусловной минимизации (см. разд. 7.4.1.1). Однако *применять для решения последних обычные методы оптимизации без ограничений не рекомендуется* (см. Сиссер (1981)). Следует отметить также, что понижения размерности, характерного для методов активного набора, при использовании схемы замены переменных не будет.

Когда все ограничения задачи простые, а ее целевая функция квадратична, поиск решения можно организовать значительно эффективнее, чем для общей задачи квадратичного программирования (см. Флетчер и Джексон (1974), Гилл и Мюррей (1976b, 1977a)). В этом случае ценой незначительных усилий можно выбрать «хороший» начальный список фиксируемых переменных. Кроме того, если матрица G положительно определена, то простому вычислению поддается выигрыш по критерию, обеспечиваемый выводом ограничения из рабочего списка при условии, что на следующей итерации этот список не придется расширить.

5.6. БОЛЬШИЕ ЗАДАЧИ С ЛИНЕЙНЫМИ ОГРАНИЧЕНИЯМИ

5.6.1. МЕТОДЫ РЕШЕНИЯ БОЛЬШИХ ЗАДАЧ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

Задачи линейного программирования принято представлять в следующей *стандартной форме*:

$$\text{стандартная форма LP: найти } \min_{x \in R^n} c^T x$$

$$\text{при ограничениях } Ax = b,$$
$$l \leqslant x \leqslant u.$$

Распространенность именно этой, а не какой-либо другой формы объясняется как историческими причинами, так и ее удобством для эффективной и экономной по памяти организации поиска решения. Отметим, что в стандартной постановке *все ограничения общего вида являются равенствами*, а неравенства допускаются только в простых ограничениях на переменные.

Между формой (5.42) и стандартной формой существует очевидная эквивалентность. Точнее говоря, любая задача, представленная в одной из форм, может быть преобразована в задачу другой формы. Приведение всевозможных задач к стандартному виду выполняют с помощью целого ряда общепринятых приемов. В частности, чтобы перевести задачу (5.42) в стандартную, надо ввести в каждое из ее ограничений свою вспомогательную переменную определенного знака и заменить неравенства равенствами. Так, например, ограничение $a^T x \geqslant b$ надо заменить на $a^T x - y = b$, наложив на y условие неотрицательности $y \geqslant 0$. Хотя при подобных преобразованиях размерность задачи может существенно возрастать, утяжеленно вычислений и увеличение объема памяти, связанные с появлением вспомогательных переменных, будут незначительны (поскольку последние не входят в целевую функцию и ограничены только простыми неравенствами).

Допустим, что в стандартной задаче есть m ограничений-равенств общего вида. При этом обычное для линейного программирования требование невырожденности матрицы ограничений рабочего списка означает, что на каждой итерации на своих граничных значениях будут фиксироваться ровно $n-m$ переменных. (Допустимую точку, где $n-m$ переменных принимают граничные значения, принято называть *базисным допустимым решением* стандартной задачи.)

Отбросив на время индекс k очередной итерации, и пусть x — текущая точка процесса решения стандартной задачи с m равенствами. Не умаляя общности рассуждений, будем считать, что в x на граничных значениях зафиксированы $n-m$ последних переменных. Тогда x есть решение системы вида

$$\hat{A}x = \begin{pmatrix} B & N \\ 0 & I_{n-m} \end{pmatrix} \begin{pmatrix} x_B \\ x_N \end{pmatrix} = \begin{pmatrix} b \\ b_N \end{pmatrix}. \qquad (5.67)$$

Здесь B есть невырожденная $m \times m$-подматрица, составленная из ряда столбцов A и именуемая *базисной*. *Базисными* называют также формирующие B столбцы A и отвечающие им переменные задачи (в (5.67) вектор этих переменных обозначен через x_B). Столбцы, относящиеся к N, и переменные x_N, ассоциируемые с ними, принято называть *небазисными*. Компонентами вектора b_N из (5.67) будут граничные значения небазисных переменных, т. е. соответствующие компоненты либо l, либо u в зависимости от того, на какой из своих границ зафиксирована небазисная переменная.

Точка x является оптимальной для задачи с ограничениями-равенствами (5.67). Чтобы определить, будет ли она оптимальной и для исходной задачи, надо вычислить в ней значения множителей Лагранжа. Разобьем составленный из них вектор на m-мерный подвектор π (в него войдут множители исходных ограничений-равенств) и $(n-m)$-мерный σ (отвечающий зафиксированным переменным).

Тогда уравнения (5.43) для множителей можно будет записать так:

$$A^T\lambda = \begin{pmatrix} B^T & 0 \\ N^T & I_{n-m} \end{pmatrix}\begin{pmatrix} \pi \\ \sigma \end{pmatrix} = \begin{pmatrix} B^T\pi \\ N^T\pi + \sigma \end{pmatrix} = \begin{pmatrix} c_B \\ c_N \end{pmatrix}. \qquad (5.68)$$

Здесь c_B и c_N суть векторы, образованные из компонент c, отвечающих соответственно базисным и небазисным переменным.

Из (5.68) видно, что вектор π будет решением невырожденной линейной системы

$$B^T\pi = c_B. \qquad (5.69)$$

Заметим, что знаки компонент π в условиях оптимальности для исходной задачи не оговариваются. После того как π вычислен, множители зафиксированных переменных легко определяются по второй группе уравнений в (5.68). Ее решение дается формулой

$$\sigma = c_N - N^T\pi.$$

В терминологии линейного программирования множители σ_j называются *оценками замещения* или *приведенными ценами*.

В оптимальной точке компоненты вектора σ должны иметь определенные знаки. Точнее, те из них, которые соответствуют переменным, зафиксированным на нижних границах, должны быть неотрицательны, а остальные — неположительны. Коль скоро это требование в x выполнено, можно утверждать, что x — решение задачи. В противном случае одна из небазисных переменных будет освобождена, т. е. одно из простых ограничений будет выведено из рабочего списка. Предположим, что такой переменной является x_s, имеющая нижнее граничное значение, и представим очередное направление поиска p в виде $(p_B\ p_N)$. Тогда компонента вектора p_N, отвечающая x_s, должна быть единичной, и это — его единственная ненулевая компонента. Что же касается p_B, то он определится из уравнений типа (5.44), (5.45), которые в рассматриваемой ситуации выглядят так:

$$\begin{pmatrix} B & N \\ 0 & I_{n-m} \end{pmatrix}\begin{pmatrix} p_B \\ p_N \end{pmatrix} = e_s.$$

Соответственно p_B есть решение системы

$$Bp_B = -a_s, \qquad (5.70)$$

где через a_s обозначен s-й *столбец* матрицы A.

Двигаться вдоль p, уменьшая значение целевой функции, надо до тех пор, пока какая-то из базисных переменных не примет свое граничное значение, т. е. пока не проявится какое-то из отсутствующих в текущем рабочем списке простых ограничений. Оно войдет в рабочий список для следующей итерации. Как следствие, матрицы B и N изменятся, а точнее, они будут набираться из других столбцов A. В результате полной итерации симплекс-метода происходит обмен ролями для некоторой пары базисной и небазисной переменных.

В терминологии линейного программирования соответствующие изменения рабочего списка называют сменами базиса. Отметим, что в отличие от схемы, изложенной в разд. 5.5.2, где во время итераций в матрице $\hat{A}_{FR}$ могут происходить замены, отбрасывания и присоединения столбцов и строк, размеры матрицы B в симплекс-методе решения стандартной задачи *фиксированы*.

В большинстве современных реализаций симплекс-метода, рассчитанных на большие задачи линейного программирования, для решения систем (5.69) и (5.70) используется LU-факторизация матрицы B. На первом шаге она дает представление

$$L^{-1}B = M_{m-1}\ldots M_2M_1B = U, \tag{5.71}$$

где каждая M_j есть элементарная нижняя треугольная, а U — верхняя треугольная матрицы (ради простоты мы предположили, что перестановок строк и столбцов, которые могли бы понадобиться для обеспечения численной устойчивости процедуры факторизации, не потребовалось; ср. с (2.40)). Имея (5.71), для поиска решения системы типа $Bx=b$ надо вычислить $y=M_{m-1}\ldots M_2M_1b$, а затем подстановками назад решить систему $Ux=y$. (Здесь уместно отметить, что подстановки вперед и назад организуются одинаково просто независимо от того, как в памяти машины хранится матрица — по столбцам или по строкам.) Расчет вектора y состоит в прямом поочередном применении преобразований M_j. Расчет x по y требует обратного прохода по столбцам или строкам матрицы U. (Аналогичные процедуры используются также для решения систем типа $B^Tz=c$.)

Возможность эффективного решения задачи линейного программирования с очень большим числом переменных существует только тогда, когда матрица A достаточно разрежена (т. е. число ее ненулевых позиций достаточно мало). В этом случае для записи ненулей факторов LU-разложения матрицы B потребуется разумный объем памяти. Надо иметь в виду, что количество этих ненулей сильно зависит от расстановки строк и столбцов B. Следовательно, есть смысл тратить определенные усилия на поиск такой нумерации переменных и ограничений, при которой число дополнительных ненулей в LU-разложении B будет умеренным. Соответствующие процедуры иногда называют *подбором упорядочения ведущих элементов*. В идеале хотелось бы, чтобы перестановкой строк и столбцов матрицу B можно было преобразовать в нижнюю треугольную — тогда для решения систем (5.69) и (5.70) никакой факторизации не понадобится. В действительности же базисные матрицы прикладных задач обычно «почти» преобразуемы в треугольные, что и используется существующими методами упорядочения ведущих элементов.

При обмене столбцами (в результате смен базиса) между B и N факторы LU-разложения B должны пересчитываться. Во время таких пересчетов в список определяющих L элементарных матриц добавляются новые $\{M_j\}$ и формируются новые столбцы типа $L^{-1}a_j$.

Как правило, каждый пересчет приводит к появлению в U новых ненулевых позиций. Память, отводимую для хранения U, организуют так, чтобы вновь возникающие ненули помещались в конец текущей записи U, но при этом были легко доступны на соответствующих этапах решения систем (5.69) и (5.70) (познакомиться с такими способами хранения U можно по литературе, указанной в замечаниях).

По мере выполнения итераций симплекс-метода и связанных с ними пересчетов LU-разложения базисной матрицы объем памяти, занимаемой факторами этого разложения, растет. Обычно после серии пересчетов он оказывается много больше того, который потребовался бы для записи факторов LU-разложения, построенного в текущей точке «с нуля» на какой-нибудь из разумных схем с выбором упорядочения ведущих элементов. Поэтому через определенные промежутки времени возникает необходимость построить разложение заново. Как правило, такие обновления, именуемые *повторными обращениями*, осуществляются с фиксируемой частотой, но могут выполняться и раньше намеченного срока, если память, отведенная для записи факторов разложения B, исчерпывается.

5.6.2. Большие задачи с линейными ограничениями и нелинейными критериями

В этом разделе рассматриваются задачи вида

$$\begin{aligned} &\text{найти } \min_{x \in \mathcal{R}^n} F(x) \\ &\text{при ограничениях } Ax = b, \\ &\qquad\qquad\qquad l \leqslant x \leqslant u. \end{aligned} \tag{5.72}$$

Будем считать, что количество переменных в (5.72) «велико», а матрица A разрежена. Заметьте, что структура ограничений из (5.72) точно такая же, как в стандартной форме LP. Соответственно и описанные ниже методы будут сильно опираться на технику, разработанную для решения линейных задач большой размерности.

Как правило, возможности выбора способов решения большой задачи относительно небогаты. Дело в том, что многие вычислительные процедуры, применимые для широкого круга малых задач, при увеличении размерности становятся чрезмерно трудоемкими или требуют слишком много памяти. В то же время оптимальная среди оставшихся возможностей становится менее очевидной, поскольку специфика конкретной задачи и детали реализации доступных схем приобретают существенно больший вес.

При переходе к большим задачам нередко приходится менять представления о том, какую стратегию решения считать «идеальной». Подход, заведомо неэффективный в случаях малой размерности, для некоторых больших задач может оказаться наилучшим. Требуют пересмотра и определенные стандартные предположения об

относительной трудоемкости разных этапов процесса поиска решения. Например, мерой эффективности обычного алгоритма безусловной оптимизации принято считать количество обращений к программируемым пользователем процедурам вычисления функций (целевой функции, ее градиента), необходимое для отыскания решения с заданной точностью. Хотя такой способ измерения эффективности может показаться чрезмерно упрощенным, для большинства малых задач он вполне пригоден. Это объясняется тем, что для них объем не связанных с подсчетом функций арифметических операций, выполняемых на каждом шаге, относительно мал (его максимальный порядок есть n^2) и трудоемкость манипуляций с памятью незначительна. Однако уже для задач *средних размеров* затраты чисто алгебраических блоков оптимизационных схем и процедур организации доступа к данным становятся существенными.

На каждой итерации схемы активного набора, примененной к (5.72), матрица A будет содержать все строки A и какие-то строки единичной матрицы, отвечающие переменным, фиксируемым *на* граничных значениях. При этом в отличие от линейного случая заранее не известно, сколько переменных должны достигать в решении своих границ, т. е. число фиксируемых переменных может меняться в процессе поиска. Следовательно, любое обобщение из (5.72) техники деления переменных на базисные и небазисные должно допускать вариации числа небазисных (фиксируемых) переменных. Предположим, что на очередной итерации это число равно r. Тогда матрицу A (с точностью до перестановки столбцов) можно будет расчленить так:

$$A=(B\ \ S\ \ N). \tag{5.73}$$

Здесь N есть $m\times r$-матрица, столбцы которой отвечают небазисным переменным; B — «базисная» невырожденная $m\times m$-матрица, набранная из столбцов A, соответствующих базисным переменным; S — матрица, состоящая из $n-m-r$ столбцов A, отвечающих остальным переменным, которые впредь будем называть *супербазисными*. Число супербазисных переменных представляет собой количество степеней свободы, оставшихся после закрепления небазисных (поскольку в рабочем списке будет $m+r$ ограничений). В линейном случае, рассмотренном в разд. 5.6.1, матрица S отсутствует, а число столбцов всегда равно $n-m$.

На рассматриваемой итерации ограничения из рабочего списка (записанные в удобной форме) будут выглядеть следующим образом:

$$\hat{A}x=\begin{pmatrix} B & S & N \\ 0 & 0 & I \end{pmatrix}\begin{pmatrix} x_B \\ x_S \\ x_N \end{pmatrix}=\begin{pmatrix} b \\ b_N \end{pmatrix}. \tag{5.74}$$

Компоненты вектора b_N совпадают с соответствующими компонентами l или u в зависимости от того, на каком из граничных значений фиксируется переменная.

Чтобы получить способ расчета направления спуска Zp_Z для задачи (5.72), надо задать формулы вычисления его базисной, небазисной и супербазисной составляющих. Матрицу Z, ортогональную строкам $\hat{A}$, определяют так:

$$Z=\begin{pmatrix} -B^{-1}S \\ I \\ 0 \end{pmatrix} \tag{5.75}$$

Ни саму Z, ни B^{-1} в явном виде не вычисляют. Для подсчета матрично-векторных произведений с Z и Z^T достаточно иметь треугольное разложение квадратной матрицы B.

При построении Z по формуле (5.75) расчленение вектора переменных на три составляющие находит прямое отражение в процедуре вычисления направления поиска p. Представив $p=Zp_Z$ в виде $(p_B\ p_S\ p_N)$, из (5.75) получим, что $p_N=0$ и

$$Bp_B=-Sp_S. \tag{5.76}$$

Равенство (5.76) показывает, что p_B однозначно выражается через p_S. Таким образом, минимизацию можно организовать как поиск в подпространстве супербазисных переменных. В рассматриваемой схеме расчета p вектор p_S играет в точности такую же роль, которая была у p_Z в методах, описанных выше, и полностью определяет p. После того как p_S построен, p_B находят решением системы (5.76).

5.6.2.1. Выбор супербазисной составляющей направления поиска. Соображения, лежащие в основе методов выбора «хорошего» вектора p_S, аналогичны приведенным в разд. 5.1.2 относительно вектора p_Z. В наиболее эффективных методах p_S определяется по *квадратичной аппроксимации* целевой функции с учетом ограничения (5.76). Точнее говоря, p_S подбирается так, чтобы построенное по нему направление поиска было решением квадратичной задачи вида

$$\begin{aligned} &\text{найти } \min_{p\in\mathbb{R}^n} g^Tp+\frac{1}{2}p^THp \\ &\text{при ограничениях } \hat{A}p=0, \end{aligned} \tag{5.77}$$

где H — некоторая аппроксимация матрицы Гессе функции $F(x)$. Для вычисления такого вектора p_S нужна только спроектированная матрица Z^THZ. Он будет решением линейной системы

$$Z^THZp_S=-Z^Tg, \tag{5.78}$$

где Z определена формулой (5.75). Заметим, что $Z^Tg=-S^TB^{-T}g_B+g_S$.

Реальная возможность решить систему (5.78) с помощью факторизации ее матрицы существует лишь при условии, что размерность

матрицы достаточно мала. Дело в том, что, как правило, она будет сильно заполненной даже при разреженных A и H. Значит, предлагаемый способ расчета p_s практичен только в случаях, когда заранее известно, что число ограничений в рабочем списке будет достаточно велико на каждой итерации. К счастью, для многих больших задач это условие соблюдается, причем можно дать априорную нижнюю оценку количества ограничений, которые будут выполнены в решении как равенства. Предположим, к примеру, что целевая функция нелинейна по малому числу переменных (эта ситуация типична, когда нелинейности возникают в результате уточнения изначально линейной задачи). Такая $F(x)$ будет выглядеть следующим образом:

$$F(x) = f(\tilde{x}) + c^T x,$$

где $\tilde{x} = (x_1, x_2, \ldots, x_q)^T$ — вектор небольшой размерности и $f(\tilde{x})$ — некоторая дифференцируемая нелинейная функция. Если известно, что в рассматриваемом случае оптимальная точка удовлетворяет достаточным условиям второго порядка, то можно утверждать, что в ней размерность спроектированной матрицы Гессе не превзойдет q.

Даже при умеренных размерностях систем (5.78) методы ньютоновского типа с факторизацией G_Z для больших задач, как правило, оказываются непрактичными; слишком много вычислений требуется для построения спроектированных матриц Гессе заново на каждом шаге (напомним, что всякая операция с Z предполагает решение системы уравнений с матрицей B). В отличие от них квазиньютоновские методы адаптируются к большим задачам очень эффективно. Имеется в виду обсуждавшаяся в разд. 5.2.4.2 схема расчетов с пошаговой корректировкой разложения по Холесскому квазиньютоновского приближения спроектированной матрицы Гессе.

Среди прикладных задач большой размерности встречаются и такие, в решениях которых число активных ограничений значительно меньше числа переменных. Например, иногда большая часть ограничений вводится, чтобы исключить «неразумные» решения, и оказывается неактивной в «разумном». Если памяти для факторизации $(n-l)$-мерной матрицы из (5.78) не хватает, решение (5.78) можно искать методами сопряженных градиентов типа рассмотренных в разд. 5.1.2.5. В частности, если вторые производные доступны и матрица Гессе G разрежена, реально воспользоваться схемой ньютоновского типа, в которой p_s определяется в результате итерационного поиска решения системы (5.78) с $H = G$. Надо только, чтобы подсчет значительного числа произведений вида $Z^T G Z v$ при поэтапной организации, когда сначала вычисляется $v_1 = Zv$, затем $v_2 = Gv_1$ и наконец $v_3 = Z^T v_2 = Z^T G Z v$, не был слишком трудоемок. Аналогичную процедуру можно использовать и в случаях, когда доступна не сама матрица Гессе, а ее разреженная аппроксимация. К итерационному

методу поиска решения системы (5.78) предъявляются два требования: во-первых, он должен быть быстросходящимся и, во-вторых, должен давать приемлемое направление спуска даже при знаконеопределенной Z^THZ.

***5.6.2.2. Изменения в рабочем списке.** Пока норма $\|Z^Tg\|$ «велика», оптимизируются значения только базисных и супербазисных переменных. Когда какая-нибудь из них выходит на свою границу, ее «переводят» в разряд небазисных, рабочий список корректируется соответствующим образом и счет продолжается дальше.

Допустим теперь, что величина $\|Z^Tg\|$ стала «малой» (ср. с (5.2)). Это рассматривается как признак «близости» очередной точки к оптимальной для текущего рабочего списка. Здесь надо определить, можно ли еще изменить значение целевой функции, если освободить одну из небазисных переменных. Для этого требуется вычислить оценки множителей Лагранжа. Определяющая их система уравнений выглядит так:

$$\begin{pmatrix} B^T & 0 \\ S^T & 0 \\ N^T & I \end{pmatrix} \begin{pmatrix} \pi \\ \sigma \end{pmatrix} \approx \begin{pmatrix} g_B \\ g_S \\ g_N \end{pmatrix}. \tag{5.79}$$

В качестве искомых π и σ берут

$$\pi = B^{-T} g_B, \tag{5.80}$$

$$\sigma = g_N - N^T\pi. \tag{5.81}$$

Они будут точными значениями соответствующих частей вектора множителей Лагранжа, если $Z^Tg = 0$. Действительно, в этом случае

$$g_N = S^TB^{-T}g_B = S^T\pi,$$

т. е. π и σ из (5.80), (5.81) при $Z^Tg = 0$ обеспечивают в (5.79) точное равенство. Вектор σ составлен из оценок множителей, отвечающих простым ограничениям на небазисные переменные. В решении его компоненты должны иметь определенные знаки. Если же правило знаков для какой-то компоненты в σ нарушено, отвечающую ей небазисную компоненту переводят в разряд супербазисных и оптимизация продолжается.

Заканчивая разбор методов решения больших задач, просуммируем их отличия от рассмотренных прежде обычных методов активного набора. Эти отличия существенны, хотя идеи, лежащие в основе и тех и других методов, одинаковы. Во-первых, нуль-пространство матрицы $\hat{A}$ определяется теперь путем *разбиения переменных*, а не факторизацией, приводящей к Z с ортогональными столбцами. При этом из выражения (5.75) для Z видно, что плохая обусловленность базисной матрицы B может сильно снизить точность всех построений алгоритма и круто

изменить веса переменных. Если столбцы Z ортонормальны, выполняется неравенство $\|Z^Tg\| \leqslant \|g\|$; в противном случае вектор Z^Tg оказывается «неотмасштабированным». Поскольку применение ортонормальных Z для больших задач, как правило, непрактично (и по памяти, и по трудоемкости), при построении p в приближений спроектированной матрицы Гессе в методах решения таких задач нередко возникают дополнительные численные трудности.

Далее оценки множителей Лагранжа, найденные по формулам (5.80) и (5.81), точны только при $Z^Tg=0$, в окрестность, в которой они будут иметь правильные знаки, зависит от обусловленности B. Значит, когда норма $\|Z^Tg\|$ просто «мала» (а не очень мала»), освобождать небазисную переменную с неподходящей по знаку компонентой вектора σ может оказаться бесполезным. Хотя какое-то направление спуска будет получено, не исключено, что на одной из ближайших итераций освобожденная переменная выйдет за прежнюю границу и ее снова придется зафиксировать. При решении малых задач, когда оценки множителей вычисляются с применением ортонормальной Z, аналогичная проблема стоит куда менее остро, поскольку области корректности знаков оценок будут зависеть только от обусловленности A. Уменьшение надежности оценок множителей Лагранжа при переходе от ортогональных Z к (5.75) неизбежно, и это всегда надо принимать во внимание.

Наконец, следует сказать о значительной трудоемкости вычислений и операций над памятью, связанных с построением и пересчетом разложений матрицы B. Для многих больших задач затраты алгебраических блоков процедуры минимизации намного перекрывают затраты на вычисление значений целевой функции.

Замечания и избранная библиография к разделу 5.6

В настоящее время есть немало коммерческих реализаций симплекс-метода, рассчитанных на задачи с тысячами переменных. Интересная история раннего этапа разработок коммерческих пакетов линейного программирования хорошо изложена в статье Орчард-Хейса (1978а). За деталями проектирования и реализации математического обеспечения линейных задач большой размерности мы отсылаем читателя к работам Орчард-Хейса (1968, 1978b, c), Била (1975), Бенишу и др. (1977), Гринберга (1978a, b) и Муртафа (1981).

Главные различия в реализациях симплекс-метода относятся к способам решения уравнений (5.69) и (5.70). Первые программы оперировали явным представлением матрицы, обратной к базисной (см., например, Орчард-Хейс (1968)). Более поздние работают с LU-разложением (Бартелс и Голуб (1969), Форрест и Томлин (1972)), с LQ-разложением без запоминания Q (Гилл и Мюррей (1973c)), с

LQ-разложением, где Q хранится в виде произведения диагональной и ортогональной матриц (Гилл, Мюррей, Сондерс (1975)).

Прогресс в линейном программировании в основном стимулировался достижениями в области методов решения больших разреженных систем линейных уравнений. Одна из первых схем выбора упорядочения столбцов и строк базисной матрицы B была разработана Марковицем (1957). В ней выбор очередного ведущего элемента во время LU-факторизации осуществляется путем локальной минимизации возможного заполнения в строках и столбцах, которые еще предстоит факторизовать. Схема Марковица была реализована Райдом (1975, 1976). Холлерман и Рарик (1971, 1972) предложили метод перестановок, предназначенный для преобразования базисной матрицы *до того*, как начнется ее факторизация. Перестановки строк и столбцов в этом методе сначала подбираются так, чтобы переставленная матрица отличалась от нижней треугольной только несколькими диагональными блоками, именуемыми *горбами* (см. также Тарьян (1972), Дафф и Райд (1978)). После этого каждый горб обрабатывается до тех пор, пока не станет отличаться от нижней треугольной матрицы лишь некоторыми столбцами. Их «над-диагональные ненулевые выросты» принято называть *спайками*. Можно показать, что при последующей LU-факторизации заполнение возможно только в тех столбцах, где есть спайки.

Наиболее эффективные из современных реализаций симплекс-метода, предназначенных для решения больших задач, оперируют LU-разложением базисной матрицы B. Два основных способа *пересчета* факторов этого разложения при замене столбца B кэонструированы Форрестом и Томлином (1972) (см. также Томлин (1975а)) и Бартелсом и Голубом (см. Бартелс (1971)). Конечным итогом пересчета фактора U по методу Форреста — Томлина является замена в нем одного из столбцов. Поэтому метод лучше всего согласуется с процедурами, предполагающими запись U по столбцам. Основное его достоинство в том, что он может быть эффективно реализован в случаях, когда для хранения L и U используется внешняя или виртуальная память.

Метод Бартелса — Голуба в отличие от метода Форреста — Томлина включает определенные действия, направленные на обеспечение численной устойчивости пересчета. В нем используются перестановки строк U, и в некоторых из строк могут появляться новые ненулевые элементы. Последнее означает, что способ записи ненулей U должен быть таков, чтобы ее можно было дополнять новыми элементами, не переупорядочивая старых. Эта особенность метода предполагает хранение U целиком в оперативной памяти. Эффективная ФОРТРАН-реализация метода Бартелса — Голуба с компактной записью U по строкам разработана Райдом (1975, 1976). В ней ненули каждой строки размещаются в памяти рядом, но соседние строки могут быть записаны в несмежных областях. Программа Райда хорошо работает и в реальной, и в виртуальной

памяти. Сондерс (1976) предложил иной способ реализации метода Бартелса — Голуба. Он показал, что если перед первой факторизацией применить процедуру Хеллермана — Рарика, то удается предсказать блоки U, в которых при последующих пересчетах будут возникать новые ненули. Их можно хранить в оперативной памяти в явном виде, и тогда доступ к элементам U во время пересчетов упрощается до предела.

Некоторые важные в прикладном отношении задачи линейного программирования большой размерности характеризуются специальными структурами заполнения матриц ограничений; например, матрицы моделей динамических систем имеют «лестничную» структуру. Обсуждение таких задач выходит за рамки настоящей книги. Библиографию по этому поводу читатель найдет в сборнике под редакцией Данцига и др. (1981).

Значительный интерес в среде специалистов во всем мире вызвала недавняя публикация советского математика Хачияна (1979), где доказано, что некий алгоритм эллипсоидов находит допустимую точку системы линейных неравенств с рациональными коэффициентами за число шагов, полиномиально зависящее от (вычисляемой специальным способом) размерности системы. В этом алгоритме, принципиально разработанном Шором (см. Шор (1970, 1977), Шор и Гершович (1979)) и Немировским и Юдиным (1979), сначала строится эллипсоид, содержащий кусок допустимой области ненулевого объема, а затем – последовательность сокращающихся в объеме эллипсоидов, каждый из которых содержит в себе этот кусок. Доказательством от противного можно установить, что в конце концов центр одного из эллипсоидов окажется допустимой точкой (иначе пришлось бы признать возможным существование эллипсоида, объем которого меньше, чем объем содержащейся в нем части допустимого множества). Поскольку задачи линейного программирования всегда могут быть сведены к задачам на поиск допустимой точки, результат Хачияна о полиномиальной сходимости распространяется и на них. В то же время оценка числа итераций симплекс-метода экспоненциальна, причем есть примеры, на которых эта оценка достигается. Естественно возник вопрос: является ли теоретическое преимущество метода эллипсоидов, следующее из упомянутых оценок, в практическое? Оказалось, что, *несмотря на большое значение этого метода для теории*, виды на разработку на его основе эффективных пакетов решения больших задач линейного программирования плохие (результаты численных экспериментов с ним на некоторых задачах большой размерности можно найти у Гиэла и др. (1981а)). После появления на свет статьи Хачияна о методе эллипсоидов писали многие, и в том числе Эспвол и Стоун (1980), Гач и Ловас (1981), Гэлфон (1980), Лоулер (1980) и Вулф (1980а, b).

Одним из немногих практичных алгоритмов решения больших задач с линейными ограничениями и нелинейной целевой функцией

является метод аппроксимации (см. Гриффит и Стюарт (1961)). Его часто называют также методом последовательного линейного программирования. Известно много разных реализаций этого метода. Выбор версии обычно определялся доступной библиотекой оптимизационных программ; о методе писали Бейкер и Вентикер (1980), Бил (1974, 1978), Бэтчелор и Бил (1976).

Термин «супербазисные переменные» введен Муртафом и Сондерсом (1978). Они — авторы схемы решения больших задач с линейными ограничениями, изложенной в разд. 5.6.2. Метод Муртафа — Сондерса (разработанное ими конкретное воплощение этой схемы) оперирует разложением по Холесскому квазиньютоновской аппроксимации (в соответствии с BFGS-формулой) спроектированной матрицы Гессе, а когда памяти для записи факторов не хватает, он переключается на традиционный метод сопряженных градиентов. Переносимую ФОРТРАН-программу MINOS, реализующую этот метод, можно приобрести в Лаборатории оптимизации систем Стэнфордского университета (см. Муртаф и Сондерс (1977)).

Маршал и Шанно (1979) (см. также Маршал (1978)) предложили метод, аналогичный методу Муртафа — Сондерса и отличающийся от последнего только тем, что вместо BFGS-формулы для аппроксимации спроектированной матрицы Гессе в нем используется квазиньютоновский метод с ограниченной памятью. При этом рестарт в момент ввода в рабочий список нового ограничения не обязателен. Бакли (1975) построил квазиньютоновский метод решения больших задач, использующий $n \times n$-матрицу T вида (5.31). Лучше всего этот метод работает в случаях, когда число ограничений общего вида превышает число переменных.

*5.7. ПОИСК НАЧАЛЬНОЙ ДОПУСТИМОЙ ТОЧКИ

Все представленные в данной главе алгоритмы решения задач с линейными ограничениями были «методами допустимой точки». Это значит, что они могут стартовать только с допустимого начального приближения и генерируют последовательность точек, которые также удовлетворяют всем ограничениям. Следовательно, чтобы иметь возможность применять их в ситуациях, когда ни одной допустимой точки заранее не известно, нужно построить методы, с помощью которых такие могли бы быть найдены.

В пакетах линейного программирования задача отыскания начального допустимого приближения решается алгоритмом, именуемым *фазой I симплекс-метода*. Введем для системы из m линейных неравенств $Ax \geqslant b$ *искусственную целевую функцию* вида

$$F(x) = -\sum_{i \in \mathcal{I}} (a_i^T x - b_i),$$

где $\mathcal{I}$ есть набор индексов ограничений, нарушенных в x. Функция F представляет собой сумму «недопустимостей» и при

фиксированном $\bar{F}$ линейна. Заметьте, что $\bar{F}$ равна нулю в любой допустимой точке и положительна в любой недопустимой. Следовательно, найти допустимую точку можно, минимизируя $\bar{F}(x)$ при ограничениях $a_j^T x - b_j \geqslant 0$, $j \notin \mathcal{N}$.

Для решения поставленной оптимизационной задачи обычно используют слегка модифицированный симплекс-метод. Мы укажем две распространенные модификации, касающиеся правила выбора шага вдоль направления поиска. Во-первых, можно делать максимальный шаг, не нарушающий ни одного из выполненных в начале итерации ограничений. Обозначим через x и p текущую допустимую точку и выбранное в ней направление поиска; пусть далее $\mathcal{N}$ — набор индексов ограничений, выполненных в x как неравенства. Тогда формула расчета шага α, о котором идет речь, будет выглядеть так:

$$\alpha = \min_i \left\{ \frac{b_i - a_{ix}}{a_i^T p} \,\middle|\, i \in \mathcal{N},\ \ a_i^T p < 0 \right\}.$$

Ограничение с некоторым индексом r из $\mathcal{N}$, определяющее минимум справа, как обычно, войдет в рабочий список следующей итерации. Второй способ выбора шага — находить его из условия минимума $\bar{F}$ вдоль p. Тогда, например, если при α, вычисленном предыдущим способом, производная от $\bar{F}$ в $x+\alpha p$ по направлению p, равная

$$-\sum_{i \in \mathcal{N}} a_i^T p - a_0^T p,$$

окажется отрицательной, то шаг будет увеличен.

В обоих случаях на одной итерации могут удовлетвориться сразу несколько из нарушенных ранее ограничений. Поскольку угрозы зигзага для задачи минимизации $\bar{F}$ нет, ограничение с отрицательным множителем Лагранжа можно выводить из рабочего списка немедленно. При определенных требованиях к исходным ограничениям доказывается, что предлагаемая процедура придет за конечное число шагов в точку, которая либо будет допустимой, либо окажется вершиной вспомогательной задачи с неотрицательными множителями Лагранжа. Последнее является признаком отсутствия допустимых точек.

Когда исходная задача линейна, m будет больше n. В данном случае можно набрать n линейно независимых ограничений и определить тем самым некую (недопустимую) вершину. Соответственно для поиска допустимой точки можно будет воспользоваться симплекс-методом. В момент обнуления вспомогательной целевой функции мы получим допустимую вершину, в которой и запустим симплекс-метод решения исходной задачи.

Если $m<n$ (что нередко случается в задачах с нелинейными F), непосредственно применить симплекс-метод нельзя, так как у допустимого множества вспомогательной задачи не будет вершин.

В такой ситуации можно создать искусственные вершины, поставив дополнительные простые ограничения. Правда, чтобы не было риска исключить этими ограничениями все допустимые точки, соответствующие границы придется брать очень свободными, а это значит, что симплекс-метод найдет «неразумное» допустимое приближение (так как для него какие-то из дополнительных ограничений обязательно обратятся в равенства).

В силу сказанного при $m < n$ для поиска допустимой точки предпочтительной может оказаться несимплексная стратегия, упомянутая в разд. 5.3.1. Когда $m \ll n$, ее разумно реализовать без построения Z (см. разд. 5.4). В этом случае спроектированное направление наискорейшего спуска $p = -ZZ^T\bar{c}$ (где $\bar{c}$ — градиент вспомогательной целевой функции) вычисляют по формуле $p = -(I - YY^T)\bar{c}$, в которой строками Y служат векторы базиса пространства, натянутого на строки матрицы ограничений рабочего списка.

Замечания и избранная библиография к разделу 5.7

В современных коммерческих пакетах линейного программирования фаза I симплекс-метода полностью вытеснила применявшийся ранее способ модификации исходной задачи, позволявший сразу указать допустимое базисное приближение. Эта модификация состоит в том, что в задачу вводится ряд искусственных неотрицательных переменных, которые и становятся базисными на первом шаге. Чтобы введение этих переменных не повлияло на решение, их включают в целевую функцию с большим множителем; поэтому по мере выполнения итерации они постепенно замещаются в базисе настоящими переменными и в решении оказываются равными нулю. Данный прием преобразования линейных задач известен под названием *M-метод* (см. Данциг (1968)). На практике он не применяется.

Детали реализации фазы I симплекс-метода в коммерческих пакетах математического программирования можно найти в работах Орчард-Хейса (1968) и Гринберга (1978).

*5.8. РЕАЛИЗАЦИЯ МЕТОДОВ АКТИВНОГО НАБОРА

*5.8.1. ОПРЕДЕЛЕНИЕ НАЧАЛЬНОГО РАБОЧЕГО СПИСКА

Хотя поиск решения задачи с линейными ограничениями осуществляется в две фазы, на первой из которых определяется допустимая точка, а на второй — собственно решение, его организуют так, что всю работу фактически выполняет один алгоритм активного набора, но только сначала ему предлагается вспомогательная целевая функция, а после того, как он найдет допустимое приближение,— настоящая. Исходная точка x_0 может быть произвольной.

Важно отметить, что в момент смены целевых функций *рабочий список сохраняется*.

Эффективность любого из упомянутых в этой главе методов активного набора сильно зависит от количества ограничений в рабочем списке. Методы нуль-пространства будут работать тем лучше, чем больше строковая размерность A; для методов ранг-пространства предпочтительно, чтобы она была малой. При этом ограничения вводятся и выводятся из рабочего списка по одному, т. е. его состав меняется довольно медленно. Соответственно большое значение имеет размер начального рабочего списка.

Состав начального рабочего списка определяется перед запуском первой фазы оптимизации с помощью специальных процедур, именуемых стартовыми. Единственное обязательное требование к этому списку состоит в том, что он должен включать все линейно независимые ограничения-равенства. Прочее в основном зависит от того, к какому классу относится метод расчета направления поиска, применяемый на второй фазе оптимизации. Если это — метод нуль-пространства, в начальный рабочий список стараются ввести как можно больше ограничений. Если же на второй фазе используется метод ранг-пространства, никакие другие ограничения, кроме упомянутых выше равенств, в начальный список не включаются.

Обозначим через $\hat{A}_r$ матрицу, чьи t_r линейно независимых строк отвечают ограничениям из стартового рабочего списка. Если $t_r < n$, то найдется бесконечно много точек, для которых

$$\hat{A}_r x = \hat{b}_r. \tag{5.82}$$

Первая фаза оптимизации обычно запускается с точки $\hat{x}_0$, удовлетворяющей (5.82) и в каком-то смысле являющейся ближайшей к заданному пользователем начальному приближению x_0. Например, в качестве $\hat{x}_0$ может выступать $x_0 + p$, где p — вектор минимальной квадратичной нормы, удовлетворяющий равенству $\hat{A}_r(x_0 + p) = \hat{b}_r$. Имея LQ-разложение $\hat{A}_r$, такой вектор p вычисляют по формулам

$$p = Y_r v, \quad \text{где} \quad L_r v = b_r - \hat{A}_r x_0.$$

Здесь Y_r и L_r — соответствующие блоки LQ-разложения.

Подбор стартового рабочего списка для метода нуль-пространства можно организовывать многими способами, например включать в него все линейно независимые равенства и те из линейно независимых неравенств, для которых $|a_i^T x_0 - b_i| \leqslant \delta$, где δ — некоторое малое число.

Самые сложные стартовые процедуры применяются для больших задач линейного программирования, которые обычно ставятся в стандартной форме (см. разд. 5.6.1). Здесь требуется найти начальную базисную матрицу B_1. При этом, как правило, стартовая процедура ищет не ту матрицу, которая определяет точку, близкую

к заданной пользователем, а ту, которая удобна для треугольного разложения. Например, стартовый базис может быть набран только из вспомогательных переменных или в него войдут основные переменные, столбцы которых образуют треугольную или почти треугольную матрицу B_c. Во многих промышленных пакетах стартовые процедуры работают в несколько этапов, используя приведенные цены, отвечающие уже построенной легко обратимой базисной матрице, для выбора следующего, более подходящего начального базиса.

*5.8.2. ЛИНЕЙНО ЗАВИСИМЫЕ ОГРАНИЧЕНИЯ

Линейная зависимость ограничений — явление для прикладных задач рядовое, и надо сразу сказать, что она не порождает каких-либо серьезных вычислительных трудностей. Когда в некоторой точке x_k ограничения, выполненные как равенства, линейно зависимы, говорят, что имеет место *вырождение*, и x_k называют *вырожденной точкой*.

Важная черта модельной схемы решения задач LIP, данной в разд. 5.2.1, состоит в том, что при выборе направления движения из текущей допустимой точки x_k не обязательно должны учитываться все ограничения, обращающиеся в x_k в равенства. Если каких-то из этих ограничений в рабочем списке нет, одно из них может попасть туда на следующей итерации, но лишь при условии, что выбранное направление выводит вне его допустимого множества. Затем, возможно, потребуется ввести в рабочий список еще одно из указанных ограничений и т. д. Таким образом, ограничения вводятся в рабочий список по одному, даже если не учтенных в нем равенств несколько. Эта стратегия расширения рабочего списка гарантирует, что содержащиеся в нем ограничения всегда будут линейно независимыми. В самом деле, предположим, что вектор a_{l+1} представляет собой нормаль некоторого ограничения, которое обращается в x_k в равенство, но которого нет в рабочем списке, причем a_{l+1} есть линейная комбинация строк $\bar{A}_k$. Тогда, так как выбранное направление p будет удовлетворять равенству $\bar{A}_k p=0$, получим $a_{l+1}^T p=0$. Значит, движение вдоль p не нарушает этого ограничения, и оно не попадет в рабочий список.

Рассмотрим теперь ситуацию, возникающую в вырожденной точке *при выводе* ограничения из рабочего списка. Здесь p не будет удовлетворять равенству $\bar{A}_k p=0$, и соответственно любой положительный шаг вдоль p может нарушать одно из неучтенных в рабочем списке линейно зависимых активных ограничений; его придется включить туда, сделав нулевой шаг вдоль p. Если x_k не вершина, двигаться можно, не сокращая рабочего списка. В вершине же какое-то ограничение обязательно должно быть отброшено. При наличии вырождения шаг вдоль полученного в результате направле-

или может оказаться нулевым, рабочий список снова придется расширять и т. д. Не исключено, что, выводя и вводя ограничения в одной и той же вырожденной точке, мы через несколько итераций вернемся к исходному варианту рабочего списка, после чего все начнется сначала. Это явление называется *зацикливанием*. Отметим, что само по себе вырождение неприятностей не доставляет, и если бы не вероятность зацикливания (как его следствия), то о нем можно было бы просто не говорить.

Перебрав в вырожденной вершине достаточное количество возможных рабочих списков, всегда можно найти такой, при котором будет получено направление, позволяющее уйти из нее с уменьшением целевой функции. Однако это — комбинаторная задача, которая в принципе может оказаться очень трудоемкой. Здесь были бы основания для беспокойства, но, к счастью, зацикливание редко встречается на практике, а при использовании некоих элементарных эвристических приемов не наблюдается вообще.

5.8.3. НУЛЕВЫЕ МНОЖИТЕЛИ ЛАГРАНЖА

Когда точные значения некоторых из множителей Лагранжа задачи с линейными ограничениями-равенствами, соответствующей текущему рабочему списку, оказываются нулевыми или очень близкими к нулю, решение вопроса о целесообразности сокращения рабочего списка может существенно осложниться. Понятно, что близость к нулю *оценок* каких-то множителей приводит к затруднениям далеко не всегда. Например, если очередная точка x_k еще далека от стационарной, то всегда можно подождать с выводом ограничений в надежде, что по мере приближения к промежуточному оптимуму модули близких к нулю оценок множителей возрастут. Проблем не будет и в том случае, когда наряду с почти нулевыми есть существенно отрицательные оценки (например, если $\lambda_k = (1, 0, -1)^T$). Здесь ясно, какое ограничение отбросить.

Трудности возникают тогда, когда близка к нулю минимальная из оценок множителей и при этом «почти равен нулю» спроектированный градиент. Чтобы гарантировать оптимальность условно стационарной точки с неотрицательными множителями Лагранжа, среди которых есть нулевые, нужно исследовать вторые производные, а они доступны далеко не всегда. Если отрицательность множителя говорит о том, что, исключив соответствующее ограничение из рабочего списка, мы наверняка получим точки с меньшим значением целевой функции, то равенство множителя нулю означает лишь отсутствие возможности предсказать существование лучших точек по информации первого порядка. Непосредственная же проверка направлений, получающихся при сокращении рабочего списка, дает определенный ответ только при условии, что будут просмотрены все комбинации ограничений с нулевыми множителями.

Когда таких ограничений много, этот просмотр потребует огромной работы.

Представленный ниже пример показывает, что проверка ограничений с нулевым множителем по одному проблемы не решает.

Пример 5.5. На рис. 5а изображены линии уровня целевой функции двумерной задачи вида

$$\text{найти} \quad \min_{x \in \mathbb{R}^2} -x_1 x_2$$

$$\text{при ограничениях} \quad 0 \leqslant x_i \leqslant 10, \quad i = 1, 2.$$

Для нее точка начала координат является допустимой вершиной (и соответственно условно стационарной точкой), причем множители Лагранжа обоих обращающихся в ней в равенства условий неотрицательности равны нулю. При этом, хотя ни одно из этих условий не будет активным в решении, вывод любого из них из рабочего списка, если сохранить другое, не дает возможности уменьшить значение критерия.

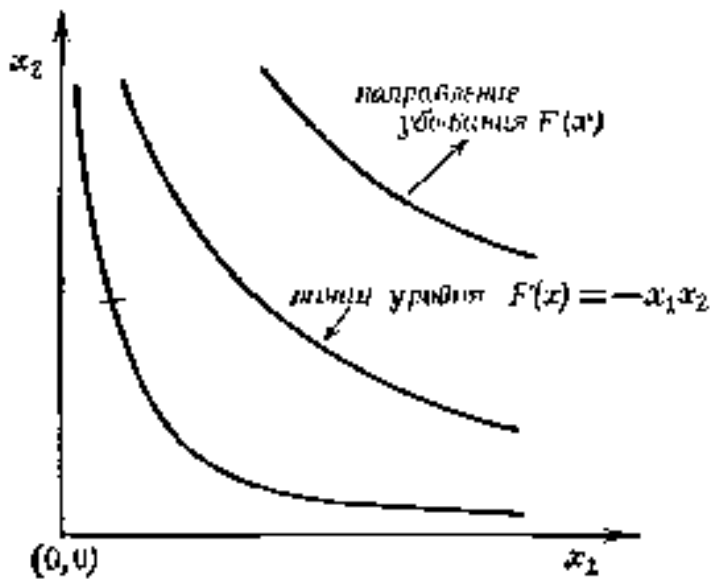

Рис. 5а. Случай одновременного обращения в нуль двух множителей Лагранжа.

Трудности с нулевыми множителями усугубляются ошибками машинных вычислений. Из-за этих ошибок существует конечный порог точности численных оценок множителей. Поэтому мы в принципе не можем отличить нулевые множители от «малых» и не можем выяснить истинных знаков последних. Таким образом, отрицательность малой по модулю оценки множителя не является гарантией целесообразности вывода соответствующего ограничения из рабочего списка, даже если есть уверенность, что точка, где подсчитана эта оценка, близка к условно стационарной.

Замечания и избранная библиография к разделу 5.8

Процедура выбора начального рабочего списка с использованием ортогональной факторизации реализована в библиотеке ФОРТРАН-программ для решения задач с линейными ограничениями, которую можно приобрести в Лаборатории оптимизации систем Стэнфордского университета (см. Гилл и др. (1980)). Эти программы обра-

бавляют простые ограничения и ограничения общего вида раздельно и оперируют ортогональным разложением матрицы $A_{B,D}$ (см. разд. 5.5.2). (Отметим, что подобные средства очень легко присоединяются для задач минимизации, в которых есть только простые ограничения.)

Подробные сведения о стартовых процедурах, применяемых в коммерческих пакетах линейного программирования, читатель найдет в книге Орчард-Хейса (1968). Стартовая процедура для алгоритма решения задач с нелинейными целевыми функциями, относящегося к классу методов ранг-пространства, должна генерировать вершину с максимальным числом искусственных ограничений (определение искусственных ограничений дано в разд. 5.3.2.2).

Среди приемов борьбы с зацикливанием при наличии вырождения надо упомянуть метод возмущений Чарнеса (см. Чарнес (1952)), лексикографический метод Данцига, Ордена и Вулфа (1955) и метод Вулфа (1963а) (см. также Харрис (1973) и Бенишу и др. (1977)). Блэнд (1977) предложил защищенную от зацикливания версию симплекс-метода, основанную на применении при выборе ведущего элемента правила «минимального индекса». Пероль (1981a, b) показал, что при смене базисов в вырожденной точке в процедуре пересчета разложения базисной матрицы возможна некоторая экономия вычислений.

Методы разбора ситуаций с близкими к нулю множителями Лагранжа, предназначенные для задач с произвольными гладкими целевыми функциями, даны в работе Гилла и Мюррея (1977b). В терминологии линейного программирования нулевые множители иногда называют *вырожденными двойственными переменными*. Заметим, что для линейных задач проблема нулей стоит не так остро, поскольку неотрицательности множителей достаточно, чтобы утверждать, что найдено решение. Худшее, что здесь возможно,— это ненужные итерации с выводом из рабочего списка ограничений, имеющих «малые» множители.

ГЛАВА 6

ЗАДАЧИ С НЕЛИНЕЙНЫМИ ОГРАНИЧЕНИЯМИ

Свобода с ограничениями есть две стороны необходимости.
Антуан де Сент-Экзюпери, «La Citadelle» (1948)

В этой главе рассматриваются задачи минимизации нелинейных функций при нелинейных ограничениях. Они намного сложнее задач с линейными ограничениями, и арсенал применяемых к ним методов, естественно, отражает это. В нем нет того концептуального единства, которое присуще схемам предыдущей главы, причем даже близкие по идее методы могут значительно различаться в деталях.

Как и в гл. 4 и 5, здесь будут описаны лишь те методы, в эффективности которых авторы неоднократно убедились на собственном опыте, и те, которые нужны для пояснения каких-то важных моментов или в качестве основы для последующего изложения. Ссылки на литературу, где обсуждаются иные методы или содержатся более подробные сведения о представленных, вынесены в замечания, завершающие каждый большой раздел. Там же называются имена авторов различных схем и результатов.

Чтобы упростить изложение, мы не станем касаться задач с условиями смешанного типа, а будем разбирать только два случая: первый — все ограничения являются *равенствами* и второй — все они *неравенства*. Так уже делалось в гл. 3 и 5. Мы надеемся, что способы адаптации предлагаемых схем к задачам со смесью нелинейных равенств и неравенств читателю будут понятны и без наших пояснений.

В большей части этой главы все ограничения трактуются одинаково — как нелинейные. На самом же деле, если какие-то из равенств или неравенств конкретной задачи линейны, их почти наверняка стоит выделить и в оптимизационном процессе обрабатывать приемами предыдущей главы.

Итак, в дальнейшем речь будет идти о задаче с *ограничениями-равенствами*

$$\text{NEP} \quad \text{найти} \quad \min_{x \in \mathcal{R}^n} F(x)$$
$$\text{при ограничениях} \quad c_i(x) = 0, \quad i = 1, \ldots, t,$$

и о задаче с *ограничениями-неравенствами*

$$\text{NIP} \quad \text{найти} \quad \min_{x \in \mathcal{R}^n} F(x)$$
$$\text{при ограничениях} \quad c_i(x) \geqslant 0, \quad i = 1, \ldots, m.$$

Везде, где не оговорено иное, *функции ограничений* $\{c_i\}$ или $\{c_i\}$ и целевая функция F считаются дважды непрерывно дифференцируемыми. Когда употребляется термин «функции задачи», имеются в виду и $\{c_i\}$ (или $\{c_i\}$), и F одновременно. Условия оптимальности для NEP и NIP получены в разд. 3.4.

Напомним, что градиенты функций ограничений $\tilde{c}_i(x)$ ($c_i(x)$) принято обозначать через $a_i(x)$ ($a_i(x)$), а матрицы Гессе — через $G_i(x)$ ($G_i(x)$). Под $A(x)$ ($A(x)$) понимается матрица, чья i-я строка есть $a_i(x)^T$.

6.1. ОБЩИЕ ОПРЕДЕЛЕНИЯ

6.1.1. ФУНКЦИЯ ВЫИГРЫША

При обсуждении условий оптимальности для задач NEP и NIP было отмечено, что движение по прямой из допустимой точки, где некоторое нелинейное ограничение обращается в равенство, вообще говоря, приведет к нарушению этого равенства независимо от того, как направлена прямая (т. е.— в терминах разд. 3.3.2 — *для нелинейного ограничения не существует удерживающих направлений*). Это обстоятельство сильно осложняет учет нелинейных ограничений и оказывается решающим при разработке алгоритмов.

Возвращаясь к представленным в предыдущей главе методам решения задач, ограничения которых линейны, отметим, что все они исходят из принципа сохранения допустимости, т. е. после того, как с помощью процедуры первой фазы найдено допустимое начальное приближение, генерируют последовательности допустимых точек. Такой принцип берется в основу организации поиска, в частности, потому, что его воплощение не требует чрезмерных усилий: (i) ограничения из рабочего списка выдерживаются благодаря выбору подходящих направлений движения; (ii) возможность нарушить прочие ограничения исключается правилами расчета длин шагов. При этом качество приближений характеризуется только значениями в них целевой функции.

Поддерживать допустимость приближений при решении задач, среди ограничений которых есть нелинейные, намного сложнее (а подчас и просто невозможно). Если же допустимость не обеспечивается, то для того, чтобы определить, является ли точка x_{k+1} «лучшей», чем x_k, мало сравнить $F(x_{k+1})$ и $F(x_k)$ — нужно как-то сопоставить и соответствующие невязки ограничений. Точнее говоря, нужна *функция выигрыша*, значения которой отражали бы (обычно конфликтующие) стремления к уменьшению F и соблюдению ограничений и потому могли бы быть использованы как показатели качества. Некая функция выигрыша явно или неявно обязательно присутствует в каждом алгоритме для NEP и NIP, генерирую-

щем подпустимые точки. Как будет видно из содержания данной главы, разные алгоритмы используют существенно разные функции выигрыша.

6.1.2. КЛАССИФИКАЦИЯ ПОДЗАДАЧ

6.1.2.1. Адаптивные и детерминированные подзадачи. В оптимизационных методах последовательности приближений строятся на основе решения подзадач, каким-то образом связанных с исходной задачей. В частности, если говорить о методах безусловной минимизации и минимизации при линейных ограничениях, то можно выделить две определяющие итерацию подзадачи: подзадачу расчета направления поиска и подзадачу выбора длины шага. Первая обычно *детерминирована* в том смысле, что число требуемых для ее решения обращений к процедурам вычисления исходной целевой функции F и ее производных *заранее и не зависит от получаемых значений функции*. Так, например, для расчета направления поиска в дискретном методе Ньютона, рассчитанном на функции с плотной матрицей Гесса, всегда требуется вычислить ровно n конечных разностей градиентов, и больше никакие связанные с F величины не нужны. Что же касается подзадачи выбора длины шага, то она, как правило, является *адаптивной*. Это значит, что заранее не известно, сколько раз придется вычислить F (или произвольные F) по ходу ее решения, и все будет зависеть от того, как поведет себя F в пробных точках. (Здесь уместно напомнить о критериях «существенного убывания» (см. разд. 4.3.2.1), которые включают F и производные от F.)

Среди методов решения задач с нелинейными ограничениями тоже есть такие, в которых построение очередной точки сводится к детерминированному вычислению направления поиска и адаптивному выбору длины шага. Однако наряду с этими методами существуют и другие, со значительно более сложными подзадачами. К примеру, очередное приближение может определяться как точка безусловного минимума или минимума при линейных ограничениях функции общего вида. Разумеется, подзадача отыскания этой точки относится к категории адаптивных, причем ее решение требует серии итераций со своими адаптивными подзадачами. Таким образом, здесь имеются адаптивные подзадачи двух уровней — внешние и внутренние.

В дальнейшем, разбирая методы с последовательной минимизацией функций общего вида (без ограничений или с линейными ограничениями), мы будем касаться вопроса о том, как эту минимизацию осуществлять, только тогда, когда это будет отражаться на основных определениях. Вообще же предпочтительный способ зависит от доступности производных исходных функций, размерности и т. д.

6.1.2.2. Нормальные и дефектные подзадачи. Второй признак классификации подзадач — их разрешимость. Дело в том, что в некоторых методах подзадачи могут не иметь удовлетворительных решений *даже при условии, что исходная задача поставлена вполне корректно*. Соответственно выделяют *нормальные* и *дефектные* подзадачи. К первым относят те подзадачи, решения которых существуют и хорошо определены (например, подзадачи расчета вектора, удовлетворяющего системе линейных уравнений с невырожденной матрицей); ко вторым — подзадачи, не имеющие решений или имеющие бесконечные решения (например, подзадачи безусловной минимизации квадратичной функции со знакоопределенной матрицей Гессе или решения несовместной системы уравнений).

Дефектные подзадачи — не редкость для многих методов минимизации при нелинейных ограничениях. Если они идентифицируемы, на случай их появления предусматривают какие-то замены в формулировках. Однако бывают и такие подзадачи, выявить дефектность которых невозможно; к примеру, не существует универсального способа, позволяющего устанавливать существование или отсутствие конечного безусловного минимума у нелинейных функций общего вида. Тогда приходится действовать по-другому, а именно ограничивать усилия на решение одной подзадачи, с тем чтобы не тратить время попусту, если она окажется дефектной.

Наконец, надо отметить, что *дефектность* подзадачи может быть следствием порочности исходной постановки. Заканчивая на этом обсуждение затронутых вопросов, мы надеемся, что у читателя создалось впечатление о том, какие трудности могут возникнуть при решении сложных нелинейных задач.

6.2. МЕТОДЫ ШТРАФНЫХ И БАРЬЕРНЫХ ФУНКЦИЙ

Один из подходов к решению задачи с нелинейными ограничениями состоит в том, чтобы сконструировать функцию, безусловный минимум которой совпадает с x^* или связан с x^* определенным образом. Тогда к x^* можно прийти, решив одну или много подзадач *безусловной минимизации*. Целевые функции таких подзадач впредь будем обозначать через Φ_U.

6.2.1. МЕТОДЫ ГЛАДКИХ ШТРАФНЫХ И БАРЬЕРНЫХ ФУНКЦИЙ

Методы, рассматриваемые в этом разделе, имеют не очень хорошую репутацию и, как правило, не столь эффективны, как те, о которых речь пойдет в дальнейшем. Однако идеи, лежащие в их основе, представляют интерес, поскольку находят и более удачные воплощения, а знание свойств этих методов важно для понимания родственных им эффективных алгоритмов.

6.2.1.1. Квадратичная штрафная функция. Обычно точка безусловного минимума целевой функции F задачи с ограничениями лежит за пределами допустимого множества либо F вообще не ограничена снизу. Поэтому рассчитывать на сходимость последовательностей решений вспомогательных задач безусловной минимизации к x^* можно лишь в том случае, если Φ_ρ включает какой-то член, обеспечивающий в пределе допустимость. В частности, таким членом может быть дифференцируемая функция, чьи значения в недопустимой области положительны и имеют смысл «штрафов» за нарушение ограничений.

Основные идеи рассматриваемого подхода мы обсудим на примере *квадратичной штрафной функции*

$$P_Q(x, \rho) \equiv F(x) + \frac{\rho}{2} \tilde{c}(x)^T \tilde{c}(x). \tag{6.1}$$

В этой записи $\tilde{c}(x)$ есть вектор левых частей нарушенных в x ограничений. (Ограничения-равенства условимся считать нарушенными в любой точке.) Неотрицательное число ρ называют *параметром штрафа*, а скалярное произведение $\tilde{c}(x)^T \tilde{c}(x)$ *штрафным членом*.

Штрафной член в (6.1) является непрерывно дифференцируемой функцией. Его вторые производные разрывны, причем разрывы возникают в тех точках, где хотя одно из неравенств исходной задачи обращается в равенство. Таким образом, если в искомом решении x^* какие-то из ограничений-равенств активны, это решение будет точкой разрывности вторых производных от P_Q. Чтобы устранить этот дефект P_Q, можно условиться считать неравенство $c_j(x) \geqslant 0$ нарушенным, если $c_j(x) < \varepsilon$ для некоторого малого положительного ε. При таком определении «нарушенных ограничений» ни в точке x^*, ни в некоторой ее окрестности разрывов вторых производных не будет. (Хотя есть сколько угодно неквадратичных штрафных членов, имеющих в x^* непрерывные вторые производные, такими членами не пользуются, поскольку они теоретически, ни практически это не дает никаких преимуществ.)

При очень слабых предположениях удается доказать, что для достаточно больших ρ существует зависимость $x^*(\rho)$, определяющая точки безусловных минимумов по x функций (6.1) и обладающая тем свойством, что

$$\lim_{\rho \to \infty} x^*(\rho) = x^*. \tag{6.2}$$

Важно отметить, что для доказательства (6.2) регулярности ограничений в x^* (см. разд. 3.4.1) не требуется.

Проиллюстрируем результаты штрафного преобразования (6.1) на простом примере

Пример 6.1. Рассмотрим одномерную задачу вида

$$\text{найти } \min_{x \in \mathbb{R}^1} x^2$$

$$\text{при ограничении } x - 1 = 0.$$

Для нее

$$P_Q(x, \rho) = x^2 + \frac{\rho}{2}(x-1)^2$$

и

$$x^*(\rho) = \frac{\rho}{\rho + 2}.$$

Графики $F(x)$ и $P_Q(x, \rho)$ при $\rho = 10$ изображены штриховой и сплошной линиями соответственно на рис. 6а. Ясно, что $\lim_{\rho \to \infty} x^*(\rho) = x^* = 1$.

Глядя на соотношение (6.2), можно подумать, что естественным способом поиска хорошего приближения x^* было бы взять очень большое (заведомо достаточное) значение параметра штрафа и один раз решить задачу безусловной минимизации P_Q. Однако делать так *не рекомендуется*. Причина в том, что если количество l ограничений, активных в x^*, лежит в диапазоне $0 < l < n$, то число обусловленности матрицы Гессе $P_Q''(x, \rho)$ в точке $x^*(\rho)$ растет с увеличением ρ и в пределе становится бесконечным. Значит, при очень большом ρ мы получим очень тяжелую задачу безусловной минимизации. В двумерном случае линии уровня плохо обусловленной штрафной функции в окрестности x^* будут «почти параллельными».

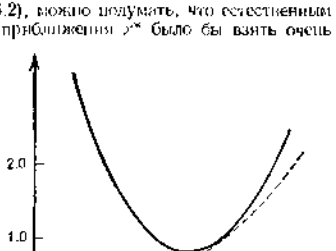

Рис. 6а. Штриховая линия — график целевой функции из примера 6.1, $F(x) = x^2$; сплошная — график квадратичной штрафной функции $P_Q(x, 10) = x^2 + 5(x-1)^2$.

Пример 6.2. Рассмотрим задачу, которая еще не раз послужит для целей иллюстрации в будущем:

$$\text{найти } \min_{x \in \mathbb{R}^2} x_1 x_2^2$$

$$\text{при ограничении } 2 - x_1^2 - x_2^2 \geqslant 0.$$

Ее единственное ограничение активно в решении $x^* = (-0.81650, -1.1547)^T$ и имеет множитель Лагранжа $\lambda^* = 0.81650$ (все величины округлены до пяти значащих цифр).

К задаче из примера 6.2 прилагаются рис. 6b и 6c. На левом фрагменте рис. 6b изображены линии уровня целевой функции и граница допустимого множества (выпуклая вниз кривая). Кре-

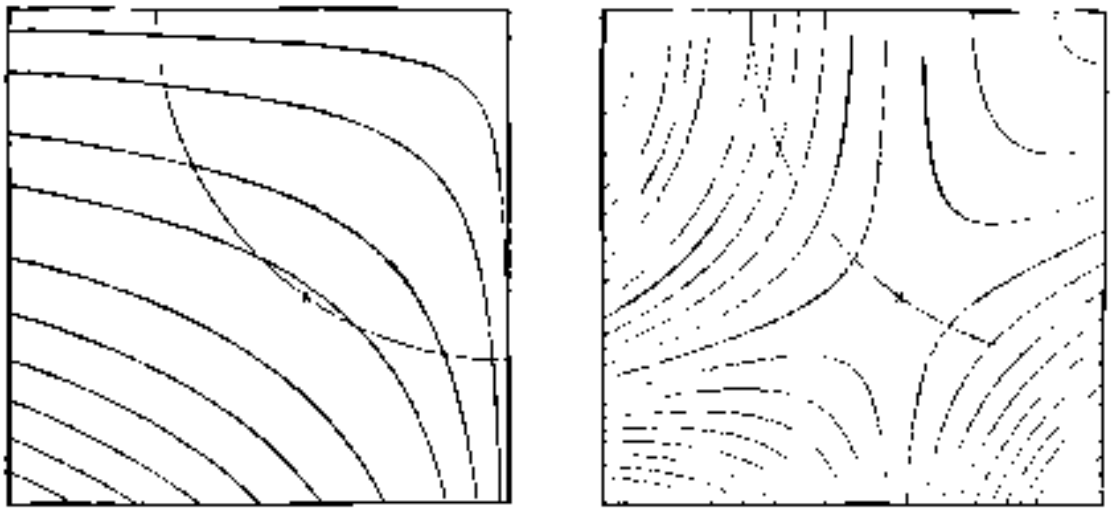

Рис. 6b. На левом фрагменте изображены линии уровня функции $f(x) = x_1 x_2^4$ и границы допустимого множества ограничения $2 - x_1^2 - x_2^2 \geqslant 0$; крестиком отмечена точка оптимума $(-0.8166, -1.1547)$. На правом фрагменте изображены линии уровня соответствующей функции Лагранжа $L(x, \lambda^*)$.

стиком отмечена точка оптимума. Видно, что обусловленность исходной задачи не оставляет желать лучшего. Линии уровня

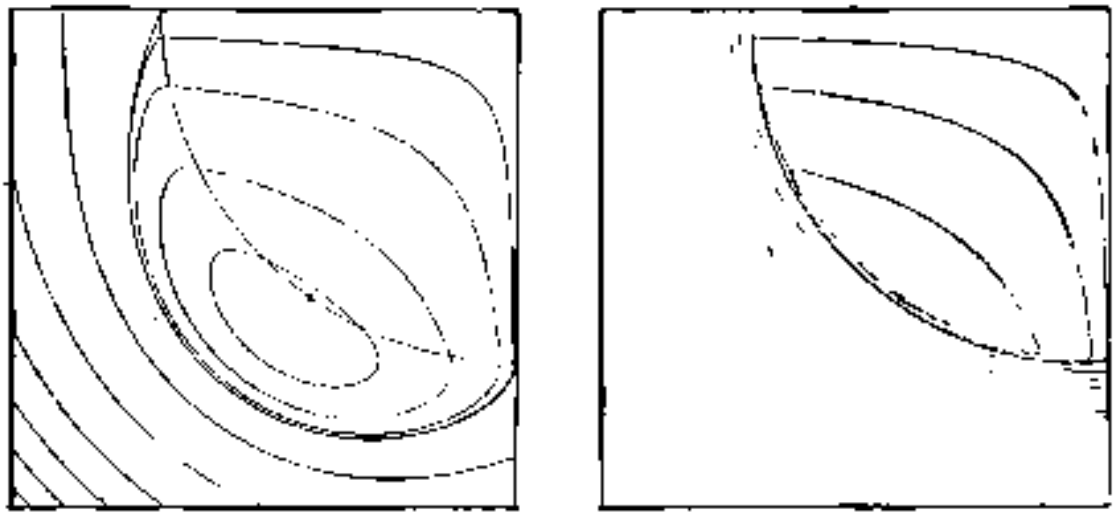

Рис. 6c. Линии уровня квадратичных штрафных функций $P_Q(x, \rho)$ с $\rho = 1$ и $\rho = 100$ для примера 6.2

отвечающих ей квадратичных штрафных функций с $\rho = 1$ и $\rho = 100$ показаны на рис. 6c. Первая обусловлена неплохо — ее линии уровня напоминают окружности. При $\rho = 100$ картина совсем иная; здесь уже явственно проявляется плохая обусловленность, сопутствующая большим значениям параметра штрафа.

По мере того как с увеличением ρ обусловленность матрицы Гессе штрафной функции ухудшается, минимум последней в нуль-пространстве градиентов активных ограничений становится все менее отчетливо выраженным. Поэтому, если сразу взять «очень большой» параметр штрафа, даже надежный алгоритм безусловной минимизации, как правило, будет испытывать во время поиска $x^*(\rho)$ очень серьезные затруднения (см. разд. 8.3). Таким образом, при использовании гладких штрафных функций в задачах NEP или NIP приходится решать *последовательность задач безусловной минимизации*, постепенно увеличивая параметр ρ. Обычно точку $x^*(\rho)$, полученную для предыдущего значения ρ, используют в качестве начального приближения для минимизации штрафной функции со следующим ρ. В конце концов должна быть найдена точка, удовлетворяющая выбранному критерию останова, и в ней вычисления будут прекращены.

Итак, чтобы отыскать решение задачи с нелинейными ограничениями методом гладких штрафных функций, надо, задав начальное значение параметра штрафа ρ_0, положительное целое число K и начальную точку x_0 и сделав присвоение $k \leftarrow 0$, выполнить следующие действия:

Алгоритм DP (*Модельная схема минимизации с использованием гладкой штрафной функции*)

DP1. [Проверка соблюдения условий окончания счета.] Если x_k удовлетворяет условиям оптимальности или $k > K$, вычисления прекратить. (В первом случае выдается признак успешного окончания счета, во втором — признак неудачи.)

DP2. [Минимизация штрафной функции.] Взяв x_k в качестве начального приближения, применить алгоритм решения подзадачи безусловной минимизации

$$\text{найти } \min_{x \in R^n} P_Q(x, \rho_k), \tag{6.3}$$

в котором должны быть приняты предосторожности на случай, если P_Q окажется неограниченной снизу. Он выдаст очередную точку x_{k+1}.

DP3. [Увеличение параметра штрафа.] Взять в качестве ρ_{k+1} некоторое превосходящее ρ_k число, сделать присвоение $k \leftarrow k+1$ и вернуться к шагу DP1.

Плохая обусловленность подзадач безусловной минимизации при $0 < t < n$ и больших ρ является первой из общих особенностей методов гладких штрафных функций. Пришло время поговорить о двух других.

Свойство локальности минимума. Обсуждая метод с гладкой штрафной функцией P_Q, нередко забывают подчеркнуть, что в общем случае точки $x^*(\rho)$ из теоремы о его сходимости — это точки

локальных минимумов. Глобальный же минимум P_Q может быть бесконечным (нетрудно привести примеры, когда P_Q не ограничена снизу при любом ρ). В таких случаях обеспечить сходимость процедуры безусловной минимизации в нужную точку бывает непросто: итерации поиска могут вывести за пределы ее «области притяжения». Сказанное означает, что применять к подзадачам метода штрафных функций стандартные схемы минимизации типа рассмотренных в гл. 4 следует с осторожностью. Последние предполагают ограниченность функции снизу и в ее отсутствие могут расходиться. Таким образом, их никак нельзя использовать в методах штрафных функций как «черный ящик». Алгоритм, предназначенный для решения соответствующих подзадач, должен включать средства, позволяющие выявить неограниченность и, если возможно, нейтрализовать ее влияние (на что и указано в п. DP2 общей схемы).

Возможность оценивания множителей Лагранжа. В силу специальной структуры штрафной функции, объединяющей в себе все функции задачи, последовательность $\{x^*(\rho_k)\}$ содержит много полезной информации. В частности, по ней можно строить оценки множителей Лагранжа (когда эти множители существуют). В точке $x^*(\rho)$, доставляющей безусловный минимум функции P_Q, должно выполняться равенство

$$g + \rho A^T c = 0.$$

Отсюда видно, что величины

$$\lambda_i(\rho) = -\rho c_i(x^*(\rho)) \tag{6.4}$$

играют в $x^*(\rho)$ ту же роль, что и множители Лагранжа в искомой точке x^*: они служат коэффициентами разложения градиента g по строкам матрицы A. При естественных предположениях можно доказать, что

$$\lim_{\rho \to \infty} \lambda_i(\rho) = \lambda_i^*, \tag{6.5}$$

причем $\|\lambda^* - \lambda(\rho)\| = O(1/\rho)$. Так, в примере 6.1, где $\lambda_1^* = 2$, формула (6.4) дает $\lambda_1(\rho) = 2\rho/(\rho+2)$.

Допустим, что метод штрафных функций используется для решения задачи с неравенствами и в ней $\lambda_i^* > 0$. Тогда из (6.4) и (6.5) следует, что в точках минимумов $P_Q(x, \rho)$ с достаточно большими ρ i-е активное ограничение будет *нарушено*. Данное свойство позволяет идентифицировать набор активных в x^* ограничений и, по существу, означает, что для метода штрафных функций «рабочий список» есть «нарушенный список». Оно исключает возможность применения метода в случаях, когда важна допустимость генерируемых приближений.

6.2.1.2. Логарифмическая барьерная функция. Методы штрафных функций идут к решению извне допустимого множества. Поэтому они не годятся для задач NIP, требующих соблюдения огра-

ничений в течение всего процесса поиска; тут нужны *методы допустимой точки*, порождающие только допустимые приближения. Таковы, в частности, *методы барьерных функций*, рассматриваемые в данном разделе (другие будут представлены в разд. 6.3 и 6.5.3.4). Основная область применения методов допустимой точки — задачи, некоторые (или все) функции которых не определены или плохо определены за пределами допустимого множества. Кроме того, они предпочтительны тогда, когда не нужна высокая точность достижения оптимума, но ограничения важно выдержать,— в этой ситуации иные методы (скажем, метод с квадратичной штрафной функцией) сэкономить время не позволяют, так как при прерывании вдали от решения дают существенно недопустимые точки.

Методы барьерных функций работают по тому же принципу, что и методы штрафных функций, описанные выше: x^* ищется как предел последовательности точек безусловных минимумов гладких модифицированных целевых функций Φ_r. Однако в них Φ_r устроены так, что эти точки лежатся *строго внутрь допустимого множества*: добавки в Φ_r к F служат «барьерами», удерживающими процедуру минимизации Φ_r от нарушения ограничений. В качестве таких «барьеров» используют взвешенные суммы непрерывных функций, обращающиеся на границе допустимого множества в бесконечность. По мере стремления весового параметра к нулю точка безусловного минимума соответствующей Φ_r приближается к x^*. Отметим, что барьерное преобразование применимо лишь к тем задачам, допустимые множества которых имеют непустые внутренности, и порождает приближения, удовлетворяющие всем ограничениям с запасом. Для задач с ограничениями-равенствами оно не определено.

Характерные черты всех методов барьерных функций мы обсудим на примере одного из них, а именно метода *логарифмической барьерной функции*

$$B(x, r) = F(x) - r \sum_{i=1}^{m} \ln (c_i(x)). \qquad (6.6)$$

Положительное число r называют *параметром барьера*. При слабых предположениях можно доказать, что существует «траектория» $x^*(r)$ безусловных минимумов функций (6.6), сходящаяся к x^* при стремлении r к нулю, т. е.

$$\lim_{r \to 0} x^*(r) = x^*.$$

Пример 6.3. Рассмотрим эффект барьерного преобразования для задачи

$$\text{найти } \min_{x \in \mathbb{R}^1} x^2$$

при ограничении $x - 1 \geqslant 0$.

В данном случае

$$B(x, r)=x^2-r\ln(x-1)$$

и

$$x^*(r)=\frac{1}{2}+\frac{1}{2}\sqrt{1+2r}.$$

Истинным решением является $x^*=1$.

У методов барьерных и гладких штрафных функций есть много общих свойств. Это видно и из представленного ниже перечня наиболее существенных особенностей метода с логарифмической барьерной функцией.

Плохая обусловленность. Пусть l — число ограничений задачи, активных в x^*, и $0<l<n$. Тогда по мере приближения r к нулю матрица Гессе барьерной функции в $x^*(r)$ будет все хуже и хуже

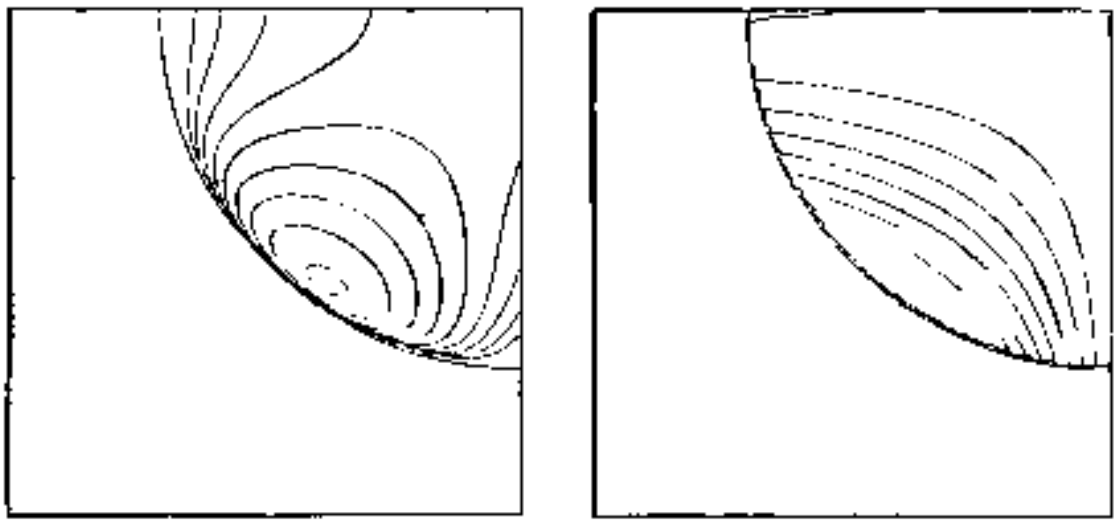

Рис. 6d. Линии уровня логарифмических барьерных функций $B(x, r)$ с $r=0.2$ и $r=0.001$ для примера 6.2.

обусловленной, и в пределе ее число обусловленности становится бесконечным. На рис. 6d изображены линии уровня логарифмических барьерных функций для задачи из примера 6.2 с $r=0.2$ и $r=0.001$. Поскольку вне допустимого множества эти функции не определены, там никаких линий не нарисовано. Отметим, что на правом фрагменте, отвечающем меньшему значению r, линии уровня почти параллельны, т. е. отчетливо проявляется плохая обусловленность. Из-за трудностей, возникающих при поиске безусловного минимума функций (6.6) с «очень малым» r, к искомому решению x^* приходится идти, последовательно минимизируя барьерные функции с постепенно уменьшаемыми значениями.

Локальность минимума. Логарифмическое барьерное пребразование, вообще говоря, порождает во внутренности допустимого множества лишь локальный минимум. При этом функция (6.6) может оказаться неограниченной снизу. Правда, такая перспе-

ность существует только в случаях, когда не ограничено допустимое множество. И все же ее нельзя сбрасывать со счетов, т. е. она должна быть учтена в методе, применяемом для безусловной минимизации (6.6).

Трудности одномерного поиска. Из-за того что барьерные функции определены не во всех точках и имеют особенности, применять в алгоритмах их минимизации стандартные процедуры одномерного поиска, как правило, неэффективно. Во-первых, эти процедуры предполагают возможность вычисления функции в любой точке и активно используют все пробные значения. Значит, обращаясь к какой-нибудь из них, придется доопределить барьерную функцию в недопустимой области и смириться с тем, что соответствующие бессмысленные значения смогут запутывать вычисления. В то же время находить для каждого из генерируемых направлений шаг до границы допустимой области и ставить его в качестве порога для одномерного поиска (как это делается в алгоритмах решения задач с линейными ограничениями) в общем случае невозможно. Во-вторых, традиционный способ построения очередной оценки длины оптимального шага исходя из одномерной полиномиальной аппроксимации (см. разд. 4.1.2.3), для барьерных функций скорее всего будет работать плохо (особенно при малых значениях параметра барьера, когда поиск идет вблизи особых точек). Короче говоря, здесь нужны специальные процедуры минимизации по направлению, и, к счастью, поскольку характер особенностей барьерных функций известен заранее, эти процедуры удается сделать довольно эффективными (см. замечания).

Оценки множителей Лагранжа. Точки $\{x^*(r)\}$ минимумов логарифмических барьерных функций с малыми r несут информацию о множителях Лагранжа исходной задачи. В $x^*(r)$ будет выполнено равенство

$$g - \sum_{i=1}^{m} a_i \frac{r}{c_i}, \tag{6.7}$$

т. е. градиент функции F в $x^*(r)$ есть неотрицательная линейная комбинация градиентов всех ограничений. Таким образом, коэффициенты при a_i в (6.7) играют в $x^*(r)$ ту же роль, что и множители Лагранжа в x^*. При слабых предположениях можно доказать, что для неактивных в x^* ограничений с уменьшением r они стремятся к нулю, а если i-е ограничение в x^* активно, то

$$\lim_{r \to 0} \frac{r}{c_i (x^*(r))} = \lambda_i^*.$$

Отметим отличие (6.7) от (6.4), где фигурируют только нарушенные в $x^*(\rho)$ ограничения.

6.2.2. МЕТОДЫ НЕГЛАДКИХ ШТРАФНЫХ ФУНКЦИЙ

Как указано в разд. 6.2.1.1, дифференцируемые штрафные функции страдают плохой обусловленностью даже для гладких, хорошо поставленных задач. Это отрицательно сказывается на характеристиках соответствующих методов и приводит к тому, что в них приходится решать длинные серии подзадач безусловной минимизации. В то же время можно строить *недифференцируемые, но хорошо обусловленные* штрафные функции с локальным минимумом в искомой точке x^*. Тогда поиск x^* сведется к однократной безусловной минимизации. Данный подход представляется более естественным, по крайней мере в отношении задач, которые сами содержат негладкие функции. Конструкцию и свойства недифференцируемых штрафных функций мы рассмотрим в разд. 6.2.2.1, а их приложение к негладким задачам — в разд. 6.2.2.2.

6.2.2.1. Абсолютная штрафная функция. Наиболее распространенной среди недифференцируемых штрафных функций является так называемая *абсолютная штрафная функция*:

$$P_A(x, \rho) = F(x) + \rho \sum_{i \in I} |c_i(x)| = F(x) + \rho |\hat{c}(x)|_1. \qquad (6.8)$$

Здесь через $\hat{c}(x)$ обозначен вектор нарушенных в x ограничений, а через I — список их индексов. (Напомним, что $|y|_1 = \sum |y_i|$.)

Первые производные от $P_A(x, \rho)$ разрывны в любой точке, где хотя бы одна из функций $c_i(x)$ обращается в нуль; таким образом, x^* будет точкой гладкости P_A только в том случае, если все ограничения в x^* неактивны. Функция P_A помимо прочего принципиально отличается от P_Q тем, что при нормальных условиях для построения с ее помощью искомого решения x^* неограниченно увеличивать ρ не нужно: существует конечное пороговое значение $\bar{\rho}$, такое, что x^* *будет точкой безусловного минимума* P_A *при любом* $\rho > \bar{\rho}$. По этой причине штрафные функции типа P_A иногда называют *точными* (в противовес дифференцируемым штрафным функциям, для которых $x^*(\rho)$ не совпадает с x^* ни при одном конечном ρ). Другие точные штрафные функции упоминаются в замечаниях к разд. 6.4.

Чтобы проиллюстрировать эффект негладкого штрафа, вернемся к задаче из примера 6.1:

$$\text{найти} \min_{x \in \mathcal{R}^1} x^2$$
$$\text{при ограничении } x - 1 = 0.$$

Для нее

$$P_A(x, \rho) = x^2 + \rho |x - 1|, \qquad (6.9)$$

и нетрудно проверить, что $x^* = 1$ является точкой безусловного минимума (6.9) при любом $\rho \geq 2$. Графики $F(x)$ и $P_A(x, \rho)$ с $\rho = 4$ изоб-

ражены на рис. 6в штриховой и сплошной линиями. Видно, что в x^* реализуется локальный минимум P_A.

Рассмотрим принципиальную схему решения гладкой задачи условной минимизации с применением недифференцируемой штрафной функции (схема для негладких задач описана в разд. 6.2.2.2). В соответствии с ней, задав начальное приближение x_0, положительное число K, параметр штрафа ρ и сделав присвоение $k \leftarrow 0$, надо выполнить следующие действия.

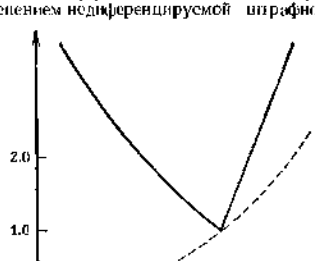

Рис. 6в. Штриховая линия — график целевой функции из примера 6.1, $F(x)=x$; сплошная — график абсолютной штрафной функции $P_A(x, 4)=x^2+4|x-1|$.

Алгоритм ЕР (*Модельная схема с точной штрафной функцией*)

ЕР1. [Проверка соблюдения условий окончания счета.] Если x_k удовлетворяет условиям оптимальности или $k > K$, вычисления прекратить. В первом случае x_k берется в качестве решения; во втором выдается признак неудачи.

ЕР2. [Увеличение параметра штрафа.] Если $k > 0$, увеличить ρ.

ЕР3. [Минимизация штрафной функции.] Использовав x_k в качестве начальной точки, применить алгоритм решения задачи

$$\text{найти } \min_{x \in E^n} P_A(x, \rho), \tag{6.10}$$

правила останова которого должны предусматривать возможность неограниченности P_A снизу; лучшее из найденных приближений взять в качестве x_{k+1}.

ЕР4. [Перевод счетчика итераций.] Сделать присвоение $k \leftarrow k+1$ и вернуться к шагу ЕР1.

В силу сказанного ранее при любом «достаточно большом» начальном ρ итерации будут прерваны при $k=1$.

Поскольку для определения x^* минимизацией негладкой штрафной функции безгранично увеличивать ρ не требуется, проблемы *неизбежной* плохой обусловленности при использовании соответствующих методов не возникает. Надо только подобрать хорошее значение ρ. К сожалению, пороговый параметр ρ зависит от величин, связанных с x^* и, следовательно, заранее неизвестных. Поэтому для выбора начального ρ приходится пользоваться какими-то эври-

ками, и на случай, если они подведут, необходимо предусмотреть возможность корректировки ρ (шаг EP2 общей схемы). При этом неудачный выбор параметра штрафа чреват серьезными осложнениями. Если значение ρ занижено, штрафная функция либо может оказаться неограниченной снизу, либо «область притяжения» точки x^* будет очень малой. Если же взять ρ слишком большим, подзадача (6.10) окажется плохо обусловленной.

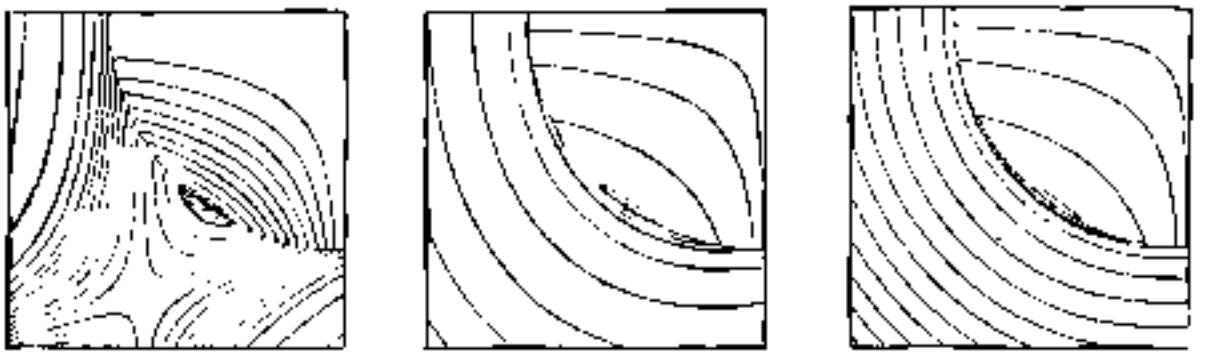

Рис. 6f. Линии уровня абсолютных штрафных функций $P_A(x, \rho)$ с $\rho = 1.2$, $\rho = 5$ и $\rho = 10$ для примера 6.2.

Влияние величины ρ проиллюстрировано рис. 6f, где изображены линии уровня P_A для задачи из примера 6.2 при трех значениях ρ ($\rho = 1.2$, $\rho = 5$, $\rho = 10$). Так как абсолютный штраф «слабее» квадратичного, проблемы, связанные с неограниченностью, стоят для P_1 острее, чем для P_Q. В частности, хотя первая из показанных на рис. 6f функций достигает в x^* локального минимума и в окрестности x^* хорошо обусловлена, процедуре ее безусловной минимизации удастся найти x^* лишь при условии, что поиск начнется с очень хорошего приближения. (Кстати сказать, P_A для примера 6.2 не ограничена снизу ни при одном ρ.) Ухудшение обусловленности P_A с увеличением ρ отчетливо проявляется на третьем фрагменте рис. 6f.

Поскольку функция P_1, как правило, недифференцируема даже в точке минимума, стандартные методы оптимизации без ограничений, рассчитанные на гладкие функции, применять к ней напрямую не следует. Для минимизации P_A в случаях, когда исходные функции дифференцируемы, разработаны специальные алгоритмы; ссылки на них даны в замечаниях. Эти алгоритмы в полной мере учитывают специфику P_A для гладких задач. Для повышения эффективности поиска минимума недифференцируемой штрафной функции и выбора разумного значения ρ может использоваться информация об исходной задаче и в том числе оценки множителей Лагранжа.

6.2.2.2. Метод решения негладких задач общего вида. Если какие-то из функций задачи NEP или NIP разрывны, или имеют разрывные производные, абсолютная штрафная функция уже не будет

композицией гладких составляющих. Поэтому для поиска ее минимума скорее всего придется использовать какой-нибудь универсальный метод негладкой оптимизации типа метода многогранника из разд. 4.2.2; правда, для некоторых задач с известной структурой разрывов удается разработать более эффективные специальные методы (см. разд. 6.8).

Подбирая параметр штрафа в P_A, с одной стороны, желательно получить функцию, достаточно хорошо обусловленную в окрестности искомой точки x^*, а с другой стороны, обеспечить этой точке хорошую область притяжения (которая, вообще говоря, тем шире, чем больше ρ). Данный принцип воплощен в предлагаемой ниже схеме решения негладких задач с произвольными структурами разрывов. Хотя она похожа на алгоритм EP (см. разд. 6.2.2.1), между ними есть и существенные различия: проверки, выполняемые в новой схеме (например, оценка «удовлетворительности минимума»), должны быть ориентированы на конкретную задачу. Кроме того, x_0 принимается в качестве решения только в том случае, если дополнительная попытка улучшить x_0 не увенчалась успехом.

Итак, задав начальную точку x_0, начальное значение параметра штрафа ρ, положительные ρ_{min} и $\gamma(\gamma>1)$ и сделав присвоение $k \leftarrow 0$, решение негладкой задачи можно искать следующим образом.

Алгоритм ND (*Модельная схема решения негладких задач с ограничениями*)

ND1. [Решение безусловной подзадачи.] Начиная с x_k, делается попытка найти приближение минимума функции $P_A(x, \rho)$ методом многогранника из разд. 4.2.2. Чтобы гарантировать остановку метода, вводится ограничение сверху на число обращений к процедуре расчета функций задачи. В качестве x_{k+1} берется лучшая из просмотренных точек.

ND2. [Увеличение параметра штрафа.] Если $k>0$, осуществляется переход к шагу ND3. Если $\rho \le \rho_{min}$, вычисления прекращаются с выдачей сообщения о неудаче. Если на предыдущем шаге удовлетворительного минимума P_A найти не удалось или точка x_{k+1} недопустима, выполняется присвоение $\rho \leftarrow \gamma\rho$ и осуществляется переход к шагу ND1. В противном случае выполняется шаг ND4.

ND3. [Выяснение возможности уменьшить параметр штрафа.] Если x_{k+1} не значительно предпочтительнее, чем x_k, вычисления прекращаются и лучшая из точек x_{k+1} и x_k берется в качестве решения.

ND4. [Уменьшение параметра штрафа.] Делаются присвоения $\rho \leftarrow \rho/\gamma$, $k \leftarrow k+1$ и осуществляется возврат к шагу ND1.

Разрывность производных P_A в точке минимума для метода многогранника — обстоятельство полезное. Поэтому именно он использован в предлагаемой схеме. Правда, маловероятно, что в x^* будут разрывны производные от P_A по всем направлениям, и, зна-

чит, есть определенный риск, что метод многогранника «замкнется в подпространстве», т. е. в некоторых направлениях минимизация будет выполнена очень неточно. Как средство, которое, возможно, поправит дело, в схеме предусмотрена возможность уменьшения параметра штрафа в шаге ND4.

Замечания и избранная библиография к разделу 6.2

Штрафные и барьерные функции имеют долгую историю. По-видимому, они использовались при решении инженерных проблем задолго до того, как получили официальное признание у математиков. У этих функций есть много разных форм, и в конкретных приложениях им часто дают специальные названия. К примеру, в кристаллографических расчетах квадратичный штраф добавляется к целевой функции как «сдерживающий фактор», гарантирующий, что решение оптимизационной задачи «не слишком сильно» нарушит ограничения. Абсолютная штрафная функция многие годы применяется в строительном и машинном проектировании.

Самым фундаментальным изданием, посвященным гладким штрафным и барьерным функциям, является книга Фиакко и Мак-Кормика (1968) «Нелинейное программирование. Методы последовательной безусловной минимизации». В ней просуммированы и развиты результаты из серии статей, опубликованных Фиакко и Мак-Кормиком в журнале Management Science. Эта книга служит главным источником сведений по методам штрафных и барьерных функций.

Квадратичная штрафная функция впервые появилась в математической литературе в работе Куранта (1943). Автором логарифмической барьерной функции считается Фриш (1955); Кэррол (1959, 1961) предложил так называемую «обратную» барьерную функцию

$$B_1(x, r) \equiv F(x) + r \sum_{i=1}^{m} \frac{1}{c_i(x)}.$$

Розенброк (1960) построил метод с модификацией целевой функции, аналогичной барьерному преобразованию, но не создающей особенностей на границе допустимого множества.

Общий обзор штрафных и барьерных функций можно найти в работах Зангвилла (1965, 1967а) и Райана (1974).

Больше всего о методах штрафных и барьерных функций говорилось в 60-х годах. Это объясняется появлением в то время мощных алгоритмов безусловной минимизации (техника поиска безусловного экстремума обсуждается в гл. 4). Программа SUMT, разработанная Фиакко и Мак-Кормиком, была тогда, пожалуй, самым широко употребляемым универсальным средством решения нелинейных задач на условный минимум.

Неизбежная плохая обусловленность матриц Гессе в методах гладких штрафных и барьерных функций впервые была отмечена Мюрреем (1967). Подробное обсуждение этого явления проводится в работах Мюррея (1969a, 1971b) и Лутсма (1969, 1970, 1972).

Специальные процедуры одномерного поиска для алгоритмов минимизации штрафных и барьерных функций читатель найдет, например, у Флетчера и Мак-Канна (1969), Мюррея (1969a), Лэсдона, Фокса и Ратнера (1973), Мюррея и Райт (1976).

Методы штрафных и барьерных функций иногда используют в качестве предварительных в «гибридных» схемах. Они дают неплохие приближения, которые затем уточняются какими-нибудь методами иного сорта с хорошими свойствами локальной сходимости. В частности, Райан (1971) и Осборн и Райан (1972) комбинировали таким образом методы квадратичной штрафной функции и модифицированной функции Лагранжа (см. разд. 6.4), а Розен (1978) и Ван-дер-Хук (1979) предложили применять метод штрафной функции как средство поиска хорошего начального приближения для метода спроектированного лагранжиана (см. разд. 6.5).

Идея построения *недифференцируемой* штрафной функции, для которой искомое решение x^* было бы точкой безусловного минимума при *конечном* значении параметра штрафа, впервые в неявной форме высказана Эблоу и Брайхемом (1955), а затем пропагандировалась в отношении выпуклых задач Пшеничным (1962) и Зангвиллом (1965, 1967a). Обзор общих свойств абсолютной штрафной функции приведен в работе Пшеничного (1969). Среди других публикаций, посвященных свойствам точных штрафных функций данного типа, можно назвать работы Эванса, Гулда и Толле (1973), Хоуна (1973), Бертсекаса (1975a), Караламбуса (1978), Хана и Мангасарьяна (1979), Коулмана и Конна (1980a).

Методы недифференцируемых штрафных функций поначалу считали непрактичными, так как обычные алгоритмы безусловной минимизации (предполагающие гладкость) в них применять нельзя, а других эффективных алгоритмов не было. Однако к настоящему времени для негладких штрафных функций (и других функций с известной структурой разрывов производных) разработано немало хороших специальных схем. Их описания можно найти в статьях Зангвилла (1967b), Конна (1973), Конна и Пшеничного (1977), Лемарешаля (1975), Мифflина (1977), Маратоса (1978), Коулмана (1979), Мэйни и Маратоса (1979). Во многих из этих схем направления поиска определяются из тех же соображений, что и в методе спроектированного лагранжиана с квадратичной подзадачей, рассматриваемом в разд. 6.5.3.

Абсолютную штрафную функцию иногда используют в качестве функции выигрыша в методах спроектированного лагранжиана (см. разд. 6.5.3.3), причем некоторые из этих методов явно ориентированы на ее минимизацию. О них еще будет сказано в замечаниях к разд. 6.5.

Многие из методов решения негладких задач с ограничениями разрабатывались ради конкретных приложений. Обычно эти методы опираются на технику прямого сопоставления значений функций (см. разд. 4.2.1), и поэтому их часто называют *методами прямого поиска*. В подавляющем большинстве они имеют эвристическую природу и не гарантируют успеха. Некоторые методы прямого поиска работают со штрафными или барьерными функциями (как, например, алгоритм ND из разд. 6.2.2.2). Желающим познакомиться с ними получше мы рекомендуем обзорную статью Свэна (1974).

6.3. МЕТОДЫ ПРИВЕДЕННЫХ ГРАДИЕНТОВ И ПРОЕКЦИЙ ГРАДИЕНТОВ

6.3.1. ОБЩИЕ СООБРАЖЕНИЯ

Методы, о которых речь пойдет ниже, получаются обобщением стандартной техники решения задач с линейными ограничениями на нелинейный случай. Все они далее называются *методами типа приведенных градиентов*. Эти методы довольно сложны и нередко бывают представлены в таком виде, что за массой алгоритмических деталей усмотреть их суть и принципиальное родство бывает непросто. Мы же начнем с изложения общей основы и только потом перейдем к формальным описаниям.

Методы типа приведенных градиентов для задач с *нелинейными* ограничениями и рассмотренные в гл. 5 методы активного набора воплощают одну и ту же идею: двигаться к решению по допустимым точкам, монотонно уменьшая целевую функцию и сохраняя равенства в некоторых ограничениях (что ведет к сокращению размерности области минимизации). Последнее требует специальных средств, позволяющих постоянно (т. е. в любой пробной точке) согласовывать значения переменных с выдерживаемыми равенствами. Когда ограничения линейны, такое согласование достигается за счет особой конструкции направлений поиска. На нелинейный случай данный подход непосредственно не обобщается. Поэтому для отслеживания изогнутой границы допустимого множества приходится применять итеративные процедуры «коррекции».

Проиллюстрируем сказанное на примере двумерной задачи с единственным ограничением вида

$$5 + x_1^2 \exp(x_1 \sin(x_2 - x_1)) = 0. \qquad (6.11)$$

Он задает функциональную связь между x_1 и x_2, в силу которой одна переменная определяет другую. Таким образом, для минимизации остается только одна степень свободы. Чтобы при известной x_1 подобрать x_2, обеспечивающую (6.11), надо применить какую-нибудь процедуру поиска нуля типа описанных в разд. 4.1.1.

Делать допустимые шаги с уменьшением целевой функции при сохранении ряда равенств можно по-разному. Разными могут быть

и правила перебора списков «замораживаемых» на равенствах ограничений. Конкретный набор соответствующих приемов дает конкретный метод типа приведенных градиентов.

Раньше термины «проекция градиента» и «приведенный градиент» употреблялись в отношении вполне определенных преобразований. Здесь, однако, они трактуются значительно более свободно и относятся к любому преобразованию «проектирования» F в «приведенное подпространство» точек, удовлетворяющих набору условий-равенств.

6.3.2. ПОИСК ПРИ ОГРАНИЧЕНИЯХ-РАВЕНСТВАХ

В этом разделе описана итерация метода типа приведенных градиентов при заданном списке фиксируемых на равенствах ограничений (обсуждение вопроса о том, как строить такие списки при решении задач с неравенствами, откладывается до разд. 6.3.3). Через $\bar{c}(x)$ будем обозначать m-мерную вектор-функцию, а через x_k — текущее приближение, причем $\bar{c}(x_k) = 0$. Следующим приближением будет точка x_{k+1}, в которой должны быть выполнены равенство $\bar{c}(x_{k+1}) = 0$ и условие «существенного убывания» F (см. разд. 4.3.2.1). Шаг из x_k в x_{k+1} обозначим через s_k.

6.3.2.1. Определение составляющей шага в нуль-пространстве градиентов ограничений. Первое, что требуется от искомого шага s_k, — это подчинение равенству

$$\bar{c}(x_k + s_k) = 0. \tag{6.12}$$

Будь вектор-функция $\bar{c}$ линейной, для конструктивного описания всех подходящих s_k мы воспользовались бы техникой, изложенной в разд. 5.1.1. При нелинейной $\bar{c}$ эта техника тоже полезна; она позволяет описать первые приближения для s_k. Возьмем линейную аппроксимацию $\bar{c}$ в окрестности x_k по Тейлору:

$$\bar{c}(x_k + s_k) \approx \bar{c}(x_k) + \bar{A}(x_k)\, s_k = \bar{A}(x_k)\, s_k. \tag{6.13}$$

Из нее видно, что множество допустимых s_k аппроксимируется подпространством векторов p_k, представляющих собой решения системы линейных уравнений вида

$$\bar{A}_k p_k = 0, \tag{6.14}$$

где через $\bar{A}_k$ обозначена матрица $\bar{A}(x_k)$. Эта система аналогична системе (5.5), определяющей допустимые направления в линейном случае, но только теперь в силу нелинейности ограничений матрица $\bar{A}_k$ от итерации к итерации будет меняться.

По аналогии с (5.6) можно утверждать, что множество p_k, удовлетворяющих (6.14), описывается формулой

$$p_k = Z_k p_Z, \tag{6.15}$$

где Z_k есть матрица, чьи $n-l$ столбцов формируют базис подпространства векторов, ортогональных строкам $\bar{A}_k$, а p_Z — произвольный $(n-l)$-мерный вектор. Таким образом, для построения приближений к s_k есть $n-l$ степеней свободы (представленных вектором p_Z).

Конкретный p_Z обычно выбирают так, чтобы обеспечить минимум какой-то аппроксимации целевой функции, выраженной через p_Z. К примеру, в ранее рассмотренных алгоритмах применялся вектор «наискорейшего спуска»

$$p_Z = -Z_k^t g_k. \tag{6.16}$$

(Отметим, что $Z_k Z_k^t g_k$ есть проекция градиента целевой функции F на подпространство векторов, ортогональных градиентам активных ограничений.) Чтобы получить более высокую скорость сходимости, при выборе p_Z надо использовать информацию о вторых производных; в частности, p_Z можно вычислять как решение уравнения

$$Z_k^t W_k Z_k p_Z = -Z_k^t g_k, \tag{6.17}$$

где W_k — некоторое приближение матрицы Гессе функции Лагранжа.

Матрицу Z_k определяют по-разному. Распространенным способом является метод исключения переменных, представленный в разд. 5.1.3.2. Алгоритмы, в которых он реализован, объединяют под названием «методы обобщенных приведенных градиентов». Напомним, что данный способ построения Z_k исходит из расчленения $\bar{A}_k$ на две составляющие, одна из которых невырожденная квадратная $t \times t$-матрица. К примеру, если первые t столбцов $\bar{A}_k$ линейно независимы, $\bar{A}_k$ можно расчленить так:

$$\bar{A}_k \equiv (V\ U),$$

где V невырожденна. Тогда соответствующей Z_k будет

$$Z_k = \begin{pmatrix} -V^{-1}U \\ I \end{pmatrix}. \tag{6.18}$$

При использовании Z_k из (6.18) процедура построения p_k становится особенно прозрачной. Разобьем, подобно $\bar{A}_k$, вектор p_k на составляющие p_V и p_U (то же сделаем и с градиентом g_k). Условие (6.14) позволяет выразить t «зависимых» переменных p_V через $n-l$ «независимых» p_U, т. е. понижение размерности до $n-l$ может быть реализовано в *терминах исходных координат*. Это и происходит, когда применяют матрицы (6.18): в данном случае p_Z совпадает с p_U. Название «методы обобщенных приведенных градиентов» объясняется тем, что просто *методом приведенных градиентов* принято называть алгоритм с $p_Z =$

$= +U^2V^{-1}g_1 - g_2$, т. е. с $p_Z = -Z_k g_k$, где Z_k определяется формулой (6.18).

6.3.2.2. Восстановление допустимости. Если бы ограничения были линейными, шаг из x_k в x_{k+1} можно было бы искать в виде γp_k, где γ — некоторое число, а p_k задается формулой (6.15). Однако *при нелинейных* ограничениях такой шаг, вообще говоря, приведет в недопустимую точку, т. е. использовать p_k как «направление поиска» в обычном понимании этого термина не удается. Стандартная для методов типа приведенных градиентов формула определения допустимого шага s_k выглядит следующим образом:

$$s_k = \gamma p_k + Y_k p_Y. \tag{6.19}$$

Здесь Y_k — матрица, составленная из векторов базиса ранг-пространства A_k^T.

Требование допустимости точки $x_k + s_k$ означает, что число γ и вектор p_Y в (6.19) будут связаны друг с другом. Проще говоря, p_Y *зависит от* γ, и, *таким образом, подбирать значения* γ — *процедура более сложная, чем обычный одномерный поиск.*

Как правило, значение γ, позволяющее получить допустимый шаг, определяется в итеративном режиме: просматриваются пробные γ, для каждого из которых делается попытка решить l нелинейных уравнений

$$\tilde{c}(x_k + s_k) \equiv \tilde{c}(x_k + \gamma p_k + Y_k p_Y) = 0 \tag{6.20}$$

относительно l неизвестных компонент вектора p_Y. Поиск p_Y есть *адаптивная подзадача*, так как количество обращений к процедурам расчета функций ограничений (и их градиентов), которое потребуется для решения системы (6.20), заранее неизвестно. Эта задача может оказаться дефектной — система (6.20), вообще говоря, будет разрешимой не для любого γ. При естественных предположениях разрешимость (6.20) гарантирована, если значение γ достаточно мало.

В методах обобщенных приведенных градиентов матрицу Y_k строят так:

$$Y_k = \begin{matrix} \left.\begin{matrix} I \\ 0 \end{matrix}\right\} \begin{matrix} l \\ n-l \end{matrix} \end{matrix}.$$

Это значит, что допустимость s_k обеспечивается подгонкой вариаций зависимых переменных x_k. (Именно таким способом в предыдущем разделе предлагалось сохранять равенство (6.11).)

6.3.2.3. Уменьшение значения целевой функции. Мало найти пару γ и p_Y, удовлетворяющую (6.20), — надо подобрать γ так, чтобы было выполнено еще и условие «существенного убывания»

$$F(x_k + s_k) \leqslant F(x_k) - \delta_k, \tag{6.21}$$

где δ_k — некоторая положительная величина (см. разд. 4.3.2.1). В слабых предположениях относительно p_Z можно доказать, что среди малых γ всегда найдутся подходящие. После того как значение γ, обеспечивающее (6.20) и (6.21), получено, в качестве x_{k+1} берут $x_k + s_k$, определяют новый вектор p_Z и т. д.

Подводя итоги, можно сказать, что построение шага s_k в методах типа приведенных градиентов предполагает «линейный» расчет p_Z и «нелинейную» подгонку p_Y. При этом решается подзадача выбора γ, которая адаптивна сама по себе и, кроме того, включает подчиненную адаптивную подзадачу поиска корня системы (6.20).

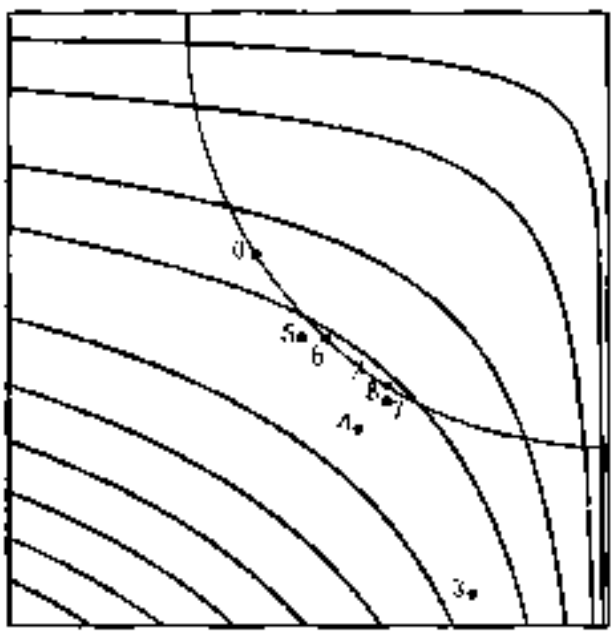

Рис. 6g. Точки двух первых итераций метода типа приведенных градиентов при решении задачи из примера 6.2 с ограничением, зафиксированным на равенстве.

На рис. 6g изображены результаты двух первых итераций метода типа приведенных градиентов для задачи из примера 6.2, единственное ограничение которой фиксируется на равенстве. Цифрой 0 помечено начальное приближение. Прочими цифрами занумерованы пробные точки метода (в том порядке, в котором они просматривались). Первая и вторая пробные точки в выделенный фрагмент не попали. Значения p_Z определялись из уравнения (6.17). Точки с номерами 6 и 8 — это два найденных методом допустимых точек приближения с уменьшающимися значениями F. В данном примере получилось так, что первые же значения γ, для которых удавалось подобрать p_Y, удовлетворяли и (6.21).

6.3.2.4. Свойства методов типа приведенных градиентов. Методы приведенных градиентов и проекций градиентов включены во многие библиотеки математического обеспечения и очень неплохо работают для задач, ограничения которых «почти линейны». Если же нелинейность ограничений значительна и начальная точка далека от оптимальной, требование соблюдения равенства (6.20) с высокой точностью может приводить к тому, что продвижение к решению будет осуществляться очень малыми шагами. Дело в том, что, чем больше кривизна допустимой поверхности, тем меньше значения γ, позволяющие восстановить допустимость и при этом не растерять выигрыш по критерию, достигнутый шагом из x_k в касательном направлении. Методы типа приведенных градиентов имеют тенденцию идти к решению «вдоль границы», и это становится не-

достатком, когда она сильно искривлена. Так проявляется неточность линейной аппроксимации ограничений, положенной в их основу.

Чтобы избежать потерь эффективности, связанных с требованием жесткого соблюдения ряда равенств, неоднократно выдвигалось предположение занижать точность их выполнения на промежуточных итерациях. Это, однако, означает отход от основного принципа методов типа приведенных градиентов, т. е. приводит к методам иной категории. Некоторые из них могут быть интерпретированы как методы спроектированного лагранжиана, рассматриваемые в разд. 6.5.

6.3.3. ОПРЕДЕЛЕНИЕ РАБОЧЕГО СПИСКА

Чтобы применить описанную выше технику для задач с неравенствами, ее надо скомбинировать с какой-нибудь «стратегией активного набора» подобно тому, как это делается в линейных случаях. Здесь есть широкое поле возможностей, но, хотя способы воплощения предлагаемого подхода бывают разными, суть всегда одна: на каждой итерации как-то прогнозируется список активных в решении неравенств и выделенные тем самым ограничения «замораживаются» на равенствах. Отметим отличие от методов барьерных функций (см. разд. 6.2.1.2), в которых все ограничения все время выполняются как неравенства и могут становиться равенствами лишь в пределе.

Стратегии активного набора реализуются введением «рабочего списка». Это — перечень ограничений, которые в данный момент трактуются как равенства, причем только они учитываются при выборе p_2 и вычислении поправки p_y. От итерации к итерации состав рабочего списка может изменяться.

Для задач с неравенствами помимо прочего усложняется процедура построения шага s_k. В частности, подбирая γ в (6.19), надо следить за неактивными в начале итерации ограничениями. Если полученное значение γ приводит к тому, что какое-то из них нарушается, γ надо уменьшить и итеративно подогнать таким образом, чтобы вновь проявляющееся ограничение стало равенством (после чего ввести последнее в рабочий список очередной итерации).

Центральным местом любой стратегии активного набора является правило вывода ограничений из рабочего списка. Эти правила чаще всего опираются на оценки множителей Лагранжа, которые подробно рассмотрены в разд. 6.6. Критерии целесообразности вывода ограничения и способы вычисления оценок множителей в разных версиях схемы типа приведенных градиентов бывают разными.

Иногда в методах типа приведенных градиентов стратегию активного набора реализуют неявно, преобразуя все нелинейные ограничения в равенства введением вспомогательных переменных, подчиняемых условиям неотрицательности. Тогда эти ограничения

всегда будут в рабочем списке; активность исходных неравенств в данном случае определяется тем, обращаются ли в нуль соответствующие вспомогательные переменные.

Замечания и избранная библиография к разделу 6.3

Идея «приведения» или «проектирования» градиентов первоначально высказывалась в отношении задач с *линейными* ограничениями. Ссылки на соответствующие работы даны в замечаниях к разд. 5.1 и 5.2. Для задач с нелинейными ограничениями первый метод «проекций градиентов» был предложен Розеном (1961), который обобщил сконструированный им же аналогичный метод минимизации при линейных ограничениях (Розен, 1960). Для определения p_Z в алгоритме Розена неявно используется ортогональная матрица Z_k.

Авторами схемы обобщенных приведенных градиентов являются Абади и Карпентье (1965, 1969). В ее первой реализации матрица (6.18) строилась в явном виде. Программа, разработанная Абади и его коллегами по Electricité de France, широко применялась для решения практических задач (см., например, Абади и Гигу (1970)). Экспериментальное сопоставление различных методов, проведенное Колвиллом (1968), показало, что методы обобщенных приведенных градиентов были одними из самых надежных и эффективных. К настоящему моменту их семейство пополнилось большим количеством новых алгоритмов: как уже было отмечено в разд. 6.3.1, для конкретизации каждого пункта схемы типа приведенных градиентов есть немало возможностей.

В последние годы над методами типа приведенных градиентов в числе прочих работали Сарджент и Муртаф (1973), Абади (1978), Лэсдон и Уорен (1978).

Известно, что при существенно нелинейных ограничениях методы типа приведенных градиентов работают плохо, поскольку сильно искривленные границы допустимого множества отслеживать тяжело (см. разд. 6.3.2). В связи с этим выдвигались разные предложения, и в частности предлагалось занижать требуемую точность соблюдения ограничений. Правда, последнее означает пересмотр принципиальных позиций и переход в класс методов спроектированного лагранжиана (см. разд. 6.5). Общий обзор методов типа приведенных градиентов с некоторыми соображениями относительно их связей с методами спроектированного лагранжиана читатель найдет в статье Сарджента (1974).

6.4. МЕТОДЫ МОДИФИЦИРОВАННЫХ ФУНКЦИЙ ЛАГРАНЖА

К классу *методов модифицированных функций Лагранжа* можно идти разными путями. Однако конечную цель всегда формулируют одинаково: свести поиск решения задач NEP или NIP к минимиза-

ции без ограничений, причем вспомогательную функцию Φ_U подобрать так, чтобы (i) не было неизбежной плохой обусловленности (как в методах гладких штрафных и барьерных функций из разд. 6.2.1) и (ii) Φ_U имела непрерывные первые производные (в отличие от негладких штрафных функций из разд. 6.2.2).

6.4.1. ОПРЕДЕЛЕНИЕ МОДИФИЦИРОВАННОЙ ФУНКЦИИ ЛАГРАНЖА

Методы, которые мы будем рассматривать здесь и в разд. 6.5, будут получены из рассмотренных ранее условий оптимальности (см. разд. 3.4) и существенно используют множители Лагранжа, а точнее их оценки. Сначала изложим основные идеи, не останавливаясь на деталях реализации. Подробное описание способов оценивания множителей в задачах с нелинейными ограничениями будет дано в разд. 6.6.

Методы модифицированных функций Лагранжа можно использовать и для задач с равенствами, и для задач с неравенствами. Принцип учета равенств всегда один и тот же, а для неравенств возможны варианты. Обычно используется какая-нибудь *стратегия просматривания активного набора*, и тогда в начале каждого цикла безусловной минимизации по некому правилу определяется, какие ограничения войдут в $\hat{c}$ (см. разд. 6.5.5). Другие схемы учета неравенств в методах модифицированных функций Лагранжа обсуждаются в замечаниях. Пока же ради простоты изложения предположим, что заранее известен правильный список активных в x^* ограничений, и через $\hat{c}(x)$ будем обозначать вектор, составленный из их функций.

В дальнейшем считается, что в искомой точке x^* выполнены достаточные условия оптимальности из разд. 3.4, в частности

$$g(x^*) = \hat{A}(x^*)^T \lambda^*. \tag{6.22}$$

Здесь $\hat{A}$ — матрица, чьи l строк представляют собой градиенты активных в x^* ограничений. Если $\hat{A}(x^*)$ имеет полный ранг, вектор λ^* определяется равенством (6.22) однозначно.

Введем функцию Лагранжа

$$L(x, \lambda) \equiv F(x) - \lambda^T \hat{c}(x). \tag{6.23}$$

(Заметьте, что определение (6.23) зависит от того, как формируется вектор $\hat{c}$.) Соотношение (6.22) означает, что при $\lambda = \lambda^*$ x^* будет ее стационарной (по x) точкой. Если бы эта стационарность означала экстремальность, функцию Лагранжа можно было бы использовать в качестве Φ_U. Однако в общем случае *x^* не доставляет минимума* $L(x, \lambda^*)$ (см. второй фрагмент рис. 6b). Значит, даже имея λ^*, рассчитывать на то, что x^* удастся найти безусловной минимизацией функции Лагранжа, нельзя.

Обозначим через $W(x, \lambda)$ матрицу вторых производных по x от $L(x, \lambda)$, т. е.

$$W(x, \lambda) \equiv G(x) - \sum_{i=1}^{t} \lambda_i \hat{G}_i(x).$$

У $W(x^*, \lambda^*)$ могут быть и отрицательные, и нулевые собственные значения, но *спроектированную матрицу Гессе* $Z^T(x^*) W(x^*, \lambda^*) Z(x^*)$ в силу сказанного ранее мы считаем положительно определенной (через $Z(x)$ принято обозначать матрицу, столбцы которой формируют базис пространства векторов, ортогональных строкам $\hat{A}(x)$). Следовательно, x^* будет *точкой минимума* $L(x, \lambda^*)$ *по* x *на линеаризации, ортогональном градиентам активных в* x^* *ограничений.*

Указанное свойство оптимальности x^* означает, что подходящую функцию Φ_U можно получить, добавив к лагранжиану слагаемое, которое, не нарушив стационарности x^*, изменит свойства матрицы Гессе относительно векторов из подпространства, натянутого на столбцы $\hat{A}(x^*)$. Чаще всего в качестве такой поправки используют квадратичный штраф, получая тем самым *модифицированную функцию Лагранжа* вида

$$L_A(x, \lambda, \rho) = F(x) - \lambda^T \hat{c}(x) + \frac{\rho}{2} \hat{c}(x)^T \hat{c}(x). \qquad (6.24)$$

Здесь ρ — положительный параметр штрафа.

В точке x^* и сам квадратичный штраф, и его градиент обращаются в нуль (поскольку $\hat{c}$ состоит из функций активных в x^* ограничений). Таким образом, при $\lambda = \lambda^*$ точка x^* является стационарной (по x) для (6.24). Матрица Гессе штрафного терма равна $\sum_i \hat{c}_i(x) \hat{G}_i(x) + \hat{A}(x)^T \hat{A}(x)$. При $x = x^*$ ненулевым в этом выражении будет только второе слагаемое $\hat{A}(x^*)^T \hat{A}(x^*)$. Это — положительно полуопределенная матрица, причем все ее собственные значения, отвечающие собственным векторам из ранг-пространства $\hat{A}(x^*)^T$, больше нуля. Значит, поправка к функции Лагранжа в (6.24) приводит к увеличению (возможно, отрицательных) собственных значений матрицы W для векторов из ранг-пространства $\hat{A}(x^*)^T$, в то время как свойства матрицы Гессе в отношении векторов, ортогональных $\hat{A}(x^*)^T$, сохраняются. Нетрудно доказать существование *конечного* значения $\bar{\rho}$, такого, что для любого $\rho > \bar{\rho}$ матрица Гессе $\nabla^2_{xx} L_A(x^*, \lambda^*, \rho)$ будет положительно определена и соответственно в x^* реализуется *сильный локальный минимум* $L_A(x, \lambda^*, \rho)$. При этом независимо от ρ для любой матрицы $Z(x^*)$, ортогональной строкам $\hat{A}(x^*)$, справедливо равенство

$$Z(x^*)^T \nabla^2_{xx} L_A(x^*, \lambda^*, \rho) Z(x^*) = Z(x^*)^T W(x^*, \lambda^*) Z(x^*),$$

т. е. на спроектированную матрицу Гессе штрафная добавка не влияет.

Проиллюстрируем эффект описанной модификации функции Лагранжа на простом примере.

Пример 6.4. Рассмотрим одномерную задачу

$$\text{найти } \min_{x \in \mathbb{R}^1} x^3$$

при ограничении $x+1=0$.

Ее единственной допустимой и, следовательно, оптимальной точкой является $x^*=-1$, причем $\lambda^*=3$. Таким образом,

$$L(x, \lambda^*) = x^3 - 3(x+1).$$

Эта функция в x^* имеет локальный максимум. В то же время при любом $\rho > \bar{\rho}$ ($\bar{\rho}=6$) модифицированная функция Лагранжа

$$L_A(x, \lambda^*, \rho) = x^3 - 3(x+1) + \frac{\rho}{2}(x+1)^2$$

в x^* достигает локального минимума. Графики для x^3 и $L(x, \lambda^*)$ изображены на рис. 6h сплошной и пунктирной линиями. Модифи-

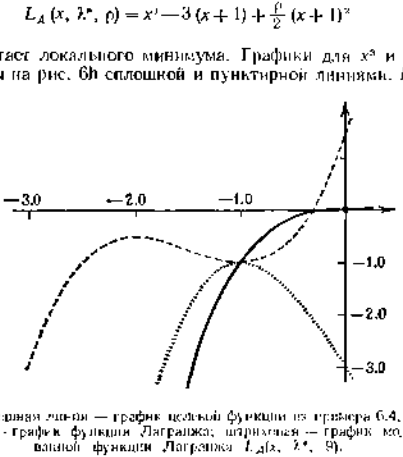

Рис. 6h. Сплошная линия — график целевой функции из примера 6.4, $F(x)=x^3$; пунктирная — график функции Лагранжа; штриховая — график модифицированной функции Лагранжа $L_A(x, \lambda^*, 9)$.

цированная функция Лагранжа с $\rho=9$ показана штриховой линией. Отметим, что она не ограничена снизу при любом значении параметра штрафа ρ.

6.4.2. СХЕМА АЛГОРИТМОВ С МОДИФИЦИРОВАННЫМИ ФУНКЦИЯМИ ЛАГРАНЖА

Из сказанного в разд. 6.4.1 следует, что в идеальном случае x^* можно найти однократной безусловной минимизацией гладкой функции (6.24). Однако этот случай практического интереса не представляет, поскольку предполагает знание вектора λ^*, а, не имея решения x^*, его вычислить нельзя. В реальной ситуации вместо истинных множителей Лагранжа используются их оценки. Центральная роль, которую они играют в методах модифицированных функций Лагранжа, объясняет, почему эти методы часто называют также *методами множителей*.

Известно несколько способов применения функций типа (6.24) для поиска условного минимума. Едины они в том, что всегда возникает n-мерная подзадача *безусловной минимизации* дифференцируемой функции, имеющей в качестве параметров оценки множителей Лагранжа и включающей штраф, назначение которого превратить x^* из безусловно стационарной точки в безусловно оптимальную.

6.4.2.1. Модельная схема. В данном разделе схематично описан один из способов поиска минимума при ограничениях-неравенствах с использованием модифицированной функции Лагранжа. Он предполагает задание начального списка включаемых в $\hat{c}$ ограничений, начальной оценки вектора множителей Лагранжа λ_0, начального приближения x_0, начального значения параметра штрафа и положительного целого числа K, которое послужит верхней границей количества итераций. Определив все это и сделав присвоение $k \leftarrow 0$, надо выполнить следующие действия.

Алгоритм **AL** (*Модель метода модифицированной функции Лагранжа*)

AL1. [Проверка соблюдения правил останова.] Если x_k удовлетворяет условиям оптимальности или $k > K$, вычисления прекратить. В первом случае x_k берется в качестве искомого решения; во втором выдается признак неудачи.

AL2. [Минимизация модифицированной функции Лагранжа.] Использовав x_k в качестве начальной точки, применить процедуру решения подзадачи

$$\text{найти } \min_{x \in \mathbb{R}^n} L_A(x, \lambda_k, \rho), \tag{6.25}$$

предусматривающую возможность неограниченности L_A снизу. Лучшее из найденных ею приближений взять в качестве x_{k+1}.

AL3. [Пересчет оценок множителей Лагранжа.] Если это уместно, изменить состав $\hat{c}$. Вычислить новый вектор оценок множителей Лагранжа λ_{k+1}.

AL4. [Увеличение, если это необходимо, параметра штрафа.] Если в точке x_{k+1} невязки ограничений задачи существенно меньше, чем в x_k, перейти к следующему шагу, а иначе увеличить ρ.

AL5. [Перевод счетчика итераций.] Сделать присвоение $k \leftarrow k+1$ и вернуться к шагу AL1.

6.4.2.2. Свойства модифицированной функции Лагранжа. Предполагая использовать какой-нибудь метод с модифицированной функцией Лагранжа L_A, надо иметь в виду следующие обстоятельства.

Локальность минимума. Как и для штрафных функций, описанных в разд. 6.2 (в отсутствие у исходной задачи специальных свойств), в точке x^* в лучшем случае реализуется лишь *локальный* минимум L_A. При этом L_A может быть не ограниченной снизу для любого значения параметра штрафа ρ. Следовательно, применяя к подзадаче (6.25) стандартный алгоритм безусловной минимизации, нужно предусмотреть в нем соответствующие меры предосторожности (на что и указано в п. AL2 модельной схемы). Обычно увеличением ρ в конце концов удается получить настолько широкую область притяжения x^*, что неограниченность l_A становится неопасной. Однако полных гарантий сходимости к x^* в общем случае дать нельзя.

Трудности выбора параметра штрафа. Подобрать подходящий параметр ρ иногда бывает непросто даже в таких ситуациях, когда проблема неограниченности снизу модифицированной функции Лагранжа не возникает. Дело в том, что среди значений ρ, обеспечивающих существование в x^* локального безусловного минимума L_A, наряду со «слишком большими» обычно есть и «слишком малые». И те и другие дают плохо обусловленные L_A. При завышенных ρ функция L_A становится плохо обусловленной по тем же причинам, что и гладкие штрафные функции. Причиной плохой обусловленности L_A при заниженных ρ является вырождение ее матрицы Гессе (если матрица Гессе обычной функции Лагранжа знакоопределена) при минимальном допустимом ρ.

На рис. 6i изображены линии уровня $L_A(x, \lambda^*, \rho)$ при трех значениях параметра ρ для задачи из примера 6.2, ограничение которой тождественно равенство. Вторая функция ($\rho = 0.2$) обусловлена хорошо. Первая ($\rho = 0.075$) и третья ($\rho = 100$) иллюстрируют последствия выбора слишком малых и слишком больших ρ.

Критическая роль оценок множителей Лагранжа. Поскольку x^* будет точкой минимума модифицированной функции Лагранжа лишь если $\lambda = \lambda^*$, для сходимости использующего ее метода при ограниченном ρ необходимо обеспечить сходимость генерируемых оценок множителей к λ^*.

На рис. 6j показаны линии уровня функции (6.24) для задачи из примера 6.2 (ограничение которой трактуется как равенство) при трех значениях λ ($\lambda = 0.5$, $\lambda = 0.9$ и $\lambda = 1.0$). Параметр штрафа был взят равным 0.2. Такой выбор ρ обеспечивает наилучшую обусловленность функции (6.24) с $\lambda = \lambda^*$. Из приведенных фрагментов

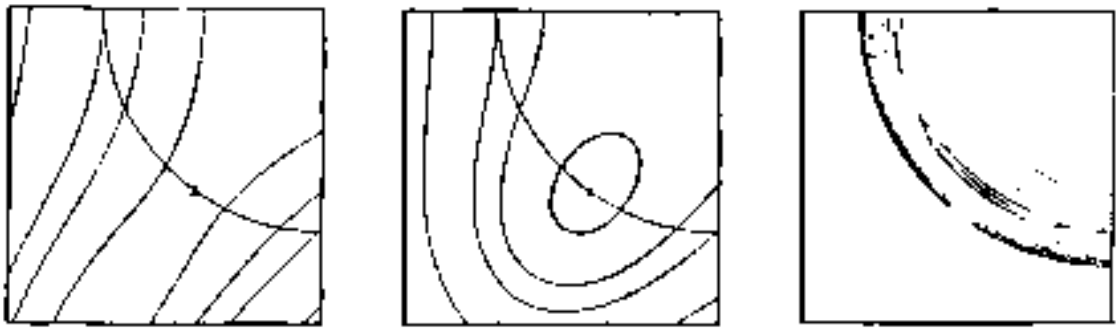

Рис. 6i. Линии уровня модифицированных функций Лагранжа $L_д(x, \lambda^*, \rho)$ с $\rho = 0.075$, $\rho = 0.2$ и $\rho = 100$ для задачи из примера 6.2 с ограничением, трактуемым как равенство.

видно, как сильно иногда влияют на характер поведения модифицированной функции Лагранжа даже малые вариации оценок множителей.

Можно показать, что приближения $\{x_k\}$ из модельной схемы разд. 6.4.2.1 будут сходиться к x^* не быстрее, чем оценки $\{\lambda_k\}$ к

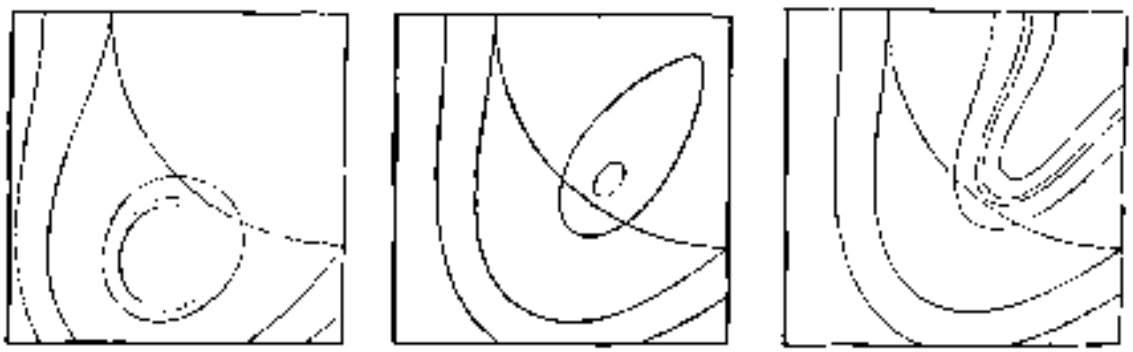

Рис. 6j. Линии уровня модифицированных функций Лагранжа $L_д(x, \lambda, \rho)$ с $\lambda = 0.5$, $\lambda = 0.9$ и $\lambda = 1.0$ для задачи из примера 6.2 с ограничением, трактуемым как равенство.

λ^*. Отсюда, в частности, следует, что квадратичная скорость сходимости к x^* достижима только тогда, когда для оценивания используется какой-нибудь метод второго порядка (см. разд. 6.6.2).

В первых реализациях схемы модифицированных функций Лагранжа правило пересчета λ_k формулировали, исходя из обеспеченного в решении x подзадачи (6.25) равенства

$$g(\bar{x}) - \hat{A}(\bar{x})^T \lambda_k + \rho \hat{A}(\bar{x})^T \hat{c}(x) = 0. \tag{6.26}$$

Оно говорит о том, что вектор

$$\lambda_{k+1} = \lambda_k - \rho \hat{c}(\bar{x}) \tag{6.27}$$

играет в точке $\bar{x}$ ту же роль, что λ^* в x^*, т. е. задает разложение градиента $g(\bar{x})$ по строкам матрицы $A(\bar{x})$. Поэтому представляется естественным взять (6.27) в качестве формулы пересчета λ_k, что и делалось. (Как будет показано в разд. 6.6.1, при некоторых условиях (6.27) определяет линейные оценки множителей.)

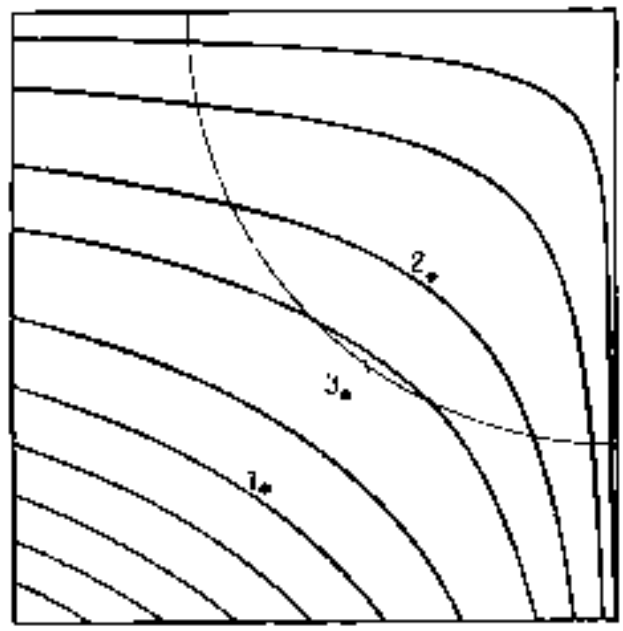

Рис. 6k. Три первые точки, найденные методом модифицированной функции Лагранжа для задачи из примера 6.2 с ограничением, трактуемым как равенство; каждая есть результат безусловной минимизации.

На рис. 6k изображены три первые точки, полученные алгоритмом AL для задачи примера 6.2 (с ограничением, трактуемым как равенство), причем исходной оценкой для λ^* послужила $\lambda_0 = 0$, а последующие оценки вычислялись по формуле (6.27). Параметр штрафа ρ был зафиксирован равным 0.6. Последовательность $\{\lambda_k\}$ из (6.27) в общем случае сходится лишь линейно, и потому алгоритм AL с пересчетом оценок множителей по формуле (6.27) тоже оказывается линейно сходящимся. Для примера 6.2 набор $\{\|x_k - x^*\|_2\}$, $k = 3, \ldots, 8$, выглядит так: 1.69×10^{-1}, 9.17×10^{-2}, 4.61×10^{-2}, 2.41×10^{-2}, 1.23×10^{-2}, 6.39×10^{-3}. Ошибка уменьшается приблизительно вдвое на каждой итерации. Если же применить к задаче примера 6.2 алгоритм AL, в котором по-прежнему $\lambda_0 = 0$ и $\rho = 0.6$, но множители оцениваются методом второго порядка (см. разд. 6.6.2), то значениями $\{\|x_k - x^*\|_2\}$, $k = 1, \ldots, 6$, будут 5.61×10^{-1}, 1.88×10^{-1}, 2.75×10^{-2}, 6.59×10^{-4}, 3.87×10^{-7} и 3.19×10^{-13}. Здесь очевидна квадратичная сходимость.

*6.4.3. ВАРИАЦИИ СТРАТЕГИИ ПОИСКА

Схема разд. 6.4.2.1 представляет собой не более чем один из многих способов применения модифицированной функции Лагранжа. Понятно, что идти к равенству (6.22) можно и по-другому.

Когда значение параметра штрафа еще не подогнано или оценки множителей Лагранжа существенно неточны, усилия, затрачиваемые на поиск аккуратного решения подзадачи (6.25) в алгоритме AL, могут оказаться неоправданными. Более того, не исключено, что эффективность поиска в целом повысится, если уменьшить точность безусловной минимизации на всех шагах (повысив тем самым «частоту» пересчета оценок λ_k). Исходя из этих соображений,

предлагались методы модифицированных функций Лагранжа с различными степенями точности решения (6.25). Среди них — «крайний» метод, в котором x и λ пересчитываются синхронно; точнее говоря, выполняется одна итерация спуска для (6.25), затем пересчитывается λ, делается один шаг безусловной минимизации при новом λ и т. д. В последнем случае на каждой итерации ставится новая подзадача. При этом шаги спуска по x обычно выбирают на основании квадратичной аппроксимации L_A, и, коль скоро это так, рассматриваемый метод можно проектировать в терминах данной аппроксимации, а не самой L_A. Тогда его уместно будет отнести к классу методов спроектированного лагранжиана с квадратичной подзадачей (которые мы рассмотрим в разд. 6.6.3).

Замечания и избранная библиография к разд. 6.4.

Кем и когда идея модификации функции Лагранжа была высказана впервые, не ясно. Во всяком случае в середине 40-х годов она уже была достаточно популярна (см. Хестенс (1946, 1947)). С историей этого вопроса можно познакомиться по докладу Хестенса (1979). В практику численной оптимизации модифицированные функции Лагранжа вошли относительно недавно. Подход, представленный выше, в основном следует построениям Хестенса (1969). Другой вывод методов модифицированных функций Лагранжа был независимо дан Пауэллом (1969). Он предложил применять квадратичный штрафной терм, в который каждое ограничение входит со своим «сдвигом» и «весом». В алгоритме Пауэлла (для задачи NLP) целевая функция подзадачи безусловной минимизации выглядит так:

$$F(x)+\frac{1}{2}\sum_{i=1}^{r}\sigma_i\,(c_i(x)-\theta_i)^2,$$

где σ_i — положительный вес, а θ_i — сдвиг для i-го ограничения. Если взять все веса равными, она будет отличаться от (6.24) только не зависящим от x слагаемым. Для пересчета оценок множителей как Хестенс, так и Пауэлл рекомендовали формулу (6.27). В качестве еще одной из первых работ по методам модифицированных функций Лагранжа можно назвать статью Хаархоффа и Байса (1970). Обзоры свойств этих методов читатель найдет у Бертсекаса (1975b, 1976a, b), Мангасаряна (1975) и Флетчера (1977). Общий взгляд на модифицированные функции Лагранжа излагается в статье Хестенса (1980b). Отличные от (6.24) функции данного класса описаны, например, в работе Бога и Толле (1980).

Учитывать в методах модифицированных функций Лагранжа ограничения-неравенства можно разными способами. Байс (1972) и Рокафеллар (1973a, b, 1974), например, предположили искать решение задачи с неравенствами последовательной безусловной мини-

мизацией по x обобщающей (6.24) функции вида

$$L_A(x, \lambda, \rho) = F(x) + \sum_{i=1}^{m} \begin{cases} -\lambda_i c_i(x) + \frac{\rho}{2} c_i(x)^2, & \text{если } c_i(x) \leqslant \frac{\lambda_i}{\rho}, \\ -\frac{\rho}{2} \lambda_i^2, & \text{если } c_i(x) > \frac{\lambda_i}{\rho}. \end{cases} \tag{6.28}$$

В данной записи λ представляет собой «расширенный» вектор множителей Лагранжа. Как сама функция (6.28), так и ее градиент непрерывны всюду, в том числе в точках, где какое-то из «активных» ограничений (т. е. тех, для которых $c_i \leqslant \lambda_i/\rho$) становятся «неактивными» ($c_i > \lambda_i/\rho$). Вторые производные от L_A разрывны в этих точках, но есть надежда, что последние будут лежать вдали от решения, и поэтому указанная разрывность не осложнит работы алгоритма безусловной минимизации L_A. Аналог формулы Пауэлла – Хестена (6.27) для модифицированной функции Лагранжа (6.28) таков:

$$\bar{\lambda}_i = \lambda_i - \min(\rho c_i, \lambda_i).$$

Здесь через $\bar{\lambda}_i$ обозначено обновленное значение оценки i-го множителя. Эта формула гарантирует неотрицательность λ_i для «активных» неравенств и дает нулевые λ_i для «неактивных». Иные формулы пересчета, предназначенные для методов с более частыми сменами λ_i, приводятся, например, в диссертации Байса (1972). К задачам с неравенствами можно применять также стратегии проецирования активного набора (см. разд. 6.5.5), когда для каждой подзадачи безусловной минимизации определяется, какие из ограничений в ней учесть, и эти ограничения трактуются как равенства.

Характер сходимости любого метода модифицированных функций Лагранжа существенно зависит от качества используемых в нем оценок множителей. Способ их построения разумно подбирать так, чтобы скорости сходимости образуемой ими последовательности и последовательностей точек, генерируемых при решении подзадач (6.25), были одинаковы. В частности, решая (6.25) процедурой ньютоновского типа, для вычисления λ_i имеет смысл взять метод второго порядка (см. разд. 6.6.2). Тогда можно будет достичь квадратичной сходимости процесса в целом. Если же к (6.25) применяется квазиньютоновская процедура, то достаточно, чтобы формулы пересчета оценок множителей обеспечивали им сверхлинейную сходимость. Характеристики методов с разными способами вычисления оценок множителей анализируются в работах Байса (1972), Бертсекаса (1975b, 1976b), Корта (1975), Берла (1976, 1978), Корта и Бертсекаса (1976), Гилла (1977, 1978), Глэда (1979) и Глэда и Полака (1979).

Много исследований было посвящено методам модифицированных функций Лагранжа с относительно неточным решением подзадач (6.25). С ними можно познакомиться по ссылкам, перечисленным в предыдущем абзаце.

Когда между двумя последовательными пересчетами оценок множителей выполняется только одна итерация спуска в (6.25), вся процедура становится похожей на метод спроектированного лагранжиана с квадратичной подзадачей (см. разд. 6.5.3). В частности, Танка (1978) показал, что при определенных условиях приближения, генерируемые комбинацией квазиньютоновского метода для (6.25) и некой формулы пересчета оценок, просто идентичны получаемым с помощью одного из методов спроектированного лагранжиана. И все же не надо забывать о концептуальном различии между методами модифицированных функций Лагранжа, базирующихся на подзадачах безусловной минимизации, и методами спроектированного лагранжиана, в основе которых лежат подзадачи с линейными ограничениями.

Модифицированные функции Лагранжа неоднократно использовались для построения комбинированных алгоритмов. Из них стоит упомянуть комбинации с методами типа приведенных градиентов, описанные в статьях Миеле, Крэгга, Айера и Леви (1971), Миеле, Крэгга и Леви (1971).

Предметом пристального внимания многих исследователей был вопрос о выборе и интерпретации параметра штрафа ρ. Особый интерес представляет связь между ρ и областью сходимости (см. Бертсекас (1979)).

Подходу к решению задач условной оптимизации, рассмотренному выше, близка идея построения *гладкой точной штрафной функции*. Так как x^* является точкой безусловного минимума (6.24) с $\lambda=\lambda^*$, можно показать, что при любой дважды дифференцируемой $\lambda(x)$, удовлетворяющей равенству $\lambda(x^*)=\lambda^*$, для достаточно больших ρ в x^* реализуется минимум

$$F(x)-\lambda(x)^T\tilde{c}(x)+\frac{\rho}{2}\tilde{c}(x)^T\tilde{c}(x). \tag{6.29}$$

Следовательно, функция (6.29), как и абсолютная штрафная (см. разд. 6.2.2.1), может послужить точной штрафной функцией. Первым ее предложил Флетчер (1970b), (см. также Флетчер и Лилл (1970), Лилл (1972) и Флетчер (1973)). К сожалению, $\lambda(x)$ определяется через производные от F и $\tilde{c}$, так что подсчет градиента (6.29) оказывается весьма трудоемкой процедурой. Дальнейшие подробности относительно методов точных гладких штрафных функций читатель найдет по приведенным выше ссылкам.

6.5. МЕТОДЫ СПРОЕКТИРОВАННОГО ЛАГРАНЖИАНА

6.5.1. ПРЕДВАРИТЕЛЬНЫЕ СООБРАЖЕНИЯ

6.5.1.1. Описание подзадачи с линейными ограничениями. Если в искомой точке x^* выполнены достаточные условия из разд. 3.4, то в ней будет достигаться минимум функции Лагранжа на множестве векторов, ортогональных градиентам активных в x^* ограниче-

ний. Последнее наводит на мысль искать x^* как решение подзадачи с *линейными ограничениями*, целевая функция которой (впредь обозначаемая через Φ_{LC}) привязана к функции Лагранжа, а условия подобраны так, чтобы минимизация происходила в нужном подпространстве. При этом непосредственное «усечение» пространства минимизации избавит от необходимости пользоваться штрафными поправками. Алгоритмы, состоящие в решении последовательности подзадач, ограничения которых линейны, а целевые функции Φ_{LC} строятся на основе функции Лагранжа, принято называть *методами спроектированного лагранжиана*. Как и Φ_L из методов модифицированных функций Лагранжа, Φ_{LC} будут включать оценки множителей.

Коль скоро приближения к x^* решено определять из подзадач с ограничениями, естественно формулировать их так, чтобы множители Лагранжа этих подзадач служили приближениями для λ^*. Обозначив через λ_k оценку вектора λ^*, используемую при постановке подзадачи в очередной точке x_k, это соображение можно оформить в виде следующего требования: из равенств $x_k = x^*$ и $\lambda_k = \lambda^*$ должно следовать, что x^* — решение подзадачи, а λ^* — ее вектор множителей Лагранжа.

6.5.1.2. Формулировка подзадачи. Ради простоты начального изложения допустим, что набор активных в x^* ограничений каким-то образом выявлен; через $\bar{c}$ в дальнейшем обозначается вектор, составленный из их функций. О методах определения активного набора речь пойдет в разд. 6.5.5.

Пусть q — шаг из неоптимальной точки x_k в x^*. Он должен удовлетворять равенству

$$\bar{c}(x^*) - \bar{c}(x_k + q) = 0. \tag{6.30}$$

Чтобы сформулировать подзадачу для расчета приближения q, воспользуемся тейлоровским разложением функции $\bar{c}$ в окрестности x_k:

$$\bar{c}(x^*) = \bar{c}_k + \bar{A}_k(x^* - x_k) + O(\|x^* - x_k\|^2) = 0, \tag{6.31}$$

где через $\bar{c}_k$ и $\bar{A}_k$ обозначены $\bar{c}(x_k)$ и $\bar{A}(x_k)$. Пренебрегая нелинейным слагаемым, получаем, что приближения к x^* можно искать среди $\bar{x}$, для которых

$$\bar{A}_k(\bar{x} - x_k) = -\bar{c}_k. \tag{6.32}$$

Система ограничений (6.32) задает множество точек, обнуляющих линейную аппроксимацию нелинейной функции $\bar{c}$, построенную в x_k. Она аналогична системе, используемой для вычисления шага в ньютоновской процедуре решения уравнений (6.30) (см. (4.72) в разд. 4.7.6).

Итак, для выбора очередного приближения предлагается решать следующую подзадачу:

$$\begin{aligned} &\text{найти } \min_{x \in \mathbb{R}^n} \Phi_{LC} \\ &\text{при ограничениях } \tilde{A}_k x = -\tilde{c}_k + \tilde{A}_k x_k. \end{aligned} \tag{6.33}$$

Не исключено, что ее ограничения окажутся несовместимыми, т. е. она будет дефектной. Однако это возможно только тогда, когда $\tilde{A}_k$ имеет неполный ранг или когда размерность $\tilde{c}$ превосходит n. Заметим также, что решение подзадачи (6.33) зависит от x_k, так как по x_k определяются параметры ее ограничений.

Считая, что подзадача (6.33) нормальна, обозначим ее решение и отвечающий ему вектор множителей Лагранжа через x_k^* и λ_k^* соответственно. Они будут подчиняться условиям оптимальности первого порядка:

$$\tilde{A}_k x_k^* = -\tilde{c}_k + \tilde{A}_k x_k \tag{6.34a}$$

и

$$\nabla \Phi_{LC}(x_k^*) = \tilde{A}_k^T \lambda_k^*. \tag{6.34b}$$

Кроме того, гарантирована положительная полуопределенность матрицы $Z_k^T \nabla^2 \Phi_{LC}(x_k^*) Z_k$ (через Z_k, как обычно, обозначена матрица, столбцами которой служат векторы базиса подпространства, ортогонального строкам $\tilde{A}_k$).

Функцию Φ_{LC} для подзадачи (6.33) можно строить по-разному. В двух последующих разделах будут рассмотрены два типа Φ_{LC}, порождающие два класса методов спроектированного лагранжиана.

6.5.2. ПОДЗАДАЧА С ЦЕЛЕВОЙ ФУНКЦИЕЙ ОБЩЕГО ВИДА

6.5.2.1. Формулировка целевой функции. Казалось бы, в качестве Φ_{LC} можно взять текущую аппроксимацию функции Лагранжа $L(x, \lambda^*)$:

$$F(x) = \lambda_k^T c(x). \tag{6.35}$$

Если $x_k = x^*$ и $\lambda_k = \lambda^*$, ее минимум достигается в x^*, так что одно из требований к Φ_{LC} в данном случае выполнено. Однако, поскольку градиент (6.35) с $\lambda_k = \lambda^*$ в x^* обращается в нуль, вектор λ_k^* тоже будет равен нулю (а не λ^*), т. е. (6.35) в качестве Φ_{LC} не подходит.

Очень близкая к (6.35) функция, удовлетворяющая всем предъявляемым к Φ_{LC} требованиям, выглядит так:

$$F(x) = \lambda_k^T \tilde{c}(x) + \lambda_k^T \tilde{A}_k x. \tag{6.36}$$

Поскольку на допустимом множестве задачи (6.33) она отличается от (6.35) не зависящим от x слагаемым, при переходе

в (6.33) от (6.35) к (6.36) решение x_k^* сохранится. Как видно из условия (6.34b), вектор λ_k^* при этом получит приращение, равное λ_k. Соответственно, задавая Φ_{LC} в виде (6.36) при $x_k = x^*$, $\lambda_k = \lambda^*$, имеем $x_k^* = x^*$, $\lambda_k^* = \lambda^*$.

6.5.2.2. **Упрощенная модельная схема.** Чтобы сделать изложение более предметным, сформулируем упрощенный алгоритм метода спроектированного лагранжиана, который, однако, не следует воспринимать как некую универсальную схему. Ниже будут предложены вариации стратегии поиска, которые приводят к более надежным алгоритмам.

Задав начальную точку x_0 и начальный вектор λ_0 и сделав присвоение $k \leftarrow 0$, точку x^* можно попытаться найти следующим образом.

Алгоритм SP (*Упрощенная схема спроектированного лагранжиана*)

SP1. [Проверка соблюдения правил останова.] Если x_k удовлетворяет условиям оптимальности, вычисления прекращаются и x_k берется в качестве решения.

SP2. [Решение подзадачи с линейными ограничениями.] Начиная с x_k в качестве исходного приближения, запускается процедура решения подзадачи

$$\text{найти} \min_{x \in \mathbb{R}^n} F(x) - \lambda_k^T \tilde{c}(x) + \lambda_k^T \hat{A}_k x$$
$$\text{при ограничениях } \hat{A}_k x = -\hat{c}_k + \hat{A}_k x_k, \tag{6.37}$$

защищенная на случай неограниченности.

SP3. [Пересчет оценок множителей.] Лучшая из найденных на предыдущем шаге точек берется в качестве x_{k+1}, λ_{k+1} полагается равным вектору множителей Лагранжа подзадачи (6.37) (здесь считается, что (6.37) — нормальная подзадача), счетчик итераций переводится на единицу, т. е. $k \leftarrow k+1$, и осуществляется возврат к шагу SP1.

Решение подзадачи. Для поиска оптимума в подзадаче (6.37) можно использовать какой-нибудь из универсальных методов минимизации при линейных ограничениях-равенствах типа описанных в разд. 5.1. Конкретный выбор будет определяться доступностью информации о производных функций (возможностью и трудоемкостью вычисления аналитических значений производных первого и второго порядков) и размерностью задачи.

В сравнении с методом модифицированных функций Лагранжа, описанным ранее, алгоритм SP может показаться переусложненным — ведь с технической точки зрения подзадача (6.37) сложнее, чем (6.25). Но, во-первых, введение условий-равенств понижает размерность пространства минимизации, и это — положительный

эффект, а, во-вторых, если среди исходных ограничений есть линейные и они будут обрабатываться приемами гл. 5, то никакого дополнительного усложнения конструкции всего алгоритма при переходе от (6.25) к (6.37) не понадобится. В последнем случае подзадачи класса (6.37) возникают независимо от способа учета нелинейных ограничений.

Локальная сходимость. В предположении, что искомая точка x^* удовлетворяет достаточным условиям оптимальности из разд. 3.4, можно доказать, что для любого мало отличающегося от (x^*, λ^*) начального приближения (x_0, λ_0) генерируемая алгоритмом SP последовательность $\{(x_k, \lambda_k)\}$ квадратично сойдется к (x^*, λ^*). При этом, если нормы $\|x_k - x^*\|$ и $\|\lambda_k - \lambda^*\|$ есть величины порядка ε, то аналогичные нормы с x_{k+1} и λ_{k+1} будут пропорциональны ε^2. Таким образом, применение в (6.37) оценок множителей первого порядка не снижает скорости сходимости до линейной (как в методах модифицированных функций Лагранжа, обсуждавшихся в разд. 6.4.2.2).

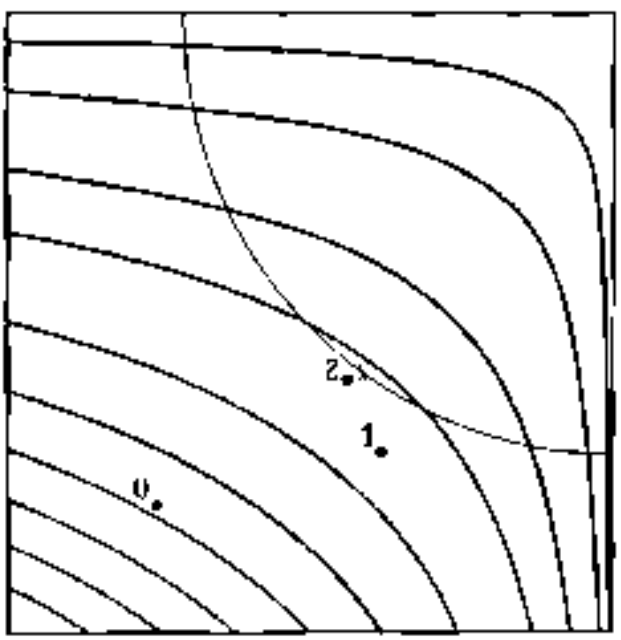

Рис. 61. Три первые точки, вычисленные методом спрямленного лагранжиана для задачи из примера 6.2; каждая есть результат решения подзадачи с линейным ограничением.

На рис. 61 изображены три первых приближения, полученные алгоритмом SP для задачи из примера 6.2 при $x_0 = (-1.5, -1.6)^T$ и $\lambda_0 = 0$. Каждое из них найдено решением соответствующей подзадачи (6.37), т. е. является результатом нескольких «внутренних» итераций. Последние на рисунке не показаны. Значения нормы $\{\|x_k - x^*\|\}$, $k = 1, \ldots, 4$, таковы: 2.7×10^{-1}, 3.20×10^{-2}, 3.57×10^{-4}, $4.50 \times \times 10^{-8}$. Уже с первых шагов отчетливо проявляется квадратичная сходимость.

***6.5.2.3. Усовершенствования модельной схемы.** Несмотря на великолепные свойства локальной сходимости алгоритма SP из разд. 6.5.2.2, рекомендовать его в качестве полноценного средства решения задач на условный минимум нельзя, так как он недостаточно надежен. Когда x_0 и λ_0 значительно отличаются от x^* и λ^*, целесообразность применения подзадач (6.37) сомнительна. Вывод (6.37) опирался на условия оптимальности (6.22) и (6.30), а они выполняются только в x^*, λ^*. Таким образом, разрешимость под-

задачи (6.37) гарантирована лишь в малой окрестности этой пары. При неточных x_k, λ_k (6.37) может не иметь решения или иметь решение, расстояние от которого до x^* больше, чем от x_k. Таким образом, запуск алгоритма SP в отсутствие хорошего начального приближения x_0, λ_0 связан с большим риском аварийного останова или расходимости.

Доработку алгоритма SP с целью обеспечить сходимость в более широком диапазоне значений x_0, λ_0 можно осуществлять по-разному. Наиболее распространенный подход заключается в том, чтобы дополнять его специальными средствами отыскания точки, которая подойдет ему в качестве начального приближения.

Для поиска «подходящего» приближения можно воспользоваться *квадратичной штрафной функцией* (см. разд. 6.2.1.1) с *относительно небольшим* параметром штрафа. Есть надежда, что при разумном ρ точка ее минимума $x^*(\rho)$ будет неплохой оценкой x^*, а хорошую начальную оценку для λ^* можно будет вычислить либо по формуле (6.4), либо каким-нибудь из способов, описываемых в разд. 6.6. Другой прием — решать на начальных итерациях подзадачи (6.33) с модифицированной функцией Лагранжа (см. разд. 6.4.1) в качестве Φ_{LC}; тогда переключение на алгоритм SP сведется к обнулению параметра штрафа. (Схема такого сорта для задач большой размерности рассмотрена в разд. 6.7.1.)

В обоих случаях важно выбрать «хорошее» значение ρ. Слишком малые ρ не позволят получить приемлемого приближения, а слишком большие порождают обычную проблему плохой обусловленности. Еще один нетривиальный вопрос: когда переключаться с модифицированной функции Лагранжа на функцию (6.36)? Должны быть приняты какие-то меры, страхующие от преждевременного запуска алгоритма.

6.5.3. КВАДРАТИЧНАЯ ПОДЗАДАЧА

6.5.3.1. Обоснование. Подход, изложенный в разд. 6.5.2, обычно критикуют за то, что усилия, затрачиваемые на поиск решения сложной подзадачи (6.37), далеко не всегда окупаются качеством нового приближения x_{k+1}. Причины ясны — если x_k плохо приближает x^*, то ограничения (6.37) плохо описывают допустимую окрестность оптимума, а при λ_k, далеком от λ^*, целевая функция (6.37) мало похожа на функцию Лагранжа в решении.

Поскольку сложность (6.37) успеха не гарантирует, возникает естественное желание вывести из свойств оптимума более простую подзадачу, и желательно, чтобы она была детерминированной, а не адаптивной (см. разд. 6.1.2). Тогда по крайней мере облегчится расчет очередного приближения. При этом важно позаботиться, чтобы желанные свойства решений сложных подзадач сохранились и у решений упрощенных.

Ниже рассмотрен класс методов, в которых Φ_{LC} — квадратичная функция, т. е. методов, для которых (6.33) — задача *квадратичного программирования*. Решения последних принято выражать в терминах шага из x_k в x_{k+1}. Чтобы упростить изложение, предположим, что правильный список активных ограничений известен (обсуждение реальных способов учета неравенств откладывается до разд. 6.5.5). В общем предлагаемая техника расчета очередного приближения применима всегда, когда активный набор можно хорошо спрогнозировать. Она во многом напоминает описанные в в гл. 5 приемы решения задач с линейными ограничениями.

Общий вид подзадачи для k-й итерации теперь будет таким:

$$\text{найти } \min_{p \in \mathbb{R}^n} d_k^T p + \frac{1}{2} p^T H_k p \tag{6.38а}$$

$$\text{при ограничениях } C_k p = b_k. \tag{6.38б}$$

Методы с подзадачами этого сорта иногда называют методами *последовательного квадратичного программирования*.

Предполагая, что решение (6.38) существует, обозначим его через p_k. С ним связан вектор множителей Лагранжа η_k, удовлетворяющий равенству

$$H_k p_k + d_k = C_k^T \eta_k. \tag{6.39}$$

Конкретную формулировку квадратичной подзадачи для точки x_k из малой окрестности x^* можно получить, опираясь на те же соображения, которые привели к (6.37). В частности, линейная аппроксимация активных ограничений по тейлоровским разложениям их функций в окрестности x даст набор равенств вида

$$\hat{A}_k p = -\hat{c}_k. \tag{6.40}$$

Здесь через $\hat{c}_k$ обозначен вектор значений в x_k функций активных ограничений, а $\hat{A}_k$ — матрица их градиентов.

Далее, функция (6.38а) интерпретируется как квадратичная аппроксимация функции Лагранжа. Соответственно H_k будет иметь смысл приближения матрицы Гессе последней. Логично предположить, что при этом в роли d_k в (6.38а) выступит градиент функции Лагранжа $g_k - \hat{A}_k^T \lambda_k$, где λ_k — текущая оценка λ^*. Однако удобнее взять d_k равным g_k — градиенту F в x_k. Решение подзадачи в обоих случаях одно и то же (так как на множестве p, заданном равенствами (6.40), функции (6.38а) с $d_k = g_k - \hat{A}_k \lambda_k$ и $d_k = g_k$ отличаются друг от друга на величину, не зависящую от p), но значения вектора η_k (6.39) окажутся разными, причем η_k, отвечающий второму способу задания d_k, может служить оценкой вектора λ для исходной задачи.

6.5.3.2. Упрощенная модельная схема. Чтобы иметь основу для последующего изложения, опишем некий упрощенный алгоритм

последовательного квадратичного программирования. Как и алгоритм SP из разд. 6.5.2.2, он не может служить достаточно универсальным средством решения реальных задач и требует модификаций.

Задав исходное приближение x_0 и начальный вектор множителей λ_0 и сделав присвоение $k \leftarrow 0$, поиск x^* можно организовать по следующей схеме.

Алгоритм SQ. *(Упрощенная схема спроектированного лагранжиана с квадратичной подзадачей)*

SQ1. [Проверка соблюдения правил останова.] Если x_k удовлетворяет условиям оптимальности, вычисления прекращаются и x_k берется в качестве решения.

SQ2. [Решение квадратичной подзадачи.] Определяется вектор p_k, оптимальный для квадратичной задачи

$$\text{найти } \min_{p \in \mathbb{R}^n} g_k^{\mathrm{T}} p + \frac{1}{2} p^{\mathrm{T}} H_k p \quad \text{при ограничениях } \bar{A}_k p = -\bar{c}_k \tag{6.41}$$

(H_k зависит от x_k и λ_k).

SQ3. [Пересчет оценки решения.] Выполняется присвоение $x_{k+1} \leftarrow x_k + p_k$, вычисляется λ_{k+1}, счетчик итераций переводится на единицу, т. е. $k \leftarrow k+1$, и осуществляется возврат к шагу SQ1.

Решение подзадачи. Оптимальный в подзадаче (6.41) вектор p_k поддается прямому расчету и удобно выражается через матрицы Y_k и Z_k, первая из которых составлена из векторов базиса ранг-пространства $\bar{A}_k^{\mathrm{T}}$, а вторая — из векторов базиса его ортогонального дополнения. (Методы построения этих матриц обсуждались в разд. 5.13.) Вычисляя p_k в виде

$$p_k = Y_k p_Y + Z_k p_Z, \tag{6.42}$$

p_Y находят из ограничений (6.41), так как подстановка в них правой части (6.42) дает

$$\bar{A}_k p_k = \bar{A}_k Y_k p_Y = -\bar{c}_k. \tag{6.43}$$

Если задача (6.41) разрешима, уравнения (6.43) будут совместными. Обозначим через r ранг $\bar{A}_k$; тогда произведение $\bar{A}_k Y_k$ должно включать невырожденную $r \times r$-подматрицу и p_Y однозначно определится любыми r линейно независимыми уравнениями из (6.43). Что же касается вектора p_Z из второго слагаемого в правой части (6.42), то он должен доставлять безусловный минимум целевой функции (6.41) на соответствующем подпространстве и является решением ньютоновской системы уравнений

$$Z_k^{\mathrm{T}} H_k Z_k p_Z = -Z_k^{\mathrm{T}} (g_k + H_k Y_k p_Y). \tag{6.44}$$

Вектор множителей Лагранжа η_k в задаче (6.41) определяется заведомо совместной системой вида

$$H_k p_k + g_k = \hat{A}_k^T \eta_k. \tag{6.45}$$

Формулы (6.42), (6.43) и (6.44) корректны при любой невырожденной матрице $Z_k^T H_k Z_k$, но дают оптимальный для (6.41) вектор только в том случае, если $Z_k^T H_k Z_k$ положительно определена. Когда у нее имеются отрицательные собственные значения, конечного минимума в задаче (6.41) просто не существует. Важно отметить, что при описанном способе построения p_k определенность $Z_k^T H_k Z_k$, т. е. содержательность подзадачи (6.41), можно выяснить по ходу вычислений.

Для оптимального вектора в задаче (6.41) существует много математически эквивалентных выражений. Например, p_k можно рассматривать как составляющую решения совместной системы линейных уравнений относительно p_k и η_k

$$\begin{pmatrix} H_k & -\hat{A}_k^T \\ \hat{A}_k & 0 \end{pmatrix} \begin{pmatrix} p_k \\ \eta_k \end{pmatrix} = \begin{pmatrix} -g_k \\ -c_k \end{pmatrix}, \tag{6.46}$$

которая есть не что иное, как условия оптимальности p_k первого порядка. Заметим, что уравнения (6.46) аналогичны используемым в методах ранг-пространства для оптимизации при линейных ограничениях (см. (5.54) в разд. 5.4).

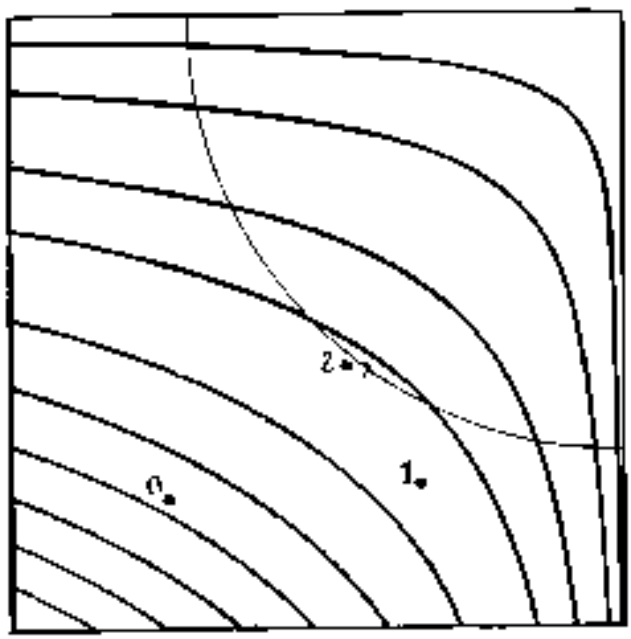

Рис. 6.5. Три первые точки, найденные по примеру 6.2 методом спроектированного лагранжиана с квадратичной подзадачей; каждая является точкой минимума квадратичной аппроксимации функций Лагранжа при линейном ограничении-равенстве.

Свойства локальной сходимости. Если в качестве H_k брать матрицу W_k — аппроксимацию матрицы Гессе функции Лагранжа в x^*, λ^*, построенную по аналитическим значениям производных, т. е.

$$H_k = W_k \equiv G(x_k) - \sum_{i=1}^{t} (\lambda_k)_i \ddot{G}_i(x_k), \tag{6.47}$$

то нетрудно доказать, что алгоритм SQ будет локально сходящимся. В частности, при таком выборе H_k из условий (i) (x_0, λ_0) мало отличается от (x^*, λ^*), (ii) в x^* выполнены достаточные условия оптимальности из разд. 6.4 и (iii) λ_{k+1} берется равным η_k из (6.45) следует квадратичная сходимость генерируемой алго-

ритмом SQ последовательности к (x^*, λ^*). Как и в случае с алгоритмом SP из разд. 6.5.2.2, первый порядок точности оценок множителей не снижает скорости сходимости до линейной; если $\|x_k - x^*\|$ и $\|\lambda_k - \lambda^*\|$ — величины порядка ε, то аналогичные нормы с x_{k+1} и λ_{k+1} будут пропорциональны ε^2.

На рис. 6п изображены три первых приближения, полученные алгоритмом SQ с H_k из (6.47) в задаче примера 6.2. Вычисления были начаты с $x_0 = (-1.5, -1.6)^T$ и $\lambda_0 = 0$. Заметим, что точки очень близки к показанным на рис. 6l, иллюстрирующем работу алгоритма SP, но теперь никаких «внутренних» итераций не подразумевается. Правда, на каждом шаге требуется вычисление матрицы Гессе всех функций задачи. Числовые значения норм $\{\|x_k - x^*\|_2\}$, $k = 1, \ldots, 4$, таковы: 2.08×10^{-1}, 1.41×10^{-2}, 8.18×10^{-5}, 4.95×10^{-10}.

Квадратичную сходимость алгоритма SQ при сформулированных предположениях можно пояснять, если интерпретировать p_k и η_k, λ_k из решения подзадачи (6.41) как шаг по x и λ метода Ньютона для поиска решения системы нелинейных уравнений

$$g(x^*) - \hat{A}(x^*)^T \lambda^* = 0 \tag{6.48a}$$

и

$$c(x^*) = 0, \tag{6.48b}$$

представляющей собой условия оптимальности x^* в исходной задаче. В самом деле, чтобы получить для (6.48) формулу расчета ньютоновского шага, надо линеаризовать функции (6.48) в окрестности (x_k, λ_k), продифференцировав их, и это даст систему (6.46) с $H_k = -W_k$.

К сожалению, подобно упрощенному алгоритму SP в разд. 6.5.2.2, алгоритм SQ хорош только при наличии хорошего начального приближения, и если запустить его с пары (x_0, λ_0), значительно отличающейся от (x^*, λ^*), он скорее всего не сойдется. Здесь снова уместно сослаться на его интерпретацию как результата применения классической ньютоновской схемы к уравнениям (6.48) — ведь известно, что последняя ненадежна даже в одномерном случае (см. разд. 4.1 и 4.4). Свойства ее локальной сходимости идеальны, но в отношении глобальной ничего хорошего сказать нельзя.

Ниже рассматриваются модификации алгоритма SQ, позволяющие повысить его надежность.

6.5.3.3. Использование функции выигрыша. Приступая к выводу алгоритма SQ, мы подчеркнули, что высказываемые соображения ориентированы на малую окрестность оптимума. Если же точка x_k удалена от x^* (или вектор λ_k существенно отличен от λ^*), целесообразность шага, определяемого подзадачей (6.41), представляется сомнительной. Область применимости этой подзадачи существенно расширится, если ее решение считать не шагом, а направлением поиска, шаг вдоль которого еще предстоит выбрать. Тогда

очередным приближением будет

$$x_{k+1}=x_k+\alpha_k p_k, \tag{6.49}$$

где p_k — оптимальный вектор из (6.41), а α_k — длина шага вдоль p_k. Значение α_k подбирают так, чтобы обеспечить существенное убывание» (см. разд. 4.3.2.1) некоторой функции выигрыша, являющейся средством оценивания качества приближений. Впредь функции выигрыша будем обозначать через Φ_M.

В роли Φ_M предлагалось использовать разные функции и в том числе квадратичную штрафную (разд. 6.2.1.1), абсолютную штрафную (разд. 6.2.2) и модифицированную функцию Лагранжа (разд. 6.4.1). Свойства, которые требуются от Φ_M, состоят в следующем. Во-первых, Φ_M должна быть разумной мерой близости к решению. Во-вторых, всегда должна существовать возможность уменьшить Φ_M положительным шагом вдоль p_k. Это означает определенную привязку Φ_M к порождающей p_k подзадаче, причем следует отметить, что ради повышения надежности метода бывает целесообразно использовать подзадачи, отличные от (6.41). В-третьих, подсчет значения Φ_M не должен быть чрезмерно трудоемкой процедурой. И наконец, желательно, чтобы условие убывания Φ_M не ограничивало скорости сходимости метода. К примеру, если в качестве H_k берется W_k, квадратичная сходимость при использовании Φ_M возможна лишь тогда, когда это условие допускает последовательность шагов $\{\alpha_k\}$, достаточно быстро стремящуюся к единице.

* **6.5.3.4. Другие постановки квадратичной подзадачи.** Поскольку формулировка задачи (6.41) связана с условиями оптимальности, т. е. с соотношениями, которые выполнены только в x^*, ожидать, что (6.41) будет иметь большой смысл в далекой от x^* точке x_k, не приходится. Ниже кратко рассмотрены постановки подзадачи, которые вдали от x^* могут существенно отличаться от (6.41).

Подзадача, ориентированная на свойства квадратичной штрафной функции. Допустим, что x_k лежит на траектории $\{x^*(\rho)\}$, порождаемой методом квадратичных штрафных функций (см. разд. 6.2.1.1). Тогда при необременительных предположениях устанавливается, что шаг p из x_k в другую точку $x^*(\rho)$ этой траектории, отвечающую большему значению ρ, приблизительно совпадает с решением задачи

$$\begin{aligned} &\text{найти } \min_{p\in\mathbb{R}^n} g_k^T p+\frac{1}{2}\,p^T H_k p \\ &\text{при ограничениях } \hat{A}_k p=-\hat{c}_k-\begin{pmatrix}0\\ \rho^{-1}\end{pmatrix}\lambda_k. \end{aligned} \tag{6.50}$$

Здесь H_k — аппроксимация матрицы Гессе функции Лагранжа, а λ_k — текущая оценка λ^*. Таким образом, можно предложить алгоритм, в котором p_k определяется из (6.50), а в качестве Φ_M используется квадратичная штрафная функция. Заметим, что при

бесконечном ρ подзадача (6.50) переходит в (6.41). Следует также подчеркнуть, что никакой плохой обусловленности при больших ρ здесь (в отличие от методов разд. 6.2.1.1) не возникает. В подзадаче (6.50) значение штрафного параметра назначают в соответствии с оценкой расстояния до x^* и неограниченно увеличивают по мере приближения к оптимуму.

Подзадача, ориентированная на свойства логарифмической барьерной функции. Когда ограничения исходной задачи являются неравенствами и важно выдержать допустимость всех приближений, квадратичные подзадачи для расчета p_k строят, опираясь на свойства траектории точек $x^*(r)$, из метода барьерных функций (см. разд. 6.2.1.2). Если x_k принадлежит этой траектории и выполнены некие естественные условия, шаг p в точку $x^*(r)$, отвечающую меньшему значению r, будет аппроксимироваться решением задачи

$$\begin{aligned} &\text{найти} \quad \min_{p \in \mathbb{R}^n} g_k^T p + \frac{1}{2} p^T H_k p \\ &\text{при ограничениях} \quad \hat{A}_k p = -\hat{c}_k + \delta_k, \end{aligned} \tag{6.51}$$

где H_k — приближение матрицы Гессе функции Лагранжа, а вектор δ_k вычисляется по формуле $(\delta_k)_i = r/(\lambda_k)_i$, в которой λ_k — текущая оценка λ^*. Следовательно, определяя p_k из (6.51) и используя в качестве Φ_{r_k} при выборе α_k в (6.49) логарифмическую барьерную функцию, можно получить алгоритм, который при допустимом начальном приближении x_0 будет генерировать только допустимые точки x_k. Значение r для (6.51) должно отражать степень близости x_k к x^*. Когда параметр r равен нулю, подзадача (6.51) совпадает с (6.41). Отметим, что никакой плохой обусловленности при малых r не возникает.

Интерпретация предлагаемых подзадач. Хорошую формулировку ограничений подзадачи метода последовательного квадратичного программирования отличают два свойства. Во-первых, по мере приближения x_k к x^* определяемые ею ограничения достаточно быстро стремятся к (6.40), и это обеспечивает методу наилучшие характеристики локальной сходимости. Во-вторых, она является достаточно гибкой, т. е. пригодна для различных ситуаций, которые могут возникать по ходу решения. В подзадачах (6.50), (6.51) подобная гибкость достигается введением в правые части линеаризованных ограничений специфических сдвигов. Общие соображения в пользу таких сдвигов приведены ниже.

Траектории методов квадратичных штрафных и логарифмических барьерных функций *не касательны* к допустимому множеству в x^*. При этом, если x_k — точка, лежащая на какой-то из них, то единичный шаг вдоль вектора p_k, полученного решением (6.50) или (6.51), будет короче, чем требуется для обнуления линеаризованных функций ограничений. Учитывая оба указанных обстоятельства,

можно говорить о том, что введение сдвигов повышает надежность линейной аппроксимации ограничений, на основе которой выбирается p_k.

Для иных точек x_k целесообразность сдвигов обосновывается по-другому. Обозначим через $\bar{c}_i$ функцию ограничения, которое на основе некого разумного прогноза (см. разд. 6.5.5) предполагается активным в искомом решении, и пусть в текущей точке x_k значение $\bar{c}_i$ очень близко к нулю, хотя ясно, что до x^* еще далеко. Тогда делать шаги с сохранением почти нулевого $\bar{c}_i$ скорее всего неэффективно: это было бы «отслеживанием нелинейной границы линейными средствами», что чревато существенным замедлением сходимости (см. замечания к разд. 6.3). Именно так работал бы метод с (6.41), в то время как подзадачи (6.50) и (6.51) в этой ситуации дадут направления, уводящие от границы.

*6.5.4. Стратегии для дефектных подзадач

Ни одна из упомянутых выше формулировок подзадачи для метода спроектированного лагранжиана не исключает возможности, что эта подзадача окажется дефектной. На такие случаи надо иметь альтернативные формулировки. Краткий обзор их приводится ниже; более подробные сведения читатель найдет по ссылкам, данным в замечаниях.

6.5.4.1. Несовместные линейные ограничения. Описывая в гл. 5 методы минимизации при линейных ограничениях, мы не обсуждали проблемы несовместности условий подзадач для выбора направлений поиска, поскольку такая несовместность означала бы отсутствие допустимых точек из исходной задачи, и поэтому обсуждать здесь просто нечего. По-иному дело обстоит в случаях с нелинейными ограничениями: допустимое множество подзадачи типа (6.37) или (6.38) может оказаться пустым и тогда, когда нелинейные ограничения совместны. Это предполагает линейную зависимость строк матрицы $\bar{A}_k$ (т. е. наличие у нее дефекта ранга или превышение числа ее строк над числом столбцов) и на практике, к сожалению, не является редкостью.

Если ограничения $\bar{A}_k p_k = \bar{d}_k$ подзадачи метода спроектированного лагранжиана оказались несовместными, в качестве p_k можно взять, например, вектор, минимизирующий норму $\|\bar{A}_k p - \bar{d}_k\|_2$. Он будет удовлетворять набору возмущенных равенств

$$\bar{A}_k p_k = \bar{d}_k + \hat{e}_k$$

с «минимальной» поправкой $\hat{e}_k$. Таков один из способов разрешения проблемы несовместности модификаций ограничений подзадачи. Есть и другие. В частности, можно переопределять вектор $\bar{c}_k$, непосредственно контролируя линейную независимость

градиентов включаемых в него функций. Если исходная задача имеет хорошо обусловленное решение, подобные модификации потребуются только вдали от x^* и потому не повлияют на асимптотическую скорость сходимости метода.

Неразрешимость подзадачи с матрицей $\hat{A}_k$, имеющей линейно зависимые строки, служит признаком того, что при плохо обусловленной $\hat{A}_k$ возможны осложнения. В действительности их два: становится ненадежным вычисляемое значение составляющей направления поиска из ранг-пространства $\hat{A}_k$, т. е. вектора p_Y в (6.43); теряют точность оценки множителей Лагранжа (разд. 6.6).

*** 6.5.4.2. Плохая аппроксимация функции Лагранжа.** Вторая вероятная форма дефектности подзадачи выбора p_k — это неограниченность ее решения.

При реализации метода, подобного рассмотренному в разд. 6.5.2, угроз отсутствия у подзадачи (6.37) (или подобной ей) конечного оптимума можно учесть лишь специальными пунктами в правилах останова применяемого к ней алгоритма. В частности, следует предусмотреть его прерывание после выполнения некоторого числа итераций. Больше здесь ничего сделать нельзя, так как не существует конечной процедуры, которая в общем случае позволила бы установить, ограничено решение (6.37) или нет.

С представленными выше методами последовательного квадратичного программирования картина ясней. Квадратичная подзадача будет иметь бесконечное решение тогда, когда матрица $Z_k^T H_L Z_k$ не является знакоопределенной. Тут ситуация аналогична возникающей в методах ньютоновского типа для поиска минимума без ограничений и с линейными ограничениями. Аналогичен и набор средств, пригодных для модификации подзадачи. Например, p_t можно строить по генерируемой на основе $Z_k^T H_k Z_k$ положительно определенной матрице (разд. 4.4.2). Правда, как и в ньютоновских методах, полученный в результате вектор не будет отмасштабированным, причем дело осложняется тем, что его образует не только связанный с $Z_k^T H_L Z_k$ вектор p_Z, но и p_Y, который от этой матрицы не зависит и полностью определяется линеаризованными ограничениями. Поэтому есть опасность, что плохо отмасштабированный p_Z «подавит» вполне разумный p_Y (или наоборот). Проблема несоразмерности масштабов p_Z и p_Y возникает при любом способе модификации подзадачи.

*6.5.5. ПОСТРОЕНИЕ АКТИВНОГО НАБОРА

В этом разделе речь пойдет о формировании подзадач выбора p_k в методах спроектированного лагранжиана для отыскания минимума при ограничениях-неравенствах. Будем считать, что полная

совокупность исходных ограничений представлена записью $c(x) \geqslant 0$, где $c(x)$ — векторная функция. Через $A(x)$ ниже обозначается матрица, чья i-я строка есть градиент от $c_i(x)$.

Мы рассмотрим два «крайних» подхода к формированию подзадач; разумеется, можно предложить и массу промежуточных. Первый состоит в том, чтобы ставить подзадачи, все ограничения которых имеют вид линейных равенств и каждый раз отражают какую-то часть исходных ограничений. Предыдущий материал разд. 6.5 относится именно к этому подходу. С другой стороны, можно ставить подзадачи с линейными неравенствами, всегда учитывая все исходные ограничения.

***6.5.5.1. Подзадачи с ограничениями-равенствами.** Построение подзадачи с равенствами для расчета очередного направления поиска начинается с определения того, какие из исходных неравенств включить в рабочий список (т. е. трактовать как равенства) на данной итерации. Лучше всего было бы, если бы последний совпал со списком ограничений, активных в искомом решении. Поэтому выбор рабочего списка должен расцениваться как попытка угадать эти ограничения. Соответственно правила, регламентирующие формирование составов ограничений подзадач, принято называть *стратегиями прогнозирования активного набора.*

В методах минимизации при линейных ограничениях, изложенных в гл. 5, основным признаком, по которому ограничения отбираются в рабочий список, служит обращение их в равенства. Для методов спроектированного лагранжиана этот признак не годится: в них нелинейные ограничения могут стать равенствами лишь в пределе. Поэтому здесь при составлении рабочего списка приходится пользоваться более сложными критериями. Типичная стратегия прогнозирования активного набора исходит из стремления добиться, чтобы выделяемые ограничения удовлетворяли в текущей точке x_k тем требованиям, которым активные в x^* ограничения должны удовлетворять в окрестности x^*. Можно учитывать также свойства функции выпрыша Φ_ρ; например, если в качестве Φ_ρ используется квадратичная штрафная функция (6.1), то активные в x^* ограничения разумно будет искать среди нарушенных (см. разд. 6.2.1.1). Иногда для очередного прогноза активного набора удается использовать множители Лагранжа из предыдущей подзадачи. Наконец, применяют «двухступенчатые» правила, когда сначала выбирается некий пробный список, а затем по множителям Лагранжа соответствующей подзадачи определяется окончательный. Здесь происходит нечто подобное «выводу ограничений из рабочего списка» в методах минимизации при линейных неравенствах.

Главным доводом в пользу постановки подзадач с линейными равенствами является их относительная простота. При этом, однако, важно обеспечить, чтобы неверный прогноз активного набора не мог послужить причиной отказа алгоритма. В частности, в функции

выигрыша (см. разд. 6.5.3.3) всегда должны быть отражены все ограничения, а не только те, которые попали в рабочий список.

***6.5.5.2. Подзадачи с ограничениями-неравенствами.** На основе любой из упомянутых выше в разд. 6.5 постановок естественно определяется соответствующая подзадача расчета p_k с ограничениями вида

$$A_k p \geqslant d_k, \tag{6.52}$$

где через A_k обозначена матрица $A(x_k)$. В данном случае можно говорить о *прогнозировании списка активных в x^* ограничений активным набором решения подзадачи*.

При некоторых предположениях доказано, что подзадача метода спроектированного лагранжиана, построенная в близкой к x^* точке x_k и содержащая в качестве ограничений систему неравенств (6.52) с $d_k = -c_k$, будет иметь тот же набор активных в решении ограничений, что и исходная задача. Это позволяет распространить на методы, использующие условия (6.52), сформулированные ранее теоремы сходимости, т. е. обосновать их. Однако любая разумная стратегия прогнозирования активного набора в мало отличающейся от x^* точке x_k тоже даст правильный результат. Поэтому сам по себе указанный факт не является основанием, чтобы предпочитать методы с условиями (6.52). Надо еще выяснить, могут ли последние правильно определять активный набор там, где иные стратегии его прогнозирования ошибаются (см. комментарии в разд. 6.6.3).

Рассмотренная в разд. 6.5.4 проблема дефектности подзадач с равенствами сохраняется и в подзадачах с неравенствами, причем пути ее разрешения усложняются. В частности, для определения совместности ограничений (6.52) потребуется итерационная процедура, и не ясно, как лучше всего модифицировать их, когда они несовместны.

Замечания и избранная библиография к разд. 6.5

Идею линеаризации ограничений эксплуатируют многие алгоритмы поиска условного оптимума (в том числе и описанные в разд. 6.3 методы типа приведенных градиентов). Одно из ее первых воплощений — метод Гриффита и Стюарта (1961), в котором на каждом шаге решается подзадача LP, формулируемая на основе линейной аппроксимации исходных ограничений и целевой функции. Подобные методы называют *методами последовательного линейного программирования*. Их общий дефект состоит в том, что решение подзадачи LP всегда является вершиной линеаризованного допустимого множества, в то время как искомая точка x^* аналогичной особенностью скорее всего обладать не будет. Чтобы «сгладить» это несоответствие, в подзадачу приходится вводить дополнительные условия (например, двусторонние ограничения на переменные).

По принципу последовательного решения подзадач с линейными ограничениями, аппроксимирующими исходные, работают также методы отсекающих плоскостей (см., например, Келли (1960), Тонкис и Вайнот (1967)), относящиеся к арсеналу выпуклого программирования (очень коротко о задачах выпуклого программирования будет сказано в разд. 6.8.2.1). Надо заметить, что в настоящее время их практически не используют: они медленно сходятся и к тому же страдают численной неустойчивостью из-за ухудшения от итерации к итерации обусловленности линейных ограничений подзадачи.

Методы, основанные на подзадачах типа (6.33) с целевыми функциями общего вида, предлагались Розеном и Кройзером (1972) и Робинсоном (1972). Доказательства их квадратичной сходимости (при определенных условиях) приведены в статьях Робинсона (1972, 1974). Розен (1978) предложил также двухфазную схему, в которой сначала методом квадратичных штрафных функций отыскивается разумное приближение искомой точки x^*, а затем происходит переключение на метод спроектированного лагранжиана с подзадачей (6.37). В работе Беста и др. (1981) описана схема такого же типа, но допускающая возврат к первой фазе, если вторая не дает ожидаемой скорости сходимости. Муртаф и Сондерс (1980) разработали алгоритм, в котором целевая функция подзадачи подобна модифицированной функции Лагранжа (6.24); о нем еще будет говориться в разд. 6.7.

Методы спроектированного лагранжиана с последовательной минимизацией при ограничениях-равенствах, выбираемых на основе стратегии прогнозирования активного набора, читатель найдет в статьях Ван-дер-Хука (1979), Беста и др. (1981).

Насколько нам известно, идея применения квадратичной подзадачи для поиска минимума при нелинейных ограничениях впервые была высказана Уилсоном (1963) в его неопубликованной диссертации. Впоследствии метод Уилсона был описан и проинтерпретирован Билом (1967b). В этом методе на каждой итерации решается подзадача с линейными ограничениями-неравенствами вида

$$A_k p \geqslant -c_k. \tag{6.53}$$

Ее квадратичная целевая функция представляет собой аппроксимацию функции Лагранжа и строится по точным матрицам Гессе для F и $\{c_i\}$, причем множители Лагранжа *из предыдущей подзадачи* используются в качестве оценок компонент λ^* для последующей. Очередное приближение определяется как сумма $x_k + p_k$, где p_k есть решение подзадачи k-й итерации. Таким образом, одномерный поиск в методе Уилсона отсутствует.

Подобная (6.50) подзадача с ограничениями-неравенствами, ориентированная на предельные свойства траектории точек минимумов квадратичных штрафных функций, была построена Мюрреем (1969a, b). Он предложил для нее несколько вариантов определения

H_k, включая квазиньютоновскую аппроксимацию, и несколько способов вычисления оценок множителей. Выдвигалась также идея «частично» решать подзадачу, т. е. выполнить несколько (возможно, всего лишь одну) итераций процедуры ее решения; при этом линеаризованные неравенства с отрицательными оценками множителей должны «исключаться» из рабочего списка. Решение подзадачи в методе Мюррея используется в качестве *направления поиска*. Шаг вдоль него выбирается из соображений минимизации квадратичной штрафной функции (которая служит «функцией выигрыша»).

Бигс (1972, 1974, 1975) предложил модификацию метода Мюррея, основанную на подзадачах вида (6.50) с ограничениями-равенствами. Он тоже описал некоторые специальные способы оценивания множителей.

Среди методов спроектированного лагранжиана с квадратичной подзадачей есть немало использующих квазиньютоновскую технику. Так, например, Гарсиа Паломарес и Мангасарьян (1976) вывели метод такого сорта, применив эту технику для решения системы нелинейных уравнений (6.48). Еще один пример - метод Хана (1976, 1977a, b). Возвращаясь к идее расчета направления поиска решением квадратичной подзадачи с неравенствами типа (6.53), он предложил матрицу ее целевой функции строить как квазиньютоновское приближение матрицы Гессе функции Лагранжа и дал соответствующие формулы пересчета. При этом делалось предположение, что сама матрица Гессе функции Лагранжа будет всюду положительно определенной. В методе Хана множители Лагранжа прошлой подзадачи служат оценками λ^* для последующей. При некоторых условиях доказана его сверхлинейная сходимость. В качестве функции выигрыша в процедуре одномерного поиска Хан использовал негладкую «точную» штрафную функцию P_A из (6.8).

Процедура с последовательным решением квадратичных подзадач с неравенствами, аналогичная методу Хана, но обеспечивающая сохранение положительной определенности квазиньютоновской аппроксимации матрицы Гессе функции Лагранжа и тогда, когда эта матрица не является знакоопределенной, описана Пауэллом (1977b, 1978). При некоторых предположениях она тоже сходится сверхлинейно.

Как лучше всего использовать квазиньютоновскую технику в методах спроектированного лагранжиана с квадратичными подзадачами, не ясно. В сравнении со случаем, когда исходные ограничения линейны (см. разд. 5.1.2.4), здесь есть по крайней мере два осложнения. Во-первых, L_{H_k} вообще говоря, полностью преобразуется на каждой итерации, *даже если рабочий список неизменен*. Во-вторых, при вариациях рабочего списка функция, которую надо аппроксимировать, меняется скачком. Однако, несмотря на все это, разработано немало вариантов применения квазиньютоновских формул пересчета для аппроксимации матрицы Гессе (и спроектированной матрицы Гессе) функции Лагранжа, причем практика подтвердила

их работоспособность. О них можно прочесть, например, в работах Пауэлла (1977b), Мюррея и Райт (1978) и Шиттковского (1980).

Некоторые вопросы, связанные с выбором в качестве функции выигрыша абсолютной штрафной функции P_1, обсуждаются в работе Чемберлена и др. (1980). Определенные трудности, к которым может привести такой выбор, описаны Маратосом (1978) и Чемберленом (1979). Сопоставление достоинств разных формулировок квадратичной подзадачи для метода спроектированного лагранжиана проведено Мюрреем и Райт (1980).

Метод допустимой точки с подзадачей (6.51) изложен в диссертации Райт (1976) и работе Мюррея и Райт (1978). Реализация процедур, конструируемых на основе (6.50), (6.51) для задач со смесью линейных и нелинейных ограничений, обсуждается в опубликованном докладе Гилла и др. (1980).

Как уже было сказано в замечаниях к разд. 6.2, многие методы минимизации недифференцируемой штрафной функции организованы подобно методам спроектированного лагранжиана с квадратичной подзадачей, в направление поиска определяется в них в виде (6.42). В частности, законы методы Коллмайя (1979) и Коллмана и Конна (1980b, c).

Идея расщепления направления поиска на две ортогональные составляющие, как в (6.42), эксплуатируется многими алгоритмами минимизации при нелинейных ограничениях, например методами Барда и Гринштадта (1969), Лейбергеря (1974), Мэйна и Полака (1976) и Хита (1978). Она реализуется и в методах типа приведенных градиентов, обсуждавшихся в разд. 6.3.

Подход к решению задач с нелинейными ограничениями, основанный на применении метода Ньютона к последовательности систем нелинейных уравнений (см. (6.48)), обсуждается в статьях Таппа (1974a, b).

В качестве обзорных работ по методам спроектированного лагранжиана можно порекомендовать статьи Флетчера (1974, 1977) и Мюррея (1976).

6.6. ОЦЕНКИ МНОЖИТЕЛЕЙ ЛАГРАНЖА

Эффективность методов из разд. 6.3, 6.4 и 6.5 в значительной степени определяется качеством фигурирующих в них оценок множителей Лагранжа. Напомним, что в задачах с линейными ограничениями оценки множителей, как правило, используются, только если среди этих ограничений есть неравенства, и привлекаются, когда нужно выяснить, стоит ли сокращать рабочий список и каким образом это можно сделать; целевая функция подзадачи их не включает, и на асимптотической скорости сходимости поиска они не сказываются. Значительно большая роль отводится оценкам множителей в методах минимизации при нелинейных ограничениях с применением функций Лагранжа (типа рассмотренных в разд. 6.3,

6.4 и 6.5). Здесь они нужны, даже если все ограничения являются равенствами, входят в определение целевой функции подзадачи и, как уже было отмечено, сильно влияют на скорость сходимости.

Вектор множителей Лагранжа λ^* образуется коэффициентами разложения $g(x^*)$ по градиентам вектор-функций $\hat{c}$ активных в x^* ограничений:

$$g(x^*) = \hat{A}(x^*)^T \lambda^*. \tag{6.54}$$

Здесь $\hat{A}$ — составленная из этих градиентов матрица. По определению

$$\hat{c}(x^*) = 0, \tag{6.55}$$

и ясно, что x^* будет решением, а λ^* — отвечающим ему вектором множителей Лагранжа в задаче

$$\begin{aligned} &\text{найти } \min_{x \in \mathbb{R}^n} F(x) \\ &\text{при ограничениях } \hat{c}(x) = 0. \end{aligned} \tag{6.56}$$

(Она совпадает с исходной, если в той нет неравенств.) Поскольку при нелинейных F, $\hat{c}$ левая часть и матрица в (6.54) суть функции от x^* (что и отражено в обозначениях), не зная x^*, найти вектор λ^* нельзя. В произвольной точке x_k можно построить лишь его оценку λ_k.

В методах активного набора из гл. 5 оценки множителей Лагранжа требовалось вычислять в допустимых точках, и отвечали они ограничениям, выполненным как равенства. Теперь же точки x_k будут недопустимыми, и скорее всего никакие ограничения в равенства в x_k не обратятся.

6.6.1. ОЦЕНКИ ПЕРВОГО ПОРЯДКА

Здесь и в разд. 6.6.2 через $\hat{c}_k$ и $\hat{A}_k$ будем обозначать вычисленный в x_k вектор функций ограничений, которые предполагаются активными в x^*, и матрицу их градиентов; имеется в виду, что при x_k, близких к x^*, прогноз активных в x^* ограничений верен.

Исходя из равенства (6.54) в качестве оценки в x_k для λ^* можно предложить решение λ_L задачи о наименьших квадратах вида

$$\text{найти } \min_{\lambda} \| \hat{A}_k^T \lambda - g_k \|_2. \tag{6.57}$$

(Она аналогична (5.27), но сейчас $\hat{A}_k$ зависит от x_k.) Такая оценка всегда определена и состоятельна в смысле (5.26); правда, решение (6.57) будет единственным, только если строки $\hat{A}_k$ линейно независимы. (Численно устойчивые методы расчета λ_L описаны в разд. 2.2.5.3.)

Когда переопределенная система

$$\bar{A}_k^{\mathrm{T}}\lambda \approx g_k \tag{6.58}$$

совместна, минимум в (6.57) равен нулю. Однако в отсутствие равенства $\bar{c}(x_k)=0$ вектор λ_l и в этом случае останется лишь *оценкой* для λ^*. Мы подчеркиваем это, поскольку *совместность системы* (6.58) *автоматически обеспечивается некоторыми алгоритмами*. В частности, она гарантирована, если x_k выбирается как точка минимума модифицированной функции Лагранжа (6.24); тогда λ_l совпадает с λ_{k+1} из (6.27). (Оценкой первого порядка, отражающей отличие c_k от нуля, может послужить $\lambda_l - (\bar{A}_k\bar{A}_k^{\mathrm{T}})^{-1}\bar{c}_k$.)

Как и для задач с линейными ограничениями (разд. 5.1.5), наряду с λ_l оценкой λ^* первого порядка будет решение λ_l системы уравнений

$$V^{\mathrm{T}}\lambda_l = g_V, \tag{6.59}$$

где V есть невырожденная $t\times t$-подматрица $\bar{A}_k$, а g_V — вектор, набранный из соответствующих компонент g_k. Если система (6.58) совместна, λ_l и λ_l совпадают.

В предположении полноты ранга матрицы $\bar{A}(x^*)$ нетрудно доказать, что для близких к x^* точек x_k нормы $\|\lambda_l - \lambda^*\|$ и $\|\lambda_l - \lambda^*\|$ соразмерны с $\|x_k - x^*\|$. Отсюда и термин — *оценки первого порядка*. При этом уклонения λ_l и λ_l от λ^*, грубо говоря, пропорциональны числам обусловленности $\bar{A}_k$ и V соответственно. Кроме того, ошибки λ_l и λ_l зависят от кривизны функций задачи. Пример, иллюстрирующий характер представленных оценок, читатель найдет в разд. 5.1.5.

6.6.2. ОЦЕНКИ ВТОРОГО ПОРЯДКА

Как и в случае с линейными ограничениями (см. разд. 5.1.5.2), при определенных условиях λ^* удается оценивать с ошибкой, пропорциональной квадрату расстояния от x_k до x^*. Обозначим (неизвестный) шаг из x_k в x^* через q. Тогда (6.54) можно переписать так:

$$g(x_k+q) = \bar{A}(x_k+q)^{\mathrm{T}}\lambda^*.$$

Разлагая левую и правую части этого равенства по Тейлору, получим

$$g_k + W_k(\lambda^*)\,q = \bar{A}_k^{\mathrm{T}}\lambda^* + O(\|q\|^2), \tag{6.60}$$

где $W_k(\lambda)$ — матрица Гессе функции Лагранжа с множителем λ, т. е.

$$W_k(\lambda) = G(x_k) - \sum_{i=1}^{t}\lambda_i\bar{G}_i(x_k).$$

Непосредственно оценить λ^* по равенству (6.60) нельзя, поскольку второе слагаемое в его левой части содержит неизвестный вектор q, который к тому же умножается на матрицу, вычисляемую по λ^*. Однако, имея приближение p для q и какую-то предварительную оценку λ для λ^*, из (6.60) нетрудно вывести соотношение, которое подскажет формулу уточнения λ:

$$g_k + W_k(\lambda)\,p = \bar{A}_k^T\lambda^* + O(\|q\|^2 + \|\lambda - \lambda^*\|\,\|q\| + \|p - q\|). \quad (6.61)$$

На основании (6.61) в качестве оценки η_L для λ^* естественно предложить решение задачи

$$\text{найти } \min_{\eta} \|\bar{A}_k^T\eta - (g_k + W_k(\lambda)p)\|_2^2. \quad (6.62)$$

Если $\bar{A}_k$ имеет полный ранг, то можно добиться, чтобы при $\|q\| = O(\varepsilon)$ норма $\|\eta_L - \lambda^*\|$ была пропорциональна ε^2, т. е. η_L была оценкой второго порядка. Для этого достаточно обеспечить соотношения $\|\lambda - \lambda^*\| = O(\varepsilon)$, $\|p - q\| = O(\varepsilon^2)$, т. е. подбирать λ и p как оценки по меньшей мере первого порядка для λ^* и второго для q соответственно.

В разд. 5.1.5.2 представлен пример задачи с линейными ограничениями, в которой аналогичная η_L из (6.62) оценка оказалась крайне ненадежной; разумеется, такое случается и при нелинейных ограничениях.

Когда в качестве p используется решение квадратичной подзадачи вида

$$\begin{aligned} &\text{найти } \min_{p \in \mathbb{R}^n} g_k^T p + \frac{1}{2} p^T W_k(\lambda)\, p \\ &\text{при ограничениях } \bar{A}_k p = \bar{d}_k, \end{aligned} \quad (6.63)$$

то независимо от выбора $\bar{d}_k$ минимум в (6.62) будет нулем, причем η_L из (6.62) оказывается *точным вектором* множителей Лагранжа задачи (6.63). Если к тому же $\bar{d}_k = -c_k$, при определенных условиях (как уже было отмечено в разд. 6.5.3.2) можно утверждать, что $\|p - q\| = O(\varepsilon^2)$ и, следовательно, η_L является *оценкой второго порядка*. Этим обусловлена квадратичная сходимость q к нулю в алгоритме SQ, а из нее вытекает квадратичная сходимость η_L к λ^*. Последняя хорошо иллюстрируется набором значений $|\eta_L - \lambda^*|$, полученным применением алгоритма SQ к задаче из примера 6.2 (см. рис. 6m): первые четыре значения в этом наборе равны 2.22×10^{-1}, 2.12×10^{-2}, 2.23×10^{-4} и 1.12×10^{-8}.

Оценки λ^* второго порядка порождаются и методами спроектированного лагранжиана, рассмотренными в разд. 6.5.1. В них таковыми служат векторы λ_k^* множителей Лагранжа подзадач типа (6.37). Как указано в разд. 6.5.2.2, при некоторых предположениях последовательность λ_k^* сходится к λ^* квадратично, и это есть следствие того, что для x_k, близких к x^*, уклонение

λ_k^* от λ^* пропорционально квадрату нормы $\|x_k - x^*\|$. Квадратичная сходимость λ_k^* к λ^* отчетливо проявляется на последовательности оценок, генерируемых алгоритмом SP в задаче из примера 6.2 (см. рис. 61): значения $|\lambda_k^* - \lambda^*|$, $k = 0, \ldots, 3$, равны 1.46×10^{-1}, 1.02×10^{-2}, 1.85×10^{-4} и 7.25×10^{-9}.

*6.6.3. ОЦЕНКИ МНОЖИТЕЛЕЙ ДЛЯ ОГРАНИЧЕНИЙ-НЕРАВЕНСТВ

Компоненты вектора λ^*, отвечающие тем ограничениям исходной задачи, которые являются неравенствами, неотрицательны (см. разд. 3.4.2). Более того, обычно они оказываются положительными, и тогда соответствующие компоненты λ_L и λ_V для достаточно близких к x^* точек x_k наверняка будут иметь правильные знаки. Однако даже в таких случаях вблизи от x^* аналогичных гарантий дать нельзя. В то же время для некоторых методов неотрицательность оценок множителей неравенств существенна. Значит, здесь нужно как-то изыскать способы оценивания.

Простейший способ построения неотрицательных оценок — вычислять вектор λ_L или λ_V по обычным формулам, а затем обнулять его отрицательные компоненты, относящиеся к неравенствам. Правильность знаков можно обеспечить также усложнением задачи (6.57), а точнее, заменой ее задачей вида

$$\begin{aligned} &\text{найти } \min_{\lambda} \|\hat{A}_k^T \lambda - g_k\|_2^2 \\ &\text{при ограничениях } \lambda_i \geqslant 0, \ i \in \mathscr{L}, \end{aligned} \tag{6.64}$$

где через $\mathscr{L}$ обозначен список индексов учитываемых в $\hat{c}_k$ неравенств; для решения (6.64) надо использовать метод типа описанного в разд. 5.3.3. Фактически переход на (6.64) приводит к «выводу» неравенств с отрицательными оценками множителей из рабочего списка.

В методах минимизации при ограничениях-неравенствах оценки для λ^* второго порядка (см. разд. 6.6.2) обычно определяются как множители Лагранжа из подзадач типа (6.37) или (6.63), условия которых соответствуют прогнозу активного набора для исходной задачи. Казалось бы, чтобы обеспечить правильность знаков оценок, логичнее использовать подзадачи с неравенствами. Однако, хотя при этом неотрицательность оценок гарантирована, их надежность еще остается под вопросом. Они будут точны только при соблюдении ряда требований, и, в частности, нужна близость используемого для построения целевой функции подзадачи вектора λ_k к λ^*. Но такая близость предполагает, что выявлен правильный список активных ограничений, а тогда подзадача с равенствами должна дать тот же результат. Если же верность прогноза активных ограничений сомнительна, полагаться на оценки множителей из подзадачи с неравенствами нельзя.

6.6.4. ПРОВЕРКИ СОСТОЯТЕЛЬНОСТИ

В методах минимизации при нелинейных ограничениях оценки множителей Лагранжа используются для формирования подзадачи выбора очередного приближения или направления поиска и существенно влияют на сходимость. Поэтому, если возможно, стоит предусматривать *проверки их состоятельности*. Такие проверки особенно полезны, когда применяются оценки второго порядка, и служат как для контроля точности оценок, так и для повышения надежности составленных по ним прогнозов активного набора.

Оценка λ^* второго порядка, построенная в точке x_k, обычно имеет смысл *приближения* оценки первого порядка в x_{k+1}. Вычислив последнюю независимым образом (в соответствии с каким-то прогнозом активного набора в x_{k+1}), мы получаем возможность выявить качество этого приближения и тем самым состоятельность обеих оценок (и та и другая в идеале должны совпадать с λ^*). Подобный контроль состоятельности особенно полезен при прогнозировании активного набора.

Чаще всего оценки λ^* второго порядка в x_k оказываются хуже, чем оценки первого порядка в x_{k+1}. Поэтому, если расчет последних не требует серьезных дополнительных затрат, именно их стоит использовать в роли параметров целевых функций подзадач. Эта рекомендация относится, например, к методам с построением направлений поиска на основе LQ-разложения $\bar{A}_k$: в них ценой ничтожных дополнительных усилий можно вычислить λ_{k+1}.

Важно подчеркнуть, что оценки первого порядка сходятся к λ^* с той же скоростью, что и x_k к x^*, причем, как уже было отмечено ранее, применение таких оценок в методах спроектированного лагранжиана скорости сходимости этих методов не снижает. В частности, при определенных условиях приближения $\{x_k\}$, полученные алгоритмом SQ (см. разд. 6.5.3.2), сходятся к x^* квадратично, и тогда последовательность вычисляемых в x_k оценок первого порядка *тоже будет квадратично сходиться*. Для иллюстрации сказанного приведем начальный отрезок последовательности уклонений от λ^* оценок $(\lambda_I)_k$, полученных при решении алгоритмом SQ задачи из примера 6.2. Здесь условия квадратичной сходимости $\{x_k\}$ выполнены и значения $\|(\lambda_I)_k - \lambda^*\|$, $k = 1, \ldots, 4$, таковы: 8.57×10^{-2}, 7.67×10^{-3}, 4.02×10^{-5} и 1.36×10^{-9} (ошибки оценок η_I для той же задачи и в тех же точках даны в разд. 6.6.2; обратите внимание на то, что $(\lambda_I)_k$ всякий раз оказывается точнее, чем $(\eta_I)_{k-1}$).

В заключение отметим, что разумные оценки множителей Лагранжа информативны и в далеких от решения точках. Так, например, компоненты вектора λ_I из (6.57), будучи коэффициентами приближенного представления g_k в виде линейной комбинации градиентов ограничений рабочего списка, позволяют судить о чувствительности целевой функции к вариациям в ограничениях.

Замечания и избранная библиография к разделу 6.6

Публикации, специально посвященные процедурам оценивания множителей Лагранжа, можно пересчитать по пальцам. Обычно же эти процедуры описываются в качестве блоков алгоритмов условной оптимизации. В частности, какой-то способ построения оценок множителей определен в каждой из упомянутых в замечаниях к разд. 6.3, 6.4 и 6.5 работ.

Различные оценки множителей Лагранжа, а также методы их вычисления и проверки состоятельности подробно рассмотрены в статье Гилла и Мюррея (1979b).

6.7. ЗАДАЧИ БОЛЬШОЙ РАЗМЕРНОСТИ

В данном разделе обсуждаются задачи вида

$$\text{найти} \min_{x \in R^n} F(x)$$

$$\text{при ограничениях } Ax = b,$$

$$c(x) \left\{ \begin{matrix} = \\ \geqslant \end{matrix} \right\} 0, \tag{6.65}$$

$$l \leqslant x \leqslant u.$$

Здесь A есть $m_1 \times n$-матрица, а $c(x)$ — вектор дважды непрерывно дифференцируемых функций $\{c_i(x)\}$, $i = 1, \ldots, m_2$. Будем считать, что переменных и ограничений в (6.65) «много» (т. е. обычные методы из-за дефицита машинных ресурсов неприменимы), что процент нелинейных ограничений мал и что матрица A разрежена.

Описываемые ниже методы опираются на стандартную технику линейного программирования, а она предполагает задание ограничений общего вида в форме *равенств* (см. разд. 5.6.1). Именно поэтому в постановке (6.65) все линейные ограничения, кроме простых, представлены равенствами.

В начале разд. 5.6.2 были изложены некие общие соображения относительно задач большой размерности с линейными ограничениями. Они справедливы и для (6.65).

*6.7.1. ИСПОЛЬЗОВАНИЕ ПОДЗАДАЧИ С ЛИНЕЙНЫМИ ОГРАНИЧЕНИЯМИ

Существование развитого аппарата решения больших задач оптимизации при линейных ограничениях (см. разд. 5.6.2) наводит на мысль положить в основу поиска минимума в (6.65) подзадачи именно этого сорта. В частности, можно воспользоваться каким-нибудь методом спроектированного лагранжиана типа рассмотренного в разд. 6.5.2. Чтобы проиллюстрировать данный подход, вкратце опишем один из соответствующих алгоритмов.

Пусть x_k и λ_k — текущее приближение искомой точки x^* и текущая оценка отвечающего ей вектора множителей Лагранжа; индекс k у других величин означает, что они вычислены в x_k. В качестве следующего приближения x_{k+1} предлагается брать решение подзадачи вида

$$\text{найти} \min_{x \in \Omega} F(x) - \lambda_k^T \hat{c}(x) + \frac{\rho}{2} \hat{c}(x)^T \hat{c}(x)$$

$$\text{при ограничениях } Ax = b,$$

$$A_k(x - x_k) \begin{Bmatrix} - \\ \geqslant \end{Bmatrix} - c_k, \tag{6.66}$$

$$l \leqslant x \leqslant u.$$

В данном случае через $\hat{c}(x)$ обозначен вектор, составленный некоторыми из разностей $(c(x) - c_k - A_k(x - x_k))_i$.

Целевая функция в (6.66) имеет форму модифицированной функции Лагранжа и строится с учетом только нелинейных ограничений, причем не всех, а лишь тех, которые считаются активными на k-й итерации. В то же время в A_k и c_k включаются градиенты и значения функций всех нелинейных ограничений. Штрафной член в (6.66) введен для улучшения работы метода при его запуске с плохой начальной точки. Когда становится ясно, что искомое решение близко, параметр штрафа ρ обнуляют; тем самым удается достигать квадратичной сходимости.

В рассматриваемом методе очередное приближение x_k определяется как результат серии «внутренних» итераций поиска минимума в подзадаче (6.66). Этот поиск можно вести любой из универсальных процедур решения больших задач с линейными ограничениями. Надо позаботиться только о том, чтобы в правилах останова процедуры было предусмотрено безусловное прерывание по выполнении определенного числа итераций. Ведь если хорошее приближение к оптимуму в (6.66) не удается получить достаточно быстро, то продолжать поиск скорее всего бессмысленно: долгие блуждания уведут далеко от точки x_k, а там не годятся ни оценки λ_k, ни параметры линеаризованных в x_k ограничений. Эффективность метода в конкретных случаях сильно зависит от выбора величины ρ. Соображения, которыми надо руководствоваться при назначении ρ, аналогичны высказанным ранее в отношении метода модифицированных функций Лагранжа (см. разд. 6.4.2.2).

В качестве λ_k для (6.66) можно брать вектор оценок множителей Лагранжа из предыдущей подзадачи, полученной на последней итерации процесса ее решения. Если эта итерация стала последней потому, что удалось удовлетворить условиям оптимальности, компоненты λ_{k_i}, отвечающие неравенствам, будут неотрицательны. Отметим, что рассматриваемый вектор λ_k связан со «старой» матрицей Якоби A_{k-1}, т. е. вычисляется не по самым свежим данным и потому

оценивает λ^* не лучшим образом. Особенно же грубой эта оценка будет в том случае, когда итерации решения $(k-1)$-й подзадачи прерываются раньше, чем выполнятся условия оптимальности.

*6.7.2. ИСПОЛЬЗОВАНИЕ КВАДРАТИЧНОЙ ПОДЗАДАЧИ

Ниже описаны методы решения (6.65) с квадратичными подзадачами вида (6.41). Линейные ограничения из (6.65) в этих методах обрабатываются приемами гл. 5 и соответственно выполняются в течение всего процесса поиска. Матрица $\hat{A}_k$ в (6.41) будет включать A из (6.65), строки, отвечающие активным простым ограничениям на переменные, и градиенты функций $\{c_i(x)\}$ в x_k. Заметим, что от итерации к итерации может изменяться только *часть* коэффициентов $\hat{A}_k$, и это надо использовать для экономной организации вычислений.

В разд. 6.5.3 решение (6.41) было представлено через матрицы Y и Z из LQ-разложения $\hat{A}$ (см. (6.42)). В случаях большой размерности тоже применяют подобные представления. Мы рассмотрим одно из них, аналогичное полученному в разд. 5.6.2 для больших задач с линейными ограничениями (см. (5.73) и (5.74)).

Для удобства записи индекс k текущей итерации в дальнейшем опущен. Все исходные переменные разобьем на базисные, супербазисные и небазисные (см. разд. 5.6.2). В соответствии с этим делением рассортируем столбцы $\hat{A}$ и компоненты векторов p и g. Поскольку небазисные переменные фиксируются на своих граничных значениях, составляющая p_N вектора p будет нулевой (и ее можно опустить). Что же касается составляющих p_B и p_S, то ограничения (6.41) означают, что они должны быть связаны равенством вида

$$(B \quad S)\begin{pmatrix} p_B \\ p_S \end{pmatrix} = d, \tag{6.67}$$

где d — вектор, состоящий из нулей (они отвечают линейным ограничениям из (6.65)) и компонент вектора c, а B есть невырожденная квадратная $t \times t$-матрица. Это равенство эквивалентно такому:

$$Bp_B = d - Sp_S. \tag{6.68}$$

На (6.68) можно смотреть как на систему уравнений с неизвестным p_B, решение которой при любом p_S дает в комплекте с p_S вектор, допустимый для (6.41) и для линейных ограничений исходной задачи.

Составляющая p_S решения подзадачи (6.41) определяется безусловной минимизацией ее целевой функции после исключения с помощью (6.68) переменной p_B. Выражение этой функции через p_N и p_S выглядит так:

$$\tfrac{1}{2} p_N^T H_{NN} p_N + p_B^T H_{BS} p_S + \tfrac{1}{2} p_S^T H_S p_S + g_B^T p_B + g_S^T p_S, \tag{6.69}$$

где H_B, H_{BS} и H_S — блоки матрицы

$$H=\begin{pmatrix} H_B & H_{BS} \\ H_{BS}^T & H_S \end{pmatrix}.$$

Подстановкой $p_B=B^{-1}(d-Sp_S)$ в (6.69) получим квадратичную форму о p_S, условие минимальности которой сводится к аналогичному (6.44) равенству

$$Z^THZp_S = -Z^Tg + Z^TH\begin{pmatrix} B^{-1}d \\ 0 \end{pmatrix} \qquad (6.70)$$

с матрицей Z вида (5.75). Оно и задает искомый вектор p_S.

***6.7.2.1. Представление матрицы, обратной к базисной.** В данном разделе рассматривается техника представления матрицы B^{-1} в методах решения задачи (6.65) с квадратичной подзадачей. Явно B^{-1} никогда не формируют, так что под «представлением B^{-1}» мы подразумеваем организацию вычисления решений уравнений типа (6.68) и, в частности, выбор способов разложения B и пересчета соответствующих факторов по ходу поиска оптимума.

В дальнейшем используется следующее разбиение B на блоки:

$$B=\begin{pmatrix} B_1 & B_2 \\ B_3 & B_4 \end{pmatrix}\begin{matrix} \} t_1 \\ \} t_2 \end{matrix} \qquad (6.71)$$

Будем считать, что первые t_1 строк B (матрицы B_1 и B_2) отвечают линейным ограничениям (6.65), а последние t_2 строк (матрицы B_3 и B_4) — нелинейным. Блоки B_1 и B_4 квадратные.

От итерации к итерации в матрице B, во-первых, иногда (при сменах базиса) заменяются столбцы и, во-вторых, всякий раз обновляются элементы нижних t_2 строк. Из-за изменений второго типа проблема представления B^{-1} для (6.65) значительно сложнее, чем для задач с линейными ограничениями. При этом «каноничные способы» ее разрешения вроде LU-разложения B заново на каждом шаге не приемлемы (как чрезмерно трудоемкие); к треугольному разложению B «с нуля» прибегают лишь раз в несколько итераций ради повышения точности факторов и сокращения записи их представлений.

Когда число t_2 и количество ненулевых позиций в последних t_2 строках B малы, изменения в B из-за нелинейности ограничений затрагивают только несколько столбцов. В этом случае может оказаться практичным использовать LU-разложение B с пересчетом факторов стандартными методами. Правда, так как на каждой итерации будет применяться несколько элементарных преобразований, к LU-разложению «с нуля» (из соображений повышения точности и экономии памяти) придется прибегать чаще, чем обычно.

Разбиение на подматрицы. Поскольку матрица B_1 меняется только при сменах базиса, ее LU-разложение «поддерживать» не-

трудно. Этим можно воспользоваться, организовав решение систем уравнений с B и B^T на основе разложений B_1 и матрицы размеров B_4: решение системы $Bx=b$ представимо в виде

$$x=\begin{pmatrix} u_1 \\ u_2 \\ v_2 \end{pmatrix},$$

где u_1, u_2 и v_2 определяются из уравнений

$$B_1 u_1 = b_1, \quad Dv_2 = b_2 - B_3 u_1, \quad B_1 u_2 = -B_2 v_2. \qquad (6.72)$$

В них $D=B_4-B_3B_1^{-1}B_2$, а b_1 и b_2 — составляющие разбиения вектора правых частей в соответствии с (6.71).

Вычисление $B^{-1}b$ последовательным решением подсистем (6.72) фактически представляет собой процедуру блочного гауссова исключения с B_1 в качестве первого блока. Иногда ее называют *методом с разбиением B^{-1} на подматрицы*. Известно несколько эффективных способов экономной реализации формул (6.72) с учетом специфики B.

Аппроксимация обратной матрицы; итеративное уточнение. Один из подходов к преодолению трудностей пересчета разложения матрицы B в связи с изменениями в ее последних строках — выполнять его не на каждой итерации. По сути дела это означает «шаг к методам» разд. 6.7.1, в которых параметры линеаризованных ограничений сохраняются постоянными, пока не будет найдено решение подзадачи (6.66).

Возможны разные реализации рассматриваемого подхода. Мы укажем на две из них. Пусть $\bar{B}^{-1}$ — *аппроксимирующая* B^{-1} матрица, которая соответствует той итерации, где разложение B было откорректировано в последний раз. Тогда, во-первых, можно вместо истинных решений систем типа $Bx=b$ использовать векторы $\bar{B}^{-1}b$ (т. е. для ряда итераций заменить B на $\bar{B}$), а, во-вторых, имея B и какое-то представление $\bar{B}^{-1}$, эти решения можно оценивать по *схеме последовательного уточнения* с $\bar{B}^{-1}b$ в роли начальных приближений.

Предлагаемый способ огрубления методов с квадратичной подзадачей вполне допустим, так как в них точность аппроксимации нелинейных ограничений важна только вблизи x^* (разд. 6.5.3.4). Что же касается удаленных от x^* точек x_p, то здесь выбор параметров последних l_2 ограничений в (6.67) достаточно произволен и важно лишь обеспечить существенное убывание функции выигрыша.

Поскольку между сменами базиса первые l_1 строк B не меняются, у $\bar{B}$ они будут теми же, что и у B. Следовательно, равенства из (6.67), отвечающие линейным ограничениям исходной задачи, при расчете направления p с использованием $\bar{B}$ вместо B сохраняются.

Точнее говоря, если в

$$Bp_B=d-Sp_S$$

вместо p_B подставить решение $\hat{p}_B$ системы

$$B\hat{p}_B=d-Sp_S,$$

то первые l_1 равенств не нарушаются.

***6.7.2.2. Расчет направления поиска в пространстве супербазисных переменных.** После того как техника представления B^{-1} (а следовательно, и Z) оговорена, для определения конкретного метода решения задачи (6.65) с использованием квадратичной подзадачи надо установить, как будет вычисляться p_S из (6.70). Трудности здесь те же, что упоминались в разд. 5.6.2.1 при обсуждении больших задач с линейными ограничениями: может не хватать памяти для записи и требоваться слишком много вычислений для построения матриц Z^THZ (и Z^TH). Матрица H размера $n\times n$ наверняка целиком не поместится в памяти.

Зачастую заранее ясно, что размеры Z^THZ будут умеренными на всех итерациях, хотя число переменных n велико. Тогда строить Z^THZ и решать (6.76) можно методами, которые используются для задач малой размерности. Например, в качестве Z^THZ можно брать квазиньютоновскую аппроксимацию спроектированной матрицы Гессе функции Лагранжа, генерируемую (пересчитываемую от шага к шагу) какой-нибудь из процедур, аналогичных тем, что применяются в случаях с линейными ограничениями. Вопросы, возникающие в связи с этим подходом, не являются характерными исключительно для больших задач и упоминались в замечаниях к разд. 6.5. Надо только отметить, что эффективный при небольшом числе переменных прием с вычислением конечных разностей вдоль столбцов Z теперь становится слишком трудоемким, так как формирование каждого из этих столбцов требует решения системы линейных уравнений с матрицей B. По той же причине, даже располагая матрицей W, приходится отказываться от построения Z^TWZ.

Если из-за ограничений по памяти или по трудоемкости явно представить Z^THZ нельзя, но при этом удается эффективно вычислять произведения Z^THZ на вектор, то для определения p_S можно применить линейный метод сопряженных градиентов с улучшением обусловленности (см. разд. 4.8.6). Этот рецепт годится, например, когда матрица Гессе функции Лагранжа W является разреженной с известной структурой заполнения; в данном случае хорошо работает прием расчета Z^TWZv, описанный в разд. 5.1.2.5. Если же матрица W не разрежена (а, так как W компонуется из нескольких матриц вторых производных, это бывает чаще, чем в задачах без ограничений), то не исключено, что положение удается спасти, используя в Z^TWZv вместо векторов WZv их оценки по конечным раз-

постям градиентов функции Лагранжа вдоль Zv. Очевидно, такой выход приемлем лишь при условии, что вычисление упомянутых градиентов — относительно недорогая процедура.

Замечания и избранная библиография к разделу 6.7

Авторами рассмотренного в разд. 6.7.1 алгоритма являются Муртаф и Сондерс (1980). Он реализован в ФОРТРАН-пакете MINOS/AUGMENTED, который можно приобрести в Лаборатории оптимизации систем Стэнфордского университета. Розен (1978) предлагал использовать для больших задач типа (6.65) комбинированную схему, когда сначала процедурой «фазы 1» отыскивается неплохое приближение, а затем запускается алгоритм с подзадачами (6.37). В качестве средства решения последних рекомендовалась программа MINOS (см. разд. 5.6.2).

Очень часто к большим задачам применялась и применяется схема последовательного линейного программирования Гриффита и Стюарта (1961). Чем она привлекает, понятно — можно непосредственно использовать доступные коммерческие пакеты решения больших линейных задач. Однако при этом возникает неприятная проблема выбора эвристических правил подгонки искусственных двусторонних ограничений на переменные подзадачи (см. замечания к разд. 6.5). Методы последовательного линейного программирования обсуждаются в работах Бейкера и Венткера (1980), Бэтчелора и Била (1976), Била (1974, 1978).

Каждая итерация схемы обобщенных приведенных градиентов (см. разд. 6.3.1) фактически включает решение квадратичной подзадачи и может осуществляться теми же средствами, которые используются в пакете MINOS. Джейн, Лэсдон и Сондерс (1976) описали соответствующую реализацию этой схемы, предназначенную для больших задач.

Вопросы оптимизации при нелинейных ограничениях и большом числе переменных с использованием квадратичных подзадач рассмотрены в работе Эскудеро (1980). Там же предлагается метод, основанный на квадратичной подзадаче с неравенствами. Идеи разд. 6.7.2 подробно изложены Гиллом и др. (1981b). Схема с квадратичной подзадачей является более гибкой, чем схема к подзадаче (6.66), но зато реализовать последнюю, располагая пакетом типа MINOS, намного проще.

Процедура (6.72) использует *дополнения Шура*, подробно рассматриваемые в статье Коттла (1974). Прочие ссылки по поводу методов решения разреженных линейных систем и пересчета представлений их матриц даны в замечаниях к разд. 5.6. Упомянутый в разд. 6.7.2.1 метод последовательного уточнения досконально разобран в книге Уилкинсона (1965).

6.8. ЗАДАЧИ СПЕЦИАЛЬНЫХ ТИПОВ

Среди существующих методов безусловной минимизации и минимизации при линейных или нелинейных ограничениях есть много специальных, рассчитанных на особые категории задач. Некоторые из них были представлены выше, например методы для задач линейного и квадратичного программирования (см. разд. 5.3.1, 5.3.2 и 5.6.1) и для безусловной задачи о наименьших квадратах (см. разд. 4.7). Говорилось также, хотя и без описания способов решения, о задачах безусловной минимизации негладких функций особой структуры.

В данном разделе мы отнюдь не собираемся обсуждать (или хотя бы упомянуть) все те бесчисленные ранее специальные типы задач оптимизации — их слишком много и каждому можно было бы посвятить отдельную книгу. Здесь вкратце представлены лишь некоторые наиболее важные из них и даны ссылки на литературу, содержащую более подробные и полные сведения. Следует отметить, что многие специальные методы являются модификациями описанных в гл. 4, 5 и 6 базовых схем, эксплуатирующими структуру или известные свойства задач определенной категории.

6.8.1. СПЕЦИАЛЬНЫЕ ЗАДАЧИ МИНИМИЗАЦИИ НЕГЛАДКИХ ФУНКЦИЙ

На практике нередко встречаются задачи безусловной минимизации недифференцируемой функции F, составленной из гладких функций (негладкость F объясняется способом объединения составляющих). О них уже шла речь в разд. 4.2.3, и там, в частности, было указано на то, что структуры разрывов производных соответствующих F очень специфичны. Возьмем, например, задачу вида

$$\text{найти } \min_{x \in R^n} \max \{f_1(x), f_2(x), \ldots, f_m(x)\}, \tag{6.73}$$

в которой $\{f_i(x)\}$ — гладкие функции. В данном случае производные от F терпят разрывы в точках, где $f_i = f_j = F$, $i \neq j$. Столь же просто характеризуются и точки разрывов F из (4.4): в каждой из них при некоторых i выполняются равенства $f_i = 0$.

Особый подкласс составляют задачи типа (6.73) и (4.4) с *линейными* $\{f_i\}$. При помощи преобразований, описанных в разд. 4.2.3, они сводятся к *задачам линейного программирования* и могут быть решены симплекс-методом. Применением техники последовательной линеаризации этот подход обобщается и на (6.73), (4.4) с нелинейными $\{f_i\}$, но для них он не самый эффективный.

Современные методы решения нелинейных задач (4.4) и (6.73) работают по такому принципу: на каждой итерации (i) прогнозируется список равенств вида $f_i = f_j = F$ (для (6.73)) или $f_i = 0$ (для

(4.4)), выполненных в x^*; (ii) решением квадратичной подзадачи из схемы спроектированного лагранжиана (см. разд. 5.6.3) для минимизации F при условии соблюдения выделенных равенств определяется направление поиска; (iii) делается шаг спуска вдоль найденного направления. При этом для выбора длины шага и оценивания множителей Лагранжа, фигурирующих в подзадаче расчета направления поиска, можно предложить специальные процедуры.

Описанный подход фактически есть приложение схемы спроектированного лагранжиана к эквивалентным (4.4) и (6.73) задачам с ограничениями (см. разд. 4.2.3). Он применим не только к (4.4) и (6.73), но и к другим задачам безусловной минимизации негладких функций особой структуры. Например, его можно использовать для поиска минимума абсолютной штрафной функции (6.8) (см. замечания к разд. 6.2 и 6.5).

6.8.2. Специальные задачи с ограничениями

6.8.2.1. Выпуклое программирование. В данном разделе говорится о выпуклых и вогнутых функциях. Общие определения таковых здесь не приведены, но если предполагать существование непрерывных вторых производных, что мы и будем делать, то выпуклыми (вогнутыми) можно называть такие функции, матрицы Гессе которых всюду положительно (отрицательно) полуопределены. (Заметим, что линейные функции являются и выпуклыми, и вогнутыми одновременно.)

Задачи типа NCP (см. разд. 1.1), в которых $F(x)$ выпукла, функции равенств линейны, а неравенств вогнуты, обладают многими специфическими свойствами. На этом основании их группируют в отдельный класс, именуя *задачами выпуклого программирования* или просто *выпуклыми задачами*. С теоретической точки зрения они прежде всего хороши тем, что в них любой локальный минимум с необходимостью оказывается глобальным (см. разд. 3.1), причем положительная определенность $G(x^*)$ гарантирует единственность решения.

Выпуклость задачи исключает целый ряд алгоритмических проблем и открывает дополнительные возможности организации поиска оптимума. Мы укажем на три важных момента. Во-первых, в методах выпуклого программирования не надо предусматривать защиты на случай знаконеопределенности матрицы Гессе или неограниченности решения подзадачи (см., например, разд. 4.4.2 и 6.5.4.2). Нет нужды также модифицировать функцию Лагранжа (см. разд. 6.4.1), чтобы сделать x^* точкой безусловного минимума.

Во-вторых, для выпуклых задач нет проблемы несовместности линеаризованных неравенств, получающихся аппроксимацией исходных по Тейлору (см. разд. 6.5.4.1): если функция $c_i(x)$ вогнута, независимо от $\bar{x}$ из $c_i(x) \geqslant 0$ следует, что $c_i'(x)^T(x-\bar{x}) \geqslant -c_i(\bar{x})$.

В-третьих, в выпуклом программировании развита мощная теория двойственности (задачи, двойственные к линейным и квадратичным, кратко описаны в разд. 3.3.2.2 и 3.3.2.3), порождающая методы, ориентированные на двойственную задачу или комбинацию ее с прямой.

К сожалению, универсальных вычислительных приемов, которые позволяли бы выявлять выпуклость функций общего вида, не существует. Поэтому методы выпуклого программирования используют только тогда, когда выпуклость удается установить аналитическим путем (как, например, для задач из разд. 6.8.2.3).

6.8.2.2. Сепарабельное программирование. Функцию Φ называют сепарабельной, если

$$\Phi(x) = \sum_{i=1}^{n} \varphi_i(x_i),$$

т. е. если Φ — сумма функций одной переменной. Ясно, что матрица Гессе сепарабельной функции диагональна и что задача ее минимизации без ограничений (или при простых ограничениях на переменные) распадается на n одномерных.

Термин «сепарабельная задача» обычно употребляют в отношении постановок вида

$$\text{найти} \min_{x \in \mathbb{R}^n} F(x)$$

$$\text{при ограничениях } c_i(x) \geqslant 0,$$

где F и $\{c_i\}$ — сепарабельные функции. Многие из методов, разработанных для решения сепарабельных задач, предполагают линейность ограничений и опираются на технику линейного программирования.

6.8.2.3. Геометрическое программирование. Термин «геометрическое программирование» возник благодаря известному неравенству между геометрическим и арифметическим средними, играющему важную роль в задачах с позиномиальными функциями. *Позином* — это функция *положительного векторного аргумента* x ($x \in \mathbb{R}^n$) вида

$$\Phi(x) = \sum_{i=1}^{N} v_i(x),$$

где $v_i(x) = \alpha_i x_1^{a_{i1}} x_2^{a_{i2}} \dots x_n^{a_{in}}$, $i = 1, \dots, N$, и $\alpha_i > 0$, $i = 1, \dots, N$.

Задачу позиномиального (геометрического) программирования обычно ставят так:

$$\text{найти} \min_{x \in \mathbb{R}^n} F(x)$$

$$\text{при ограничениях } c_i(x) \leqslant 1, \quad i = 1, \dots, m;$$

где F и $\{c_i\}$ — позиномы. Можно показать, что она эквивалентна некоторой сепарабельной выпуклой задаче.

К позиномиальным близки так называемые *сигномиальные* задачи. Отличие состоит в том, что в них не требуется неравенств $\alpha_i \geq 0$. Они могут и не сводиться к выпуклым задачам.

Замечания и избранная библиография к разделу 6.8

По методам решения задач, рассмотренных в разд. 6.8.1, существует обширнейшая литература. Одна из самых старых публикаций на эту тему — статья Полна (1913), где описан алгоритм для задачи (6.73) с линейными функциями. Наиболее эффективными для линейного случая являются методы, построенные по схеме Штифеля (1960).

Нелинейные задачи (6.73) чаще всего решают сведением к последовательности линейных, причем обычно решение линеаризованной задачи используется в качестве направления поиска, т. е. предполагается еще и одномерный спуск. Так работают, например, алгоритмы Зуховицкого, Полака и Примака (1963), Осборна и Уотсона (1969). Иначе устроены методы Зангвилла (1967b), Караламбуса и Конна (1978). В них направление поиска определяется в результате решения нескольких систем линейных уравнений, формируемых в процессе идентификации набора функций, «активных» в текущей точке. Обобщения на задачи (6.73) алгоритма Левенберга — Маркардта (см. разд. 4.7.3) предлагались Мадсеном (1975), Андерсеном и Осборном (1977). В статье Уотсона (1979) описан двухэтапный гибридный метод. Методы решения задачи (6.73), родственные схеме спроектированного лагранжиана из разд. 6.5.3, читатель найдет у Хана (1978a, b), Конна (1979), Мюррея и Овертона (1980a); в них направления поиска интерпретируемы как решения неких квадратичных подзадач.

Линейная задача (4.4) безусловной минимизации суммы модулей тоже имеет довольно долгую историю. На ее связь с линейным программированием впервые указано в статье Чарнеса, Купера и Фергюсона (1955). Эта связь неоднократно использовалась для разработки алгоритмов; см., например, Бэрродейл и Роберте (1973), Армстронг и Годфри (1979). Нелинейную задачу (4.4) можно сводить к последовательности линейных. Соответствующий алгоритм описан у Осборна и Уотсона (1971). Эль-Аттар, Видьясагар и Дутта (1979) предложили для нее метод со штрафной функцией, а Мак-Лин и Уотсон (1979) — метод, основанный на процедуре Левенберга — Маркардта. Способ решения (4.4) с нелинейными функциями переходом к эквивалентной задаче с ограничениями (см. разд. 4.2.3) и применением к последней специального метода спроектированного лагранжиана с квадратичной подзадачей описан у Мюррея и Овертона (1980b).

Исчерпывающие сведения о всех аспектах выпуклости читатель найдет в книге Рокафеллара (1970), а тому, кого интересуют лишь принципиальные для выпуклого программирования результаты, мы рекомендуем книгу Мангасарьяна (1969).

Основной подход к решению задач сепарабельного программирования с линейными ограничениями (см. Хартли (1961) и Миллер (1963)) состоит в том, чтобы нелинейные составляющие целевой функции аппроксимировать кусочно-линейными и применять к полученной подзадаче специальную версию симплекс-метода (разд. 5.3.1). Подобные схемы реализованы во многих коммерческих пакетах линейного программирования.

Полный обзор ранних работ по геометрическому программированию содержится в книге Даффина, Петерсона и Зенера (1967). Первоначально термин «геометрическое программирование» применялся только в отношении позиномиальных задач; теперь его употребляют в значительно более широком смысле (см. статью Петерсона (1976)). Две прекрасные обзорные статьи по практическим и вычислительным аспектам геометрического программирования опубликованы Дембо (1978) и Эккером (1980). В обеих приведены обширные списки литературы.

ГЛАВА 7

МОДЕЛИРОВАНИЕ

...кто может сказать,
какое из обличий верно отражает ее сущность?
Уильям Батлер Йитс, «A Bronze Head» (1933)

7.1. ВВЕДЕНИЕ

Рассматриваемые в предыдущих главах алгоритмы относятся к средствам численного моделирования (моделирования с применением вычислительных машин). К нему прибегают для изучения явлений действительности, не подлающихся разбору только аналитическими методами. Нередко численная модель оказывается наиболее эффективным, а то и просто единственным инструментом оценивания альтернатив: эксперимент с реальной системой, например, может быть слишком дорогим, опасным или вообще недопустимым.

Среди всевозможных численных моделей выделяются открытые, соотношения которых оставляют некую свободу выбора значений переменных. Открытая модель, как правило, служит для определения таких величин варьируемых параметров объекта исследования, которые оптимизируют некую «меру благополучия» (скажем, максимизируют его устойчивость или обеспечивают его оптимальное функционирование при конкретных внешних условиях). Здесь и нужны методы предыдущих глав.

Формируя модель, иногда заботятся лишь о максимальном приближении ее к действительности, а решение вопроса о том, как проводить по ней оптимизационные расчеты, откладывают до момента, когда она будет в основном построена. Однако потом нередко оказывается, что среди готовых алгоритмов нет подходящего, который позволил бы эффективно работать с моделью.

С другой стороны, желание облегчить поиск оптимума не может оправдать чрезмерного упрощения модели. В связи с этим следует сказать о тенденции описывать линейными соотношениями даже существенно нелинейные процессы. Оно объясняется тем, что до недавних пор не было программ решения больших нелинейных задач (соответствующие методы рассмотрены выше в разд. 6.6.2 и 6.7), в то время как пакеты линейного программирования, рассчитанные на очень большие задачи, уже давно входят в библиотеки математического обеспечения. Однако, чтобы линеаризация не слишком огрубила модель, часто приходится сильно увеличивать размерность; к тому же при линеаризации меняется характер решения — у линейной задачи оно (если единственно) обязательно будет

вершиной допустимого множества, а для нелинейной, как правило, это не так (см. разд. 5.3.1).

При разработке оптимизационной модели следует стремиться к разумному балансу ее точности (повышение которой обычно достигается ценой усложнения формулировок) и простоты оптимизации. Один из способов достичь такого баланса — «пошаговое» уточнение модели, когда одна за другой просчитываются ее постепенно усложняемые версии. Отслеживая влияние каждой модификации на оптимизационный процесс, разбираться в возникающих трудностях значительно легче, чем тогда, когда сразу просчитывается сложная модель. Этот прием особенно полезен при конструировании моделей, содержащих много сложных взаимосвязанных блоков.

В данной главе рассмотрен ряд аспектов моделирования, важных с вычислительной точки зрения (вопросов соответствия моделей реальности мы касаться не будем). В частности, обсуждаются предпочтительные варианты постановок оптимизационных задач. Опыт показывает, что здесь есть нюансы, сильно влияющие на трудоемкость решения, даже если используются самые совершенные программы. При этом формулировка модели зачастую оказывается результатом произвольного выбора среди множества альтернатив, абсолютно равноценных в смысле адекватности исследуемому объекту и в то же время существенно различных для алгоритмов.

Ниже описаны стандартные (и по большей части простые) приемы преобразования оптимизационных задач с целью облегчить поиск их решений. Почти все эти приемы успешно применялись нами на практике. Будет сказано также о неких предосторожностях, позволяющих избегать характерных вычислительных затруднений.

Разумеется, мы не претендуем на полноту изложения всех аспектов численного моделирования, имеющих значение в контексте оптимизации,— слишком велико их многообразие. Однако хотелось бы надеяться, что представленный материал убедит читателя принимать во внимание вычислительные проблемы с самого начала создания оптимизационной модели, причем вряд ли когда-нибудь будет создано математическое обеспечение, которое избавит от этих забот.

7.2. КЛАССИФИКАЦИЯ ОПТИМИЗАЦИОННЫХ ЗАДАЧ

В гл. 1 была введена следующая общая постановка оптимизационной задачи:

$$\text{NCP найти } \min_{x \in \mathbb{R}^n} F(x)$$

при ограничениях $c_i(x) = 0,\ i = 1, 2, \ldots, m';$

$$c_i(x) \geqslant 0,\ i = m'+1, \ldots, m.$$

Ограничения на переменные могут принимать и другие формы; например, иногда каким-то переменным разрешается пробегать лишь

конечное множество значений. В последнем случае найти оптимум, как правило, оказывается намного сложнее, чем в NCP; некоторые подходы к решению задач с дискретными переменными упоминаются в разд. 7.7.

Подбирая алгоритм для конкретной задачи типа NCP, обязательно надо принимать во внимание ее специфику. Если она достаточно отчетлива, обращаться к алгоритму, рассчитанному на самый общий случай, неразумно. Рациональный выбор определяется различными свойствами функций задачи. По некоторым совокупностям этих свойств выделяются классы задач, каждому из которых отвечает своя предпочтительная схема поиска оптимума. Типовая классификация представлена в разд. 1.2.

Одни характеристики задачи очень сильно влияют на сложность ее решения, другие — намного слабее. Возьмем, к примеру, размерность. Ее увеличение приводит к разному возрастанию сложности только «в двух точках»: во-первых, при переходе от одномерных задач к многомерным и, во-вторых, когда а) данные задачи перестают умещаться в оперативной памяти машины и b) для повышения эффективности счета приходится начинать эксплуатировать слабую заполненность (большую долю нулевых элементов) матриц производных. Что же касается «отрезка» между этими «точками», то на нем усилия, требуемые для решения типичной задачи, грубо говоря, зависят от ее размерности как полином умеренной степени. Поэтому, скажем, безусловная минимизация по двенадцати переменным обычно немногим сложнее, чем по девяти.

Совсем иначе отражается на трудоемкости задачи форма ее условий. Лучше всего, когда их вообще нет или когда все они суть простые ограничения на переменные. (Надо сказать, что переход от безусловной минимизации к минимизации при простых ограничениях иногда даже ускоряет решение.) Достаточно заменить последние линейными ограничениями общего вида — и поиск оптимума ощутимо усложнится. Появление нелинейных ограничений повысит сложность еще на порядок (см. гл. 6); поэтому иногда, если это возможно, стоит переформулировать модель так, чтобы в ней не было нелинейных ограничений (см. разд. 7.4).

Среди свойств функций оптимизационной задачи, облегчающих ее решение, самым фундаментальным, по-видимому, является *дифференцируемость*. Она позволяет эффективно аппроксимировать функции по их локальным характеристикам, и это используется в алгоритмах. К примеру, если существуют непрерывные вторые производные, то локализовать решение намного легче, чем в случае с недифференцируемыми функциями. Поэтому большинство программ оптимизационного математического обеспечения рассчитано именно на гладкие задачи, и, если можно сформулировать модель в терминах гладких функций, так и надо сделать (см. разд. 7.3). Выбор алгоритма решения гладкой задачи конкретного типа зависит от того, какая информация о производных доступна, насколько

велика трудоемкость их вычисления и т. д. При этом общая тенденция такова, что с увеличением объема используемой информации о производных надежность и эффективность алгоритмов возрастают (см. разд. 8.1).

7.3. ИСКЛЮЧЕНИЕ НЕОБЯЗАТЕЛЬНЫХ РАЗРЫВНОСТЕЙ

Эпитет «необязательных» в заголовке этого раздела фигурирует потому, что, строго говоря, результаты вычислений с конечной точностью никогда не составят непрерывной функции. К тому же обычное математическое определение непрерывности, включающее произвольные малые приращения функции и ее аргумента, при операциях над конечным множеством чисел, представимых в формате с плавающей запятой, просто неприменимо. В общем «машинная версия» любой функции разрывна по своей природе. Однако, если сама функция гладкая и хорошо отмасштабирована, эта разрывность несущественна — она не затрудни! работы предполагающих гладкость алгоритмов оптимизации; к плохо отмасштабированным функциям сказанное не относится. Проблема масштабирования будет кратко обсуждаться в разд. 7.5 и подробнее — в разд. 8.7.

Поскольку задачу с недифференцируемыми функциями общего вида решать тяжело, при формировании оптимизационной модели их по возможности следует избегать. Другое дело — специальные случаи типа рассмотренных в разд. 4.2.3 и 6.8; с ними трудностей не возникает. Мы хотим подчеркнуть, что важно отличать недифференцируемые функции от функций, производные которых (по каким-то причинам) недоступны. У первых просто нет производных в некоторых точках; например, у функции $\max\{x_1, x_2\}$ их нет при $x_1=x_2$. У вторых они есть всюду, и сам факт их существования уже имеет значение для выбора алгоритма.

Из сущности модели обычно не так уже трудно понять, дифференцируемы составляющие ее функции или нет. Если у моделируемого процесса есть критические точки, скажем может переполниться какой-то резервуар или допустимо переключение производства с одного ресурса на другой, то разрывы производных возможны. Иначе их скорее всего не будет. Даже если в отношении дифференцируемости ничего определенного сказать нельзя, то обычно для начала стоит предположить, что она есть: неудачная попытка решить задачу методом, рассчитанным на гладкие функции, обойдется не так уже дорого по сравнению с теми усилиями, которые потребуются для ее решения алгоритмом, не предполагающим гладкости.

7.3.1. РОЛЬ ТОЧНОСТИ ВЫЧИСЛЕНИЯ ФУНКЦИЙ МОДЕЛИ

На практике нередки оптимизационные задачи, которые не надо решать с высокой точностью (она, например, ни к чему, когда в порождающей задачу модели заведомо не отражены какие-то сущест-

венные свойства исследуемого объекта или ее параметры подобраны по весьма приближенным данным), и бытует заблуждение, что в подобных случаях точно вычислять функцию задачи тоже бессмысленно. Считают, что их достаточно оценивать лишь слегка поточнее, чем требуется оценить оптимум.

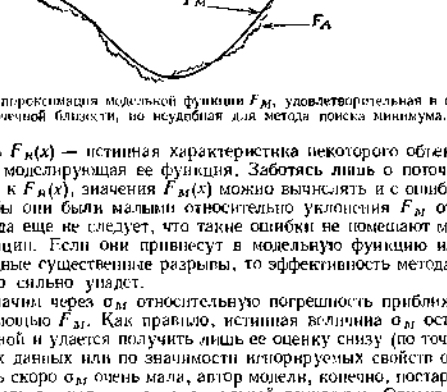

Рис. 7а. Аппроксимация модельной функции F_M, удовлетворительная в смысле поточечной близости, но неудобная для метода поиска минимума.

Пусть $F_R(x)$ — истинная характеристика некоторого объекта, а $F_M(x)$ — моделирующая ее функция. Заботясь лишь о поточечной близости к $F_R(x)$, значения $F_M(x)$ можно вычислять и с ошибками, только бы они были малыми относительно уклонения F_M от F_R. Но отсюда еще не следует, что такие ошибки не помешают методу оптимизации. Если они привнесут в модельную функцию или ее производные существенные разрывы, то эффективность метода скорее всего сильно упадет.

Обозначим через σ_M относительную погрешность приближения F_R с помощью F_M. Как правило, истинная величина σ_M остается неизвестной и удается получить лишь ее оценку снизу (по точности исходных данных или по значимости игнорируемых свойств объекта). Коль скоро σ_M очень мала, автор модели, конечно, постарается реализовать вычисления с максимальной точностью. Однако чаще σ_M лежит в диапазоне 0,1%—5,0%. Допустим, что при этом есть две вычислимые аппроксимации F_A и F_B, уклонения которых от F_M, допускаемые ради облегчения расчетов, не превосходят величины σ_c. Если σ_c существенно меньше, чем σ_M (например, $\sigma_c \approx 0{,}01\%$), то в смысле близости к F_R функции F_A и F_B будут приближениями того же качества, что и F_M.

На рис. 7а и 7b рассматриваемая ситуация показана (в несколько утрированном для наглядности варианте) для одномерного случая. Если судить только по уклонениям $|F_R(x)-F_A(x)|$ и $|F_R(x)-F_B(x)|$, изображенные функции F_A и F_B моделируют F_R одинаково хорошо. В то же время для метода минимизации они совершенно различны. В частности, F_B является гладкой, а у F_A и ее производных есть

много разрывов. Последние чреваты целым рядом неприятностей. Прежде всего у F_A есть точки локальных минимумов, далекие от точки минимума F_M, и поиск вполне может сойтись в какую-то из них. Трудности иного рода возникают при использовании для минимизации F_A конечно-разностных приближений производных: если

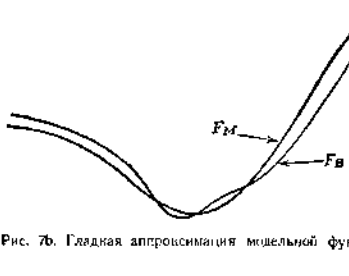

Рис. 7b. Гладкая аппроксимация модельной функции F_M.

интервал аппроксимации захватит точку разрыва, полученная оценка производной будет негодной, даже когда градиент в текущей точке хорошо определен.

Казалось бы, высказанные предостережения должны быть адресованы только узкой аудитории самых непосвященных, поскольку умышленно вводить в модель ненужные разрывы, разумеется, никто не станет. Однако практика показывает, что заблуждение относительно безобидности «малых» неточностей подсчета функций распространено шире, чем можно ожидать.

7.3.2. АППРОКСИМАЦИЯ ПО РЯДАМ И ТАБЛИЦАМ

По нашему опыту одной из самых частых причин потери гладкости является обращение при расчете модельной функции $F(x)$ к некоторым вспомогательным функциям $W(\gamma)$. Поскольку машины могут выполнять только элементарные действия, эти функции всегда как-то аппроксимируются. Часто приближениями служат отрезки рядов, выбор которых зависит от аргумента. Таким образом, может, например, случиться, что для оценивания $W(\gamma)$ будут использованы две формулы — одна при малых $|\gamma|$, а другая при больших. Граничные γ, вообще говоря, оказываются точками разрывов. Даже если $W(\gamma)$ при всех γ аппроксимируется на основе одного ряда, длины выбираемых отрезков для различных γ могут быть разными; скажем, при $\gamma=4.7$ могут использоваться четыре первых члена ряда, а при $\gamma=4.7+10^{-16}$ пять.

Чтобы избежать указанных разрывностей (или по крайней мере свести их последствия к минимуму), рекомендуется

(i) по возможности исключать смены формул (например, фиксируя длину отрезка ряда, которым приближается функция);

(ii) если смены формул допускаются, то обеспечивать их стыковку по значениям генерируемой функции (и, если удается, ее первых производных).

К сожалению, прибегая к стандартным программам вычисления вспомогательных функций, пользователь обычно не может контролировать смену формул. К примеру, стандартная библиотечная процедура подсчета функции Бесселя $J_n(y)$ реализует два метода и переключается с одного на другой при $y=11$. По этой причине иногда приходится заменять библиотечные программы своими.

Еще один источник негладкости, схожий по природе с рассмотренным,— применение табличных функций. Допустим, $F(x)$ зависит от величины $V(y)$, затабулированной для набора значений $y=0(0.01)1$. Возникает вопрос: как определять $V(y)$ в промежуточных точках? Проще всего брать в качестве $V(y)$ табличное значение из ближайшей к y точки, кратной 0.01. Тогда, например, если требуется величина $V(0.6243)$, вместо нее будет использована $V(0.62)$, и в смысле близости к F подобный подход вполне приемлем. Однако фактически это означает замену $V(y)$ кусочно-постоянной функцией, разрывность которой может привести к осложнениям. Линейная интерполяция $V(y)$ между узлами сетки дает непрерывную функцию, но первые производные будут разрывными. Лучшее же (и всегда реализуемое) решение — вообще отказаться от таблиц и заменять их гладкими аппроксимирующими функциями типа сплайнов. Даже двумерные таблицы удается эффективно представлять гладкими поверхностями.

7.3.3. ОПРЕДЕЛЕНИЕ ФУНКЦИИ ПОДЗАДАЧЕЙ

Более запутанная ситуация возникает, когда значение модельной функции определяется решением некоторой подзадачи, например интегрированием дифференциальных уравнений или какой-то другой функции. Построение такого решения с полной машинной точностью (что, кстати сказать, не всегда возможно) обычно требует значительных усилий. Поэтому, как правило, его считают непозволительной роскошью, имея в виду, что и дифференциальные уравнения, и интеграл, и любая другая математическая конструкция все равно есть лишь приближенное описание чего-то реального.

Возьмем для примера часто встречающуюся задачу безусловной минимизации по x интеграла

$$F(x) = \int_c^b f(x, t)\, dt. \tag{7.1}$$

Аналитически проинтегрировать $f(x, t)$ обычно не удается, и приходится использовать какую-нибудь численную квадратурную схему. Последняя даст вместо истинной величины $F(x)$ ее приближение, равное взвешенной сумме значений f в избранных точках:

$$\int_a^b f(x, t)\,dt \approx I(t, \omega) \equiv \sum_{j=1}^{M} \omega_j f(x, t_j). \qquad (7.2)$$

Здесь $\{\omega_j\}$ — веса, а $\{t_j\}$ — набор абсцисс, удовлетворяющих соотношениям $a \leqslant t_1 \leqslant \ldots \leqslant t_M \leqslant b$. Ошибка оценки (7.2) зависит от производных f по t порядка выше первого, числа точек $\{t_j\}$ и их расположения на отрезке $[a, b]$.

Наиболее совершенные методы численного интегрирования являются *адаптивными*. В них t_j в (7.2) подбираются по ходу вычислений в соответствии с наблюдаемым поведением f; общий принцип состоит в том, что, чем сложнее ведет себя f, тем «плотнее» надо располагать точки. Есть немало хороших реализаций адаптивных методов, и, коль скоро какая-то из соответствующих программ доступна, естественно возникает желание вычислить F в (7.1) с ее помощью. Однако при минимизации F это чревато теми же неприятными последствиями, что и применение стандартных программ расчета приближений функций по рядам (см. разд. 7.3.2). В силу своей итеративной сущности адаптивная квадратурная схема может выбрать сильно различающиеся наборы $\{t_j\}$ даже для близких значений x. Это значит, что генерируемое приближение F будет иметь нежелательные разрывы, хотя его равномерная близость к F гарантируется. Кривая оценок интеграла, вычисленных адаптивным методом, вполне может напоминать F_A с рис. 7а.

Следует подчеркнуть, что адаптивное численное интегрирование в рассматриваемом случае непригодно только потому, что (по крайней мере вдали от решения) гладкость аппроксимации F здесь важнее, чем ее точность.

При поиске минимума (7.1) лучше пользоваться квадратурными формулами с фиксированными $\{\omega_j\}$ и $\{t_j\}$. Они определяют гладкие функции I. Подав на вход метода минимизации такую I, мы получим точку x, как-то приближающую x^*. Скорее всего x недостаточно хорошо оценит x^*, и поэтому надо выполнить еще один шаг — применить более надежную квадратурную формулу (скажем, с увеличенным, но по-прежнему фиксированным числом точек M) и повторить минимизацию, взяв $\bar{x}$ в качестве начального приближения. Если $f(x, t)$ — достаточно хорошая функция, разумный выбор абсцисс позволит получить улучшенную оценку минимума F при умеренном увеличении их числа. Кроме того, так как $\bar{x}$ все же неплохое приближение x^*, много раз подсчитывать I для повторной минимизации не придется. В результате будет найдена точная оценка x^*, и, если нужна не только она, но и точная оценка величины $F(x^*)$, в завершение всей процедуры можно один раз вычислить интеграл

каким-нибудь адаптивным методом. Разобранный пример подтверждает высказанное ранее положение, что есть смысл чередовать этапы оптимизации и моделирования: переход к более совершенной квадратурной формуле — это не что иное, как уточнение модели.

Замечания и избранная библиография к разделу 7.3

Проблема ошибок вычисления функций задачи подробнее рассматривается в разд. 8.5.

О методах интерполяции и аппроксимации данных гладкими функциями можно прочесть в сборнике под редакцией Хайеса (1970) и в статье Пауэлла (1977a). Современный (на уровне 1977 г.) обзор и обширная библиография по этим методам даны Коксом (1977).

Основные квадратурные формулы и описание их свойств приводится в любом учебнике по численному анализу, например у Дальквиста и Бьёрка (1974). За более полными сведениями рекомендуем обратиться к работе Лайнесса (1977b). Подробное обсуждение трудностей, связанных с определением целевой функции задачи оптимизации по адаптивной квадратурной схеме, читатель найдет в другой статье того же автора (Лайнесс (1976)).

7.4. ПРЕОБРАЗОВАНИЯ ЗАДАЧ

7.4.1. УПРОЩЕНИЕ ИЛИ ИСКЛЮЧЕНИЕ ОГРАНИЧЕНИЙ

Еще не так давно алгоритмы безусловной минимизации были значительно более многочисленны и совершенны, чем алгоритмы минимизации при ограничениях. Сегодня ситуация иная, и современная техника решения задач с простыми и линейными ограничениями сравнима по эффективности с техникой поиска безусловного минимума. Поэтому практически никогда не стоит избавляться от простых ограничений (более того, они часто облегчают поиск) и очень редко имеет смысл преобразовывать задачи с линейными ограничениями общего вида.

7.4.1.1. Исключение простых ограничений. Любое преобразование задачи следует применять с большой осторожностью. В частности, некоторые «народные средства» облегчения поиска оптимума могут на самом деле усложнить его. В качестве примера рассмотрим один из самых «древних» способов преобразования оптимизационных постановок, предназначенный для задач вида

$$\text{найти} \min_{x \in \mathbb{R}^n} F(x)$$

$$\text{при ограничениях } x_i \geqslant 0, \quad i = 1, 2, \ldots, n.$$

Идея состоит в том, что заменить их задачами *без ограничений*

$$\text{найти } \min_{\omega \in R^n} \mathcal{F}(\omega),$$

где $\mathcal{F}(\omega)$ получается из $F(x)$ подстановкой вместо x_i квадрата новой переменной ω_i, т. е. $x_i = \omega_i^2$.

Пример 7.1. Возьмем в качестве первого объекта приложения рассматриваемой замены переменных квадратичную задачу вида

$$\text{найти } \min_{x \in R^2} (x_1 + 1)^2 + (x_2 + 1)^2$$

$$\text{при ограничении } x_1 \geqslant 0.$$

Градиент и матрица Гессе ее целевой функции равны

$$g(x) = \begin{pmatrix} 2(x_1 + 1) \\ 2(x_2 + 1) \end{pmatrix}, \quad G(x) = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}.$$

Решением является точка $(0, -1)^T$. В любом методе минимизации при простых ограничениях типа описанного в разд. 5.5.1 переменная x_1 в конце концов будет зафиксирована на нуле и после этого быстро определится минимум от $(x_2 + 1)^2$ по x_2.

Замена x_1, x_2 на ω_1^2, ω_2^2 соответственно приводит к следующей задаче:

$$\text{найти } \min_{\omega \in R^2} (\omega_1^2 + 1)^2 + (\omega_2^2 + 1)^2.$$

Первые и вторые производные ее целевой функции таковы

$$\nabla \mathcal{F}(\omega) = \begin{pmatrix} 4\omega_1(\omega_1^2 + 1) \\ 2(\omega_2 + 1) \end{pmatrix}, \quad \nabla^2 \mathcal{F}(\omega) = \begin{pmatrix} 4(3\omega_1^2 + 1) & 0 \\ 0 & 2 \end{pmatrix},$$

и к ней можно применить любой метод безусловной минимизации. Хотя функция $\mathcal{F}$ несколько «более нелинейна», чем F, у нее есть только одна стационарная точка — там достигается искомый минимум и матрица Гессе хорошо обусловлена.

Примеры, подобные приведенному, служат рекламой «квадратичного» преобразования, и именно благодаря им оно приобрело популярность. Однако далеко не всегда получается столь удачная функция $\mathcal{F}$.

Пример 7.2. Возьмем слегка отличную от предыдущей задачу:

$$\text{найти } \min_{x \in R^2} (x_1 - 1)^2 + (x_2 + 1)^2$$

$$\text{при ограничении } x_1 \geqslant 0.$$

Ее решением является точка $(1, -1)^T$, так что условие на x_1 теперь не является ограничительным. Отметим, что матрица Гессе целевой функции всюду положительно определена. Градиент и матрица Гессе преобразованной целевой функции выглядят следующим об-

разом:

$$\nabla \mathscr{F}(w) = \begin{pmatrix} 4w_1(w_1^2 - 1) \\ 2(w_2 + 1) \end{pmatrix}, \quad \nabla^2 \mathscr{F}(w) = \begin{pmatrix} 4(3w_1^2 - 1) & 0 \\ 0 & 2 \end{pmatrix}.$$

Очевиден дефект — у $\mathscr{F}$ есть седловая точка $(0, -1)^T$. Это значит, что, применив к $\mathscr{F}$ алгоритм безусловной минимизации, гарантирующий лишь стационарность предела поиска (а таковыми являются все методы, кроме модифицированных ньютоновских), можно получить ложный оптимум. В то же время при решении исходной задачи алгоритмом минимизации при простых ограничениях подобная неприятность исключена: если он и придет в точку $(0, -1)^T$, то по знаку множителя Лагранжа в ней выяснится, что переменную x_1 следует считать свободной, и затем поиск будет осуществляться по обеим переменным. Таким образом, комбинаторная проблема определения правильного рабочего списка, возникающая в методах активного набора, квадратичным преобразованием, вообще говоря, не снимается, а заменяется тоже комбинаторной и даже более сложной проблемой отбраковки неоптимальных стационарных точек.

К сожалению, описанная трудность не единственна.

Пример 7.3. Рассмотрим задачу

$$\text{найти } \min_{x \in \mathbb{R}^2} x_1^{5/2} + (x_2 + 1)^2$$

$$\text{при ограничении } x_1 \geqslant 0,$$

имеющую решение $(0, -1)^T$. Ее преобразованный аналог таков:

$$\text{найти } \min_{w \in \mathbb{R}^2} (w_1^2)^{5/2} + (w_2 + 1)^2.$$

Прежде всего следует отметить, что нельзя вычислять $\mathscr{F}$ как $w_1^5 + (w_2 + 1)^2$, поскольку в этом случае минимума в $w_1 = 0$, $w_2 = -1$ не будет (на самом деле у представленного полинома вообще нет конечных минимумов). Обязательно надо сначала возводить w_1 в квадрат, а уже затем брать корень — тогда $w_1 = 0$, $w_2 = -1$ оказывается экстремалью (и единственной стационарной) точкой $\mathscr{F}$. При этом, однако, матрица Гессе функции $\mathscr{F}$ равна

$$\nabla^2 \mathscr{F}(w) = \begin{pmatrix} 20(w_1^2)^{3/2} & 0 \\ 0 & 2 \end{pmatrix},$$

т. е. в $(0, -1)^T$ вырождена. Последнее отрицательно отразится на скорости сходимости любого алгоритма безусловной минимизации, предназначенного для функций с гладкими производными (см. разд. 8.3.1). Если же не преобразовывать исходную задачу, а применить к ней метод минимизации при простых ограничениях, никаких трудностей не возникает, так как переменная x_1 будет зафиксирована на граничном значении и соответственно исключена из спроектированных градиентов и матриц Гессе.

7.4.1.2. Исключение неравенств. В настоящее время общепринята точка зрения, что задачи с ограничениями-неравенствами тяжелее задач с равенствами. Ее трудно оспаривать, но отсюда вовсе не следует, что искусственным преобразованием неравенств в равенства всегда можно упростить дело. К примеру, популярная замена ограничений вида $c_i(x)\geqslant 0$ на $c_i(x)-y_i^2=0$, где y_i — специально вводимая вспомогательная переменная, вряд ли помогла многим, так как обладает теми же недостатками, что и преобразование, рассмотренное в предыдущем разделе. Если уж заменять неравенства равенствами, то лучше всего делать это введением новой переменной, подчиняемой условию неотрицательности, т. е. вместо $c_i(x)\geqslant 0$ использовать пару ограничений $c_i(x)-y_i=0$, $y_i\geqslant 0$.

7.4.1.3. Общие трудности с преобразованиями. В ряде случаев переход с помощью каких-то преобразований к задаче минимизации без ограничений или с простыми ограничениями существенно помогает поиску решения. Иногда для этого достаточно разумно выбрать независимые переменные модели. Однако далеко не всякая замена переменных, снимающая или упрощающая ограничения, дает задачу, решать которую будет легче, чем исходную. В частности, некоторые замены чреваты следующими неприятностями:

(a) по недосмотру утрачивается свойство экстремальности искомой точки;

(b) существенно возрастает степень нелинейности целевой функции;

(c) ухудшается обусловленность задачи;

(d) у новой целевой функции появляются особенности, которых не было у исходной;

(e) теряется гладкость;

(f) матрица Гессе получается вырожденной или плохо обусловленной в решении;

(g) возникают дополнительные стационарные и локально экстремальные точки;

(h) зависимость целевой функции от новых переменных оказывается периодической.

Универсальные рекомендации в отношении того, как избегать перечисленных трудностей, сформулировать непросто. Впрочем, на основании своего опыта можем сказать, что осложнения чаще всего возникают при тригонометрических и экспоненциальных преобразованиях, и особенно когда число переменных велико.

Сгладить неприятные последствия, связанные с периодичностью преобразованной целевой функции, можно двумя способами. Первый — модифицировать применяемый к ней алгоритм безусловной минимизации: если в новых переменных $\{y_i\}$

$$F(y+j\alpha_i e_i)=F(y), \quad j=\pm 1,\ \pm 2,\ \ldots,$$

где e_i есть i-й единичный вектор и немодифицированный алгоритм для очередной точки y дает шаг p, то в качестве следующего значе-

ния i-й переменной брать $y_i+\bar{p}_i$ с p_i, получающимся увеличением или уменьшением p_i на величину, кратную α_i и подбираемую так, чтобы было $|\bar{p}_i|<\alpha_i$. Второй способ — вводить подходящие простые ограничения. Например, если y_i имеет смысл некоторого угла, то естественно потребовать соблюдения неравенств $0\leqslant y_i\leqslant 2\pi$ и использовать для минимизации алгоритм типа описанного в разд. 5.5.1.

Другие трудности, которые могут возникать при нелинейных заменах переменных, проиллюстрированы на следующем простом примере.

Пример 7.4. Рассмотрим задачу

$$\text{найти } \min_{x\in\mathbb{R}^n} F(x) \tag{7.3a}$$

$$\text{при ограничении } \sum_{i=1}^{n} x_i^2=1. \tag{7.3b}$$

Если $n>1$, определено преобразование

$$x_1=\sin y_1 \dots \sin y_{n-1},$$
$$x_i=\cos y_{i-1}\sin y_i \dots \sin y_{n-1},\ i=2,\ \dots,\ n-1,$$
$$x_n=\cos y_{n-1}.$$

Полученные по этим формулам x_i удовлетворяют (7.3b) автоматически (при любых y_i). С другой стороны, каждый допустимый для (7.3b) вектор x выражается через y. Таким образом, предлагаемая замена переменных дает эквивалентную (7.3) задачу вида

$$\text{найти } \min_{y\in\mathbb{R}^{n-1}} \bar{F}(y), \tag{7.4}$$

т. е. позволяет избавиться от нелинейного ограничения. Однако у (7.4) есть нежелательные свойства. Помимо очевидной периодичности новая целевая функция плоха тем, что при близком к нулю значении y_i ($i>1$) практически не зависит от $y_1,\ \dots,\ y_{i-1}$; стало быть, задача (7.4) оказывается очень плохо отмасштабированной.

Иной способ автоматически удовлетворить ограничению (7.3b) — воспользоваться таким преобразованием:

$$a=\pm\left(1+\sum_{i=1}^{n-1} y_i^2\right)^{1/2}, \tag{7.5}$$

$$x_i=\frac{y_i}{a},\quad i=1,\ \dots,\ n-1 \tag{7.6}$$

и

$$x_n=\frac{1}{a}. \tag{7.7}$$

Оно приводит к задаче вида

$$\text{найти } \min_{y\in\mathbb{R}^{n-1}} \min\{\bar{F}_P(y),\ \bar{F}_N(y)\},$$

где $\bar{F}_P(y)$ — функция, отвечающая верхнему знаку в (7.5), а $\bar{F}_N(y)$ — нижнему.

Если заранее известно, что некоторая переменная x_i из (7.3) в решении будет иметь определенный знак, то замена ее, а не x_n по аналогичной (7.7) формуле позволит избавиться от необходимости введения двух функций. Среди нескольких x_i известного знака «в роли x_n» лучше использовать ту, оптимальное значение которой сильнее отличается от нуля: при близкой к нулю правой части (7.7) новые переменные будут плохо отмасштабированы.

7.4.1.4. Тригонометрические преобразования. Несмотря на свои недостатки, замены переменных с использованием тригонометрических функций иногда все же целесообразны. Рассмотрим, например, задачу, в которой надо оптимизировать расположение k точек трехмерного пространства на сферической поверхности заданного радиуса. Формально она ставится так:

$$\begin{aligned} &\text{найти} \quad \min_{x,\ y,\ z \in R^k} \bar{F}(x, y, z) \\ &\text{при ограничениях } x_i^2 + y_i^2 + z_i^2 = r^2, \quad i = 1, 2, \ldots, k. \end{aligned} \tag{7.8}$$

В ней $3k$ переменных и k ограничений.

Правильное исключение l независимых ограничений-равенств из задачи с n неизвестными должно приводить к понижению размерности до $(n-l)$ (см. разд. 6.3). В частности, чтобы обеспечить автоматическое соблюдение (7.8), достаточно перейти к $2k$ угловым координатам $\{\theta_i, \psi_i\}$, связанным с исходными следующим образом:

$$x_i = r \sin\theta_i \cos\psi_i, \quad y_i = r \sin\theta_i \sin\psi_i, \quad z_i = r \cos\theta_i. \tag{7.9}$$

Для нейтрализации неприятных следствий периодичности получающейся в результате целевой функции, ее лучше минимизировать при простых ограничениях $0 \leqslant \theta_i \leqslant 2\pi$ и $0 \leqslant \psi_i \leqslant 2\pi$. В конкретной задаче эти ограничения, возможно, удастся усилить из каких-то априорных соображений. Всегда надо стремиться к тому, чтобы внизу границы были *как можно ближе к верхним*.

Переход к угловым координатам полезен и тогда, когда минимизацию требуется осуществлять на шаре радиуса r. В данном случае можно ввести k вспомогательных переменных $\{d_i\}$, имеющих смысл расстояний точек от нуля, что позволит записать условия их принадлежности шару парами соотношений вида

$$\begin{aligned} &x_i^2 + y_i^2 + z_i^2 = d_i^2, \\ &0 \leqslant d_i \leqslant r. \end{aligned} \tag{7.10}$$

Верхние удовлетворяются заменой переменных $x_i = d_i \sin\theta_i \cos\psi_i$, $y_i = d_i \sin\theta_i \sin\psi_i$, $z_i = d_i \cos\theta_i$, т. е. в координатах $\{d_i, \theta_i, \psi_i\}$ задача сводится к минимизации при простых ограничениях на $\{d_i\}$.

Если преобразование дает возможность снять не все, а только некоторые нелинейные ограничения, то скорее всего прибегать к нему не стоит. Однако нет правила без исключений: сказанное не относится к большим задачам, для которых цепой каких-то замен удается автоматически удовлетворить условиям, включающим много переменных, и тем самым добиться, чтобы матрица Якоби функций ограничений стала разреженной.

7.4.2. Задачи с функциональными переменными

Важный класс оптимизационных задач, непосредственно не вписывающихся в конечномерную постановку задачи NCP, образуют задачи поиска экстремума на *функциональных* множествах. Например, может потребоваться найти минимум интеграла

$$\int_a^b f(x(t),\ t)\,dt \tag{7.11}$$

на пространстве гладких функций $x(t)$ аргумента t из отрезка $[a, b]$. Во многих случаях подобные задачи удается «сводить» к конечномерным; принцип соответствующих преобразований мы проиллюстрируем на (7.11).

Чтобы обрабатывать на машине функции $x(t)$, их надо представлять конечными наборами данных, причем ясно, что хранить значения $x(t)$ в огромном числе всех машинно-представимых точек отрезка определения нереально. Значит, нужен способ генерирования удовлетворительного приближения $x(t)$ по информации разумного объема. Последнюю можно интерпретировать как определение новой функции $\tilde{x}(t)$, получающейся какой-то интерполяцией $x(t)$ в незатабулированных точках. Близость $\tilde{x}$ к x будет зависеть от гладкости x и $\tilde{x}$, числа и расположения точек интерполяции и т. д.

Итак, под удовлетворительным решением задачи минимизации интеграла (7.11) будем понимать некую функцию $\bar{x}(t)$. Пусть все $\bar{x}$ строятся в виде

$$\bar{x}(t) = \sum_{j=1}^{q} c_j w_j(t), \tag{7.12}$$

где $\{c_j\}$ — набор чисел, а $\{w_j(t)\}$ — комплект так называемых базисных функций; в качестве w_j часто используют (i) степенные функции $w_j = t^{j-1}$; (ii) полиномы Чебышева: $w_j = T_{j-1}(t)$; (iii) B-сплайны: $w_j = M_j(t)$. Тогда, подставив правую часть (7.12) вместо x в (7.11), мы приходим к конечномерной задаче с неизвестными $\{c_j\}$. В зависимости от структуры f интеграл (7.11) либо можно будет взять аналитически, либо при каждом наборе $\{c_j\}$ придется оценивать численно (соображения по поводу использования численных методов интегрирования приведены в разд. 7.3.3).

Замечания и избранная библиография к разделу 7.4

Детальный анализ квадратичной замены переменных проведен Сиссером (1981). Использование кусочнополиномиальной интерполяции для решения (квадратичных) функциональных задач очень подробно обсуждается в работе Сьярле, Шульца и Варги (1967); см. также Гилл и Мюррей (1973а). Определения полиномов Чебышева и B-сплайнов даны, например, у Кокса (1977).

7.5. МАСШТАБИРОВАНИЕ

Есть ряд давно осознанных вычислительных проблем, говоря о которых неизменно употребляют термин «масштабирование». Тем не менее четкого определения этого термина нет. Поэтому не удивительно, что с масштабированием связано немало путаницы и что в публикациях его стараются не касаться.

В данном разделе мы обсудим лишь простые замены переменных и специальный прием масштабирования для нелинейных задач о наименьших квадратах. Более подробно вопросы масштабирования рассмотрены в разд. 8.7. Там даны рекомендации как для задач на безусловный экстремум, так и для задач с ограничениями.

7.5.1. МАСШТАБИРОВАНИЕ ЗАМЕНОЙ ПЕРЕМЕННЫХ

Масштабирование заменой переменной состоит в переходе от первичных единиц измерения, обычно отражающих физическую суть задачи, к новым, более подходящим для метода ее решения. Этот прием не следует смешивать с преобразованиями, которые обсуждались в разд. 7.4.1 и изменяли тип задачи.

Первая основная цель масштабирования — добиться, чтобы в представляющей интерес области модули значений преобразованных переменных имели одинаковые порядки. Дело в том, что в оптимизирующих программах допуски на нарушения условий экстремума в правилах останова и другие критерии по необходимости используют неявные определения понятий «мало», «много» и, когда из-за большого разброса модулей переменных эти определения утрачивают универсальность, возможны осложнения.

Возьмем, к примеру, задачу (оптимизации теплообменника), некоторые из переменных которой перечислены с указанием единиц измерения и характерных значений в табл. 7а. Последние имеют сильно различающиеся порядки, и в этом нет ничего удивительного, так как рассматриваются разные физические величины. Из сопоставления третьего и четвертого чисел видно, что большой разброс могут иметь и переменные одинаковой размерности.

Как правило, для масштабирования следует применять линейные замены переменных (хотя иногда допустимы и нелинейные).

Чаще всего обращаются к заменам вида

$$x=Dy,$$

где $\{x_j\}$ — исходные переменные, $\{y_j\}$ — преобразованные, D — диагональная матрица. В качестве ее диагональных элементов используют характерные значения модулей соответствующих переменных. Для задачи, к которой относится табл. 7а, при таком подходе d_1 надо взять равным 1.1×10^3.

Таблица 7а. Характерные значения неотмасштабированных переменных

Переменная	Интерпретация	Единицы измерения	Характерное значение
x_1	Поток газа	фунт/ч	11000
x_2	Поток воды	фунт/ч	1675
x_3	Термическое сопротивление пара	$(\text{БТЕ}/(\text{ч}\cdot\text{фут}^2\cdot{}^\circ\text{F}))^{-1}$	100
x_4	Потери	$(\text{БТЕ}/(\text{ч}\cdot\text{фут}^2\cdot{}^\circ\text{F}))^{-1}$	6×10^{-4}
x_5	Излучение от газа	$\text{БТЕ}/(\text{ч}\cdot\text{фут}^2\cdot{}^\circ\text{R}^4)$	5.4×10^{-10}

К сожалению, указанное преобразование иногда может приводить к специфическим потерям точности. Допустим, например, что исходная переменная x_j должна лежать в диапазоне [200.1242, 200.1806]. Тогда, масштабируя ее по характерному значению 200.1242, для преобразованной переменной получим диапазон, который с точностью до семи значащих цифр равен [1.0, 1.0000282]. Если мантисса представления числа на машине содержит семь десятичных разрядов, именно это приближение будет использовано в вычислениях. В нем область допустимых значений новой переменной y_j отражена с точностью до трех значащих цифр. В то же время диапазон для x_j представим без погрешностей.

Масштабирование диагональным преобразованием ненадежно и тогда, когда в процессе минимизации модули переменных существенно изменяются. В этом случае те d_j, которые естественно взять в одной точке, могут совершенно не годиться в другой.

Если известен вероятный *диапазон* значений переменной, обе упомянутые выше трудности преодолеваются одним приемом. Допустим, каким-то образом установлено, что x_j в течении всей процедуры решения задачи будет удовлетворять неравенствам $a_j\leqslant x_j\leqslant b_j$. Тогда новую переменную y_j можно ввести по формуле

$$y_j=\frac{2x_j}{b_j-a_j}-\frac{a_j+b_j}{b_j-a_j}. \tag{7.17}$$

В матричных обозначениях (7.17) записывается так:

$$x=Dy+c, \tag{7.18}$$

где D — диагональная матрица, чей j-й диагональный элемент равен $1/2(b_j - a_j)$, а c — вектор, составленный из полусумм $1/2(a_j + b_j)$. Гарантировано, что, какова бы ни была величина x_j из $[a_j, b_j]$, отвечающее ей в силу (7.18) значение y_j не выйдет из диапазона $-1 \leqslant y_j \leqslant +1$. В описанном выше примере *с потерей точности* формула (7.17) принимает следующий вид:

$$x_j = 0.0282 y_j + 200.1524.$$

При этом диапазон для y_j, соответствующий отрезку [200.1242, 200.1806], представляется ограничениями $-1 \leqslant y_j \leqslant +1$ без погрешностей.

Мы хотим подчеркнуть, что преобразование (7.17) применимо, только если известны хорошие приближения крайних значений x_j. Имея лишь очень грубые оценки последних (т. е. оценки, возможно отличающиеся от настоящих границ на несколько порядков), использовать его нельзя ни в коем случае.

При переходе от x к y по формуле (7.18) наряду с переменными масштабируются и производные. Пусть g, G и g_y, G_y — градиенты и матрицы Гессе до и после преобразования соответственно. Тогда $g_y = Dg$, $G_y = DGD$. Отсюда видно, что даже «умеренные» изменения масштабов переменных типа $x_j = 10 y_j$ сильно отражаются на вторых производных и потому могут существенно влиять на скорость сходимости оптимизационного процесса.

7.5.2. МАСШТАБИРОВАНИЕ В НЕЛИНЕЙНЫХ ЗАДАЧАХ О НАИМЕНЬШИХ КВАДРАТАХ

Нелинейные задачи о наименьших квадратах чаще всего возникают, когда нужно подобрать вектор параметров x модельной функции $y(x, t)$, обеспечивающей наилучшее совпадение ее значений в выделенных точках $\{t_i\}$ с наблюдаемыми в них значениями $\{y_i\}$ реальной величины, зависимость которой от t моделируется. Методы решения задач данного класса обсуждались в разд. 4.7.

Многие задачи о наименьших квадратах характерны тем, что масштабы их переменных автоматически определяются выбором масштаба независимой величины t. Как это происходит, мы покажем на примере упрощенной формулировки некоторой реальной задачи (сохранив ее специфические обозначения). Эта формулировка звучит так: требуется подобрать параметры $\{A_j\}$, $j = 0, \ldots, J$, и $\{B_k, \bar{p}_k, \sigma_k\}$, $k = 1, \ldots, K$, функции

$$y(p) = \sum_{j=0}^{J} A_j p^j + \sum_{k=1}^{K} B_k \exp\left(-\frac{(p - \bar{p}_k)^2}{2\sigma_k^2}\right), \qquad (7.19)$$

обеспечивающие минимум суммы

$$\sum_{i=1}^{m} \left(\frac{y(p_i) - Y_i}{\Delta y_i}\right)^2,$$

где величины $\{Y_i\}$ и $\{\Delta y_i\}$ фиксированы. Речь идет о настройке на данные эксперимента комбинации K гауссовых кривых и полиномиального фона (первого слагаемого в правой части (7.19)). Опорные точки и результаты подгонки параметров (7.19) при $K=4$ показаны на рис. 7с; все p_k (оценки координат пиков) лежат в диапазоне [566, 576].

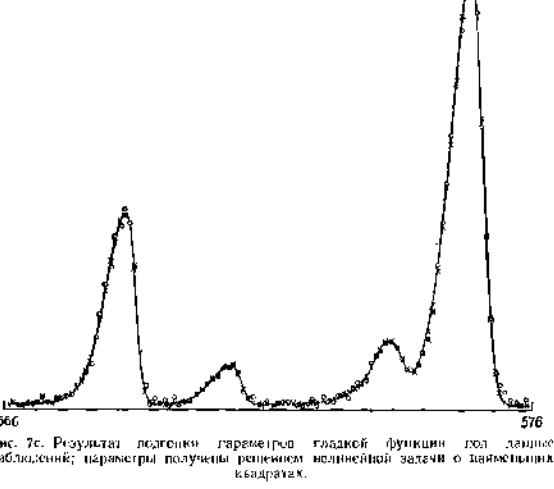

Рис. 7с. Результат подгонки параметров гладкой функции под данные наблюдений; параметры получены решением нелинейной задачи о наименьших квадратах.

При решении поставленной задачи без предварительного масштабирования неизбежны большие трудности, так как даже для умеренных j коэффициенты при A_j оказываются огромными и соответственно оптимальные A_j должны получаться очень близкими к нулю. Скажем, при $j=3$ переменная A_3 в каждой компоненте $y(p_i)$ минимизируемой суммы умножается на число, нижней оценкой которого служит $566^3 \approx 10^8$. Проблема частично снимается, если заменить p_i на z_i, связанные с p_i формулой $p_i = 576 z_i$. Тогда модули новых A_j будут больше модулей старых в 576^j раз. Однако при таком изменении масштаба независимой величины p_i, как и в случае, разобранном в предыдущем разделе, неоправданно теряется точность представления диапазона ее значений. Чтобы этого не произо-

шло, лучше воспользоваться преобразованием $p-5z\,|\,571$, при котором все z_i также ложатся на отрезок $[-1, +1]$. В преобразованной задаче место $y(p)$ займет функция

$$\Phi(z) \equiv \sum_{i=0}^{J} \overline{A}_j z^j + \sum_{k=1}^{K} B_k \exp\left(\frac{-(z-\overline{z}_k)^2}{2\overline{\sigma}_k^2}\right).$$

После того как оптимальные $\{\overline{A}_j\}$, $\{\overline{z}_k\}$ и $\{\overline{\sigma}_k\}$ найдены, решение исходной задачи (т. е. оптимальные $\{A_j\}$, $\{\overline{p}_k\}$ и $\{\sigma_k\}$ для функции (7.19)) вычисляется по несложным формулам. Впрочем, если сами по себе $\{A_j\}$, $\{\overline{p}_k\}$ и $\{\sigma_k\}$ не нужны, такие вычисления не понадобятся, поскольку значения функции $y(p)$ можно (и лучше) определять по $\Phi(z)$ на основании равенства $y(p) = \Phi((p-571)/5)$.

7.6. ПОСТАНОВКА ОГРАНИЧЕНИЙ

7.6.1. ВЫРОЖДЕНИЕ

Неудачная формулировка модели с ограничениями на переменные может привести к плохо поставленной оптимизационной задаче даже тогда, когда, по существу, решение хорошо определено. Многообразие совершаемых оплошностей такого сорта довольно обширно и даже слишком обширно, чтобы перечислять их в этой книге. К примеру, нередко вводят избыточные ограничения, которые представимы линейными комбинациями основных и предназначаются только для того, чтобы заставить алгоритм автоматически вычислять какие-то сводные показатели. Пользователь знает, что решение от этого не изменится, и полагает, что на процессе поиска это тоже не отразится. Однако данное предположение оправдывается далеко не всегда.

Часто неприятности возникают, когда между переменными задачи и характеристиками изучаемого явления нет прямого соответствия. Это положение мы проиллюстрируем примером из собственной практики. Рассматривалась статистическая модель заболеваемости раком легких. Основой исходного описания служили трехиндексные вероятности $\{p_{ijk}\}$, $i=1, \ldots, I$; $j=1, \ldots, J$; $k=1, \ldots, K$, и минимизируемая функция зависела только от них. При обсуждении одной из постановок было решено выражать $\{p_{ijk}\}$ произведениями трех двухиндексных вероятностей, т. е. пользоваться представлением

$$p_{ijk} = f_{ij} g_{jk} h_{ik}. \tag{7.20}$$

Для $\{f_{ij}\}$, $\{g_{jk}\}$ и $\{h_{ik}\}$ удалось выписать разумные ограничения, и тем самым была поставлена нелинейная задача на условный экстремум с $\{f_{ij}\}$, $\{g_{jk}\}$, $\{h_{ik}\}$ в качестве переменных.

Некорректность постановки вскрылась после того, как в численных экспериментах выявились систематически серьезные за-

трудности со сходимостью. Они оказались настоящим сюрпризом, поскольку применялся весьма эффективный алгоритм, от которого ждали, что он будет сходиться квадратично. Чтобы определить источник затруднений, одну и ту же задачу решили с нескольких начальных приближений: всякий раз получался новый набор $\{f_i\}$, $\{g_{jk}\}$ и $\{h_{lm}\}$, но все комплекты $\{p_{ijk}\}$ оказались одинаковыми. Это заставило внимательно проанализировать формулировку задачи, и выяснилось, что она сильно вырождена. Из (7.20) видно, что величина p_{ijk} не изменится, если, например, f_{ij} и g_{jk} заменить на αf_{ij} и g_{jk}/α, где α — некоторое ненулевое число. Оказалось, что ограничения тоже инвариантны к подобным заменам. Поэтому все системы уравнений для расчета направлений поиска получались линейно зависимыми, и не будь используемый алгоритм так надежен, он вообще не справился бы с задачей. После того как все это было обнаружено, вырождение легко было устранено дополнительными нормировочными условиями типа $\sum_k g_{jk}=1$.

Описанная история не столь уж необычна, как может показаться на первый взгляд, и служит еще одним подтверждением тезиса о целесообразности интерактивного объединения этапов моделирования и оптимизации. Второй общий вывод таков: всегда надо стремиться, чтобы решение формализованной задачи было столь же хорошо определено, как и оптимум порождающей ее содержательной постановки.

7.6.2. ИСПОЛЬЗОВАНИЕ ОГРАНИЧЕНИЙ С ДОПУСКАМИ

Ограничения-равенства в оптимизационных задачах бывают разного происхождения. Нередко они вытекают из самого смысла переменных; например, если $\{x_i\}$ суть какие-то доли или вероятности, то по определению должно быть $\sum_i x_i=1$. В таких случаях говорят об «истинных» равенствах, подчеркивая тем самым, что в искомом численном решении их надо соблюсти с максимально возможной точностью. Однако иногда в виде равенств формулируют ограничения, которые можно было бы ставить и не столь жестко, т. е., по существу, эти равенства точно выполнять не обязательно (скажем, известно, что описываемые связи включают некую неопределенность). Фактически имп заменяют «узкополосные» двусторонние неравенства, которые принято называть *ограничениями с допусками*. Так, например, требуют, чтобы было

$$a^T x=b,$$

хотя в действительности достаточно удовлетворить условиям

$$b-\varepsilon_1\leqslant a^T x\leqslant b+\varepsilon_2, \qquad (7.21)$$

где ε_1 и ε_2 — малые, но все же ощутимо отличные от нуля положительные величины (сказанное применимо и к нелинейному случаю).

Замена ограничений с допусками типа (7.21) соответствующими равенствами в некоторых случаях может приводить к несовместности или к сильному изменению решения: при плохой обусловленности системы ограничений переход от приближенных равенств к точным влечет существенное сокращение допустимого множества.

Приведенные соображения и некоторые полезные преобразования мы проиллюстрируем на примере несколько упрощенной (ради наглядности) постановки задачи, которая возникла при проектировании конвертера выхлопных газов автомобильного двигателя. Требовалось решать систему уравнений, описывающих химические реакции в конвертере. Она распадается на две подсистемы: нелинейные уравнения для скоростей реакций и линейные уравнения балансов масс. Всего рассмотрим 13 уравнений (восемь нелинейных и пять линейных).

Переменными $\{x_1, \ldots, x_8\}$ являются парциальные давления следующих газов: пропана, угарного газа, окиси азота, окиси углерода, кислорода, водяного пара, азота и водорода. Естественно, что они должны быть неотрицательными, т. е. в разумном решении должно получаться $x_i \geqslant 0$, $i = 1, \ldots, 8$. Восемь нелинейных уравнений с параметрами $\{K_1, \ldots, K_8\}$, имеющими смысл констант реакций, таковы:

$$\frac{x_1^2 x_5^5}{x_4^3 x_7^{10}} - K_1 = f_1(x) = 0, \tag{7.22}$$

$$\frac{x_4^8 x_6^9}{x_1 x_5^{12}} - K_2 = f_2(x) = 0, \tag{7.23}$$

$$\frac{x_3^9 x_8^{10}}{x_1 x_5^{13}} - K_3 = f_3(x) = 0, \tag{7.24}$$

$$\frac{x_4 \sqrt{x_7}}{x_2 x_3} - K_4 = f_4(x) = 0, \tag{7.25}$$

$$\frac{x_4 x_8}{x_2 x_6} - K_5 = f_5(x) = 0, \tag{7.26}$$

$$\frac{x_6 \sqrt{x_7}}{x_3 x_8} - K_6 = f_6(x) = 0, \tag{7.27}$$

$$\frac{x_4}{x_5 \sqrt{x_7}} - K_7 = f_7(x) = 0, \tag{7.28}$$

$$\frac{x_4}{x_2 \sqrt{x_7}} - K_8 = f_8(x) = 0. \tag{7.29}$$

Линейные балансовые уравнения с параметрами $\{a_1, \ldots, a_5\}$, представляющими собой парциальные давления реагентов на выходе из двигателя, выглядят следующим образом:

Баланс кислорода

$$x_2 + x_3 + 2x_4 + 2x_5 + x_6 - a_2 - a_3 - 2a_5 - a_8 = f_9(x) = 0. \tag{7.30}$$

Баланс углерода

$$3x_1 + x_2 + x_4 - 3a_1 - a_2 - a_4 = f_{10}(x) = 0. \tag{7.31}$$

Баланс водорода

$$8x_1 + 2x_2 + x_5 - 8a_1 - 2a_2 - a_5 = f_{11}(x) = 0. \qquad (7.32)$$

Баланс азота

$$x_3 + 2x_7 - a_3 - 2a_7 = f_{12}(x) = 0. \qquad (7.33)$$

Общий баланс

$$\sum_{i=1}^{8} x_i - \sum_{i=1}^{8} a_i = f_{13}(x) = 0. \qquad (7.34)$$

Обычный метод решения переопределенной системы уравнений — искать минимум суммы квадратов их невязок. В нашем случае он приводит к задаче

$$\text{найти } \min_{x \in R^8} \sum_{i=1}^{13} f_i^2(x), \qquad (7.35)$$

в которой разнородные уравнения (7.22)—(7.34) «свалены в одну кучу», и это не хорошо. Более естественно разделить их, например заменить (7.35) задачей

$$\text{найти } \min_{x \in R^8} \sum_{i=1}^{8} f_i^2(x) \qquad (7.36)$$

$$\text{при ограничениях } f_i(x) = 0,\ i = 9, \ldots, 13.$$

Однако и ее нельзя признать удачной. Дело в том, что восемью описанными реакциями отнюдь не исчерпывается их полный список (около 30 второстепенных процессов отброшены), и поэтому настаивать на точном соблюдении равенств (7.30)—(7.34) рискованно: можно получить бессмысленное решение с отрицательными компонентами. Чтобы избежать этого, следовало бы вместо (7.30)—(7.34) поставить соответствующие ограничения с допусками, отражающими «материалоемкость» неучтенных процессов.

В некоторых случаях ослабление ограничений-равенств дает удовлетворительную постановку задачи и больше никаких преобразований не требуется. Рассматриваемый же пример сложнее и требует дополнительного масштабирования. Прежде всего оно необходимо из-за огромного разброса констант реакций $\{K_i\}$ (в частности, K_1 имеет порядок 10^{250}). Один из приемов, который может помочь,— заменить $f_i(x)$, $i = 1, \ldots, 8$, функциями

$$F_i = \ln(f_i(x) + K_i) - \ln K_i$$

и минимизировать сумму $\sum_{i=1}^{8} F_i^2(x)$. Однако всех проблем это преобразование не решает. Остается еще как-то избавиться от чрезмерной чувствительности целевой функции к малым вариациям переменных в окрестности нуля — ведь оно сильно меняется, скажем, при переходе от $x_1 = 10^{-14}$ к $x_1 = 10^{-20}$, в то время как для любого стандартного алгоритма такие значения наверняка будут «эквивалентными».

Чтобы получить менее чувствительную функцию, необходимо сделать нелинейную замену переменных, т. е. приходится терять линейность ограничений (7.30)—(7.34). К счастью, эту утрату удается скомпенсировать тем, что линейными станут функции $\hat{F}_i$. Для этого надо взять

$$x_i = e^{y_i}. \tag{7.37}$$

Заметим, что попутно автоматически обеспечивается неотрицательность x_i.

Проиллюстрируем эффект замены (7.37) на функциях F_8 и f_{11}. Первая преобразуется в

$$\hat{F}_8(y) = \frac{1}{2}\, y_7 + y_6 - y_3 - y_2 - \ln K_8,$$

а вторая в новых переменных будет выглядеть так:

$$\hat{f}_{11}(y) = 8e^{y_6} + 2e^{y_4} + e^{y_1} - b_3 = 0.$$

Аналогичная (7.36) задача теперь формулируется следующим образом:

$$\begin{aligned} &\text{найти } \min_{y \in R^8} \sum_{i=9}^{18} \hat{f}_i^2(y) \\ &\text{при ограничениях } \hat{F}_i(y) = 0,\ i = 1, \ldots, 8. \end{aligned} \tag{7.38}$$

Важно подчеркнуть, что в решении (7.38) (если оно существует) $\hat{f}_i(y)$, вообще говоря, будут ненулевыми, и поэтому, если требуется точное соблюдение равенств (7.30)—(7.34), такая постановка не подходит.

Ограничения задачи (7.38) представляют собой систему из восьми линейных уравнений с восьмью неизвестными, причем ее матрица вырождена. На совместность таких уравнений при оценивавемых экспериментально и, следовательно, неточных константах $\{K_i\}$ рассчитывать нельзя. Поэтому их надо заменить ограничениями с допусками, подбираемыми в соответствии с оценками погрешностей измерения $\{K_i\}$. Интересно отметить, что при «слегка» ошибочных $\{K_i\}$ уравнения в (7.38) тоже будут лишь «слегка» несовместными.

Замечания и избранная библиография к разделу 7.6

Оптимизационная задача, возникающая из статистической модели заболеваемости раком легких, была предложена нам профессором Стэнфордского университета А. Уинтмором.

Литература по применению методов оптимизации в конкретных исследованиях слишком обширна, чтобы сколько-нибудь полно цитировать ее здесь. Выборочные сведения читатель найдет у Брэкена и Мак-Кормика (1968), а также в сборниках конспектов статей

под редакцией Аврицля, Рейкарта и Уайлда (1973), Балинского и Лемарешаля (1978), Аврицля и Дембо (1979). Вопросы численного моделирования, возникающие в контексте технической оптимизации, хорошо изложены в книге Уайлда (1978).

7.7. ЗАДАЧИ С ДИСКРЕТНЫМИ И ЦЕЛОЧИСЛЕННЫМИ ПЕРЕМЕННЫМИ

Среди прикладных задач нередко встречаются такие, в которых наряду с условиями вида равенств и неравенств ставится требование, чтобы значения некоторых (или всех) переменных принадлежали конечным множествам. Эти переменные называют *дискретными* или *целочисленными*. Примеров можно привести сколько угодно; в частности, x может иметь смысл характеристики конструкции (типа мощности насоса, размера балки и т. д.), выпускаемой в определенной номенклатуре, или служить каким-то счетчиком (скажем, числа поездок коммивояжера).

Непосредственным применением стандартных методов минимизации к задаче с дискретными переменными ее, как правило, не решить (исключение составляет ряд особых случаев, когда оптимальные значения переменных, освобожденных от требования целочисленности, удовлетворяют ему автоматически). Здесь используется своя техника. Наиболее развит аппарат решения целочисленных задач, целевые функции и функции ограничений которых линейны; большая часть соответствующих методов опирается на схему «ветвей и границ». Для некоторых других специальных задач удается успешно применять методы динамического программирования. Мы, однако, не будем касаться ни первых, ни вторых, а ограничимся кратким обсуждением общего случая с нелинейными ограничениями и смесью непрерывных дискретных переменных.

Существует два типа дискретности, и их важно различать, поскольку каждому соответствует свой класс предпочтительных алгоритмов. Мы обсудим эти типы на примере двух конкретных задач.

7.7.1. ПСЕВДОДИСКРЕТНЫЕ ПЕРЕМЕННЫЕ

Рассмотрим задачу проектирования городской сети сброса дождевых вод. Содержательная постановка такова: на некой площади известным образом размещены сточные люки (размещение диктуется потребностями доступа и географией улиц); они связываются между собой прямолинейными трубами; к каждому люку подходит несколько входных труб (идущих от вышерасположенных точек) и одна выходная; никаких насосов не предусматривается, т. е. вода будет течь только под действием силы тяжести. Надо подобрать характеристики сети так, чтобы обеспечить ей опреде-

ленную пропускную способность и при этом условии минимизировать затраты на ее строительство.

Перейдем теперь к описанию математической постановки задачи. Переменными являются диаметры трубопроводов и углы их наклонов к горизонтали. Ограничения таковы: угол наклона каждого трубопровода должен лежать в заданных пределах (определяемых в соответствии с топографией поверхности и желаемой пропускной способностью); при движении «сверху вниз» вдоль каждой ветви сети диаметр последующего трубопровода должен быть не меньше диаметра предыдущего; интенсивности потоков в трубопроводах при некоторой расчетной «стационарной» нагрузке на сеть (нелинейные функции диаметров и углов) должны иметь значения из фиксированных диапазонов. При этих ограничениях минимизируется сумма стоимостей труб и затрат на рытье траншей (прочие издержки строительства относительно малы, и ими можно пренебречь). Следует отметить, что указанные стоимости и затраты являются взаимодополняющими: чем меньше диаметры труб, тем они дешевле, но тем большие углы и соответственно тем более глубокие траншеи понадобятся, чтобы обеспечить требуемую пропускную способность.

Казалось бы, поставленная задача относится к классу нелинейных задач на условный экстремум с непрерывными переменными. Однако на самом деле это не так: промышленность производит трубы определенных диаметров, и поэтому значения отвечающих им переменных должны выбираться из конечного множества. В данном случае можно говорить о *псевдодискретности* – задача имеет вполне осмысленное решение и в непрерывной постановке. Просто оно не реализуемо по чисто внешним причинам.

Методы, применяемые к задачам с псевдодискретными переменными, исходят из предположения, что требование дискретности не сильно сказывается на решении. В частности, в примере с проектированием сточной сети скорее всего так и будет (поскольку номенклатура выпускаемых труб достаточно широка). Близость же «непрерывного» оптимума к «дискретному» позволяет опереться на стандартный аппарат минимизации.

Поиск x^* в задаче с псевдодискретными переменными можно начать с решения ее «непрерывного аналога». Это даст точку x^c, для которой $F(x^c) \leqslant F(x^*)$. Если значение x_1 в действительности должно выбираться среди $d_1, d_2, \ldots, d_r$ и $d_s < x_1^c < d_{s+1}$, то затем естественно зафиксировать x_1 на d_s или d_{s+1} (обычно берут ближайшую к x_1^c величину), найти минимум F по переменным $x_2, \ldots, x_m$ считая их непрерывными, выбрать значение следующей дискретной переменной и т. д. При этом каждая последующая минимизация будет требовать меньше усилий, чем предыдущая (хотя бы потому, что понижается размерность). В конце концов будут зафиксированы значения всех дискретных переменных и определится некая допустимая точка исходной задачи x^r.

Ее оптимальность описанной процедурой не гарантируется, но, если $F(x')$ и $F(x'')$ окажутся «близкими», x'' можно принять в качестве «удовлетворительного решения». В противном случае разумно попытаться найти лучшую точку; для этого просматривают альтернативные значения некоторых из дискретных переменных, причем обычно варьируют те переменные, чьи недопустимые «непрерывные» значения оказались примерно посередине между допустимыми дискретными. Для многих прикладных задач описанный подход быстро приводит к успеху.

В некоторых случаях фиксация одной дискретной переменной сильно сужает возможность варьирования других. К примеру, в задаче о сети водосточных труб этот эффект возникает благодаря условию неубывания диаметров вдоль каждой ветви. Поэтому, скажем, решением $x_1 = d_{11}$ иногда однозначно определится x_2.

Нередко значения дискретных переменных подбирают по принципу сохранения величины $F(x)$. Так, в задаче минимизации веса фермы моста увеличение одних ее размеров с целью поднять грузоподъемность можно связывать с уменьшением других, позволяющим избежать существенного прироста веса.

Мы хотим еще раз подчеркнуть, что представленный выше подход, будучи по существу эвристическим, эффективен только тогда, когда решения используемых в нем непрерывных задач и соответствующих дискретных оказываются близкими. К счастью, последнее характерно для многих прикладных постановок и позволяет получать приемлемое приближение ценой построения двух-трех точек x', причем трудоемкость определения дополнительных x' обычно оказывается меньше, чем поиска x' на первом шаге. Если это не так, то скорее всего дискретное решение будет сильно отличаться от непрерывного, и тогда, возможно, стоит пересмотреть постановку задачи, с тем чтобы сблизить их. В частности, если дискретность связана с ограниченностью номенклатуры предлагаемых поставок (как в задаче о сети трубопроводов), то не исключено, что имеет смысл поискать более подходящих поставщиков.

7.7.2. ЦЕЛОЧИСЛЕННЫЕ ПЕРЕМЕННЫЕ

В этом разделе речь пойдет о задачах с такими дискретными переменными, нецелочисленные значения которых не допускают разумной интерпретации. Например, x может иметь смысл количества штучных изделий или переключений с одной технологии производства на другую. Решать задачи с дискретностью подобного рода, как правило, намного труднее, чем рассмотренные в предыдущем разделе. Лучше всего, когда целочисленных переменных очень мало, скажем меньше пяти, и количества их возможных значений тоже исчисляются единицами. Тогда можно применить комбинаторный подход. В данном контексте он состоит в том, чтобы для каждой приемлемой комбинации значений целочисленных

переменных найти минимум целевой функции по остальным (непрерывным) переменным и выбрать из полученных величин наименьшую. Иногда неперспективность отдельных комбинаций ясна априори и поэтому их просматривать не придется. Кроме того, в задачах с ограничениями среди мыслимых комбинаций, как правило, есть много недопустимых (не позволяющих удовлетворить ограничениям), и нередко большую часть таковых удается отбраковать с помощью очень простых вычислений.

К сожалению, уже при умеренном числе дискретных переменных количество сочетаний их значений становится астрономически большим, и поэтому трудоемкость комбинаторного подхода оказывается неприемлемой. Бывает, что в этой ситуации представление об искомом решении удается получить из непрерывной задачи, формулируемой специальным образом. Этот прием проиллюстрирован ниже на примере задачи оптимизации конструкции ректификационной колонны.

Рис. 7d. Слева изображена схема обычной ректификационной колонки; справа — схема ее непрерывной модели.

Условная схема обычной колонны изображена слева на рис. 7d. Внизу колонны имеются отверстия, через которые подается пар. Он поднимается вверх, частично конденсируясь. Конденсат стекает вниз и вновь попадает в испаритель.

В среднюю часть колонны (непрерывный процесс) подается разделяемая смесь. Пар при контакте с ней обогащается легколетучими компонентами, жидкость — труднолетучими. Легколетучие компоненты отбираются из колонны, труднолетучие возвращаются в колонну. Поскольку разделяемая смесь и пар движутся навстречу друг другу, по высоте колонны имеется градиент концентрации летучего компонента, для расчета которого используется понятие «теоретические ступени разделения», или «теоретические тарелки». Проблема состоит в том, чтобы за счет выбора ступени, на которую будет подаваться жидкость, добиться минимальной себестоимости процесса при заданных характеристиках выхода.

На первый взгляд поставленная задача оптимизации по номеру ступени не имеет непрерывного аналога. Однако, если слегка переформулировать ее, такой аналог становится очевидным. Надо для каждой ступени ввести свою целочисленную переменную со значениями {0, 1}, имеющую смысл метки: если она равна единице, жидкость закачивается на этой ступени, если нулю — на другой. Потребовав, чтобы сумма новых переменных была единичной, мы получим задачу, эквивалентную исходной. При этом i-ю целочисленную переменную x_i можно интерпретировать и как долю общего объема жидкости, поступающую на i-ю ступень. Тем самым придается смысл непрерывной задаче, в которой условие целочисленности x_i заменяется простыми ограничениями $0 \leqslant x_i \leqslant 1$, и если в ее решении при каком-то i окажется $x_i = 0.9$, то скорее всего искомый номер равен i. На рис. 7е показана типичная картина распределения входного потока для непрерывной модели; наиболее вероятно, что оптимум в дискретной задаче достигается при $x_3 = 1$.

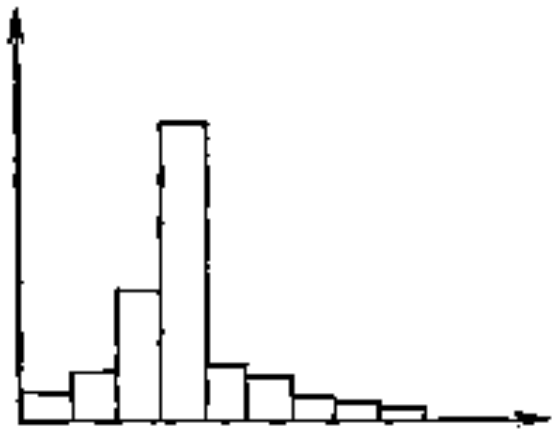

Рис. 7е. Типичное распределение входного потока по отсекам, получающееся из непрерывной модели.

Интересно отметить, что численные эксперименты с моделью ректификационной колонны заставили задуматься над конструкциями, которые раньше не применялись. В некоторых случаях из непрерывной постановки следовало, что закачка жидкости одновременно на две ступени может дать существенную экономию.

В связи с описанным и подобными ему приемами стоит упомянуть о модификации целевых функций непрерывных задач, иногда способствующей более четкому выделению предпочтительных дискретных решений. Скажем, для этого в примере с колонной к исходной F можно было бы добавить $1/\Sigma x_i^2$.

Замечания и избранная библиография к разделу 7.7.

Желающим познакомиться с алгоритмами целочисленного линейного программирования мы рекомендуем обзорную статью Била (1977). Подробности программного воплощения схем дискретной оптимизации можно найти, в частности, в работах Джонсона (1978) и Джонсона и Пауэлла (1978).

Задача с ректификационной колонной решалась Сарджентом и Гаминибандарой (1976).

ГЛАВА 8

ПРАКТИЧЕСКИЕ ВОПРОСЫ

Когда делаешь выбор, разбирай каждый случай, он окажется верным.
Альбер Камю, «The Notebooks 1935—1942» (1963)

Ты можешь ошибаться, со временем, кто все правильно.
Джон Леннон и Пол Мак-Картни, «We can work it out» (1965)

В этой главе мы обсудим ряд аспектов оптимизации, которые редко освещаются в публикациях и в то же время важны для практики. По существу, речь пойдет о неких «житейских мудростях», хотя и не являющихся панацеями от всех бед, но тем не менее помогающих чаще, чем можно было бы ожидать. Затронуты вопросы, возникающие и до, и во время, и после решения задачи на машине. Еще раз подчеркнем, что предлагаемые рекомендации не дают стопроцентных гарантий успеха и в каждом конкретном случае требуют критической оценки.

8.1. ПРИМЕНЕНИЕ БИБЛИОТЕЧНЫХ ПРОГРАММ

В данном разделе будут рассмотрены отдельные вопросы использования оптимизационного математического обеспечения в прикладных работах. Часто комментарии будут относиться к «гипотетической» библиотеке программ, включающей реализации всех методов из гл. 4, 5, 6. По нашему мнению, она была бы «идеальной» на современном уровне развития аппарата оптимизации, так что, говоря о ее применении, мы моделируем «идеальную» ситуацию. Существующие библиотеки беднее, и, хотя уже сегодня многие алгоритмы предыдущих глав доступны в виде хорошо документированных программных продуктов, возможности каждого отдельного пользователя, наверное, всегда останутся ограниченными. Поэтому в главу включен разд. 8.1.5, где обсуждается, как использовать несовершенные средства.

8.1.1. ВЫБОР МЕТОДА

Когда оптимизационная задача поставлена, приходит черед выбирать метод ее решения. Прежде всего при этом следует учесть основные характеристики целевой функции и функций ограничений. По ним все задачи разбиваются на классы, каждому из которых отвечает своя группа предпочтительных алгоритмов (см. разд. 1.2). Однако речь идет именно о группе, так что проблема выбора остается и после определения класса задачи. На каком методе следует остановиться, зависит от того, какую информацию о производных можно предоставить, каков имеющийся объем ма-

шинной памяти и как соотносятся трудоемкости вычисления функций и алгебраических блоков сопоставляемых схем.

Самое общее правило выбора звучит следующим образом: *чем больше информации о производных, которую можно получить ценой приемлемых затрат, будет использовано при решении задачи, тем лучше.* В частности, отказываться от процедуры, требующей аналитические значения градиентов, только потому, что для подсчета этих значений придется писать отдельную программу, неразумно. Такая экономия усилий скорее всего обернется потерями — ведь методы поиска экстремума без применения производных сходятся медленнее и не столь надежны, как градиентные. Аналогично дело обстоит и с матрицами Гессе: если их можно вычислять и если это не в сотни раз сложнее, чем считать градиенты, то пренебрегать ими не следует.

8.1.1.1. Выбор метода безусловной минимизации. Пусть известно, что целевая функция дважды дифференцируема, а число переменных n таково, что $n \times n$-матрица свободно помещается в оперативной памяти машины. В этой ситуации методы поиска минимума без ограничений из гл. 4 можно ранжировать следующим образом:

— модифицированные ньютоновские с вычислением вторых производных;

— модифицированные ньютоновские без вычисления вторых производных;

— квазиньютоновские с вычислением градиентов;

— квазиньютоновские без вычисления градиентов;

— методы сопряженных градиентов с вычислением первых производных;

— методы сопряженных градиентов без вычисления первых производных;

— метод многогранника.

Чем выше позиция метода в этом списке, тем больше число задач указанного класса, которые он успешно решает.

Первенство держат ньютоновские методы (см. разд. 4.4 и 4.5.1). Использование информации второго порядка обеспечивает им высокие скорости сходимости и позволяет делать качественные выводы относительно найденного численного решения. Например, по соответствующей матрице Гессе можно провести анализ чувствительности (см. разд. 8.3.3). (Здесь уместно отметить, что квазиньютоновские схемы (см. разд. 4.5.2) аналогичных возможностей не предоставляют, поскольку не гарантируют точности вычисляемых приближений матрии вторых производных.) Еще одно важное преимущество модифицированных ньютоновских методов в том, что только они застрахованы от сходимости в седловые точки.

При переходе от аналитических значений вторых производных к их конечно-разностным оценкам надежность ньютоновских методов и их эффективность в большинстве случаев практически не

меняются. В основном сохраняется и состав вычислений. Иначе дело обстоит с понижением порядка квазиньютоновских методов (см. разд. 4.6.2). Во-первых, даже хороший конечно-разностный квазиньютоновский метод может отказать в области малых $|g(x)|$. Во-вторых, по составу вычислений он будет значительно отличаться от своего прототипа, рассчитанного на точные производные; в частности, в конечно-разностных квазиньютоновских методах применяются своеобразные процедуры одномерного поиска, критерии «существенного убывания» (см. разд. 4.3.2.1) и правила останова. Общие вопросы конечно-разностной аппроксимации первых производных подробно рассмотрены в разд. 8.6.

В приведенном выше списке самым неприязненным в смысле требуемой информации относительно F является метод многогранника. Он же и самый ненадежный. Как уже было указано в разд. 4.2.1, *этот и подобный ему методы следует использовать, только когда нет альтернативы.*

На рис. 8а изображена схема выбора программы безусловной минимизации из типовой библиотеки математического обеспечения. Из нее видно, на чем следует остановиться в зависимости от характеристик F (доступности производных и размерности аргумента). Наряду с основными в схеме упоминаются и некие вспомогательные процедуры. О них речь пойдет в разд. 8.1.2.2.

Современный уровень развития аппарата поиска безусловного минимума при большом числе переменных не позволяет давать столь четких рекомендаций по выбору методов, как для задач малой размерности. В тех случаях, когда их удается применить, очень эффективными здесь оказываются дискретные ньютоновские методы (см. разд. 4.8.1). Значительно хуже своих обычных аналогов работают квазиньютоновские схемы с сохранением слабой заполненности. Иногда удается успешно использовать технику сопряженных градиентов (особенно в сочетании с приемами предварительного улучшения обусловленности).

8.1.1.2. Выбор метода для задачи с линейными ограничениями. В основе каждого из рассмотренных в гл. 5 методов решения небольших задач с гладкими целевыми функциями и линейными ограничениями лежит какая-то из типовых стратегий поиска безусловного минимума. При хорошем воплощении метода именно она будет определять его качества. Поэтому здесь остается справедливой развернутая в предыдущем разделе. Надо только отметить, что дифференциация становится более ощутимой. К примеру, присутствие в задаче простых ограничений на переменные для квазиньютоновских методов, как правило, не помеха, а о следующих за ними в списке методах сопряженных градиентов этого сказать уже нельзя.

Размер задачи. Существующие программы минимизации при линейных ограничениях распадаются на две категории: одни рас-

считаны на большие размерности, другие — на малые и умеренные. Стало быть, подбирая средства для решения конкретной задачи, прежде всего следует решить, считать ли ее «большой» или нет.

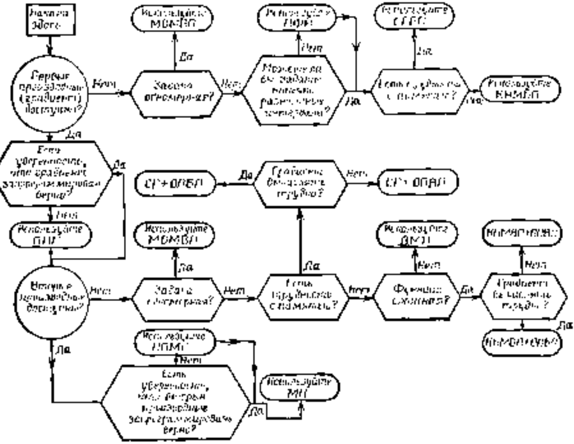

Рис. 8а. Схема выбора метода решения задачи безусловной минимизации с умеренным числом переменных.

МОМБП — метод одномерной минимизации без вычисления производных
ПФИ — программа формирования конечно-разностных интервалов
СГБП — метод сопряженных градиентов с конечно-разностной аппроксимацией производных
КНМБП — квазиньютоновский метод с конечно-разностной аппроксимацией производных
СГ + ОПБП — метод сопряженных градиентов с процедурой одномерного поиска без вычисления производных
СГ + ОПВП — метод сопряженных градиентов с процедурой одномерного поиска с вычислением производных
ППГ — программа проверки правильности вычисления градиента
МОМВП — метод одномерной минимизации с вычислением производных
ДМП — дискретный метод ньютона
КНМВП + ОПВП — квазиньютоновский метод с вычислением градиентов и процедурой одномерного поиска с использованием производных
ППМГ — программа проверки правильности вычисления матрицы Гессе
МН — метод ньютоновского типа
КНМВП + ОПБП — квазиньютоновский метод с вычислением градиентов и процедурой одномерного поиска без вычисления производных

Разработка математического обеспечения для оптимизации при больших размерностях всегда была делом коммерческим, в то время как методы и программы для небольших задач создавались в «академических» целях. Отсюда ряд характерных различий. В частности, средства, ориентированные на большие задачи (особенно коммерческие пакеты линейного программирования), намного удобнее в обращении.

Чтобы запустить библиотечную программу какого-то универсального метода, рассчитанного на разовые размерности, пользователь обычно должен сам сформировать в памяти машины двумерный числовой массив, содержащий все (включая нулевые) элементы матрицы условий и вектора правых частей. Кроме того, нередко требуется, чтобы он задал солидный список значений параметров метода (этот момент подробнее рассмотрен в разд. 8.1.3). Совсем иначе организуется общение с программами, предназначенными для больших задач. Здесь посителями исходной информации служат *текстовые входные файлы*, составляемые по специальным правилам. Наиболее распространенным правилом построения входных файлов является так называемый *MPS-формат*. Его суть в следующем: ограничения и переменные именуются; матрица условий определяется последовательностью текстовых строк, каждая из которых отвечает одному коэффициенту и помимо записи его значения содержит соответствующую пару имен. При этом достаточно определить только ненулевые позиции матрицы. Для всех параметров метода предусматриваются *значения по умолчанию*. Если какие-нибудь из них пользователя не устраивают, он может их заменить, другие задавать не надо. Этот способ настройки избавляет от необходимости выбирать массу величин, многие из которых понятны только специалисту.

Средства решения разовых и больших задач различаются также способами реализации. Первые обычно доступны в виде подпрограмм на общепринятых алгоритмических языках высокого уровня (чаще всего на ФОРТРАНе), которые можно вставлять в свою программу. Вторые же, как правило, реализуются на машинно-ориентированных языках (т. е. могут работать только в составе определенных вычислительных комплексов), причем их конкретная начинка составляет предмет собственности разработчиков и от пользователя скрыта. Методы для задач большой размерности часто поставляют в составе систем математического программирования, куда помимо основных, оптимизирующих блоков входит много сервисных (например, генераторы отчетов). Как уже было сказано ранее, общение с методом в данном случае осуществляется не вызовом подпрограммы, а через входной и выходной файлы системы.

Вследствие развитой сервисной части, математическое обеспечение для больших оптимизационных задач проигрывает обычному в эффективности основных алгоритмов. Его ориентация требует экономного использования памяти и тем самым исключает возможность применения самых совершенных и надежных схем выбора направлений поиска и оценивания множителей Лагранжа (см. разд. 5.1.3 и 5.1.5). Поэтому, если нет уверенности, что задача достаточно хорошо обусловлена и размерность позволяет воспользоваться качественной программой, рассчитанной на разовые случаи, то лучше обратиться к последней. Здесь уместно также напомнить о значительных «накладных расходах», отличающих

процедуры решения задач большой размерности,— имеются в виду трудоемкие операции по поддержанию в процессе поиска оптимума разреженных представлений данных.

Из-за указанных выше различий между существующими программными средствами решения больших и обычных задач переключаться с одних на другие непросто. В будущем с появлением более совершенного математического обеспечения ситуация должна измениться. Принимая в расчет популярность и удобство для пользователей правил описания входных данных типа MPS-формата, в новые библиотеки процедур для задач небольшой размерности уже сегодня начинают включать вспомогательные программы, преобразующие буквенно-цифровые спецификации задач во внутренние структуры данных алгоритмов и выдающие результаты решения в терминах этих спецификаций. Беда только в том, что если писать эти программы ввода-вывода на языках типа ФОРТРАНа, то они неизбежно оказываются очень громоздкими.

Структура ограничений. Из содержания гл. 5 видно, что алгоритмы решения небольших задач могут использовать существенно различные внутренние представления матрицы ограничений и ориентироваться на разные предположения о ее структуре. Будучи приспособленным к одним структурам, алгоритм окажется неэффективным (или неудобным) в обращении для других.

Сказанное относится, в частности, к алгоритмам линейного программирования. Среди них есть такие, в которых все ограничения (включая простые) обрабатываются одинаково, а есть и иные, требующие постановки задачи в *стандартной форме* и учитывающие простые ограничения и ограничения общего вида по-разному (см. разд. 5.3.1 и 5.6.1). В случае когда число общих ограничений равно m, а число переменных n, причем для каждой указаны границы, алгоритмы первого типа будут работать с базисной $n \times n$-матрицей, а второго — с базисной $m \times m$-матрицей. В зависимости от того, что больше — n или m, предпочтительными оказываются либо те, либо другие.

Возьмем для примера линейную задачу с 10 неравенствами общего вида и 100 переменными; в каждой ее вершине по меньшей мере 90 переменных принимают граничные значения. Метод, не отличающий простых ограничений от общих, для решения этой задачи потребует память под запись плотной 100×100-матрицы. В то же время метод, рассчитанный на стандартную форму, будет оперировать 10×10-матрицей. С другой стороны, если в задаче 10 переменных и 100 общих неравенств, то 100×100-матрица возникает во втором методе, а в первом — матрица размера 10×10.

Коль скоро линейную задачу приходится решать с помощью метода, для которого она «неудобна», выход один — сформировать двойственную задачу (см. разд. 3.3.2.2) и применить имеющуюся программу к ней. К сожалению, когда исходная постановка сильно

структурирована, правильно выписать двойственные ограничения бывает не просто, и здесь часто допускают ошибки.

Методы нуль-пространства и ранг-пространства. Для маленьких задач трудоемкость вычисления нелинейной целевой функции и ее производных обычно значительно превышает трудоемкость алгебраических блоков оптимизационных процедур, и поэтому для них подбирать методы легко — можно не принимать во внимание специфику ограничений. Предпочтение в таких случаях следует отдавать методам нуль-пространства (см. разд. 5.2) как более надежным и устойчивым. Однако по мере увеличения размерности требуемые ими объемы памяти и вычислений быстро возрастают, так что в определенных ситуациях становится разумным использовать методы ранг-пространства. Рациональный выбор зависит в основном от числа активных в решении ограничений. (Сопоставление методов нуль- и ранг-пространства дано в разд. 5.4.)

Выбор метода, генерирующего допустимые точки. На практике нередко возникают задачи, целевые функции которых определены только в допустимых точках. Примером служит рассмотренная в разд. 1.1 задача о проектировании носовой части летательного аппарата — математическая теория подсчета лобового сопротивления (которое надо минимизировать) неприменима, если, скажем, нарушено условие неотрицательности объема (одно из ограничений). Бывает также, что целевая функция F определена вообще, но ее значения в недопустимой области бессмысленны. Здесь можно привести такой пример: пусть переменная x должна удовлетворять ограничениям $-1 \leqslant x \leqslant 1$ и входит в F через разность между опорной величиной y_i и чебышевским разложением $\Sigma_j a_j T_j(x)$; поскольку полиномы $\{T_j(x)\}$ обладают хорошими свойствами только на отрезке $[-1, 1]$, за его пределами смысла у значений F не будет, хотя F вычислима и там.

Если бы машина считала без погрешностей, в любом методе из гл. 5, не использующем конечно-разностной аппроксимации производных, целевая функция F автоматически вычислялась бы исключительно в допустимых точках. Из-за ошибок округления невязки в ограничениях порядка машинной точности ненабежны, но больших проблем из-за этого не возникает — эти невязки можно «компенсировать», заранее подправив правые части ограничений должным образом. Сложнее дело обстоит с методами, в которых применяется численное дифференцирование. Здесь обращения к процедуре подсчета F возможны в точках, где нарушения ограничений намного больше машинной точности, причем избежать этого модификацией конечно-разностных формул можно лишь в том случае, когда все ограничения простые (см. разд. 5.5.1). Следовательно, если важно, чтобы вычислений F в недопустимых точках не было, для решения задачи надо взять программу, оперирующую аналитическими производными.

8.1.1.3. Выбор метода для задачи с нелинейными ограничениями. Если основу аппарата минимизации при линейных ограничениях составляют методы, генерирующие исключительно допустимые точки, то в случае с нелинейными ограничениями ситуация обратная — среди разработанных для него методов большинство составляют такие, которые не обеспечивают соблюдения ограничений (некоторые же заведомо нарушают их; см. разд. 6.2.1.1). Поэтому здесь при выборе процедуры решения конкретной задачи прежде всего надо уяснить, важно ли сохранять допустимость в течение всего процесса поиска или нет. Иногда ответ диктуется самой природой задачи: если в недопустимой области не целевая функция не определена или бессмысленна, использовать метод, который может нарушать ограничения, нельзя.

Из методов допустимой точки для задач с нелинейными ограничениями наиболее распространен метод барьерных функций (см. разд. 6.2.1.2). Правда, «в чистом виде» он применим лишь при условии, что все ограничения являются неравенствами. Если в задаче есть также *линейные* равенства, то в качестве метода, сохраняющего допустимость, можно взять какую-нибудь процедуру, построенную на базе барьерного преобразования и обрабатывающую эти равенства приемами гл. 5.

Когда в списке общих ограничений задачи есть l нелинейных равенств и больше ничего нет, возникает соблазн «исключить» их простым приемом: разбить вектор переменных x на две составляющие x_D и x_I размерностей l и $n-l$ соответственно и вести минимизацию по x_I, все время подбирая x_D как «решение» системы ограничений. (Данный подход лежит в основе методов типа приведенных градиентов; см. разд. 6.3.) Однако мы считаем, что поддаваться подобному соблазну не стоит. Как отмечалось в разд. 6.3, стремление выдерживать нелинейные равенства в течение всего процесса поиска часто оказывается причиной медленной сходимости. Кроме того, если помимо «исключаемых» равенств в задаче есть простые ограничения на x_D, учесть их при рассматриваемом подходе не просто. В общем здесь лучше не экспериментировать, а применить какую-нибудь библиотечную программу минимизации при нелинейных равенствах.

8.1.2. Роль пользователя

Иногда высказывается мнение, что цель математического обеспечения — «избавить пользователя... от всякой необходимости думать над решением своей задачи» (Дэвис и Рабинович, 1967). Для некоторых классов вычислительных задач эта цель достижима (в той степени, которую допускают погрешности машинной арифметики). Однако класс задач оптимизации к таковым не относится. Только самые легкие из них не требуют обдумывания. Обычно же

при постановке задачи для решения на машине возникает ряд важных вопросов, ответить на которые может только пользователь.

Обе стороны пользователи в общении с программными средствами бывают очень разными. В простейших случаях от него требуется лишь дать описания функций задачи, в сложных могут понадобиться начальное приближение и привлекаемого метода. Роль пользователя частично отражается объемом дополнительных данных, которые он сообщает программе.

Для оптимизационных алгоритмов характерна значительная свобода в выборе возможностей реализации. Как лучше всего использовать ее, зависит от конкретной задачи. Имея это в виду, разработчики соответствующих программ передают право выбора пользователю, предоставляя ему возможность назначать некоторые параметры метода. Их бывает довольно много, и поэтому, чтобы облегчить жизнь неопытным и незаинтересованным лицам, обычно организуют дело так, что все параметры задавать не обязательно. Если метод реализуется в виде библиотечной процедуры, то для него пишут две программы — основную и «упрощенную»; команда вызова первой содержит полный список параметров, а команда вызова второй — лишь те, которые подобрать автоматически никак нельзя. Если же контакт с методом осуществляется через входной файл, то всё устраивают еще удобнее: требуют, чтобы некоторые из спецификаций присутствовали в нем непременно, а прочие позволяют включать или не включать в него по усмотрению пользователя.

Коль скоро значение какого-то параметра пользователем не зафиксировано, оно должно выбираться автоматически. Делается это одним из двух способов: либо параметр получает «дежурное» значение, либо он вычисляется по какому-то принципу в соответствии с известными значениями других параметров и характеристик задачи.

8.1.2.1. Параметры по умолчанию. При ряде предположений о задачах, к которым будет применяться метод, для некоторых из его параметров можно заранее подобрать разумные значения и сделать так, что в случаях, когда пользователь не установит других, будут приняты эти. Их называют *значениями по умолчанию*.

Например, для всех «хорошо отмасштабированных задач» (см. разд. 8.7) безусловной минимизации функций, вычисляемых с полной машинной точностью, можно взять единый порог в критерии останова (о критериях останова речь пойдет в разд. 8.2.3).

Значения по умолчанию считаются «безопасными» — их выбирают в предположениях, справедливых для большинства практических задач, и, как правило, они не становятся причиной существенного ухудшения работы метода. Однако в некоторых случаях эти значения оказываются не слишком удачными, а изредка могут и просто

испортить дело. Так бывает, когда задача не согласуется с «моделью», на основании которой они выбраны. Применять значения по умолчанию не следует также, если решается большая серия похожих друг на друга трудоемких задач: скорее всего в процессе подобного эксперимента алгоритм можно «настроить» и поточнее. (Рекомендации по выбору параметров даны в разд. 8.1.3.)

8.1.2.2. Сервисные программы. Бывают параметры, для которых нельзя подобрать хороших универсальных значений (уж очень они привязаны к задаче), но в то же время можно построить хорошие универсальные правила их вычисления. Как и параметры по умолчанию, их не включают в список обязательно требуемых от пользователя, и, хотя лучше, если, хорошо понимая свою задачу и алгоритм, он определит их сам, за ним оставляют право не делать этого и предусматривают специальные *сервисные программы* для их расчета. Значения, выданные сервисной программой, как правило, оказываются удачнее, чем «выбранные по наитию» неопытным пользователем.

Возьмем, к примеру, квазиньютоновский метод с аппроксимацией первых производных (см. разд. 4.6.2). Ему нужен набор конечно-разностных интервалов, причем их наилучшие значения зависят от задачи, а плохие чреваты серьезными затруднениями. Если пользователь не хочет сам задать эти интервалы, их можно вычислить по данным, относящимся к начальной точке, с помощью сервисной программы (см. разд. 8.6.2). При условии что в процессе минимизации масштабные характеристики задачи изменятся не слишком сильно, найденные значения будут вполне приемлемыми до конца поиска.

Хорошие системы математического обеспечения включают также сервисные модули для проверки правильности программ расчета функций, составляемых пользователем, например для тестирования программ вычисления производных (как это делается, кратко описано в разд. 8.1.4.2). Коль скоро обращение к методу осуществляется командой вызова из программы пользователя, то при «короткой команде» (см. выше) сервисные модули данного типа обычно подключаются автоматически. Если же доступ организован через входной файл, их вызов чаще оставляют на усмотрение пользователя, резервируя для этого специальные предложения раздела спецификаций.

8.1.3. ВЫБОР ПАРАМЕТРОВ ПОЛЬЗОВАТЕЛЕМ

Назначая параметры метода при обращении к стандартной программе, пользователь может «настраивать» его на свою задачу и передавать ему какую-то дополнительную полезную информацию.

Смысл одних параметров очевиден; чтобы понять роль других, надо хорошо знать метод. К первым относится, например, началь-

ное приближение матрицы Гессе для квазиньютоновского поиска. По умолчанию в качестве этого приближения обычно берется единичная матрица, но, конечно, будет лучше, если на основании каких-то соображений пользователь задаст более правдоподобную оценку.

Параметры второго типа имеют смысл управляющих. В восьми следующих разделах мы обсудим некоторые из них. Рекомендации по выбору конечно-разностных интервалов даны в разд. 8.6.

8.1.3.1. Точность вычисления функций. Часто от пользователя требуется оценка точности, с которой будут вычисляться функции задачи. Она может понадобиться: (i) для подсчета ошибки условий при конечно-разностном приближении производных (разд. 4.6.1.1); (ii) для масштабирования переменных (см. разд. 8.7); (iii) для критерия останова (знание этой точности позволяет существенно уменьшить вероятность преждевременного прерывания и ненужного затягивания счета) (см. разд. 8.2); (iv) для определения минимального просвета между пробными точками в процедуре одномерного поиска (разд. 4.3.2.1 и 8.4.2).

Отметим, что при решении оптимизационных задач оценки точности подсчета функций нужны для *неизвестных заранее значений переменных* (в отличие от других вычислительных задач, где они строятся только для одной заданной точки). Получив информацию, относящуюся к начальному приближению, оптимизирующая программа должна на ее основе сама определить оценки для последующих итераций (некоторые способы оценивания описаны в в разд. 8.5).

Как правило, у пользователя запрашивается оценка сверху для абсолютной ошибки вычисления функции в окрестности x_0, т. е. число ε_A, такое, что для всех x, близких к x_0, будет

$$fl(F(x)) = F(x) + \sigma,$$

где $|\sigma| \leqslant \varepsilon_A$ (см. разд. 2.1.6) или, что эквивалентно,

$$|fl(F(x)) - F(x)| \leqslant \varepsilon_A.$$

Пожалуй, более принято думать о точности в *относительных* терминах, а проще говоря, в терминах числа правильных значащих цифр в выданном машиной значении. Однако (см. разд. 8.5) соответствующая форма задания ошибки при малых по модулю $F(x)$ не подходит. Все же для некоторых F можно использовать и ее, характеризуя аккуратность расчетов *верхней границей относительной погрешности*: это — число ε_R, характерное тем, что из

$$fl(F(x)) = F(x)\ (1+\varepsilon)$$

следует неравенство $|\varepsilon| \leqslant \varepsilon_R$.

При работе на машине с длинным словом абсолютную ошибку вычислений можно оценивать довольно грубо. Один-два порядка

здесь существенной роли не играют, и если есть основания полагать, что примерно r значащих цифр машинного значения F верны, то можно смело брать $\varepsilon_F = 10^{-r}$ и $\varepsilon_A = 10^{-r}(1+|F|)$. Аккуратные оценки точности важны при расчетах на машине с коротким словом; тут и излишний пессимизм, и неоправданный оптимизм чреваты серьезными потерями.

Мы хотим подчеркнуть, что ошибки, о которых идет речь, определяются только дефектами машинной арифметики и никоим образом не должны смешиваться с погрешностями моделирования функцией F какой-то реальной зависимости (см. также разд. 7.3.1 и 8.5.1.1).

Бывает, что пользователь не может обоснованно оценить аккуратность вычисления своих функций. Тогда для определения ε_A надо применить сервисную программу (к вопросу о выборе ε_A мы еще вернемся в разд. 8.5).

8.1.3.2. Алгоритмы одномерного поиска. На типичной итерации поиска безусловного минимума сначала рассчитывается направление движения, а затем срабатывает алгоритм выбора длины шага (см. разд. 4.3.2.1). Среди таких алгоритмов наиболее распространены алгоритмы, использующие квадратичную или кубическую аппроксимацию; перебирая пробные шаги, первые требуют подсчета только соответствующих значений функции (в начальной точке итерации могут понадобиться и производные), а вторым, кроме того, нужны градиенты.

В некоторых случаях сразу ясно, какому из двух указанных способов одномерного поиска отдать предпочтение. Скажем, при обращении к процедурам ньютоновского типа всегда прибегают к несколько более надежной кубической аппроксимации. Затратами на вычисление дополнительных градиентов здесь можно пренебречь, так как построение точной или приближенной матрицы Гессе обходится дороже. Если же речь идет о какой-то процедуре с конечно-разностным приближением первых производных, то надо брать квадратичную схему - слишком уж накладно считать и значений функций (для оценки градиента) при каждом пробном шаге.

Рациональный выбор алгоритма одномерного поиска для программы квазиньютоновского метода с точными производными зависит от характера функции. Коль скоро саму ее вычислять значительно проще, чем ее градиент, лучше обратиться к квадратичному алгоритму: с ним дело, скорее всего, пойдет быстрее. Однако, если при одномерной минимизации возникнут трудности, надо будет переключиться на (более надежную) кубическую аппроксимацию.

8.1.3.3. Точность одномерного поиска. Методом выбора шага стараются отыскать локальный минимум функции по направлению, и делают это с точностью, которая контролируется специальным параметром η из диапазона $0 \le \eta < 1$ (см. разд. 4.3.2.1). Чем меньше η, тем точнее будет выполняться одномерная минимизация. При

этом с уменьшением η, вообще говоря, возрастает среднее число пробных точек на одну основную итерацию и сокращается общее количество итераций, требуемых для удовлетворения критерию сходимости. Трудоемкость процесса в целом с приближением η к нулю сначала убывает, а потом начинает расти. К примеру, при переходе от $\eta = 0.9$ к $\eta = 10^{-2}$ число основных итераций обычно уменьшится по крайней мере вдвое, а суммарное количество обращений к процедуре подсчеты целевой функции увеличится раза в полтора. Дальнейшее же уменьшение η, скажем до 10^{-6}, скорее всего приведет только к увеличению количества вычислений функции, а на числе итераций практически не отразится. Указанная тенденция особенно отчетлива, когда одномерный поиск осуществляется на основе квадратичной аппроксимации по трем точкам; дело в том, что такая аппроксимация вблизи минимума по направлению теряет качество.

Лучшее значение η (то, при котором решение будет найдено за минимальное время) зависит от задачи. Однако для многих задач и алгоритмов оптимальные η оказываются близкими. Поэтому, если брать вместо них некое среднее, выбранное на основании большого опыта вычислений, серьезных потерь эффективности, как правило, не будет; это среднее в библиотечных программах служит значением по умолчанию (см. разд. 8.1.2.1).

Само собой разумеется, встречаются и задачи с существенно отличающимися от значения по умолчанию оптимальными η. В частности, к ним относятся задачи с простыми функциями. Для них предпочтительные η меньше «среднего», поскольку его принято подбирать по таким функциям, трудоемкость минимизации которых в основном определяется необходимостью их вычисления в пробных точках. В простых же случаях увеличение при занижении η числа обращений к процедуре подсчета функции с лихвой окупится экономией суммарных затрат алгебраических блоков метода (имеются в виду решение линейных систем, пересчет матричных разложений и т. д.).

Заниженные значения η оправданны и тогда, когда производные считать намного труднее, чем функцию. Здесь тоже желательно сократить число итераций, пусть даже ценой просмотра большого числа пробных точек при выборе шагов (для одномерного поиска надо использовать процедуру, не нуждающуюся в производных).

Труднее всего подбирать параметр η в алгоритмах сопряженных градиентов (см. разд. 4.8.3). Для них оптимальные η, отвечающие разным задачам, разбросаны особенно сильно. Поэтому, если при помощи программы метода сопряженных градиентов предполагается решить большую серию схожих задач, то для определения хорошего η имеет смысл поставить специальный численный эксперимент.

8.1.3.4. Максимальная длина шага. Во многих алгоритмах полезно ограничивать изменение x, допускаемое на одной итерации (см. разд. 4.3.2.1). Соответствующее ограничение вводит через параметр Δ, подчиняя длину шага α вдоль выбранного направления p неравенству $\|\alpha p\| \leqslant \Delta$.

Существует ряд причин, по которым выбор Δ разумно предоставить пользователю. Хорошо понимая свою задачу, назначением правильного Δ он может

1) устранить угрозу переполнения при подсчете функций задачи для пробных шагов процедуры одномерной минимизации,

2) повысить эффективность поиска, обеспечив вычисление функций только в «разумных точках»;

3) исключить бессмысленно большие шаги, провоцируемые несоблюдением критерия «существенного убывания» при меньших шагах (например, из-за того что целевая функция не ограничена снизу вдоль выбранного направления);

4) стимулировать сходимость к решению, лежащему вблизи начальной точки.

Следует отметить, что тех же целей можно достичь (и даже с большей вероятностью) введением подходящих двусторонних ограничений на переменные. Не обязательно, чтобы они были очень точными, хотя, чем верней указаны границы, тем эффективнее работают алгоритмы.

8.1.3.5. Максимальное число обращений к процедуре подсчета целевой функции. Данный параметр вводится для того, чтобы дать пользователю возможность остановить метод после вычисления определенного количества значений целевой функции. Эта возможность полезна во всех тех ситуациях, когда нет уверенности, что избранным методом можно решить задачу. Бывает, например, что, запуская программу безусловной минимизации, мы не знаем, ограничена функция снизу или нет, и, коль скоро она не ограничена, а прерывание не предусмотрено, машина может долго и бессмысленно считать — до тех пор, пока не произойдет переполнение.

Если количество вычислений функции, нужное для решения задачи, разумно оценить не удается и при этом есть способ легко возобновлять работу программы после прерывания, то для начала лучше запланировать значение параметра, о котором идет речь. Анализ промежуточных результатов обычно позволяет правильно спрогнозировать перспективы сходимости и избежать бесполезных затрат машинного времени, когда задача оказывается некорректной.

В случае с безусловной минимизацией число необходимых обращений к процедуре подсчета целевой функции можно оценивать снизу по запросам метода при работе с положительно определенной квадратичной формой. Например, квазиньютоновскому методу, использующему аналитические значения градиентов, для отыскания точки ее минимума нужно $n+1$ итераций, причем на каждой ее

придется вычислить по меньшей мере один раз. Значит, всего потребуется не менее $n+1$ вычислений. То же можно сказать и в отношении неквадратичных функций. (На самом деле в квазиньютоновских методах их обычно приходится вычислять более чем в $5\times n$ точках.)

8.1.3.6. Локальный поиск. Если вторые производные недоступны, то проверить, соблюдаются ли в найденной точке достаточные условия оптимальности (см. гл. 3), вообще говоря, нельзя, и есть риск, что она будет седловой, а не оптимальной. В связи с этим иногда прибегают к локальному поиску — попытке уменьшить значение целевой функции с помощью средств, радикально отличающихся от реализованных в основном методе. Ее безрезультатность считают признаком истинности проверяемого численного решения.

Типичные процедуры локального поиска работают по принципу «случайного» сканирования и соответственно требуют многократных дополнительных вычислений функции. Поэтому, если считать ее трудно, а сходимость в седловую точку (в силу специфики задачи) практически исключена, от локального поиска можно и отказаться. Однако в ситуациях, когда вероятность ложного решения надо свести к минимуму, пренебрегать им не следует.

8.1.3.7. Параметр штрафа в модифицированной функции Лагранжа. Программы, реализующие методы модифицированных функций Лагранжа, обычно требуют, чтобы пользователь задал начальное значение параметра штрафа ρ. На случай, если ход решения покажет, что оно выбрано неудачно, предусматривается возможность автоматической замены (как правило, ρ может только увеличиваться).

Выбор ρ, вообще говоря, очень сильно влияет на работоспособность метода. Если взять ρ слишком малым или слишком большим, подзадача безусловной минимизации модифицированной функции Лагранжа может оказаться плохо обусловленной (см. рис. 61). Более того, бывает, что при заниженных ρ (т. е. при ρ меньших, чем некоторый порог) у этой функции нет конечных минимумов.

Поскольку оптимальное значение ρ зависит от матрицы Гессе функции Лагранжа в искомой точке, хороших универсальных рекомендаций по выбору ρ не существует. Можно лишь сказать, что для практических задач (с правильно отмасштабированными функциями) более-менее приемлемым обычно является $\rho = \max\{10, 10\|F(x_0)\|_\infty\}$. Ясно также, что если при заведомо хорошем начальном приближении x_0 нет возможности точно оценить множители Лагранжа, то ρ лучше взять большим (чтобы метод не удалялся от x_0).

Когда методом модифицированной функции Лагранжа поочередно решаются несколько схожих задач, окончательное значение параметра штрафа для предыдущей задачи естественно учитывать

при выборе его начального значения для последующей. Коль скоро первое было получено в результате настройки ρ самой программой, с него и надо начать в новом цикле итераций. Если же оно было задано в качестве исходного, причем программа может только увеличивать ρ, стоит попробовать уменьшенное (скажем, в 10 раз) значение.

Напоследок отметим, что введение разумных простых ограничений на переменные существенно сглаживает непристойные последствия неудачного выбора ρ.

8.1.3.8. Параметр штрафа для негладких задач. Негладкие задачи с ограничениями чаще всего решают методами, подобными описанному в разд. 6.2.2.2. Обращаясь к соответствующей программе, пользователь должен определить начальное значение параметра штрафа ρ. Как и в случае с модифицированной функцией Лагранжа, обсуждавшемся выше, программа впоследствии может изменить это значение, причем допускаются поправки обоих знаков.

К счастью, метод с негладким штрафом обычно менее чувствителен к выбору исходного ρ, чем метод с модифицированной функцией Лагранжа. Здесь самая безопасная стратегия — брать относительно большие ρ: тем самым уменьшается вероятность, что у штрафной функции не будет нужного локального минимума. Разумеется, при этом есть риск получить плохо обусловленную подзадачу, но даже если это случится, то в дальнейшем, имея возможность уменьшать ρ, программа сама поправит дело.

Опыт показывает, что на практике обычно подходит $\rho = \max\{100, 100\,|F(x_0)|\}$. Как и для методов модифицированных функций Лагранжа, надо выяснить ситуации, когда известно хорошее начальное приближение x_0; только теперь предпочтительны *заниженные*, а не завышенные значения ρ, скажем $\rho = \max\{1, |F(x_0)|\}$. Снова следует отметить и положительное воздействие простых ограничений, в присутствии которых метод слабее реагирует на неудачные ρ. (Эти ограничения учитываются одинаково с общими, т. е. через соответствующие слагаемые в штрафном терме.)

8.1.4. ОШИБКИ В ПРОГРАММАХ ПОЛЬЗОВАТЕЛЯ

Даже самый совершенный метод едва ли способен найти верное решение неверно запрограммированной задачи, и очень часто причиной непонятных затруднений при работе со стандартными процедурами являются дефекты программ пользователя. Поэтому, столкнувшись с такими затруднениями, прежде всего надо поискать ошибки в своих программах.

8.1.4.1. Ошибки в блоках вычисления функций. При неожиданном отказе стандартного алгоритма прежде всего следует проверить правильность блоков подсчета функций задачи. Очевидный способ — заставить машину вычислять их при тех значениях аргу-

менов, для которых верный ответ известен заранее, и посмотреть, что она выдаст. Это — хороший прием, но выбирать пробные точки надо аккуратно: просто удивительно, как часто берут $x=0$ и $x=1$ и как часто всем специфики функций соответствующие тесты оказываются бессмысленными.

Особенно внимательно следует относиться к случаю, когда подпрограмме расчета функции нужны некоторые вспомогательные данные, передаваемые ей через массив-параметр или (в ФОРТРАНе) через COMMON-память. Иногда эти данные первоначально затираются, причем после того, как используются впервые, поэтому результат первого вычисления функции оказывается верным, а остальные — ошибочными. (Это одна из причин, по которой, приступая к решению задачи на машине, имеет смысл для начала проделать короткий цикл «пробных итераций».)

Ошибки в программах подсчета функций бывают весьма тонкого свойства и тогда приводят к слабым уклонениям от правильных значений. Например, точность представления функции может оказаться меньше полной машинной точности того, что неаккуратно вычисляется какая-то вспомогательная величина. Очень распространенная опошлость — привлекать в расчетах, проводимых с двойной точностью, числа, представленные с одинарной. Обрабатывая выражение, куда они входят, транслятор организует дело так, что зависящие от них результаты арифметических операций тоже будут иметь одинарную точность, т. е. половина правильных значащих цифр теряется.

Некоторые алгоритмы слабо реагируют на малые относительные ошибки вычисления функций. Обращаясь к какому-нибудь из них, приемлемое решение задачи скорее всего удастся получить даже в присутствии таких ошибок. Однако есть и алгоритмы, очень чувствительные к малым неточностям. Это — алгоритмы с конечно-разностной аппроксимацией первых производных. Когда функции считаются неаккуратно, они могут долго работать без ощутимого прогресса, а то и просто застрявают в начальной точке (разд. 8.4). Признаком подобной ситуации может послужить резкое изменение работы программы при смене конечно-разностных интервалов. На нее указывает также переключение на симметричную формулу аппроксимации, когда градиент далеко еще не равен нулю.

8.1.4.2. Ошибки в блоках вычисления производных. Практика показывает, что чаще всего пользователи ошибаются при программировании производных. Эти ошибки почти никогда не бывают незначительными и сбивают с толку любой алгоритм. Вот почему мы настоятельно рекомендуем применять какие-нибудь тесты правильности дифференцирования в программах.

Самый прямой способ выявить ошибочность вычисленного значения производной — сравнить его с результатом конечно-разностной аппроксимации. Обозначим через x точку, где проводится ис-

пытание, и пусть h — малое число, а p — случайный вектор единичной длины с одинаковыми по модулю компонентами. В нормальной ситуации должно выполняться приближенное равенство

$$F(x+hp)-F(x)\approx hg(x)^T p. \tag{8.1}$$

Если оно окажется нарушенным, то (i) либо градиент вычислен неверно, (ii) либо при подсчете F сильно сказываются ошибки округления, (iii) либо функция F плохо отмасштабирована (см. разд. 8.7.1). Последнюю из трех перечисленных альтернатив можно распознать с помощью соотношения

$$F(x+hp)-F(x)\approx \tfrac{1}{2}(F(x+hp)-F(x-hp)). \tag{8.2}$$

Справедливость (8.2) при нарушенном (8.1) является надежным признаком того, что в программе вычисления градиента допущена ошибка. Если же (8.2) не выполняется, причем правые части (8.2) и (8.1) близки, ошибок, вероятнее всего, нет. Наконец, в случае, когда не соблюдаются ни (8.1), ни (8.2) и правые части (8.1) и (8.2) существенно различны, надо посмотреть, что будет при увеличенном h. Коль скоро (8.1) остается нарушенным при любом разумном h, программа вычисления g почти наверняка, неверна (поведение конечно-разностной аппроксимации производной в зависимости от h обсуждается в разд. 4.6.1 и 8.6).

Описанный способ тестирования в практическом отношении вполне эффективен для выявления больших ошибок при подсчете h, но не позволяет обнаруживать малые. В то же время и они могут стать причиной отказа алгоритма; как распознавать такие отказы, обсуждается в разд. 8.4.2.4.

Не сомневаясь в программе подсчета первых производных, правильность вычисления вторых можно проверить приемом, аналогичным описанному выше. Основой тестирования служит соотношение

$$hp^T G(x)\,p\approx (g(x+hp)-g(x))^T p. \tag{8.3}$$

Если при первом пробном h оно нарушается, надо увеличить h и снова проверить его. Обозначим выбранные значения h через h_1 и h_2. Коль скоро (8.3) не выполняется при обоих и

$$h_2(g(x+h_1p)-g(x))^T p\approx h_1(g(x+h_2p)-g(x))^T p, \tag{8.4}$$

то вторые производные, по-видимому, считаются неверно. Если же (8.4) тоже окажется нарушенным, то, как и в случае с первыми производными, надо повторить тест с другим значением h.

Небольшие погрешности при построении матрицы Гессе сходимости обычно не нарушают, но ее скорость может снизиться (см. разд. 8.3.1).

В хороших библиотечных программах правильность подсчета производных проверяется автоматически; когда обнаруживаются какие-то неувязки, пользователю выдается соответствующее сообщение.

8.1.5. РАБОТА С ОГРАНИЧЕННЫМ МАТЕМАТИЧЕСКИМ ОБЕСПЕЧЕНИЕМ

Далеко не все существующие оптимизационные библиотеки столь полны, как хотелось бы, и нередко метод для решения своей задачи приходится выбирать из таких, среди которых нет специально приспособленного к ее типу. Общая рекомендация здесь может быть только одна — *предпочтение надо отдать программе, эксплуатирующей максимум характерных особенностей задачи*. Ниже мы обсудим некоторые конкретные случаи вынужденного применения не вполне подходящих средств.

8.1.5.1. Нелинейные задачи о наименьших квадратах. В разд. 4.7 было показано, что специфика задач о наименьших квадратах позволяет строить для них методы, работающие эффективнее универсальных. Однако их реализации есть не всегда, и поэтому имеет смысл обсудить, как решать такие задачи с помощью методов безусловной минимизации, рассчитанных на целевые функции общего вида.

Обозначим через $J(x)$ матрицу Якоби системы функций, сумму квадратов которых надо минимизировать. Если в искомой точке все они близки к нулю, матрица Гессе этой суммы будет хорошо оцениваться произведением $J(x)^T J(x)$. Значит, даже отказавшись от вычисления вторых производных, для задачи о наименьших квадратах все равно можно взять программу метода ньютоновского типа, «подсунув» запрашиваемую его процедуру построения матрицы Гессе процедурой расчета $J(x)^T J(x)$; это лучше, чем обращаться к какому-нибудь общему методу первого порядка. Такой подход целесообразен и в случаях, когда недоступны аналитические значения не только вторых, но и первых производных; здесь надо применить конечно-разностную аппроксимацию $J(x)$.

Коль скоро невырожденность $J(x)$ гарантировать нельзя, вместо $J(x)^T J(x)$ предпочтительнее использовать какую-нибудь «похожую» на положительно определенную матрицу. Например, можно заменить $J(x)^T J(x)$ на $J(x)^T J(x)+\sigma I$, где σ — малое положительное число (для хорошо отмасштабированной задачи подойдет $\sigma=\sqrt{\varepsilon_m}(1+|F(x)|)$). Это избавит ньютоновский метод от затруднений, возникающих в случае, когда у матрицы системы для расчета направления поиска нет свойства существенной положительной определенности.

8.1.5.2. Аппроксимация части производных. Подавляющее большинство стандартных программ минимизации составляют такие, которые, если уж прибегают к первым или вторым производным, требуют задания их всех без исключений. В то же время бывает, что доступны не все, а «почти все» производные. Тогда есть две

вы альтернативы: можно обратиться к процедуре низкого порядка, а можно подменить недостающие истинные значения производных их конечно-разностными приближениями, т. е. для части позиций градиента или матрицы Гессе вместо аналитического дифференцирования запрограммировать численное. Второй выход предпочтительней.

Чтобы обеспечить качество конечно-разностных оценок, программируя их вычисление надо учитывать обстоятельства, на которые было указано в разд. 4.6.2. В частности, при малых по норме градиентах следует применять центральную формулу. (Сказанное справедливо и для задач с ограничениями, но тогда оно должно быть отнесено к спроектированному градиенту.)

Для аппроксимации части элементов матрицы Гессе по конечным разностям обычно прибегают к той самой программе подсчета первых производных, которая используется в блоке выбора направления поиска и строит градиент целиком. Если при этом брать в качестве конечно-разностных векторов единичные, то сэкономить усилия по сравнению со случаем с аппроксимацией всех вторых производных удается лишь при условии, что какие-то столбцы матрицы Гессе известны полностью. В то же время иногда лишнего счета можно избежать, даже если неизвестные элементы есть в каждом столбце. Для этого надо подобрать специальные конечно-разностные векторы, и здесь привлекают технику, обсуждавшуюся в разд. 4.8.1 в связи с проблемой эффективного оценивания разреженных матриц Гессе. Следует, однако, отметить, что соответствующие приемы часто оказываются довольно сложными.

8.1.5.3. Решение задач с ограничениями программой безусловной минимизации. Если понадобилось найти минимум при нелинейных ограничениях, а в вашем распоряжении есть только программа безусловной минимизации, то на ее основе можно реализовать схему с модифицированной функцией Лагранжа (см. разд. 6.4) (*Штрафными функциями лучше не пользоваться*, см. разд. 6.2.1.1.) При этом должны быть приняты определенные предосторожности.

В общем случае модифицированная функция Лагранжа для ряда (а иногда и для всех) значений параметра штрафа оказывается неограниченной снизу. Поэтому необходимо как-то застраховать применяемый в ней метод от бесконечного счета, например устанавливая относительно невысокую квоту на количество пробных точек или занижая максимальную разрешенную величину шага (см. разд. 8.1.3.4 и 8.1.3.5). Если же имеется программа минимизации при простых ограничениях, то разумнее всего ввести таковые и использовать ее — это самый надежный способ решения проблем, связанных с неограниченностью.

8.1.5.4. Учет линейных и нелинейных ограничений. По многим соображениям теоретического и практического характера (см. гл. 5) линейные и нелинейные ограничения лучше учитывать по-

равному даже при решении задач, в которых есть и те и другие. Однако бывает, что программы, рассчитанной на такие задачи, нет, а имеются только реализации методов поиска минимума при линейных ограничениях и методов, трактующих все ограничения как нелинейные. К какому же из них обратиться, если возникла задача смешанного типа? Коль скоро большинство составляют нелинейные ограничения, скорее всего следует взять метод второй группы. Некоторые потери из-за несовершенства учета линейных ограничений в данном случае оправданы тем, что нелинейные будут обрабатываться наиболее эффективно. Правда, если нужно, чтобы первые выдерживались в течение всего процесса поиска, возможно, придется действовать иначе, поскольку не все способы обработки ограничений, допускающие нелинейность, гарантируют, что в ее отсутствие неязок не будет. Тогда, как и в случае, когда почти все ограничения линейны, следует взять какой-нибудь из методов первой группы, построив на его основе процедуру с последовательной минимизацией модифицированной функции Лагранжа, включающей только нелинейные ограничения (при этом не надо забывать о предосторожностях, упомянутых в разд. 8.1.5.3).

8.1.5.5. Нелинейные уравнения. Не располагая специальными программами решения систем нелинейных уравнений, вместо них можно применять (обычно с тем же успехом) программы минимизации суммы квадратов. Коль скоро некоторые из уравнений линейны, лучше не включать их в минимизируемую сумму, а учитывать непосредственно, т. е. взять метод для задач о наименьших квадратах с линейными ограничениями (пример задачи такого сорта рассмотрен в разд. 7.6.2).

Замечания и избранная библиография к разделу 8.1

Рассмотренные выше параметры настройки стандартных программ довольно типичны, и, в частности, характерны для библиотек, поставляемых Группой численных алгоритмов (NAG) и Национальной физической лабораторией. Эти библиотеки наряду с основными включают и различные сервисные программы.

Принципы разработки и рациональная структура фортрановского математического обеспечения для нужд оптимизации приведены в статье Гилла, Мюррея, Пиксна и Райт (1979). О процедурах выбора метода, реализованных в разных библиотеках и работающих по блок-схемам типа приведенной в разд. 8.1.1.1, можно прочесть, например, у Флетчера (1972c), Свенна и др. (1974) и в «Руководстве-справочнике по ФОРТРАН-библиотеке» (1981).

Стандарт MPS-формата подробно описан в документе фирмы IBM за номером H20-0476-2 («Система математического программирования 360, версия 2, линейное и сепарабельное программирование. Руководство пользователю», стр. 141–151). Более сжатое описание и иллюстративные примеры читатель найдет в инструк-

ции по системе MINOS (Муртаф и Сондерс (1977)) и в книге Муртафа (1981).

Сервисные процедуры для проверки правильности программ вычисления производных имеются во всех больших оптимизационных библиотеках. По этому поводу см. работы Моррея (1972b), Вулфа (1976) и Море (1979a).

8.2. СВОЙСТВА ЧИСЛЕННОГО РЕШЕНИЯ

8.2.1. ЧТО ТАКОЕ ПРАВИЛЬНЫЙ ОТВЕТ?

Вопрос о том, как определить, что найденную машиной точку можно считать «верным» решением, чрезвычайно сложен, и его детальный разбор потребовал бы отдельной книги. Поэтому здесь мы лишь вкратце коснемся некоторых основных моментов.

Испытание стандартной программы на задаче с известным ответом иногда дает обескураживающий результат: она может отыскать правильное решение и тем не менее выдать сообщение об отказе или, что еще хуже, остановиться в точке, совершенно не похожей на ожидаемую, и заявить об успешном завершении поиска. Причины, по которым исключить подобные казусы невозможно, рассмотрены ниже.

В разд. 2.1 уже отмечалось, что машины не способны находить абсолютно точные решения даже для очень простых задач. Поэтому требовать от них такие решения бессмысленно, и надо довольствоваться «хорошими» приближениями. Таким образом, встает проблема оценивания качества последних.

Принципиальная трудность состоит в том, что, ориентируясь на известные свойства аналитического решения, не всегда можно вынести правильное суждение о пригодности численного решения. Формальным выражением этих свойств обычно служит равенство

$$\Delta(x^*) = 0,$$

где Δ — некая векторная функция. Однако даже для ближайшей к x^* представимой в машине точки $\bar{x}$ значение $\Delta(\bar{x})$ вычисляемой версии Δ скорее всего окажется ненулевым. Таким образом, от «корректного численного решения» разумно добиваться только подчинения неравенству

$$\|\Delta(\bar{x})\| \leqslant \delta,$$

где δ — «малое» число. Конкретную величину δ следует выбирать очень аккуратно с учетом особенностей задачи (иногда «малым» можно считать и $\delta = 10$).

К сожалению, норма вектора Δ не обязательно (а точнее, редко) *непосредственно* характеризует расстояние до искомой точки: связи

$$\|x_1 - x^*\| < \|x_2 - x^*\| \Leftrightarrow \Delta(x_1) < \Delta(x_2), \qquad (8.5)$$

как правило, нет. При этом, чем хуже обусловлена задача, тем труднее хорошо оценить близость к x^* по *вычислимым критериям*. Отсюда и возникают значительные отклонения «корректных численных решений» от аналитических. Сказанное проиллюстрировано в разд. 8.2.2.1 на примере с плохо обусловленной квадратичной формой.

Теперь несколько слов о причинах, по которым программа может выдать сообщение об отказе, хотя в действительности найден «правильный» ответ. Таковые бывают разными, но и здесь дело обычно в том, что неверно оценивается точность имеющегося приближения. Какие бы приемы для этого ни привлекались, всегда находятся ситуации, в которых они оказываются несудачными. Признак отказа, означающий, что установленных критериев близости к x^* соблюсти не удается, часто является лишь следствием выхода на предел точности, обусловленный погрешностями машинной арифметики (см. разд. 8.2.2).

8.2.2. ПРЕДЕЛЬНАЯ ТОЧНОСТЬ РЕШЕНИЯ

Из-за того что машина считает с ошибками, для каждой задачи есть предельная точность численного решения. Не исключено, что ее удастся превысить, но рассчитывать на это нелзя. Хорошо оценить предельную точность бывает непросто, однако полезно по двум обстоятельствам. Во-первых, получив соответствующую оценку, намного легче выбрать разумные параметры в критериях останова. (Требование нереальной точности — одна из самых частых причин неформального завершения работы программ.) Во-вторых, сопоставление конечного результата с выводами предварительного анализа позволяет делать ценные заключения об обусловленности и отмасштабированности задачи.

Предельная точность численного решения сильно зависит от погрешностей вычисления функций задачи. Ниже эти погрешности описываются верхней границей $\varepsilon_д$ для абсолютной ошибки подсчета F в x^* ($\varepsilon_д$ может автоматически оцениваться программой по указанному пользователем порогу ошибки подсчета F в начальной точке; см. разд. 8.5.3).

8.2.2.1. Задачи без ограничений. Для гладких задач безусловной минимизации в роли вычислимых показателей качества приближений обычно используют значения целевой функции F и ее градиента. Допустим, известно, что результатом вычисления F в x^* будет

$$fl(F(x^*)) = F(x^*) + \sigma, \tag{8.6}$$

где σ — некое число, удовлетворяющее неравенству $|\sigma| \leqslant \varepsilon_д$. Тогда все точки x, для которых

$$|F(x) - F(x^*)| \leqslant \varepsilon_д, \tag{8.7}$$

приходится признать «неразличимыми»: каждая из них может оказаться численно неулучшаемой. Следовательно, (8.7) и есть основа для оценивания предельной точности. Любую $\bar{x}$, удовлетворяющую этому неравенству, будем называть *приемлемым решением*.

Общие соображения. Посмотрим, какие $\bar{x}$ выражение (8.7) допускает в случае, когда F дважды непрерывно дифференцируема, а x^* — точка сильного локального минимума, т. е. $g(x^*)=0$, и матрица $G(x^*)$ положительно определена. Для этого воспользуемся разложением F в окрестности x^* по Тейлору. Его можно записать так:

$$F(\bar{x}) = F(x^*+hp) = F(x^*)+\frac{1}{2}h^2p^TG(x^*)p + O(h^3), \quad (8.8)$$

где $\|p\|=1$ и $|h|=\|\bar{x}-x^*\|$. Отсюда следует, что граничные $\bar{x}$ должны удовлетворять соотношению

$$|F(\bar{x})-F(x^*)| \approx \frac{1}{2}h^2|p^TG(x^*)p| \approx \varepsilon_A$$

или, что эквивалентно,

$$|\bar{x}-x^*| \approx \frac{2\varepsilon_A}{p^TG(x^*)p}. \quad (8.9)$$

Из (8.9) хорошо видно, как влияет на значения $\|\bar{x}-x^*\|$ обусловленность матрицы $G(x^*)$: при плохо обусловленной $G(x^*)$ предельные уклонения $\bar{x}$ от x^* для разных направлений p оказываются существенно различными. В частности, при p, представляющих собой линейные комбинации тех собственных векторов $G(x^*)$, которым отвечают *наибольшие* собственные значения, правая часть (8.9) будет относительно малой, в то время как вдоль собственных векторов $G(x^*)$, соответствующих *наименьшим* собственным значениям, допускаются намного более сильные уклонения. Из сказанного ясно, что для плохо обусловленной задачи ошибка приемлемого решения по некоторым направлениям может оказаться весьма ощутимой.

В связи с тем что одним из наиболее употребительных критериев качества приближений является малость нормы градиента, полезно оценить и предельную точность совпадения $g(\bar{x})$ с нулем. Разложим $g(x)$ в окрестности x^* по Тейлору:

$$g(\bar{x}) = g(x^*+hp) = hG(x^*)p + O(h^2)$$

(напомним, что $g(x^*)=0$). Чтобы получить искомую оценку, отбросим остаточный член, возведем полученное приближенное равенство в квадрат и подставим вместо h^2 правую часть (8.9). Это даст соотношение вида

$$|g(\bar{x})|^2 \approx 2\varepsilon_A\frac{p^TG(x^*)p}{p^TG(x^*)p}. \quad (8.10)$$

Снова легко просматривается влияние обусловленности $G(x^*)$. Если она плохая, то норма $\|g(\bar{x})\|$ будет большой, когда p есть линейная комбинация собственных векторов $G(x^*)$, отвечающих *наибольшим* собственным значениям, и малой, когда соответствующие собственные значения относятся к группе *наименьших*.

Пример 8.1. Для иллюстрации последней плохой обусловленности матрицы Гессе рассмотрим квадратичную форму

$$F(x_1, x_2) = (x_1 - 1)^2 + 10^{-4}(x_2 - 1)^2 + 1.$$

Ясно, что ее минимум достигается в $x^* = (1, 1)^T$. Пусть $\varepsilon_A = 10^{-6}$. Тогда точка $\bar{x} = (1, 2)^T$ будет граничным приемлемым решением для нее $F(\bar{x}) = 1 + 10^{-4}$. При этом $\|\bar{x} - x^*\|_2 = 1$. Еще одно граничное приемлемое решение — точка $\bar{x} = (1 + 10^{-3}, 1)^T$; для нее тоже имеет место равенство $F(\bar{x}) = F(x^*) + 10^{-6}$. Если судить по удаленности от x^*, то $\bar{x}$ намного лучше, чем $\bar{x}$. Однако $\|g(\bar{x})\|_2 = 2 \times 10^{-4}$, а $\|g(\bar{x})\|_2 = 2 \times 10^{-6}$, т. е. по близости градиента к нулю — стандартному критерию качества — «лучшей» оказывается $\bar{x}$. Этот пример подтверждает высказанное в разд. 8.2.1. положение о том, что вычислимые критерии качества лишь косвенно отражают расстояние до искомой точки.

Хорошо отмасштабированные задачи. Чтобы представить себе характерные порядки предельных ошибок, надо рассмотреть, какими они будут для «хорошо отмасштабированной» задачи (разд. 8.7). Имеется в виду тот случай, когда число обусловленности матрицы Гессе $G(x^*)$ не «очень велико» и $F(x^*)$, $\|x^*\|$ есть величины порядка единицы. В этих предположениях из (8.9) и (8.10) следуют оценки вида

$$\|\bar{x} - x^*\| = O(\sqrt{\varepsilon_A})$$

и

$$\|g(\bar{x})\| = O(\sqrt{\varepsilon_A}).$$

Среди прикладных программистов уже давно бытует мнение, что если F хорошо отмасштабирована, то в нормах приемлемого решения будет *примерно вдвое* меньше правильных разрядов, чем в расчетных значениях F; первое из представленных соотношений вполне согласуется с ним. (По поводу относительной точности отдельных компонент $\bar{x}$ ничего сказать нельзя.) Более того, аналогичная оценка справедлива и для нормы градиента.

Обычно $\varepsilon_A \gg \varepsilon_M$; это значит, что число правильных разрядов в $\|\bar{x}\|$, как правило, не превосходит половины длины мантиссы. Однако, если F удается вычислять с повышенной точностью, реальны и меньшие ошибки (см. разд. 8.5.1.3).

Негладкие задачи. До сих пор речь шла о дважды непрерывно дифференцируемых функциях. Если же $F(x)$ разрывна и, в частности, имеет разрыв в x^*, неравенством (8.7) будут исключаться даже некоторые из сколь угодно близких к x^* точек. Соответствующая ситуация изображена на рис. 8b. Хотя $\bar{x}_1$ и $\bar{x}_2$ отстоят от x на одинаковое расстояние, приемлемым решением можно назвать только $\bar{x}_2$. Чем «больше» окрестность x^*, в которой $F(x) \approx \approx F(x^*)$, тем больше вероятность попасть в нее; сказать что-нибудь более конкретное в общем случае нельзя.

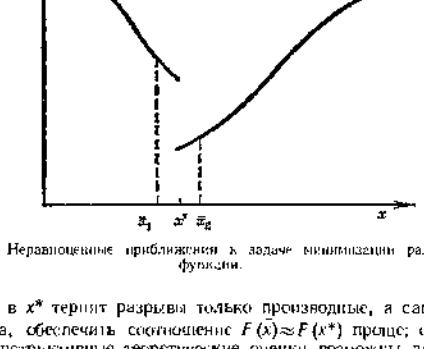

Рис. 8b. Неравноценные приближения к задаче минимизации разрывной функции.

Если в x^* терпят разрывы только производные, а сама F непрерывна, обеспечить соотношение $F(\bar{x}) \approx F(x^*)$ проще; однако и здесь конструктивные теоретические оценки возможны лишь при весьма жестких предположениях.

8.2.2.2. Точность в задачах с ограничениями. Предельную точность численного решения задачи с линейными равенствами и неравенствами можно оценить, анализируя отклонения от x^* в двух взаимно ортогональных подпространствах — в нуль-пространстве и ранг-пространстве матрицы $\hat{A}$ активных ограничений.

Точность в ранг-пространстве: линейные ограничения. Ниже показано, что предельная точность проекции численного решения на нуль-пространство $\hat{A}$ определяется не ошибкой подсчета F, а обусловленностью $\hat{A}$.

Искомая точка x^* удовлетворяет равенству

$$\hat{A}x^* = b,$$

причем, не умаляя общности, можно считать, что $\|\bar{A}\|$ и $\|\bar{b}\|$ суть величины порядка единицы. При этом, если $\bar{x}$ есть результат применения какого-нибудь из методов, рассмотренных в гл. 5, получим

$$\bar{A}\bar{x} = \bar{b} + O(\varepsilon_M). \tag{8.11}$$

Обозначим через Y матрицу, чьи столбцы формируют базис подпространства, натянутого на столбцы $\bar{A}^T$, и пусть Z — матрица базиса его ортогонального дополнения. Тогда $\bar{x}$ можно представить так:

$$\bar{x} = x^* + Zp_Z + Yp_Y.$$

Подстановка правой части этого равенства в (8.11) дает

$$\bar{A}Yp_Y = O(\varepsilon_M).$$

Отсюда видно, что порядок величины $\|Yp_Y\|$ *связан с обусловленностью* $\bar{A}$. При хорошо обусловленной $\bar{A}$ эта величина того же порядка, что и ε_M. Если же $\bar{A}$ обусловлена плохо, норма $\|Yp_Y\|$ может оказаться большой, даже когда невязки активных ограничений в $\bar{x}$ почти нулевые.

Небезынтересно посмотреть, как возмущение Yp_Y отражается на значении F. Разлагая F в ряд Тейлора в окрестности x^*, получим

$$\begin{aligned} F(x^* + Yp_Y) &= F(x^*) + g(x^*)^T Yp_Y + O(\|Yp_Y\|^2) = \\ &= F(x^*) + \lambda^{*T}\bar{A}Yp_Y + O(\|Yp_Y\|^2) \end{aligned} \tag{8.12}$$

(напомним, что $g(x^*) = \bar{A}^T\lambda^*$). В предположении хорошей обусловленности $\bar{A}$ из (8.12) следует, что

$$F(x^* + Yp_Y) - F(x^*) \approx \lambda^{*T}\bar{A}Yp_Y = O(\varepsilon_M\|\lambda^*\|).$$

При плохо обусловленной $\bar{A}$ это неверно, так как тогда остаточный член в (8.12) отбрасывать нельзя.

Точность в нуль-пространстве; линейные ограничения. Оценивание предельной точности составляющей численного решения из нуль-пространства $\bar{A}$ проводится по схеме, аналогичной использованной в разд. 8.2.2.1 для случая без ограничений. Чтобы упростить изложение, допустим, что уклонений в ранг-пространстве нет, т. е. $\bar{x} = x^* + Zp_Z$. Тогда, полностью повторяя рассуждения разд. 8.2.2.1, получим

$$\|\bar{x} - x^*\|^2 \approx \frac{2\varepsilon_A}{p_Z^T Z^T G(x^*) Zp_Z}$$

и

$$\|Z^T g(x^*)\|^2 \approx 2\varepsilon_A \frac{|Z^T G(x^*) Zp_Z|^2}{p_Z^T Z^T G(x^*) Zp_Z}.$$

Отличие этих соотношений от (8.9) и (8.10) только в том, что место полной матрицы Гессе заняла спроектированная. Таким образом, обусловленность последней сказется на точности в нуль-пространстве абсолютно так же, как обусловленность полной матрицы Гессе сказывалась в задаче без ограничений.

Когда задача хорошо отмасштабирована и хорошо обусловлена, ошибки из нуль-пространства будут доминирующими. Поэтому для такого случая приведенных соображений достаточно. Разбор ситуации с плохо обусловленной A сложнее и выходит за рамки данной книги.

Нелинейные ограничения. Для хорошо отмасштабированных и хорошо обусловленных задач с *нелинейными* ограничениями оценки достижимой точности те же, что и для задач, ограничения которых линейны (только вместо $\hat{A}$ надо брать $\hat{A}(x^*)$, а вместо $G(x^*)$ — матрицу Гессе функции Лагранжа $W(x^*, \lambda^*)$). В отношении плохо обусловленных задач этого сказать нельзя: здесь нелинейность ограничений осложняет дело.

8.2.3. КРИТЕРИИ ОСТАНОВА

8.2.3.1. Необходимость критериев останова. Почти все оптимизационные методы дают решение лишь в пределе (в том числе и те, которые в отсутствие погрешностей вычислений были бы конечными). Поэтому в реализующих их программах приходится предусматривать специальные правила прерывания счета — критерии останова. Пользователь должен хорошо понимать эти критерии, так как от него обычно требуются значения их параметров и к тому же без этого трудно объективно оценить результаты работы программы.

Оптимизационный процесс желательно прервать, (i) если достигнута требуемая точность решения; (ii) если хорошее приближение еще не найдено, но скорость продвижения к оптимуму так упала, что нет смысла считать дальше; (iii) если метод начал расходиться (из-за отсутствия у задачи приемлемого решения) или зациклился. Таким образом, хорошие критерии останова должны обеспечивать выход из программы, когда получена подходящая точка, гарантировать минимум риска преждевременного прерывания и не допускать бессмысленных затрат машинного времени.

Возможность принять x_k в качестве численного решения определяется на основе условий оптимальности и признаков приближения последовательности (x_k) к пределу. К сожалению, для многих задач удается использовать лишь некоторые из условий, причем даже их точного соблюдения требовать нельзя (см. разд. 8.2.1). Поэтому и приходится существенно опираться на тесты второго типа. Сами по себе они тоже ненадежны, признаки близости преде-

да могут быть следствием плохой работы метода и уж во всяком случае не исключают сходимости и неминимальную точку.

Важно подчеркнуть, что *критерии останова, подходящие для всех задач и всех методов, не существуют*. Каждый из них был сконструирован в некоторых предположениях, и для каждого можно указать ситуацию, где его применение приведет либо к ненужному затягиванию, либо к преждевременному завершению счета.

Ниже рассмотрены величины, используемые в критериях останова, и обсуждается ряд вопросов, связанных с оценкой качества приближений.

8.2.3.2. Критерии останова для безусловной оптимизации. Представленные в данном разделе условия окончания счета ориентированы на задачи без ограничений и включают лишь один свободный параметр. Он имеет смысл желаемой точности решения. Существуют две естественные формы запроса точности: первая — когда указывается желаемая близость к оптимуму по точке, вторая — по значению целевой функции. В определенных случаях между соответствующими уклонениями есть хорошая связь (см. разд. 8.2.2.1), и тогда не важно, какую форму принять. Однако, *если задача плохо обусловлена, постоянное значение F реализуется для точек x_k, которые очень по-разному удалены от x^**, и здесь ожидать хорошей близости x_k к x^* нельзя; рассчитывать можно только на близость F_k к $F(x^*)$. Поэтому мы предлагаем регулировать точность параметром τ_F — числом правильных разрядов F_k, которое хотелось бы получить. Подразумевается, что для F_k, не превосходящих по модулю единицы, старшие разряды после десятичной точки должны учитываться, даже если в них стоят нули. Например, выбор $\tau_F = 10^{-4}$ означает, что при $|F(x^*)| \geqslant 1$ желательно, чтобы у F_k и $F(x^*)$ совпали шесть первых значащих цифр, а при $|F(x^*)| < 1$, чтобы F_k и $F(x^*)$ различались не более чем на 10^{-4}.

Гладкие задачи. При решении задач безусловной минимизации гладких функций неплохим индикатором удовлетворения в x_k сформулированного выше запроса на точность служат неравенства

U1. $F_{k-1} - F_k < \Theta_k$;

U2. $\|x_{k-1} - x_k\| < \sqrt{\tau_x}\,(1 + |x_k|)$;

U3. $|g_k| \leqslant \sqrt[3]{\tau_f}\,(1 + |F_k|)$,

где Θ_k — мера абсолютной погрешности, вычисляемая по формуле

$$\Theta_k = \tau_F\,(1 + |F_k|) \qquad (8.13)$$

(доводы в пользу этого определения приводятся ниже). Первые два неравенства суть признаки близости последовательностей $\{F_k\}$ и $\{x_k\}$ к своим пределам. Третье представляет собой огрубление известного условия оптимальности $\|g(x^*)\| = 0$.

Отметим, что, хотя речь шла только о поиске хорошего значения F_k, а об уклонении x_k от x^* ничего не говорилось, в список включено условие U2, относящееся непосредственно к x_k. Для правильно отмасштабированных задач его можно не вводить — оно будет следствием U1. Однако для прочих задач условие U2 полезно и заставит алгоритм отыскивать более подходящие точки. Нарушение условия U2 при выполнении U1 может означать следующее: либо направление, выбранное в x_{k-1}, было почти ортогонально градиенту, либо F имеет пологий минимум. В том и другом случае имеет смысл продолжить поиск.

Суть приведенных условий прозрачна, но их конкретная формулировка требует некоторых пояснений. Прежде всего, казалось бы, естественнее взять θ_k равным $\tau_F|F_k|$. Однако тогда при малых $|F_k|$ условие U1 было бы слишком жестким. Как правило, такие F_k получаются сложением существенно отличных от нуля величин противоположных знаков и, следовательно, вычисляются с низкой относительной точностью (см. разд. 8.5.1.3). Поэтому для оценки близости F_{k-1} к F_k и при нулевом F_k и при F_k порядка единицы разумно использовать один эталон. Именно это соображение определило выбор формулы вычисления θ_k.

Исходя из результатов разд. 8.2.2.1, соблазнительно заменить правую часть условия U2 на квадратный корень из $\theta_k'\|G(x_k)\|$. Однако, так как величина $\|G(x_k)\|$ обычно недоступна, полученное условие было бы непрактичным. В то же время для хороших задач оно не так уже сильно отличалось бы от U2. Присутствие $\|x_k\|$ справа в условии U2 оправдано естественным желанием ослабить зависимость оценки близости x_{k-1} к x_k от масштаба переменных, а единичная добавка к $\|x_k\|$ полезна по тем же соображениям, что и в (8.13).

Некое недоумение может вызвать использование в условии U3 величины $\sqrt[3]{\tau_F}$ — ведь в соответствии с оценками разд. 8.2.2.1 надо было бы взять квадратичный корень. Тем не менее это не описка: практика показывает, что удовлетворять условию U3 с $\sqrt{\tau_F}$ слишком трудно и в присутствии U1, U2 его вполне можно смягчить. Аналогичного ослабления условия U2 не требуется — если подставить в U2, U3 одинаковые степени τ_F, то условие U2 почти всегда будет следствием U3 (но не наоборот).

Условия U1—U3 срабатывают, когда, попав в «хорошую» точку x_{k-1}, алгоритм делает еще один шаг. На случай, если x_{k-1} окажется настолько близкой к x^*, что выйти из нее с уменьшением F не удастся, предусматривают альтернативный останов по признаку U4. $\|g_k\| \leqslant \varepsilon_A$.

В методах ньютоновского типа наряду с приведенными полезно использовать условие существенной положительной определенности матрицы $G(x_k)$ (в дискретных ньютоновских методах — с конечно-разностной аппроксимации). Следует также отметить, что вместо

аналитических значений первых производных в условия U3 и U4 можно подставлять их численные оценки.

Какую норму лучше всего взять для U2—U4, зависит от числа переменных n. Если оно невелико, подойдет евклидова норма. При больших n разумнее использовать норму l_1 (максимум модуля по компонентам), так как при евклидовой норме удовлетворить условиям U3, U4 становится слишком тяжело. Иногда применяют также «псевдонормы», масштабируемые в соответствии с величиной n.

Негладкие задачи. Критерии останова для алгоритмов негладкой минимизации (типа метода многогранника из разд. 4.2.2) обычно сводятся к одному-единственному условию — требуется, чтобы некое изменение функции было меньше заданного порога. К примеру, на каждой итерации метода многогранника определено $n+1$ точек $x_1, x_2, \ldots, x_n, x_{n+1}$ и соответствующих значений F, причем точки упорядочиваются так, что $F_1 \leqslant F_2 \leqslant \ldots \leqslant F_{n+1}$; здесь и в качестве упомянутого условия берут неравенство

$$F_{n+1} - F_1 < \tau_F (1 + |F_1|). \tag{8.14}$$

Из-за того что у производных F могут быть разрывы, никаких более тонких тестов предложить нельзя.

8.2.3.3. Критерии останова для минимизации при линейных ограничениях. Ниже приводится перечень требований, выполнение которых может служить признаком успешного завершения поиска минимума при линейных ограничениях. Они ориентированы на методы активного набора, использующие ортогональный базис нуль-пространства матрицы $\hat{A}$ (см. разд. 5.1 и 5.2). Между ними и условиями, рассмотренными ранее, есть два принципиальных отличия. Во-первых, если хотя бы одно из ограничений обращается в x_k в равенство, то место полного градиента g_k займет *спроектированный* $g_Z(x_k)$. Во-вторых, вводится дополнительное требование к оценкам множителей Лагранжа активных неравенств; тем самым контролируется правильность финального активного набора.

Пусть t_k — полное число ограничений в рабочем списке k-й итерации; λ_k — вектор оценок их множителей Лагранжа (равенства t_k); σ_{min} — минимальная среди оценок множителей активных *неравенств*; λ_{max} — оценка множителя (равенства или неравенства), имеющая максимальную норму; θ_k — параметр, вычисляемый по формуле (8.13). Точку x_k, где есть активные ограничения (иначе надо было бы обратиться к условиям из предыдущего раздела), предлагается считать численным решением, если

LC1. $F_{k-1} - F_k < 0_k$.

LC2. $\|x_{k-1} - x_k\| < \sqrt{\tau_F}(1 + \|x_k\|)$.

LC3. $\|g_Z\| \leqslant \sqrt[3]{\tau_F}\|g_k\|$.

LC4. (Это условие надо проверять, только когда в рабочем списке есть хоть одно ограничение-неравенство.) Если $t_k > 1$, то $\sigma_{\min} \geqslant \sqrt{\tau_F}\lambda_{\min}$; если $t_k = 1$, то $\sigma_{\min} \geqslant \sqrt{\tau_F}(1 + |F_k|)$.

Аналогичным U4 алгеритивным признаком целесообразности останова теперь станет выполнение условий **LC4** и

LC5. $\|g_Z\| < \varepsilon_A$.

Отметим, что проверка малости невязок ограничений рабочего списка в представленный перечень тестов не включена: они адресуются методам с ортогональной факторизацией, в которых такие невязки заведомо будут нулевыми. Однако если иметь в виду иные методы, в частности применяемые для решения задач большой размерности, то подобная проверка необходима (см. разд. 5.6).

8.2.3.4. Критерии останова для минимизации при нелинейных ограничениях. Гладкие задачи. Поскольку методы решения задач с нелинейными ограничениями могут использовать принципиально разные стратегии поиска, в критериях их останова тоже возможны значительные варианты. Обширный класс составляют методы с адаптивными подзадачами (см. разд. 6.1.2.1). В них соотношения, по которым выносится суждение о качестве текущей точки, проверяются только в моменты завершения «основных» итераций (т. е. когда решена очередная подзадача). Обычно здесь рассматриваются изменения некоторых величин (оценок множителей Лагранжа, значений функции выигрыша и т. д.) в результате последней основной итерации. В методах с детерминированными подзадачами (разд. 6.1.2.1) применяются другие критерии, причем и они варьируются в зависимости от характерных свойств последовательностей генерируемых приближений $\{x_k\}$. Например, методы типа приведенных градиентов, сохраняющие допустимость, дают последовательности $\{x_k\}$, для которых векторы шагов из x_{k-1} в x_k оказываются к нуль-пространству матрицы активных ограничений $\bar{A}(x^*)$. Если задача хорошо отмасштабирована, из этого следует, что вблизи решения будет

$$F_{k-1} - F_k = O(\|x_k - x_{k-1}\|^2)$$

(для методов, подходящих к решению «нетангенциально», такой связи нет).

Ориентируясь на случаи, когда активный набор, соответствующий искомой точке, не пуст, можно порекомендовать три условия останова, подходящих для любого метода минимизации при нелинейных ограничениях:

NC1. $\|c_k\| \leqslant \tau_C$.

NC2. $\|g_Z\| \leqslant \sqrt{\tau_I}\|g_k\|$.

NC3. (Это условие надо проверять, только когда в рабочем списке есть хоть одно ограничение-неравенство.) Если

$t_k > 1$, то $\sigma_{\text{опт}} \geqslant \sqrt{\tau_F}/\lambda_{\max}$|; если $t_k = 1$, то $\sigma_{\text{опт}} \geqslant \sqrt{\tau_F}(1 + |F_k|)$.

Здесь θ_k, t_k, λ_{k1}, $\sigma_{\text{опт}}$ и $\lambda_{\max}$ — величины, определенные в предыдущем разделе, $\hat{c}_k$ есть вектор-функция активных в x_k ограничений, а τ_C — допуск на норму их невязок. Заметим, что в правой части аналогичного LC3 условия NC2 стоит не кубический корень из τ_F, а квадратный, т. е. оно жестче чем LC3. Это усиление целесообразно в связи с тем, что от теста типа U1 (или LC1) приходится отказаться (он годится лишь для упомянутых выше методов типа приведенных градиентов).

В программах методов с детерминированной подзадачей естественно использовать также следующее условие останова:

NC4. $\|x_{k-1} - x_k\| < \sqrt{\tau_F}(1 + \|x_k\|)$.

При этом нужно предусмотреть и альтернативный набор «одноточечных» условий. Туда можно включить, например, неравенства NC2, NC3 и

NC5. $\|\hat{c}_k\| < \varepsilon_A$

Негладкие задачи. Для решения негладких задач с полиэдральными ограничениями обычно применяют методы, подобные описанному в разд. 6.2.2.2. В частности, для него разумным признаком успешного завершения поиска будет выполнение неравенства (8.14) и

$$\sum |\hat{c}_i(x_1)| \leqslant \tau_C,$$

где x_1 — вершина с минимальным значением штрафной функции, τ_C — приемлемое значение суммы невязок активных ограничений.

8.2.3.5. Условия аварийного прерывания. Иногда оптимизирующую программу надо остановить до того, как реализуются соотношения, при которых лучшее из просмотренных приближений можно принять в качестве искомой точки. Естественно, такой останов необходим, когда в спецификациях задачи, составленных пользователем, обнаруживаются ошибки; например, параметру, указывающему число переменных, присвоено отрицательное значение. Еще одна простая причина для прерывания счета с признаком неудачи — выполнение максимального дозволенного (пользователем) числа итераций или обращений к процедуре подсчета целевой функции (см. разд. 8.1.3.5).

Разрабатывая стандартные программы, аварийный останов стараются обеспечить также в случаях, когда возможность «сущимого прогресса» при продолжении счета становится сомнительной (см. разд. 8.4.3). Поскольку понятие прогресса для разных стратегий поиска определяется по-разному, критерии останова по указанному признаку тоже бывают различными.

Почти во всех оптимизационных алгоритмах естественно ставить условие «существенного убывания» применяемой функции выигрыша на каждой итерации. Невозможность удовлетворить этому условию служит надежным сигналом о каких-то неполадках (см. разд. 8.4.2). Весьма распространенным и, по нашему мнению, неудачным критерием отказа является близость к нулю производной функции выигрыша по очередному направлению поиска. Более разумно определять в x_k минимальное приемлемое расстояние δ_k до следующей точки (величина δ_k должна быть аккуратной оценкой той нормы $\|x_{k+1}-x_k\|$, при которой гарантирован правильный знак разности вычисляемых в x_{k+1} и x_k значений функций выигрыша) и считать, что алгоритм отказал, если он не может получить лучшего, чем x_k, приближения, сделав шаг, превосходящий δ_k.

8.2.3.6. Выбор параметров в критериях останова. Ниже речь пойдет в основном о выборе допуска τ_F для представленных в разд. 8.2.3.2 условий завершения поиска минимума без ограничений. Неудачный выбор τ_F может привести к серьезным потерям, так что к назначению τ_F следует подходить ответственно.

Казалось бы, чтобы не затягивать счет понапрасну, τ_F надо брать равным желаемой точности решения, а она определяется существом задачи: если, скажем, F описывает некую реальную зависимость с относительной ошибкой 3%, то и минимум точнее искать незачем. Однако, поскольку строгих гарантий близости F_k к $F(x^*)$ условия останова не дают, получив точку, удовлетворяющую им «без запаса», трудно бывает поверить, что достигнута максимальная близость, на которую позволительно надеяться при соответствующем τ_F. Иное дело, когда допустимая погрешность в оценке $F(x^*)$ составляет 3%, а условия выполнены для $\tau_F=0.0001\%$; тут, наверное, каждый был бы уверен, что желаемая точность получена, и почти наверняка не ошибся бы.

Недостаточно жесткие требования к окончательному приближению опасны тем, что могут выполняться вдали от искомого решения. Возьмем, к примеру, функцию одной переменной, изображенную на рис. 8с. Если при поиске ее минимума в качестве критерия останова использовать неравенство $|g(x)|<\sigma(1+|F(x)|)$ и положить $\sigma=0.05$, то на роль «численного решения» будут претендовать все точки, отвечающие обведенным участкам графика. Среди них есть и не имеющие к x^* никакого отношения. Совсем иная картина возникает при $\sigma=10^{-8}$: точки, подчиняющиеся неравенству с таким параметром, удается найти лишь вблизи x^*. Разумеется, нетрудно построить аналогичную по виду, но хуже отмасштабированную функцию, для которой и условие $|g(x)|<10^{-8}(1+|F(x)|)$ реализуемо в далеких от x^* точках, да только на практике подобные функции встречаются редко. Обычно же жестким критериям останова вне малой окрестности x^* не удовлетворить.

Наименьшее возможное значение τ_F определяется предельной

достижимой точностью. Оно обеспечивает максимальную надежность численного решения, но прибегать к нему можно лишь в алгоритмах со сверхлинейной сходимостью. В них вблизи x^* точность последовательных приближений растет очень быстро, так что переход от «достаточно малого» τ_F к «минимальному» не повлечет чрезмерного затягивания счета. При этом проявление сверхлинейной

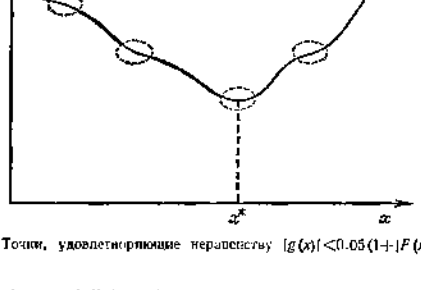

Рис. 8с. Точки, удовлетворяющие неравенству $|g(x)|\leqslant 0.05(1+|F(x)|)$.

сходимости на последних итерациях поиска «очень точного» решения послужит дополнительным подтверждением близости найденной точки к x^* (см. разд. 8.3.1). Если же алгоритм таков, что для τ_F есть порог, за которым скорость увеличения времени поиска с уменьшением τ_F резко возрастает, то использовать предельное значение τ_F не стоит: слишком долго придется ждать ответа. В частности, сказанное относится к методу многогранника — даже когда речь идет о хорошей гладкой функции, задавать ему τ_F, меньшие чем 10^{-s}, не следует.

Промежуточное положение занимают алгоритмы сопряженных градиентов, эффективные скорости сходимостей которых чаще всего линейны. Если мантисса машинного представления числа содержит $s\geqslant 10$ десятичных разрядов, для них обычно подходят τ_F порядка $10^{-s/2}$; при меньших s, как правило, можно взять $\tau_F=10^{-s}$. Такие τ_F позволяют надеяться, что у полученного в точке останова значения $F(x_k)$ будет примерно $\min\{^s/_2 s, 4\}$ верных разрядов. Для большинства прикладных задач этого вполне достаточно, но вот только полной уверенности, что надежды оправдаются, нет.

Когда аналитические значения первых производных недоступны и заменяются конечно-разностными оценками, предельная достижимая точность, вообще говоря, падает, т. е. минимальное разумное значение τ_F увеличивается. Причина в том, что погрешности ап-

проксимации градиента не позволяют правильно ориентировать направления поиска вблизи искомой точки (где градиент равен нулю).

В хорошей библиотеке оптимизационного математического обеспечения для каждого метода будет определена своя рекомендуемая величина τ_F. Кроме того, может предусматриваться программный контроль пригодности значения τ_F, заданного пользователем, и замена этого значения, коль скоро оно окажется неудовлетворительным. Тогда, например, если пользователь зафиксировал такой параметр τ_F, что θ_k получается меньше, чем оценка абсолютной ошибки вычисления F, программа сама увеличит τ_F. (Имея в виду возможность автоматической замены, не стоит прибегать к явно завышенным τ_F как к способу обеспечить быстрый останов стандартной процедуры. Обычно для этого есть специальный параметр (см. разд. 8.1.3.5).)

Теперь несколько слов о выборе τ_C в условиях останова для методов минимизации при нелинейных ограничениях. В принципе этот параметр может назначаться независимо от τ_F, и обычно его берут значительно меньшим. При этом типичная картина поиска такова: сначала обеспечивается соблюдение условия малости нормы спроектированного градиента, а затем при незначительных колебаниях последней сокращаются невязки в ограничениях.

Замечания и избранная библиография к разделу 8.2

При поиске минимума или нуля функции одной переменной используют специальные критерии останова, требующие задания финального интервала неопределенности (см. разд. 4.1). Подробности относительно этих критериев и детальное обсуждение пределов точности для одномерных задач читатель найдет в книге Брента (1973a).

Вопросы построения критериев останова для программ минимизации при нелинейных ограничениях близки вопросам оценки их функционирования; по этому поводу см. работу Гилла и Мюррея (1979c).

Об условиях завершения счета при решении систем нелинейных уравнений и нелинейных задач о наименьших квадратах можно прочесть у Деннса (1977), Деннса, Гэя и Уэлша (1977), Море (1979b). Особый класс составляют так называемые задачи с малыми невязками. Для них целевую функцию удается вычислять с необычно высокой точностью (см. разд. 8.5.1.3), и это находит отражение в соответствующих условиях.

Исчерпывающий анализ связи между приближенным решением системы линейных алгебраических уравнений и вектором их невязок дан в книге Уилкинсона (1965).

8.3. АНАЛИЗ РЕЗУЛЬТАТОВ СЧЕТА

8.3.1. ОЦЕНКА ПРИГОДНОСТИ ЧИСЛЕННОГО РЕШЕНИЯ

К сожалению, даже высококачественная оптимизационная программа может выдать сообщение об отказе, хотя найдено подходящее приближение, или признать таковым негодную точку. Поэтому к любой оценке результата счета самой программой следует относиться критически. По нашему мнению, хороший стандартный алгоритм не должен «спешить с выводом» об успешном завершении поиска. Тогда по крайней мере при нормальном останове больше шансов будет за то, что действительно получено удовлетворительное решение. Однако и здесь не помешают дополнительные тесты. Они особенно полезны, когда высокая точность минимизации не нужна, и поэтому к окончательному приближению предъявляются не очень жесткие требования.

8.3.1.1. Безусловная минимизация. Получив по окончании поиска минимума без ограничений точку x_k, имеет смысл проверить

(i) выполнено ли неравенство $\|g(x_k)\| \leqslant \|g(x_0)\|$;

(ii) достигнута ли на завершающих итерациях высокая скорость сходимости;

(iii) можно ли оценить число обусловленности матрицы Гессе (или ее приближения) небольшой величиной.

Если результаты всех трех проверок окажутся положительными, то независимо от того, как остановилась программа — с признаком отказа или успеха, x_k скорее всего будет приемлемым решением.

Из сказанного вытекает, например, что при $\|g(x_0)\|$ порядка 10^{10} вполне можно довольствоваться единичным значением $\|g(x_k)\|$. В подобной ситуации почти наверняка удовлетворится и критерий U3 из разд. 8.2.3.2. Однако неравенство U3 бывает также следствием плохой нормировки производных. Тогда при выполненном U3 проверка (i) даст отрицательный ответ. Именно в этом ее ценность. (В условия останова требование существенного уменьшения нормы градиента не включают, так как при хорошем начальном приближении оно невыполнимо.)

Многие алгоритмы (ньютоновского и квазиньютоновского типа) вблизи решения должны обеспечивать сверхлинейные скорости сходимости приближений, и проявление таких скоростей на заключительных итерациях поиска — один из самых надежных признаков его успешного завершения. С линейно сходящимися алгоритмами дело обстоит сложнее: линейная сходимость часто бывает медленной и тогда распознать ее крайне трудно; в то же время как раз эти случаи обычно оказываются спорными, поскольку отсутствие значительного прогресса в течение большого числа итераций служит общепринятым условием аварийного останова (см. разд. 8.4.3).

Скорость сходимости можно оценивать по значениям норм градиентов $\|g_k\|$ или разностей $\xi_k = F_{k-1} - F_k$ на нескольких

(скажем, пяти) последних итерациях. Соотношения $\xi_{k+1} \approx \xi_k^r$ с $r > 1$ укажут на сверхлинейную сходимость, а быстрая линейная предполагает, что $\xi_{k+1} \approx \xi_k/M$, где $M > 2$.

Идеальную картину проявления сверхлинейной (а точнее, квадратичной) сходимости дают сведенные в табл. 8a характеристики

Таблица 8a. Заключительные итерации поиска минимума функции Розенброка методом ньютоновского типа

k	F_k	$\lvert(x^*)_1 - (x_k)_1\rvert$	$\lvert(g_k)_1\rvert$
10	2.05×10^{-2}	1.1×10^{-1}	8.0
11	6.27×10^{-5}	3.0×10^{-3}	$a.0 \times 10^{-1}$
12	1.74×10^{-7}	4.0×10^{-4}	7.0×10^{-3}
13	5.13×10^{-13}	4.0×10^{-7}	2.0×10^{-6}
14	1.08×10^{-24}	6.0×10^{-13}	3.0×10^{-11}

завершающих итераций поиска минимума функции Розенброка (см. пример 4.2 и рис. 4k)

$$F(x_1, x_2) = (x_1^2 - x_2)^2 + 100\,(1 - x_1^2)$$

ньютоновским методом (см. разд. 4.4.1). Здесь все ясно и без последовательности $\{\xi_k\}$. (Отметим, что функцию Розенброка в окрестности минимума удается вычислять с высокой точностью; см. разд. 8.5.1.3.)

Таблица 8b. Итерации поиска минимума функции Розенброка методом наискорейшего спуска

k	F_k
34	1.87
35	1.83
36	1.79
37	1.71
38	1.65
99	9.20×10^{-1}
.	.
.	.
.	.
395	2.418×10^{-3}
396	2.411×10^{-3}
397	2.405×10^{-3}
398	2.397×10^{-3}

Результаты применения к той же задаче метода наискорейшего спуска (см. разд. 4.3.2.2) содержатся в табл. 8b. Это — пример очень медленной линейной сходимости. Подобная картина, вообще говоря, подсказывает (в рассматриваемом случае правильно), что до решения еще далеко.

На практике крайние ситуации типа представленных выше встречаются редко; обычно наблюдается нечто среднее между ними. В подтверждение этого тезиса приведена табл. 8c с данными решения одной прикладной задачи. Левая колонка этой таблицы похожа на правую в табл. 8b, но по разностям $\{\xi_k\}$ видно, что итерации сходятся достаточно быстро, хотя и с линейной скоростью.

При анализе последовательности $\{\xi_k\}$ не надо забывать, что, выйдя на предельную точность решения, ни одна программа не даст большего, чем медленная линейная сходимость. Если сверхлинейно сходящемуся алгоритму поставить очень жесткие условия останова,

то, достигнув своей «нормальной» скорости, он быстро наберет эту точность, а затем резко замедлится). Поэтому важно (и особенно в случае аварийного останова) просмотреть не только самые последние ξ_n, но и их предшественников. Стоит упомянуть также еще одно аналогичное явление, изредка встречающееся при использовании ме-

Таблица 8с. Заключительные итерации квазиньютоновского метода для одной прикладной задачи

F_k	ξ_k	ξ_{k+1}/ξ_k
0.986 333 666 2	—	—
0.986 056 099 9	2.8×10^{-4}	2.0×10^{-2}
0.986 050 500 0	5.6×10^{-6}	8.4×10^{-2}
0.986 050 027 8	4.7×10^{-7}	4.3×10^{-3}
0.986 050 027 4	2.0×10^{-9}	—

тодов ньютоновского типа: бывает, что из плохой точки x_{k-1} делается шаг в «очень хорошую» x_k, откуда выйти уже не удается — мешают ошибки округления. Тогда условия нормального останова могут не сработать (U1, U2, U3 из-за существенного различия между x_{k-1} и x_k, а «одноточечное условие» U4 из-за недооценки погрешности вычисления F) и программа выдаст сообщение об отказе. Эта ситуация надежно распознается по неравенству $\|g_k\| \ll \|g_{k-1}\|$ и хорошей обусловленности матрицы Гессе в x_k.

Если при решении хорошо обусловленной задачи ньютоновский метод не обеспечивает сверхлинейной сходимости, стоит проверить, нет ли вблизи x^* точек разрывов вторых производных. Последние должны быть достаточно большими (иначе они вряд ли сильно повлияли бы на сходимость), и поэтому их не трудно будет выявить. Кроме того, плохая сходимость ньютоновского метода часто бывает результатом ошибок в программе построения матрицы Гессе, так что не следует пренебрегать возможностью ее оттестировать (см. разд. 8.1.4.2).

Нелинейные задачи о наименьших квадратах. Наиболее естественным признаком успешного окончания поиска оптимума в задаче о наименьших квадратах является близость полученного значения минимизируемой суммы к нулю. Надо только сделать оговорку, что имеется в виду хорошо отмасштабированная задача — ведь умножением целевой функции на очень малое положительное число можно, не изменив решения, приблизить ее к нулю в любой точке.

Приемлемое решение не обязательно должно характеризоваться малостью F. Минимум суммы квадратов иногда оказывается большим просто потому, что в ней много слагаемых. В качестве другого примера можно назвать случай, когда в задаче «подгонки» вектора x параметров функции $\varphi(x, t)$ под результаты наблюдений $\{y_i\}$, имеющихся для точек $\{t_i\}$, какие-то квадраты $(y_i - \varphi(x, t_i))^2$ велики из-за больших производных $\varphi(x, t)$ по t. Эта ситуация проиллюстри-

ронана рис. 8d. Видно, что достигнуто хорошее согласование модели с исходными данными, но при этом одна из разностей $f_i = y_i - \psi(x, t_i)$ значительна. Чтобы устранить такой дефект, можно либо ввести в окрестность t_i новые опорные точки, либо увеличить вес f_i в целевой функции (т. е. умножить f_i на положительное число, большее единицы).

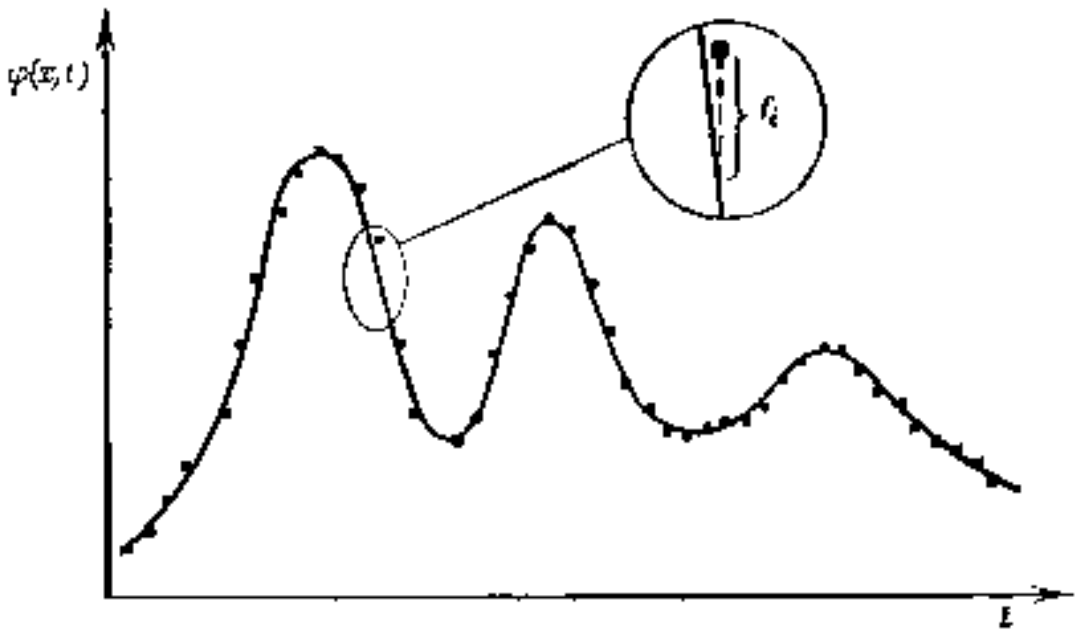

Рис. 8d. Большая невязка f_i, $y_i - \psi(x, t_i)$ при хорошем соответствии между функцией $\psi(x, t)$ и данными наблюдений (y_i).

8.3.1.2. Задачи с ограничениями. Тесты (i)—(iii) из разд. 8.3.1.1 легко обобщить на случай с ограничениями: если они линейны, надо заменить g_k и G_k на $Z^T g_k$ и $Z^T G_k Z$, а если нелинейны — на $Z(x_k)^T g_k$ и $Z(x_k)^T W(x_k, \lambda_k) Z(x_k)$, причем тогда следует обратить внимание и на достигнутую в конце поиска скорость сходимости оценок множителей Лагранжа (см. разд. 6.6).

Нулевые множители Лагранжа. Когда среди множителей Лагранжа, связанных с полученным численным решением, есть нулевые или почти нулевые, разобраться с ним нелегко. Дело в том, что *знаки* множителей активных неравенств определяют не только правильность финального рабочего списка, но и возможные способы улучшения имеющейся точки, если она неоптимальна (см. разд. 5.8.3).

Понятно, что, проводя вычисления с ограниченной точностью, выяснить правильный знак очень малого по модулю множителя непросто — незначительные возмущения (пусть даже целиком обусловленные ошибками округления) могут поменять его. Например, если численная оценка множителя равна 10^{-10}, не исключено, что его истинным значением будет -10^{-5}, а тогда соответствующее ограничение не должно быть активным. Таким образом, при близких к нулю

множителях есть опасность, что численное решение существенно ошибочно, поскольку активный набор выделен неверно.

Проблемы определения правильных знаков малых по модулю множителей трудности не исчерпываются. Допустим, известно, что множитель некоторого активного ограничения в точности равен нулю, т. е. в первом приближении слабые вариации этого ограничения на величину целевой функции не влияют. (Чтобы убедиться в оптимальности точки, где есть нулевые множители, надо исследовать вторые производные, которые не всегда доступны.) Это может означать, что оно избыточно, и решение не изменится, если его отбросить. Так будет, например, в случае, изображенном на рис. 8е: здесь избыточно ограничение $c_2(x) \geqslant 0$. Однако возможна и иная ситуация, когда в результате отбрасывания ограничения с нулевым множителем решение меняется. Она проиллюстрирована рис. 8f: равенство нулю множителя λ_1 объясняется тем, что начало координат — безусловно седловая точка, и, если убрать ограничение $c_1(x) \geqslant 0$, у задачи просто не будет конечных решений.

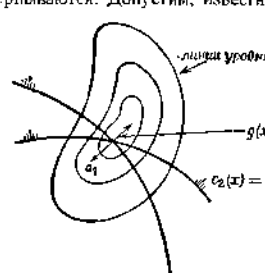

Рис. 8е. Избыточное ограничение с нулевым множителем Лагранжа.

Точка может оказаться неподходящей, даже когда все близкие к нулю множители положительны. Это видно из примера, показанного на рис. 8g. С формальных позиций $x = a$ есть корректное решение, удовлетворяющее ограничению $x \geqslant a$ как равенству, но, конечно, в подобных случаях естественно желать большего. При этом можно говорить о плохой обусловленности, поскольку малая вариация способна привести к значительному смещению точки минимума.

Существование нескольких почти нулевых множителей особенно осложняет анализ: здесь надо решить, имеет ли смысл вывести из рабочего списка какое-нибудь подмножество соответствующих ограничений. Любая процедура поиска такого подмножества неизбежно будет иметь комбинаторный характер и может потребовать большого объема вычислений.

Как поступать, когда среди множителей, отвечающих найденному численному решению, есть близкие к нулю, зависит от того, каким алгоритмом решалась задача. Вообще же малому положительному множителю доверять не следует. В частности, полезно посмотреть, что произойдет, если снять ограничение, которому он отвечает. Иной прием — попытаться варьированием правых частей ограни-

чений выяснить, убывает целевая функция при подходе к границам их допустимых множеств или растет. Кстати говоря, хорошая стандартная программа должна проводить такие тесты автоматически.

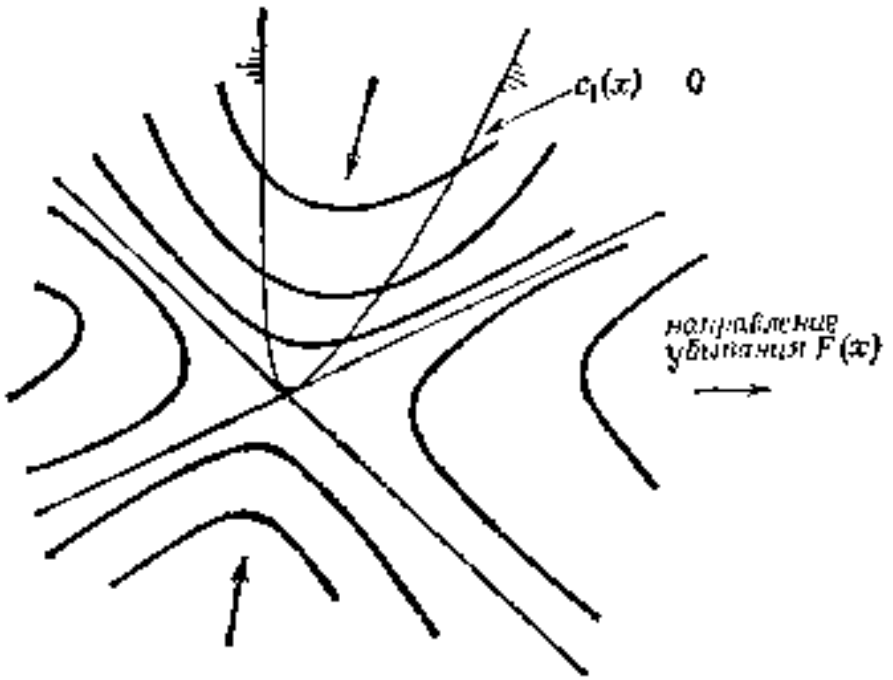

Рис. 8f. Случай, когда ограничение с нулевым множителем Лагранжа отбросить нельзя.

Коль скоро убедительного подтверждения правильности численного решения с почти нулевыми множителями получить не удается, надо попробовать переформулировать задачу, например выявить и исключить «почти избыточные» ограничения.

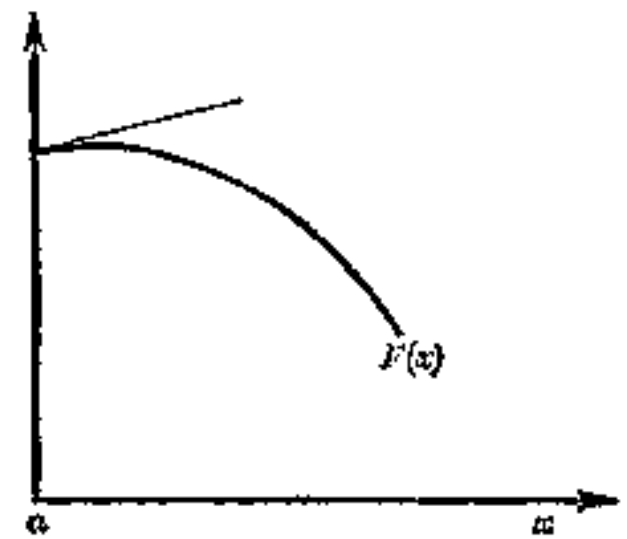

Рис. 8g. Пример неподходящего решения с близким к нулю положительным множителем Лагранжа.

Большие множители Лагранжа. Среди задач с нелинейными ограничениями есть такие, для которых множителей Лагранжа не существует (причина — дефект ранга матрицы Якоби функций активных ограничений $\hat{A}(x^*)$; см. разд. 3.4.1). К счастью, сами они в приложениях практически не встречаются, но «близкие» к ним задачи с «почти линейно зависимыми» строками у $\hat{A}(x^*)$ не редкость.

Плохая обусловленность матрицы $\hat{A}(x^*)$ обычно приводит к тому, что модули всех множителей Лагранжа оказываются намного

больше нормы $|g(x^*)|$. Однако большие множители могут объясняться и плохой нормировкой ограничений — ведь при делении функции ограничения на положительную величину w модуль его множителя увеличится в w раз (см. разд. 8.7.3). Чтобы выявить истинную причину появления больших множителей, надо оценить число обусловленности отнормированной матрицы Якоби — результата преобразования $\hat{A}(x^*)$, состоящего в делении каждой строки $\hat{A}(x^*)$ на ее евклидову норму. Оценку этого числа можно получить, например, с помощью LQ-разложения (см. разд. 2.2.5.3). Таковой послужит отношение максимального по модулю диагонального элемента фактора L к минимальному. Если оно велико, то скорее всего дело в плохой обусловленности, а иначе — в плохой нормировке. В последнем случае надо перенормировать ограничения каким-нибудь из способов, описанных в разд. 8.7.3: численное решение с большими оценками множителей ненадежно.

8.3.2. ДРУГИЕ СПОСОБЫ ПОДТВЕРЖДЕНИЯ ОПТИМАЛЬНОСТИ

Бывает, что даже после тщательного анализа выдачи программы пользователь все еще сомневается, правильно решена задача или нет. Тогда остается три способа обрести ясность:

(i) изменить параметры алгоритма;

(ii) воспользоваться другим алгоритмом;

(iii) изменить задачу.

8.3.2.1. Варьирование параметров алгоритма. Самый понятный, а потому самый подходящий для варьирования параметр — характеристика точности одномерной минимизации η (см. (4.7) и (4.10) в разд. 4.3.2.1). В стандартных программах некоторых методов значение η по умолчанию (см. разд. 8.1.2.1) берут довольно большим, допуская тем самым весьма приближенный одномерный поиск (например, для ньютоновских методов обычно рекомендуется $\eta=0.9$). Следовательно, можно попытаться улучшить имеющуюся точку, потребовав, чтобы шаги подбирались аккуратнее (т. е. задав меньшее значение η). Особенно полезно поэкспериментировать с η, если используется какой-нибудь из методов сопряженных градиентов, крайне чувствительных к точности одномерной минимизации (см. разд. 4.8.3).

Коль скоро существует возможность воспользоваться локальным поиском (см. разд. 8.1.3.6) и при первом обращении к программе этого не было сделано, для проверки полученного приближения надо вызвать ее еще раз, но уже с соответствующим запросом.

Часто причиной плохой работы квазиньютоновских методов с численным дифференцированием являются неудачные значения конечно-разностных интервалов. Поэтому, когда программа такого метода остановилась с признаком отказа, стоит пересчитать эти ин-

тервалы по данным, относящимся к последнему приближению, и запустить ее еще раз.

Иногда затруднения со сходимостью квазиньютоновского метода возникают из-за того, что с его помощью не удается построить достаточно хорошее приближение матрицы Гессе $G(x)$. Здесь дело можно поправить повторным запуском метода при использовании в качестве исходной оценки матрицы $G(x)$ ее конечно-разностную аппроксимацию. В ряде случаев полезно провести расчеты с нового начального приближения. Надо только позаботиться о том, чтобы оно существенно отличалось от приближений первого цикла итераций. Хорошо взять точку, «противоположную» полученной в результате этого цикла.

Если после какого-то из перечисленных изменений программа даст прежний ответ, у пользователя возрастет уверенность в том, что этот ответ правильный. Тем не менее надо помнить, что всегда (и особенно когда используется метод первого порядка) есть опасность повторной сходимости в неоптимальную точку.

8.3.2.2. Смена метода. Если варьированием параметров окончательная ясность не достигнута, надо попытаться применить другой метод. Как правило, лучше перейти к методу, который считается более надежным, но иногда полезно обратиться и к менее надежному — ведь бывает, например, что ньютоновские направления получаются почти ортогональными градиенту и из-за этого оказываются хуже квазиньютоновских.

Допустим, первым был использован квазиньютоновский метод с аналитическими значениями первых производных. Тогда есть по меньшей мере три разумных способа контроля, а именно: запрограммировать вторые производные и применить метод ньютоновского типа; применить ньютоновский метод с конечно-разностной аппроксимацией матрицы Гессе; применить квазиньютоновский метод без вычисления производных. Можно также посмотреть, что получится, если в исходном методе аналитическое дифференцирование заменить численным по центральной конечно-разностной формуле. Взяв для проверки метод ньютоновского типа, в качестве начального приближения ему надо задать последнюю точку предыдущего просчета: это позволит выяснить свойства отвечающей ей матрицы Гессе. В противном случае счет следует начать с какой-нибудь абсолютно не похожей на нее точки (тогда результат проверки будет более убедительным).

8.3.2.3. Изменения в задаче. Если описанные выше приемы оказываются безрезультативными, т. е. все еще нет окончательной уверенности в правильности численного решения, то это скорее всего признак каких-то дефектов формулировки задачи, скажем плохого масштабирования (см. разд. 7.6 и 8.7) или неудачного выбора переменных (см. разд. 7.6.1). Здесь остается только один выход — сформулировать задачу более подходящим образом

8.3.4. АНАЛИЗ ЧУВСТВИТЕЛЬНОСТИ

Часто желательно иметь информацию о «чувствительности оптимума» по отношению к каким-то возмущениям. Например, могут понадобиться оценки последствий малых невязок в ограничениях; когда решается задача настройки модели на результаты наблюдений, важно бывает выяснить, как сказываются на x^* погрешности исходных данных. Кроме того, иногда возникает потребность в определении вариаций x^*, приводящих к минимальным или максимальным изменениям F.

8.3.3.1. Роль матрицы Гессе. Принципы анализа чувствительности функции к варьированию аргумента вблизи оптимального значения удобно пояснить на примере квадратичной формы

$$\Phi(x) = c^T x + \frac{1}{2} x^T G x$$

с положительно определенной матрицей G. Поведение $\Phi(x)$ в окрестности минимума зависит от собственной системы последней (см. разд. 3.2.3). Обозначим собственные значения и векторы G через $\{\lambda_i\}$ и $\{u_i\}$, $i=1, \ldots, n$, причем здесь и далее будем считать, что $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_n$. Тогда числом обусловленности G (см. разд. 2.2.4.3) будет

$$\mathrm{cond}(G) = \frac{\lambda_1}{\lambda_n}.$$

При $\mathrm{cond}(G)=1$ линии уровня $\Phi(x)$ представляют собой концентрические окружности, а при прочих $\mathrm{cond}(G)$ — эллипсы. Чем больше

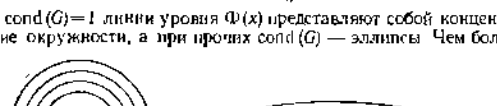

Рис. 8h. Линии уровня квадратичных функций с $\mathrm{cond}(G)=1$ и $\mathrm{cond}(G)=10$.

$\mathrm{cond}(G)$, тем сильнее они сплющены. Как видно из рис. 8h, когда число $\mathrm{cond}(G)$ велико, приращения $\Phi(x)$ при одинаковых по норме, но разных по направлению вариациях x оказываются существенно разными. Направления, вдоль которых $\Phi(x)$ меняется сильнее всего, есть соответственно u_1 и u_n.

Среди встречающихся на практике нелинейных функций большинство составляют такие, которые в окрестностях своих минимумов очень похожи на квадратичные формы; исследование их чувствительности к изменениям x, как и в случае с $\Phi(x)$, сводится к анализу матрицы Гессе (см. разд. 8.2.2). Иной класс

образуют функции, не имеющие непрерывных вторых производных, а также функции типа $F(x)=x_1^4+x_2^4$, характерные тем, что их матрицы Гессе, вычисленные в точках минимумов, состоят из одних нулей. При нулевой $G(x^*)$ особенности локального поведения F вблизи x^* можно выяснить только по производным порядка выше второго; например, отличие $x_1^4+x_2^4$ от $x_1^4+10^6x_2^4$ в окрестности начала координат определяется четвертой производной по x_2.

8.3.3.2. Оценивание числа обусловленности матрицы Гессе. Чтобы получить полное представление о свойствах матрицы Гессе, надо определить ее собственную систему, а это довольно трудоемкая вычислительная задача. Однако необходимость в столь исчерпывающих сведениях возникает редко. В то же время для построения хороших оценок значений λ_1, λ_n и отвечающих им собственных векторов достаточно иметь разложение G или аппроксимирующей ее матрицы по Холесскому (см. разд. 2.2.5.2).

Обозначим через s и r номера соответственно наибольшего и наименьшего среди диагональных элементов фактора D из LDL^T-разложения G, т. е. $d_i \leqslant d_s$ и $d_r \leqslant d_i$ при любом $i=1, 2, \ldots, n$. Их отношение $\varkappa = d_s/d_r$ будет связано с числом обусловленности $\mathrm{cond}(G)$ неравенством $\varkappa \leqslant \mathrm{cond}(G)$, причем, как правило, отличие $\varkappa$ от $\mathrm{cond}(G)$ невелико. Точнее сказать, $\varkappa$ и $\mathrm{cond}(G)$ чаще всего оказываются величинами одного порядка.

Пример 8.2. Возьмем матрицу

$$G=\begin{pmatrix} 22.3034 & 18.4384 & 7.6082 & 13.7463 \\ 18.4384 & 15.2666 & 6.2848 & 11.3694 \\ 7.6082 & 6.2848 & 2.5964 & 4.6881 \\ 13.7463 & 11.3694 & 4.6881 & 8.4735 \end{pmatrix}. \qquad (8.15)$$

Ее число обусловленности равно 1.130599×10^8, а факторами Холесского будут

$$L=\begin{pmatrix} 1 & & & \\ 0.8267 & 1 & & \\ 0.3411 & -0.2105 & 1 & \\ 0.6163 & 0.2217 & -0.1305 & 1 \end{pmatrix}$$

и $D=\mathrm{diag}\ (2.230\times 10^1,\ 2.344\times 10^{-2},\ 7.785\times 10^{-2},\ 5.613\times 10^{-8})$ (элементы того и другого выписаны с точностью до четырех значащих цифр). При этом $\varkappa = 3.9788\times 10^8$.

Ценой несложных дополнительных вычислений можно получить лучшую оценку $\mathrm{cond}(G)$. Введем векторы

$$w_r = (L^T)^{-1}e_r,\ w_s = Le_s.$$

Нетрудно показать, что $\lambda_n \leqslant d_r/\|w_r\|_2$ и $\|w_s\|_2 d_s \leqslant \lambda_1$. Значит,

$$\frac{\lambda_1}{\lambda_n} \geqslant \frac{\|w_r\|_2 \|w_s\|_2 d_s}{d_r}.$$

Величина в правой части и есть искомая оценка (обе нормы в числителе не меньше единицы, так что она не хуже ж).

Пусть далее

$$\bar{w}_r = \frac{1}{\|w_r\|_2} w_r.$$

Если $d_r/\|w_r\|_2 \approx \lambda_n$ и $\lambda_n \ll \lambda_{n-1}$, для этого вектора справедливо приближенное равенство $\bar{w}_r \approx u_n$. Если же λ_{n-1} и λ_n различаются не сильно и $\lambda_{n-1} \ll \lambda_{n-2}$, вектор $\bar{w}_r$ будет «почти линейной комбинацией» u_n и u_{n-1}. В обоих случаях $\bar{w}_r$ является направлением отрицательной кривизны.

Аналогично u_n оценивается и u_1: в предположении близости $\|w_s\|_2 d_s$ к λ_1 и неравенства $\lambda_1 \gg \lambda_2$ можно утверждать, что $u_1 \approx w_s/\|w_s\|_2$.

Для матрицы (8.15) максимальное и минимальное собственные числа равны $\lambda_1 = 4.8622797 \times 10^3$ и $\lambda_4 = 4.3006198 \times 10^{-8}$, а u_1 и u_4 выглядят следующим образом:

$$u_1 = \begin{pmatrix} 0.677217 \\ 0.560151 \\ 0.230952 \\ 0.417455 \end{pmatrix}, \quad u_4 = \begin{pmatrix} -0.437963 \\ -0.170007 \\ 0.114370 \\ 0.875332 \end{pmatrix}.$$

Векторы w_4 и $\bar{w}_4$ здесь получаются такими:

$$w_4 = \begin{pmatrix} -0.500240 \\ -0.194261 \\ 0.130467 \\ 1.000000 \end{pmatrix}, \quad \bar{w}_4 = \begin{pmatrix} -0.437898 \\ 0.170052 \\ 0.114208 \\ 0.875377 \end{pmatrix}.$$

Заметьте, что $\bar{w}_4$ и u_4 очень близки. Столь же хорошо срабатывает и формула оценивания u_1:

$$w_1 = \begin{pmatrix} 1.000000 \\ 0.826707 \\ 0.341121 \\ 0.616332 \end{pmatrix}, \quad \bar{w}_1 = \begin{pmatrix} 0.677336 \\ 0.559959 \\ 0.231054 \\ 0.417464 \end{pmatrix}.$$

Наконец,

$$\frac{\|w_r\|_2 \|w_s\|_2 d_s}{d_r} \approx 1.303443 \times 10^8,$$

и это — гораздо более точное, чем ж, приближение числа обусловленности матрицы (8.15).

59-1

8.3.3.3. Чувствительность к возмущениям ограничений. В гладкой задаче на условный минимум вариации F вблизи решения можно связать с невязками активных ограничений. Эта связь задается множителями Лагранжа.

Обозначим через $\hat{c}$ вектор-функцию активных в x^* ограничений, через $\hat{A}$ матрицу (полного ранга) градиентов ее скалярных составляющих, а через q направление, для которого

$$\hat{a}_i^T q = 1, \quad \hat{a}_j^T q = 0, \quad i \neq j. \tag{8.16}$$

Тогда, заменив в уравнении (8.12) из разд. 8.2.2.2 произведение Yp_Y на hq, для $\bar{x} = x^* + hq$ получим

$$F(\bar{x}) = F(x^*) + h\lambda_i^* + O(h^2 \| q \|^2). \tag{8.17}$$

Пусть теперь $\bar{x}$ — некая близкая к x^* точка, про которую известно только то, что $\hat{c}_j(\bar{x}) = \delta_j$, $\hat{c}_i(\bar{x}) = 0$, $i \neq j$. При этом в первом приближении должно соблюдаться равенство $\bar{x} = x^* + \delta_j q$, где q — удовлетворяющий (8.16) вектор. Значит, на основании (8.17) можно утверждать, что с точностью до величин порядка δ_j^2 разница между значениями F в x^* и $\bar{x}$ есть $\delta_j \lambda_j^*$.

Итак, на множитель Лагранжа активного ограничения можно смотреть как на меру чувствительности F к его возмущениям, и если, скажем, $\lambda_1 = 1000$, а $\lambda_2 = 10^{-3}$, то позволительно сделать вывод, что первое активное ограничение «намного важнее» второго. Правда, подобное сопоставление предполагает примерно одинаковую нормировку ограничений: нормы градиентов их функций не должны быть очень различными (см. разд. 8.7.3).

Информация о связи F с ограничениями особенно полезна, когда природа задачи допускает их варьирование, т. е. они описывают не обязательные, а лишь желаемые свойства решения. Здесь на основе этой информации иногда удается существенно улучшить постановку; например, выяснив, что малые невязки какого-то ограничения сулят значительный выигрыш, разумно будет ослабить его; с другой стороны, может оказаться полезным ужесточить некие ограничения, по отношению к которым функция не чувствительна.

Замечания и избранная библиография к разделу 8.3

Как оценивать результаты решения линейных задач по наименьших квадратам, детально описано у Лоусона и Хансона (1974). Подробности и ссылки по поводу статистической интерпретации решений нелинейных задач этого типа можно найти, например, в книге Барда (1976). Различные аспекты анализа результатов безусловной минимизации функции общего вида обсуждались Мюрреем (1972b).

Оценки обусловленности, которые точнее представленных в разд.

8.3.3.2, но требуют более сложных вычислений, даны в работах Кляйна и др. (1979), О'Лири (1980b).

Способы определения характеристик чувствительности оптимума в задачах с нелинейными ограничениями, ориентированные на методы штрафных и модифицированных функций Лагранжа, приведены у Фиакко и Мак-Кормика (1968), Фиакко (1976), Бийса и Гопина (1977).

8.4. ЧТО МОЖЕТ НЕ ПОЛУЧАТЬСЯ (И КАК ТОГДА ПОСТУПАТЬ)

Даже самые совершенные алгоритмы иногда отказывают, и надо уметь разбираться в причинах этого. В частности, уже говорилось, что нередко отказ бывает следствием ошибок в программах вычисления функций задачи. (К сожалению, такие ошибки могут проявляться не сразу, а через аварийные ситуации в каких-то блоках алгоритма.) В идеальном случае хорошая библиотечная программа сообщит пользователю о конкретной причине прерывания счета — останется «только» понять, что за этим стоит. Ниже обсуждаются наиболее характерные типы отказов, описываются причины, по которым они чаще всего происходят, и приводятся некоторые рекомендации по исправлению подобных ситуаций.

8.4.1. ПЕРЕПОЛНЕНИЕ ПРИ ПОДСЧЕТЕ ФУНКЦИЙ ЗАДАЧИ

Переполнение арифметического устройства является не самой частой, но, возможно, самой неприятной причиной аварийного останова, поскольку приводит к передаче управления операционной системе машины.

Переполнение во время процесса оптимизации может возникнуть из-за того, что имеется неограниченное решение — в результате сходимости к таковому либо абсолютная величина минимизируемой функции, либо норма очередного приближения неизбежно станет слишком большой. Подобные случаи распознаются по устойчивому прогрессу при отсутствии признаков близости предела. Если говорить о задачах на *безусловный* минимум, то, как правило, причиной неограниченности целевой функции снизу оказывается невнимательность пользователя. Правда, бывает, что о ней знают заранее и тем не менее не устраняют ее, надеясь найти некий ограниченный *локальный* минимум (а алгоритм выходит из области его притяжения). Следует также отметить, что неограниченность довольно типична для подзадач, используемых в методах минимизации при нелинейных ограничениях (см. разд. 6.2, 6.4 и 6.5).

Что делать при переполнении, вызванном сходимостью к неограниченному решению, зависит от ситуации. Коль скоро такое решение появилось из-за неудачной формулировки задачи, рецепт очевиден — надо изменить ее. Если же целевая функция действительно

должна быть неограниченной и при этом требовалось отыскать определенный локальный минимум, то успеха, возможно, удастся достичь, уменьшив максимальное разрешенное значение величины шага за одну итерацию (см. разд. 8.1.3.4) или введя простые ограничения на переменные. Наконец, с неограниченностью в подзадачах методов условной оптимизации, использующих штрафы, можно бороться увеличением штрафного параметра (см., например, рис. 6h). Однако не исключено, что и тут придется принимать меры, рекомендованные выше для стимулирования локальной сходимости,— ведь целевая функция подзадачи может быть неограниченной снизу при любом значении штрафного параметра.

Иногда переполнение возникает при минимизации заведомо ограниченных снизу функций. В частности, это бывает на начальных итерациях квазиньютоновских методов. Компоненты выбираемых здесь направлений поиска нередко оказываются очень большими, и тогда пробные шаги одномерного спуска могут приводить в точки с бессмысленно большими координатами. Когда в функции есть быстрорастущие составляющие, скажем экспоненты или высокие степени x, попытка вычислить ее там почти наверняка приведет к переполнению. (Возьмите, к примеру, $\exp(10x_1)$ и посмотрите, что получится, если поменять значение x_1 с нуля на 10^2.) В таких случаях трудности устраняются правильным выбором максимальной длины шага (см. разд. 8.1.3.4).

8.4.2. НЕДОСТАТОЧНОЕ УМЕНЬШЕНИЕ ФУНКЦИИ ВЫИГРЫША

Во многих оптимизационных алгоритмах ставится требование, чтобы применяемая функция выигрыша Φ_M «существенно уменьшалась» на каждой итерации. Обеспечить это должна процедура выбора шага (см. разд. 4.3.2.1). Если она не способна дать нужный выигрыш, это считается признаком отказа. Чаще всего именно так происходит отказ алгоритмов данного класса.

Поскольку условия, определяющие достаточность полученного выигрыша, бывают весьма разнообразными, разобрать все причины аварийных остановов мы не в силах. Кстати говоря, некоторые стандартные программы сообщают об отказе блока одномерного поиска, когда производная функции выигрыша по выбранному направлению слишком близка к нулю. На самом деле это — признак неспособности алгоритма построить хорошие направления спуска; он будет обсуждаться в разд. 8.4.6. Здесь же речь пойдет о причинах, по которым могут стать нереализуемыми неравенства вида

$$\bar{\Phi}_M(x_k + \alpha_k p_k) \leqslant \Phi_M(x_k) - \mu_k; \tag{8.18}$$

$$\delta_k \leqslant \alpha_k; \tag{8.19}$$

$$\|\alpha_k p_k\| \leqslant \Delta_k. \tag{8.20}$$

В них μ_k ($\mu_k > 0$) есть порог «существенного убывания» Φ_M; δ_k ($\delta_k > 0$) — минимальный «ощутимый» шаг вдоль p_k; Δ_k ($\Delta_k > 0$) — максимальное разрешенное расстояние между последовательными точками (см. разд. 8.1.3.4). Основное соображение при выборе δ_k состоит в том, что вариации вектора переменных, при которых изменение функции Φ_M меньше точности ε_A ее подсчета в x_k, бессмысленны. Одна из возможных формул вычисления δ_k такова:

$$\delta_k = \max\left(\frac{\varepsilon_A}{|p_k^T \nabla \Phi_M| + \sqrt{\varepsilon_A}},\quad c_M \frac{|+|x_k||}{|+|p_k|}\right). \qquad (8.21)$$

Эти и другие условия на α_k обсуждаются также в разд. 4.3.2.1 и 8.1.3.4.

8.4.2.1. Ошибки в программах пользователя. Частой причиной отказа процедуры выбора шага являются ошибки в программах подсчета функции выигрыша Φ_M и ее градиента. Обычно одномерный

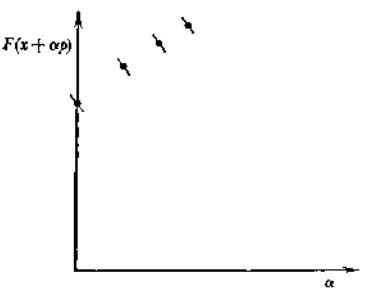

Рис. 81. Возможные последствия ошибок в программах подсчета функции и градиента; точками отмечены вычисленные значения $F(x+\alpha p)$ при разных α, черточки указывают направления касательных, построенных в соответствии с вычисленными значениями производных.

поиск осуществляется на основе простой полиномиальной аппроксимации вдоль выбранного направления; точка минимума очередного полинома берется в качестве очередного пробного шага. Если Φ_M или ее градиент считаются неверно, аппроксимация будет вестись по бессмысленным данным, и тогда вероятность выполнить какие-либо разумные условия становится ничтожной. Например, в ситуации, изображенной на рис. 81, алгоритм будет брать все меньшие и меньшие α_k, но так и не сможет добиться убывания Φ_M.

Как правило, ошибки программирования легко обнаруживаются по распечатке пробных шагов и подсчитанных для них значений функции Φ_M и ее производных по направлению. Другие способы выявления ошибок описаны в разд. 8.1.4.

8.4.2.2. Плохие масштабы. Иногда условия (8.18), (8.19) и (8.20) оказываются несовместными по одной из причин, которые можно считать проявлениями того, что функция выигрыша «плохо отмасштабирована» относительно направления p_k.

Пожалуй, самый типичный дефект рассматриваемого типа — несбалансированность изменений Φ_M и x при движении вдоль p_k. Если Φ_M почти не реагирует даже на большие вариации x (это случается при несбалансированных производных; см. разд. 8.7.1.3), то трудно обеспечить одновременное соблюдение неравенств (8.18) и (8.20). Обратная картина — сильная реакция Φ_M на незначительные изменения x — столь же нежелательна. Здесь, когда производная от Φ_M вдоль направления p_k становится настолько большой, что δ_k определяется не первой, а второй из величин в правой части (8.21), могут оказаться противоречивыми условия (8.18) и (8.19).

Угроза аварийного останова в процедуре одномерного поиска возникает также при отсутствии баланса между компонентами x_k и p_k. Допустим, например, что $x_k = (10^6, 1)^T$, $p_k = (0, 1)^T$ и расчеты ведутся на машине с $\varepsilon_M = 10^{-6}$. Тогда, хотя первая компонента x_k при движении вдоль p_k сохранится, т. е. на нее не надо было бы обращать внимания, именно из-за ее большой величины формула (8.21) даст δ_k порядка 10^{-1}, что, вообще говоря, многовато для минимальной длины шага по второй компоненте. Поэтому вполне может оказаться, что при α_k, подчиняющихся (8.19), неравенству (8.18) удовлетворить не удастся.

Коль скоро процедура выбора шага отказала по одной из перечисленных причин, в последняя в свою очередь объясняется некачественностью p_k, поиск, возможно, удастся продолжить после небольшой корректировки данных. В частности, ньютоновские направления бывают «плохо отмасштабированными» в точках x_k, где градиент функции Φ_M достаточно велик и при этом ее матрица Гессе близка к вырождению («на склоне длинного узкого оврага»). В случае соответствующего останова ньютоновский алгоритм надо запустить еще раз, задав в качестве начального приближения какую-нибудь точку, слегка отличающуюся от полученной к моменту прерывания. В квазиньютоновских алгоритмах неудачные направления чаще получаются в x_k, лежащих «на дне оврага», причем тогда, когда оценки матрицы Гессе существенно неточны. Стало быть, если речь идет о повторном запуске квазиньютоновского метода, то следует изменить оценку матрицы Гессе (например, воспользоваться конечно-разностной аппроксимацией), а точку варьировать не надо. Когда неясно, чем объяснить отказ процедуры одномерного поиска — случайным выбором неудачного направления или тем, что задача вообще плохо отмасштабирована, полезно попробовать спуститься по антиградиенту. Безрезультатность этой попытки (невыполнение условий «существенного убывания») будет признаком необходимости заново отмасштабировать задачу каким-нибудь из способов, описанных в разд. 8.7.

8.4.2.3. Чрезмерно жесткие критерии останова. Слишком сильные требования к окончательному приближению (см. разд. 8.2) тоже нередко приводят к аварийному останову в блоке выбора шага. Это бывает, когда точка x_k оказывается настолько близкой к x^*, что разница между значениями Φ_{μ} в x_k и x^* соразмерима с ошибкой вычислений. Тогда, даже если теоретическая возможность удовлетворить условию «существенного убывания» при спуске из x_k имеется, практическая вполне может отсутствовать. Именно поэтому алгоритмы иногда выдают признак отказа, хотя в действительности найдено хорошее решение.

Коль скоро процедура одномерного поиска отказала очень близко от оптимальной точки, никакие «корректировки», как правило, не нужны. Вот только, если в дальнейшем предполагается решать какую-то похожую задачу, то есть смысл пересмотреть значения параметров критериев останова, чтобы не тратить машинное время в погоне за недостижимой точностью (и не провоцировать ложных отказов).

8.4.2.4. Погрешности конечно-разностной аппроксимации. Отсутствие возможности удовлетворить условию существенного убывания бывает следствием неточности конечно-разностных оценок градиента функции Φ_{μ}. Большие относительные ошибки численной аппроксимации первых производных могут возникать по разным причинам.

Во-первых, вблизи точки безусловного минимума Φ_{μ} ни одна из простых конечно-разностных формул хорошей оценки градиента не даст (см. разд. 8.6.1), и там отказ процедуры одномерного поиска будет не чем иным, как признаком достижения предельной точности.

Во-вторых, если взять неудачные конечно-разностные интервалы, оценка $\widehat{g}_k$ градиента g_k может оказаться очень плохой и в точке x_k, далекой от оптимальной. Тогда не исключено, что та самая формула расчета p_k, которая при использовании точных значений первых производных генерирует только направления спуска (см. разд. 8.4.6), из-за замены g_k на $\widehat{g}_k$ даст «направление подъема»; немедленное следствие — аварийный останов в блоке выбора шага. В такой ситуации надо построить более подходящий набор конечно-разностных интервалов (предназначенный для этого метод описан в разд. 8.6.2).

Неточность численного дифференцирования, послужившая причиной отказа процедуры одномерного поиска, может объясняться и малыми разрывами функции Φ_{μ} (сильные разрывы обычно проявляются более просто). На вероятность их существования укажут большие (по модулю) компоненты приближения градиента. Один из способов выяснить действительно ли такие компоненты появились из-за разрывности Φ_{μ} — посмотреть, к каким последствиям приведет увеличение соответствующих конечно-разностных интервалов. Допустим, к примеру, что интервал h_i равен 10^{-7} и между x и $x + h_i e_i$

есть точка, где Φ_M скачком меняется на 10^{-8}. Тогда оценка i-й производной будет иметь порядок 10^8. Оценка той же производной при $h_i=10^{-6}$ окажется величиной порядка 10^6. Если бы не разрыв, такое изменение h_i вряд ли привело бы к столь резкому различию в оценках.

Лучшее решение проблемы малых разрывов — избавиться от них (см. разд. 7.3). В случаях, когда это невозможно, бывает, что сугубо локальные трудности численного дифференцирования удается преодолеть выбором специальных конечно-разностных формул, гарантирующих отсутствие скачков Φ_M на отрезках аппроксимации производных.

8.4.3. УСТОЙЧИВО МЕДЛЕННЫЙ ПРОГРЕСС

8.4.3.1. Безусловная минимизация. Нередко в список критериев аварийного останова программы минимизации без ограничений включают условие малости изменения целевой функции за определенное число последовательных итераций. Чаще всего останов по данному признаку объясняется тем, что вычисляемые направления спуска оказываются почти ортогональными антиградиентам. В таких случаях иногда помогают рекомендации, приведенные в разд. 8.4.2.2, а если повторный запуск программы после соответствующих корректировок к успеху не приводит, надо попытаться решить задачу другим алгоритмом, например вместо квазиньютоновского взять ньютоновский (или наоборот).

Замедление прогресса бывает связано и с тем, что поиск привел в область, где функция плохо отмасштабирована. Здесь упомянутые выше приемы могут оказаться безрезультатными, и тогда необходимо либо заново отмасштабировать задачу (см. разд. 8.7), либо вообще переформулировать ее.

8.4.3.2. Оптимизация при линейных ограничениях. При решении задач с линейными ограничениями случается, что на нескольких итерациях варьируется только состав рабочего списка, а значения переменных сохраняются неизменными. В частности, возможно зацикливание (см. разд. 5.8.2), т. е. бесконечное повторение определенной комбинации рабочих списков без смены x_k. Имея это в виду, в некоторых программах предусматривают контроль числа последовательных итераций «с нулевыми шагами» и прерывание счета с сообщением об отказе, если оно превысит заданный порог.

Аварийный останов по сформулированному признаку обычно объясняется тем, что, попав в вырожденную вершину допустимого множества, алгоритм не может выделить подходящий набор из n линейно независимых активных ограничений (см. разд. 5.8.2). Когда общее число активных ограничений невелико, есть смысл попытаться сделать это аналитическим путем. Если же их много и вырождение значительно, надо действовать иначе. Во-первых, для поиска

подходящего рабочего списка можно применить какую-нибудь комбинаторную процедуру. Коль скоро этот выход неприемлем, можно просто возобновить счет с точки останова в надежде, что алгоритму все же удастся выйти из нее. Наконец, могут помочь малые возмущения правых частей ограничений.

Серия нулевых шагов бывает также следствием обращения в нуль множителей Лагранжа некоторых активных неравенств. В данном случае она возникает в результате бесплодных «стараний» алгоритма найти такое подмножество линейно независимых активных ограничений, которому отвечала бы положительно определенная спроектированная матрица Гессе (см. разд. 5.8.3). Как и в ситуации с вырожденной вершиной, здесь встает комбинаторная проблема. Правда, существует немалая вероятность, что точка останова на самом деле окажется приемлемым решением (ведь нулевые множители Лагранжа становятся помехой лишь при условии, что спроектированный градиент близок к нулю; см. разд. 5.8.3). Коль скоро на этот счет есть сомнение, можно попытаться подобрать подходящий рабочий список из соображений аналитического плана, а если такой путь исключается, стоит попробовать вывести алгоритм из тупика с помощью небольших вариаций правых частей ограничений.

8.4.3.3. Оптимизация при нелинейных ограничениях. Алгоритмы решения задач с нелинейными ограничениями могут «застревать» вдали от оптимума по нескольким причинам. В частности, говоря о схемах с подзадачами безусловной минимизации или минимизации при линейных ограничениях, надо указать на трудности, обсуждавшиеся в разд. 8.4.3.1 и 8.4.3.2. Однако отсутствие ощутимого прогресса во время поиска минимума при нелинейных ограничениях возможно и в силу иных обстоятельств.

Трудности идентификации правильного активного набора. Методы, оперирующиеся на функции Лагранжа, иногда отказывают потому, что из-за специфики задачи не могут выделить правильный активный набор. Допустим, для некоторых неактивных в x^* ограничений $\bar{c}(x^*)=\delta$, где $\|\delta\|$ — очень малая величина. Даже хорошему алгоритму вряд ли удастся отличать такие ограничения от активных, пока не будет достигнуто неравенство $|\bar{c}(x_k)|\ll\delta$, а это обычно требует большого числа итераций. Прогресс на таких итерациях оказывается незначительным, т. е. здесь возможен аварийный останов, причем тогда на точность полученного решения рассчитывать не приходится.

Описанная трудность сродни рассмотренным в разд. 8.3.1.2, где речь шла о нулевых множителях Лагранжа, иногда означающих, что при отбрасывании соответствующих ограничений решение не изменилось бы. Отсюда, впрочем, не следует, что оценки множителей Лагранжа, связанные с $\bar{c}$, должны быть близки к нулю,— как правило, это не так.

Индикатором того, что замедление прогресса обусловлено неспособностью алгоритма выделить правильный активный набор, являются большие изменения в рабочем списке при незначительных вариациях координат текущего приближения. При этом могут сильно меняться и оценки множителей Лагранжа. Нередко процесс приобретает циклический характер — с определенной частотой происходит возврат к уже просмотренному рабочему списку и оценкам множителей, очень похожим на получавшиеся ранее.

Трудности идентификации активного набора довольно характерны для задач l_∞ и l_1 (см. разд. 4.2.3), возникающих в контексте настройки математических моделей на данные наблюдений. Настройка осуществляется по опорным точкам, среди которых есть «критические», определяющие правильный активный набор; однако наряду с ними часто есть «почти» критические точки и соответственно «почти» активные в решении ограничения.

8.4.4. ВЫПОЛНЕНИЕ МАКСИМАЛЬНОГО ЧИСЛА ИТЕРАЦИЙ ИЛИ ОБРАЩЕНИЙ К ПРОЦЕДУРЕ ВЫЧИСЛЕНИЯ ЦЕЛЕВОЙ ФУНКЦИИ

Хорошие библиотечные программы оптимизации обычно предполагают задание пользователем ограничения сверху на количество вычислений целевой функции или на число итераций. Это ограничение может оказаться полезным по нескольким причинам и предохраняет от бессмысленных затрат машинного времени при дефектах формулировок задач.

В частности, выход на верхнюю границу числа итераций иногда объясняется тем, что у задачи есть неограниченное решение (см. также разд. 8.4.1). Бывает и так, что он обусловлен замедлением сходимости в силу какого-нибудь из обстоятельств, рассмотренных в разд. 8.4.3; тогда надо следовать приведенным там рекомендациям.

8.4.5. ОТСУТСТВИЕ ОЖИДАЕМОЙ СКОРОСТИ СХОДИМОСТИ

Когда пользователь рассчитывает на высокую скорость сходимости алгоритма, а она не проявилась, он может расценить это как признак «отказа». Однако, прежде чем перейти к обсуждению соответствующих причин, мы хотели бы подчеркнуть, что об отказе здесь можно говорить лишь в случае, если надежды, возлагаемые на алгоритм, имели твердую основу. Что же касается методов оценивания достигнутой скорости сходимости, то о них кратко сказано в разд. 8.3.1.

Чаще всего «отказы» рассматриваемого сорта случаются с ньютоновскими методами, от которых в согласии с теорией ожидают квадратичной сходимости. Существуют три основные причины, по которым ньютоновский метод может сходиться медленнее. Первая, самая распространенная — ошибки в программе вычисления матрицы Гессе.

Достаточно допустить ошибку в вычислении лишь одного из ее элементов, и уже возникает риск потери квадратичной сходимости. Когда подобной сходимости не наблюдается и в то же время «матрица Гессе» хорошо обусловлена, а длины шагов, выбираемые процедурой одномерного поиска, систематически далеки от единицы, надо прежде всего поискать такие ошибки.

Сходимость ньютоновского метода с конечно-разностной аппроксимацией матрицы Гессе может замедляться из-за небольших разрывов в «машинной версии» градиента. Симптомы будут аналогичны рассмотренным в разд. 8.4.2.4 в связи с малыми разрывами функции. В частности, имеются в виду появление в генерируемой матрице очень больших элементов и сильная реакция последних на изменение конечно-разностного интервала.

Наконец, третья возможная причина — плохая обусловленность (или вырожденность) матрицы Гессе. Для квадратичной сходимости классического метода Ньютона требуется, чтобы в решении эта матрица была невырожденной, причем, чем больше число ее обусловленности, тем меньше область сходимости. Поэтому любой метод ньютоновского типа неизбежно потерпит быстродействие, если это число окажется очень большим. Когда все объясняется плохой отмасштабированностью задачи, дело можно попытаться поправить с помощью приемов, описанных в разд. 8.7.

Вырожденность матрицы Гессе в решении, как правило, указывает на то, что точка минимума определена неоднозначно (для квадратичных форм этот момент обсуждался в разд. 3.2.3). Иногда неоднозначность оказывается следствием нечеткости формулировки задачи (соответствующий пример описан в разд. 7.6.1).

8.4.6. НЕУДАЧНОЕ НАПРАВЛЕНИЕ ПОИСКА

Если на каждой итерации предполагается удовлетворять условию «существенного убывания» некой функции выигрыша Φ_M, то принято требовать, чтобы направления поиска p_k были *направлениями спуска* относительно Φ_M, т. е. чтобы

$$p_k^t \nabla \Phi_M(x_k) < 0 \tag{8.22}$$

(когда x_k — нестационарная точка). Нарушение этого неравенства считается признаком отказа алгоритма.

Вектор p_k обычно вычисляется как решение некоторой линейной системы вида

$$M p_k = -\nabla \Phi_M(x_k). \tag{8.23}$$

Коль скоро матрица M положительно определена, соблюдение (8.22) при данном p_k теоретически обеспечено. В хорошо реализованном методе M всегда будет не просто, а «существенно» положительно определенной; тем самым вероятность нарушить (8.22) исключается не только теоретически, но и практически. Отметим, что в таких реали-

зациях приходится предусматривать возможность изменений в некоторых матрицах задачи (см. разд. 4.4.2).

Ясно, что в случае знаконеопределенной M решение системы (8.23) не обязательно будет направлением спуска. Надо помнить также, что из-за ошибок округления численное решение (8.23) может нарушать (8.22) и при положительно определенной, но очень плохо обусловленной M. В частности, такое случается в квазиньютоновских алгоритмах, если в качестве гарантии положительной определенности аппроксимации матрицы Гессе (или обратной к ней матрицы) используются только условия типа (4.41). Чем это чревато, видно из примера 4.8.

Вместо (8.22) в хороших алгоритмах часто контролируется неравенство

$$p_k^T \nabla \Phi_M(x_k) < -\zeta_k, \tag{8.24}$$

где ζ_k — положительная величина. Оно может нарушаться даже для некоторых направлений спуска. Несоблюдение (8.24) обычно бывает связано с чрезмерно жесткими условиями нормального останова.

8.5. ОЦЕНКА ТОЧНОСТИ ВЫЧИСЛЕНИЯ ФУНКЦИЙ ЗАДАЧИ

8.5.1. РОЛЬ ТОЧНОСТИ

8.5.1.1. Определение точности. При использовании многих оптимизационных алгоритмов важно иметь некое представление о близости *вычисляемых* значений функций к их истинным значениям. Поскольку оптимизация подразумевает вычисления в разных точках, речь пойдет не о том, как определить величину погрешности подсчета функции при конкретном значении аргумента, а о том, как получать хорошие *оценки сверху* для погрешностей вычисления функции *во всех точках*, которые могут получаться в процессе поиска оптимума.

Применять необоснованные оценки точности не следует. В то же время, если не используется дополнительная информация, для определения достаточно надежной оценки ошибки вычисления функции в точке, как правило, требуются значительные усилия. Имея это в виду, мы предлагаем следующий, хорошо зарекомендовавший себя на практике подход. Для рассматриваемой функции Φ строится хорошая оценка погрешности вычисления в «характерной» точке (обычно в начальной точке поиска минимума); пригодные для этого процедуры описаны в разд. 8.5.2. Принимается некоторая модель поведения ошибки, и в соответствии с ней оценки для последующих точек получаются пересчетом по простой формуле (см. разд. 8.5.3).

Прежде чем перейти к описанию техники оценивания точности, приведем определение соответствующих оценок и обсудим их значение для алгоритмов. Под оценкой ошибки вычисления функции

Φ в точке x мы подразумеваем положительное число ε_A, такое, что

$$|fl(\Phi(\overline{x})) - \Phi(x)| \leqslant \varepsilon_A, \tag{8.25}$$

где $\overline{x}$ — машинное представление x. Таким образом, в ε_A включаются и ошибки подсчета Φ, и ошибки округления x. (Далее будет предполагаться, что значения x и Φ далеки от границ переполнения и антипереполнения.)

Заметим, что неравенство (8.25) не определяет ε_A однозначно (будучи выполненным при каком-то ε_A, оно сохранится и при всех *больших* ε_A), но нам это и не требуется. Даже в малой окрестности рассматриваемой точки *действительная ошибка* может принимать существенно разные значения; например, там могут найтись x, которые принадлежат представимому множеству машины и в которых Φ вычисляется без погрешности. Учитывать эти вариации ни к чему. Нам нужна величина ε_A, которая была бы хорошей оценкой для ошибок подсчета Φ в любой точке вблизи x.

Мы хотим подчеркнуть, что определение (8.25) учитывает *только погрешности при вычислении* Φ и никак не отражает точность, с которой математическая функция Φ отражает какую-то реальную зависимость. Если, к примеру, Φ согласуется с данными наблюдений только в трех старших десятичных разрядах, то это отнюдь не означает, что ε_A должна быть порядка 10^{-3}. Само собой разумеется, ошибки моделирования тоже должны приниматься во внимание, но уже на этапе содержательного анализа решения (этот момент подробнее обсуждается в разд. 7.3.1).

8.5.1.2. Роль оценок точности в оптимизационных алгоритмах. Оценка погрешностей вычислений функций задачи может учитываться во многих модулях алгоритма оптимизации. В частности, она (i) встречается в критериях останова (разд. 8.2.3); (ii) определяет минимальное приемлемое расстояние между пробными точками в процедуре одномерного поиска (разд. 4.3.2.1 и 8.4.2); (iii) входит в формулы расчета конечно-разностных интервалов (разд. 4.6.1.1 и 8.6); (iv) используется при выборе масштабирующих множителей (разд. 8.7).

В случаях (i) и (ii) рассматриваемая оценка нужна для определения момента, когда продолжение счета становится бессмысленным, поскольку вариации функции снизились до «уровня шума» (стали меньше, чем ε_A). Таким образом, заниженное значение ε_A может привести к ненужным итерациям и остановке алгоритма с признаком отказа (см. разд. 8.4.3). С другой стороны, если взять ε_A неоправданно большой, повышается вероятность преждевременного прерывания счета (см. разд. 8.2.3.6).

К счастью, работая на машине с достаточно высокой точностью представления числа, при назначении ε_A можно ошибиться на один-два порядка без риска серьезно повлиять на результаты минимиза-

зации. По-настоящему ответственно к оценке ε_A приходится относиться тогда, когда рабочая точность машины низка.

В алгоритмах с конечно-разностной аппроксимацией градиентов ε_A может определять момент переключения с правой формулы на центральную. Тогда при завышенной ε_A центральные разности начнут использоваться раньше, чем следовало бы, и это повлечет ненужные дополнительные вычисления целевой функции. Если же взять заниженную ε_A, то результатом будет падение скорости сходимости из-за плохой аппроксимации градиентов в окрестности оптимума.

8.5.1.3. Ожидаемая точность. В разд. 2.1.4 было отмечено, что при обычной реализации арифметики с плавающей точкой результат любого элементарного арифметического действия над двумя машинно-представимыми числами a и b можно выразить в виде

$$fl(a \text{ op } b) = (a \text{ op } b)\,(1+\varepsilon),$$

где $|\varepsilon|$ не превосходит $\gamma\varepsilon_M$, в котором γ – число порядка единицы. Исходя из этого, можно оценивать точность вычисления любой функции. Однако, поскольку большинство встречающихся в приложениях функций предполагает длинные цепочки вычислений, такой подход непрактичен.

Стандартная нижняя граница для ε_A. Если $\Phi(x)$ не нуль, величину ε_A можно выразить через *относительную ошибку*, т. е.

$$\varepsilon_A \sim \varepsilon_R |\Phi(x)|. \tag{8.26}$$

Однако это соотношение полезно *только при «малых» ε_R*. Когда Φ – стандартная функция типа синуса, косинуса или экспоненты, (8.26) обычно выполняется с ε_R порядка ε_M.

К сожалению, при $|\Phi|$, близких к нулю, ε_R в (8.26), вообще говоря, не будет малой. Даже для очень простых функций относительная ошибка вычисляемого значения может оказаться очень *большой*. Возьмите, к примеру, $\Phi(x) = 1 - x$ при x вблизи единицы (см. также разд. 2.1.5 и пример 8.3). Поэтому на практике разумно использовать в качестве *нижней* оценки ε_A величину

$$\varepsilon_A' \sim \varepsilon_M (1 + |\Phi|). \tag{8.27}$$

Мы будем называть ее *стандартной нижней границей* для ε_A. Из (8.27) видно, что ε_A' не может быть меньше ε_M и соответствует относительной ошибке ε_M при больших $|\Phi|$.

Функции с необычно малыми значениями ε_A. Как уже было указано выше, для стандартных функций $\Phi(x)$ ошибка ε_A может быть меньше, чем ε_A'. Другой важный класс функций, обладающих тем же свойством, образуют целевые функции *задач о наименьших квадратах с малыми невязками*. Это объясняется следующим образом. Рассмотрим

$$\Phi(x) = \sum_{i=1}^{m} f_i(x)^2,$$

считая, что для каждого i справедливо $fl(f_i)=f_i+\delta_i$, где $|\delta_i|\leqslant \varepsilon_f$. Тогда, пренебрегая ошибками, возникающими при выполнении операций сложения и возведения в квадрат, получим

$$fl(\Phi)-\Phi \sim \sum_{i=1}^{n} 2f_i\delta_i+\delta_i^2. \qquad (8.28)$$

Отсюда видно, что при «*малых*» $|\varepsilon_i|$ и $|f_i|$ величина ε_A для Φ может оказаться *намного меньше, чем* ε_A^*. Мы подчеркиваем этот факт потому, что частое использование *сумм квадратов с нулевыми невязками*, в частности функции Розенброка (пример 4.2), в качестве тестовых функций иногда порождает заблуждение относительно точности, которая достижима для функций иного сорта.

8.5.2. ОЦЕНИВАНИЕ ТОЧНОСТИ

При огромном многообразии функций, встречающихся в оптимизационных задачах, невозможно предложить разумную универсальную оценку погрешностей их вычисления. В частности, не следует думать, что все функции могут вычисляться с полной машинной точностью. В данном разделе описаны методы оценивания точности, которые во многих практических случаях дадут хорошие результаты. Важно, однако, отметить, что они основаны на определенных предположениях и соответственно качество генерируемых оценок будет зависеть от того, выполнены эти предположения или нет.

Чтобы не писать громоздких формул, мы ограничимся изложением методов, которые позволяют оценивать точность вычисления дважды непрерывно дифференцируемых функций $f(x)$ *скалярного* аргумента. (Эти методы можно использовать и в n-мерном случае, если рассматривать поведение функции вдоль некоторого направления p, $\|p\|=1$.)

Бывает, что ошибки подсчета значений f почти целиком обусловлены некоторым блоком операций, выполняемым с фиксированной известной точностью. Обычно пользователь знает об этом и может применить данную информацию для определения ε_A.

Для общего случая мы дадим четыре метода оценивания точности. Первый предполагает вычисления с увеличенной разрядной сеткой, второй и третий требуют задания аккуратных значений производных f, четвертый опирается только на вычисляемые значения f.

8.5.2.1. Оценивание точности, когда можно увеличивать длину машинного слова. Очень простой и эффективный метод оценки ε_A доступен при условии, что имеется возможность вычислений с повышенной точностью. Она существует, например, если используемый ФОРТРАН-компилятор допускает спецификации типов переменных операторами REAL*n.

Предположим, что минимизация будет проводиться с одинарной точностью, и пусть fl_s и fl_t — результаты арифметических операций, выполняемых машиной в режимах «короткого» и «длинного» представлений чисел соответственно. Тогда для определения ε_A можно вычислить f в x и нескольких соседних точках в обоих режимах и найти максимум модуля полученных разностей значений, т. е. взять

$$\varepsilon_A \sim \max_i |fl_t(f(x_i)) - fl_s(f(x_i))|.$$

Просмотр нескольких (скажем, трех-четырех) точек необходим для повышения надежности (оценка по одной точке могла бы оказаться слишком оптимистичной: представьте, что f — относительно простая функция и x записывается в «короткое» слово без ошибки округления).

8.5.2.2. Оценивание точности с использованием производных. Методы оценки ε_A, описанные в этом разделе, предполагают доступность первой производной f.

Оценка, получающаяся из тейлоровского разложения. В основе представленного здесь способа оценивания ε_A лежит предположение о том, что ошибка, с которой f вычисляется в x, имеет тот же порядок, что и приращение вычисляемого значения f при *минимальной ощутимой вариации* x.

Обозначим минимальную ощутимую вариацию x через $h_{\min}$. Если $|x|$ есть число порядка единицы или больше, $h_{\min}$ обычно задают формулой $h_{\min} = \varepsilon_M |x|$. При малых $|x|$ величина $h_{\min}$ будет зависеть от задачи. Скажем, при $\varepsilon_M = 10^{-8}$ и $x = 10^{-20}$ изменение x на 10^{-28} для одних задач будет ощутимым, а для других нет. Во многих случаях минимальная ощутимая вариация x имеет порядок ε_M.

Точное значение f в возмущенной точке $x + h_{\min}$ связано с $f(x)$ тейлоровским разложением f в окрестности x:

$$f(x + h_{\min}) = f(x) + h_{\min} f'(\xi),$$

где ξ — некое число из диапазона $x \leqslant \xi \leqslant x + h_{\min}$. Если $f'(\xi) \approx f'(x)$, то при сформулированном выше предположении двумя оценками ε_A будут

$$\varepsilon_A \sim |f(x + h_{\min}) - f(x)| \tag{8.29}$$

$$\varepsilon_A \sim h_{\min} |f'(x)|. \tag{8.30}$$

Использование ошибки конечно-разностной аппроксимации. При *очень малых* значениях конечно-разностного интервала ошибка численного дифференцирования в основном определяется составляющей (ошибкой условий), которая отражает точность вычисления f (см. разд. 4.6.1.1 и 8.6). Следовательно, зная эту ошибку, можно воспользоваться ею для оценивания ε_A.

Рассмотрим конечно-разностное приближение φ производной f', вычисленное для интервала h, который настолько мал, что ошибкой отбрасывания можно пренебречь. Коль скоро при этом ошибка условий окажется близкой к ε_A/h, получим

$$|\varphi(h)-f'(x)| \sim \varepsilon_A/h \tag{8.31}$$

и соответственно

$$\varepsilon_A \sim h\,|\varphi(h)-f'(x)|. \tag{8.32}$$

Оценка (8.32) хороша только при условии, что выполнено (8.31), а (8.31) будет нарушено, если погрешности вычислений f в x и $x+h$ близки по модулю и имеют одинаковый знак. Поэтому предлагается считать φ тремя способами (по правой, левой и центральной формулам; см. разд. 8.6.1) и по найденным φ_F, φ_B и φ_C определять ε_A следующим образом:

$$\varepsilon_A \sim \max_{\varphi_F,\varphi_B,\varphi_C} h\,|\varphi(h)-f'(x)|. \tag{8.33}$$

Ради повышения надежности оценки советуем вычислять (8.33) по меньшей мере для двух (малых) конечно-разностных шагов. Подчеркнем еще раз: эти шаги надо выбрать так, чтобы погрешности конечно-разностных аппроксимаций производной почти целиком определялись ошибками условий. Интересно отметить, что при $\varepsilon_A=0$ формулы (8.33) и (8.30), по существу, совпадают.

Проиллюстрируем применение (8.32) на простом примере. Допустим, что используется машина с двенадцатиразрядной десятичной мантиссой и что $|x|$, $|f(x)|$, $|f'(x)|$, $|f''(x)|$ — величины порядка единицы, а $\varepsilon_A \sim 10^{-10}$. Тогда мантиссы вычисленных значений $f(x)$, $f(x+h)$ и $\Delta f=f(x+h)-f(x)$, получающихся при $h=10^{-6}$, будут

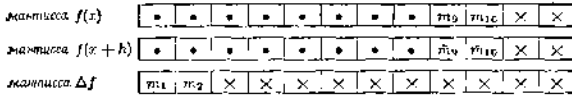

Рис. 8j. Схематическое изображение 12-разрядных мантисс операндов процедуры конечно-разностного оценивания точности вычисления f.

выглядеть так, как показано на рис. 8j. Ненадежные разряды помечены крестиками. По предположению относительно модулей f и ее производных в x восемь старших разрядов вычисленных значений $f(x)$ и $f(x+h)$ совпадут. Эти разряды помечены точками. После вычитания $f(x)$ из $f(x+h)$ останутся только четыре значащих разряда. Два младших будут ненадежными из-за неточности представления f; таким образом, в мантиссе Δf будут только две правильные цифры (десять младших разрядов Δf, помеченные крестиками ненадежны).

Деление Δf на h внесет дополнительную ошибку лишь в последний разряд. Следовательно, q_F будет отличаться от $f'(x)$ на величину порядка 10^{-2}. Так как $h=10^{-8}$, в соответствии с (8.32) получим (правильную) оценку

$$\varepsilon_\Delta \sim h|q_F - f'(x)| = 10^{-8}10^{-2} = 10^{-10}.$$

8.5.2.3. Оценивание точности по значениям функции. Если никакая информация о функции, кроме ее вычисляемых (с ошибками) значений, не доступна, оценить погрешности вычислений, вообще говоря, *нельзя* — нет точной величины, на которую можно было бы ориентироваться. Однако при определенных предположениях относительно старших производных f и статистических свойств ошибок оценивание становится возможным.

Допустим, что f подсчитана для ряда точек $\{x_i\}$, сгенерированных по правилу $x_i = x + ih$, причем $|h|$ — малая величина. По определению ε_Δ *вычисленные значения* $\bar{f}_i$ будут связаны с $f(x_i)$ следующим образом:

$$\bar{f}_i = f(x_i) + \delta_i = f(x_i) + \theta_i \varepsilon_\Delta, \tag{8.34}$$

где $|\theta_i| \leqslant 1$. Можно составить *таблицу разностей* (см. разд. 2.3.5), образовав ее первый столбец из чисел $\bar{f}_i$, а каждый последующий из разностей элементов предыдущего. В силу линейности разностного оператора в $(k+1)$-м столбце получим $\Delta^k \bar{f}_i = \Delta^k f_i + \Delta^k \delta_i$.

Как было указано в разд. 2.3.5, справедливо приближенное равенство $\Delta^k f \approx h^k f^{(k)}$. При этом, если величина h достаточно близка к нулю и f удовлетворяет необременительным требованиям, модуль $|h^k f^k|$ становится очень малым уже при умеренных k (скажем, $k \geqslant 4$). Следовательно, высшие разности вычисляемых значений f *почти целиком будут определяться ошибками* δ_i.

В некоторых предположениях относительно распределения величин $\{\theta_i\}$, трактуемых как случайные числа, разности $\Delta^k \bar{f}_i$ одного порядка должны с увеличением k *выравниваться по абсолютной величине при чередовании знаков*. Коль скоро эта тенденция отчетливо проявляется в каком-то столбце таблицы, по нему можно оценивать ε_Δ. Одна из формул, предлагавшихся для оценки ε_Δ по k-му столбцу выглядит так:

$$\varepsilon_\Delta^k \sim \frac{\max_i |\Delta^k \bar{f}_i|}{\beta_k}, \tag{8.35}$$

где

$$\beta_k = \sqrt{\frac{(2k)!}{(k!)^2}}.$$

Обычно желаемая картина наблюдается уже при k, равном 4 или 5, и крайне редко требуется строить больше десяти столбцов.

8.5.2.4. Численные примеры. В этом разделе представлены результаты применения вышеизложенных методов в конкретном случае.

Пример 8.3. Для тестирования была выбрана функция

$$f(x) = e^x + x^3 - 3x - 1.1.$$

Вычисления проводились с одинарной точностью на IBM 370 ($\varepsilon_M = 16^{-5} \approx 9.537 \times 10^{-7}$). Рассматривались две точки: $x_1 = 10$, в которой $f(x_1) = 2.29954 \times 10^4$, $f'(x_1) = 2.23235 \times 10^4$, и $x_2 = 1.115$, в которой $f(x_2) = -9.23681 \times 10^{-4}$, $f'(x_2) = 3.77924$ (все значения округлены до шести значащих цифр). Вторая точка специально подобрана так, чтобы значение функции было близким к нулю, — хотелось продемонстрировать работу методов в ситуации, когда ε_R в (8.26) существенно превосходит ε_M.

Начнем с результатов применения методов разд. 8.5.2.2 (производные f считались с одинарной точностью). В x_1 истинное значение ошибки подсчета f равно 2.5×10^{-2}. Оно меньше «стандартной» нижней границы ε_f^* (8.27) ($\varepsilon_f^* = 2.2 \times 10^{-2}$), но это не должно удивлять — ведь известно, что *действительные ошибки* могут сильно различаться даже для очень близких точек. Когда в качестве конечно-разностного интервала было задано число 1.049×10^{-3}, оценка ε_A, полученная по формуле (8.33), составила 3.7×10^{-2}; при конечно-разностном интервале 1.049×10^{-4} та же формула дала оценку ε_A, равную 6.1×10^{-3}. Формулы (8.29) и (8.30) дали 2.13×10^{-1} и 2.11×10^{-1} соответственно; эти оценки явно завышены, что объясняется большим значением производной $f'(x_1)$.

Для x_2 действительным значением ошибки вычисления f является 1.4×10^{-5}; оно превосходит «стандартную» нижнюю границу ε_A^* ($\varepsilon_A^* = 9.6 \times 10^{-7}$). Относительная ошибка в x_2 равна 1.5×10^{-2}. При конечно-разностном интервале 2.017×10^{-5} формула (8.33) дает в x_2 оценку ε_A, равную 2.9×10^{-5}. Если же судить только по φ_f (т. е. воспользоваться (8.32)), то получится величина порядка 10^{-5}; это подтверждает целесообразность операции взятия максимума по результатам применения трех конечно-разностных формул. При интервале $h = 2.017 \times 10^{-6}$ оценка ε_A, получающаяся по формуле (8.33), составила 2.7×10^{-6}. Формулы (8.29) и (8.30) дали 4.02×10^{-6} и 3.81×10^{-6} соответственно; в данном случае большого отличия от ε_A из (8.33) нет.

Перейдем теперь к методу из разд. 8.5.2.3. Столбцы разностей, генерируемые для опорной точки x_1 при $h = 10^{-2}$, частично показаны в табл. 8d. В четвертом и пятом столбцах наблюдается ожидаемое чередование знаков. Оценки ε_A, вычисленные по формуле (8.35) при $k = 4, \ldots, 7$, равны 1.02×10^{-2}, 6.40×10^{-3}, 6.30×10^{-3} и 5.93×10^{-3}. При этом *максимальная* из действительных ошибок в значениях $\bar{f}_j$, по которым строилась таблица,

Таблица 8d. Таблица разностей для примера 8.3 в x_1 при $h = 10^{-2}$

f_i	$\Delta^1 f_i$	$\Delta^2 f_i$	$\Delta^3 f_i$	$\Delta^4 f_i$	$\Delta^5 f_i$
2.300×10^{1}	2.24×10^{0}	2.02×10^{0}	-1.86×10^{-2}	$+5.05 \times 10^{-4}$	-9.38×10^{-2}
2.322×10^{1}	2.47×10^{0}	2.25×10^{0}	$+3.52 \times 10^{-2}$	-4.00×10^{-4}	$+1.21 \times 10^{-1}$
2.345×10^{1}	2.29×10^{0}	2.26×10^{0}	-7.91×10^{-3}	$+7.81 \times 10^{-4}$	-1.17×10^{-1}
2.365×10^{1}	2.30×10^{0}	2.27×10^{0}	$+7.03 \times 10^{-3}$	-6.64×10^{-4}	$+8.59 \times 10^{-2}$
2.391×10^{1}	2.34×10^{0}	2.34×10^{0}	$+2.91 \times 10^{-3}$	$+1.56 \times 10^{-3}$	-3.72×10^{-2}
2.414×10^{1}	2.36×10^{0}	2.35×10^{0}	$+1.95 \times 10^{-3}$	-1.95×10^{-3}	$+9.77 \times 10^{-4}$
2.438×10^{1}	2.38×10^{0}	2.37×10^{0}	0.0	$+3.91 \times 10^{-3}$	-1.56×10^{-2}

составила 2.5×10^{-2}, а *средняя* ошибка равна 8.1×10^{-3}. Таким образом, полученные оценки ε_A можно признать вполне удовлетворительными. Интересно отметить, что в данном примере значения θ_i из (8.34) *не были* равномерно распределены на отрезке $[-1, 1]$, так как ошибки вычисления f *во всех* точках вблизи x_1 положительны. Однако этого и не требуется — нужно только, чтобы ошибки не коррелировали и имели одинаковые варианты.

Метод из разд. 8.5.2.3 был опробован и в точке x_2. Разности, подсчитанные при $h = 10^{-3}$, сведены в табл. 8e. Здесь мы видим,

Таблица 8e. Таблица разностей для примера 8.3 в x_2 при $h = 10^{-3}$

f_i	$\Delta^1 f_i$	$\Delta^2 f_i$	$\Delta^3 f_i$	$\Delta^4 f_i$	$\Delta^5 f_i$
-6.238×10^{-2}	4.78×10^{-3}	1.24×10^{-3}	-4.57×10^{-6}	$+9.54 \times 10^{-6}$	-2.60×10^{-4}
-5.760×10^{-2}	5.80×10^{-3}	7.63×10^{-6}	$+4.97 \times 10^{-6}$	-1.05×10^{-5}	$+2.10 \times 10^{-4}$
-1.696×10^{-2}	5.80×10^{-3}	1.24×10^{-3}	$+5.75 \times 10^{-6}$	$+1.44 \times 10^{-5}$	-1.91×10^{-4}
-2.114×10^{-2}	5.82×10^{-3}	6.84×10^{-6}	$+5.72 \times 10^{-6}$	-7.63×10^{-6}	$+5.72 \times 10^{-6}$
5.696×10^{-3}	5.82×10^{-3}	1.24×10^{-3}	-1.91×10^{-6}	-1.91×10^{-6}	$+1.14 \times 10^{-5}$
0.777×10^{-3}	5.83×10^{-3}	1.05×10^{-6}	-3.82×10^{-6}	$+9.51 \times 10^{-6}$	-2.00×10^{-5}
4.361×10^{-2}	5.81×10^{-3}	6.84×10^{-6}	$+5.72 \times 10^{-6}$	-1.05×10^{-5}	$+2.01 \times 10^{-4}$

что, хотя элементы в столбцах 4 и 5 близки между собой по модулю, чередование знаков наблюдается только в группах из четырех-пяти элементов, но не в столбце в целом; такая картина типична. Для $k = 4, \ldots, 7$ значения $\varepsilon_A^{(k)}$ (вычисленные по формуле (8.35)) таковы: 1.03×10^{-5}, 1.08×10^{-6}, 1.10×10^{-6} и 1.07×10^{-6}. Максимальная среди действительных ошибок в $\bar{f}_i$ равна 5.1×10^{-5}, а средняя действительная ошибка составила 4.1×10^{-5}; таким образом, оценка ε_A снова получилась достаточно точной. Как и x_1, точка x_2 характерна тем, что вблизи нее вычисленные значения f *всегда превышают* истинные.

Наконец, для x_3 ставился эксперимент, когда все величины считались с *двойной* точностью и лишь значения f вычислялись с *одинарной*. (На IBM 370 двойной точности соответствует $\varepsilon_M = 16^{-13} \approx 2.22 \times 10^{-16}$.) Цель состояла в том, чтобы показать, как сработает метод из разд. 8.5.2.3, если ε_A намного больше

стандартной нижней границы ε_A^*. Повышение точности вспомогательных расчетов не должно было существенно сказаться на оценках ε_A. В то же время оно позволило уменьшить h до 10^{-6}. При этом в столбцах таблицы разностей, начиная с четвертого, наблюдалась устойчивая картина чередования знаков. Формула (8.35) дала для $k=4, \ldots, 7$ оценки ε_A, равные 1.14×10^{-6}, 1.20×10^{-6}, 1.25×10^{-6} и 1.29×10^{-6}. Как и ожидалось, они практически те же, что и при реализации метода с одинарной точностью.

8.5.3. ПЕРЕОЦЕНИВАНИЕ ТОЧНОСТИ

Если оптимизационному алгоритму нужны оценки погрешностей подсчета функций, то эти оценки, как правило, требуются для многих точек. Фиксировать на все время решения задачи одно значение ε_A не рекомендуется: действительные ошибки вычислений f зависят от аргумента x, а он по ходу итераций может существенно изменяться. В то же время применять на каждой итерации какой-то из методов разд. 8.5.2 слишком накладно. Разумный выход состоит в том, чтобы, получив хорошую оценку ε_A в начальной точке, использовать ее далее как опорную величину для построения последующих оценок по достаточно простому правилу.

Первое, что приходит на ум,— принять гипотезу постоянства относительной ошибки и, оценив ε_A в x_0, впоследствии пользоваться формулой

$$\varepsilon_A(x_k) = |f(x_k)|\,\frac{\varepsilon_A(x_0)}{|f(x_0)|}.$$

Однако в общем случае этот подход неправомерен: при очень малых $|f|$ соответствующие оценки ε_A могут оказываться сильно занижеными. Здесь уместно напомнить, что для многих функций величина ε_A не бывает меньше стандартной нижней границы ε_A^* (см. разд. 8.5.1).

Мы рекомендуем модель поведения ε_A, в основе которой лежит предположение о том, что для всех x «число правильных цифр» в значении f будет одинаковым, причем *при малых $|f|$ предлагается учитывать старшие нули после десятичной запятой*. Считается также, что это число не может превосходить количества разрядов, выделяемых в машинном слове для записи мантиссы. Соответствующая формула расчета оценок ε_A такова:

$$\varepsilon_A(x) = (1+|f(x)|)\max\left(\varepsilon_M,\ \frac{\varepsilon_A(x_0)}{1+|f(x_0)|}\right). \qquad (8.36)$$

Проиллюстрируем работу (8.36) на функции из примера 8.3. При $x_0=10$ стандартная нижняя граница ε_A^* больше, чем вычисляемые обычными способами оценки ε_A, и поэтому для x^* по формуле (8.36) получим $\varepsilon_A(x_*) = \varepsilon_M(1+|f(x_*)|) = 9.6\times10^{-7}$.

Если же взять $x_0 = 1.115$ и положить $\varepsilon_д(x_0) = 1.2 \times 10^{-6}$ (примерно такое значение дает метод из разд. 8.5.2.3), то для x_1 формула (8.36) дает оценку $\varepsilon_д$, равную 2.7×10^{-2}.

Замечания и избранная библиография к разделу 8.5

Тема оценивания точности обсуждалась Стюартом (1967), Брентом (1973а), Кертисом и Райдом (1974). Прием, описанный в разд. 8.5.2.3, соответствует методу Хэмминга (1962, 1973). Подробнее этот метод разобран Ляйнессом (1977а).

8.6. ВЫБОР КОНЕЧНЫХ РАЗНОСТЕЙ

Для гладких задач без ограничений и с линейными ограничениями, характеризующихся значительными размерностями и большими трудоемкостями вычисления производных целевых функций, одними из наиболее эффективных являются квазиньютоновские методы с конечно-разностной аппроксимацией градиентов (см. разд. 4.6.2 и 5.1.2.4). При этом успех применения методов данного класса сильно зависит от того, насколько точно будут аппроксимироваться градиенты — здесь точность намного важней, чем, скажем, при оценивании вторых производных (по разностям первых) в дискретных ньютоновских методах.

Ниже рассмотрена некая техника выбора конечно-разностных интервалов, обычно обеспечивающая удовлетворительное качество аппроксимаций. Правда, если задача плохо отмасштабирована или счет начинается с «нехарактерной точки», эта техника может подвести. Процедур, которые хорошо работали бы во всех случаях, не существует, но от наиболее типичных неприятностей можно застраховаться. На это и ориентированы предлагаемые ниже методы.

Ради простоты изложения мы ограничимся разбором проблемы выбора интервала для расчета оценки производной функции $f(x)$ *одной переменной*; многомерный случай кратко обсуждался в разд. 4.6.1.3. Будет предполагаться, что точность вычисления f известна (см. разд. 8.5).

8.6.1. ОШИБКИ КОНЕЧНО-РАЗНОСТНЫХ ПРИБЛИЖЕНИЙ; ХОРОШО ОТМАСШТАБИРОВАННЫЕ ФУНКЦИИ

В общем случае ошибки конечно-разностных приближений зависят от величин, которые остаются неизвестными (в частности, от значений высших производных в неких промежуточных точках). Поэтому формулы для этих ошибок прежде всего нужны для понимания их зависимости от конечно-разностного интервала. Однако при определенных предположениях относительно f и ее производных эти формулы позволяют также получить хорошие *априорные* оценки оптимальных h.

8.6.1.1. Аппроксимация по правым разностям. Просуммируем результаты разд. 4.6.1.1, касающиеся ошибок аппроксимации $f'(x)$ по правой конечно-разностной формуле

$$\varphi_F(f, h) = \frac{f(x+h)-f(x)}{h},$$

где $h>0$. Отличие *вычисляемого* значения φ_F от $f'(x)$ в основном определится суммой *ошибки отбрасывания* (игнорируются слагаемые тейлоровского разложения порядков выше первого) и *ошибки условий* (значения f вычисляются с погрешностями). Эта сумма не превосходит

$$\frac{h}{2}|f''(\xi)| + \frac{2}{h}\varepsilon_A, \tag{8.37}$$

где ξ — некоторое число из отрезка $[x, x+h]$, а ε_A — верхняя граница погрешностей вычисления f в x и в $x+h$ (см. разд. 8.5.1). Минимум (8.37) достигается при h, равном

$$h_F = 2\sqrt{\frac{\varepsilon_A}{|f''(\xi)|}}, \tag{8.38}$$

причем минимальным значением (8.37) будет $2\sqrt{\varepsilon_A|f''(\xi)|}$. К сожалению, ни производная f'', ни точка ξ не известны, и поэтому для оценки h_F на основе (8.38) нужны дополнительные предположения.

Допустим, в частности, что f не нуль и что ε_A можно выразить через известную границу ε_R *относительной ошибки*, т. е.

$$\varepsilon_A = |f|\varepsilon_R. \tag{8.39}$$

Если к тому же для всех точек отрезка $[x, x+h]$ справедливо соотношение $O(|f|) = O(|f'|)$, то из (8.38) будет следовать $h_F \sim \sqrt{\varepsilon_R}$.

Наконец, при $O(|f'|) = O(|f|)$ можно утверждать, что «минимальная» *относительная ошибка* аппроксимации $f'(x)$ имеет порядок $\sqrt{\varepsilon_R}$. Это утверждение согласуется с бытующим среди программистов мнением, что при оптимальном конечно-разностном интервале количество правильных цифр в φ_F оказывается *вдвое меньше*, чем в f.

Мы подчеркиваем, что предыдущее высказывание справедливо только при сделанных предположениях относительно значений f и ее производных. Отметим также, что если модуль $|f'|$ мал по сравнению с $|f|$ и $|f''|$ (как обычно бывает вблизи локального минимума f), то относительное уклонение φ_F от f' может оказаться большим, даже когда конечно-разностный интервал выбран наилучшим образом.

Все сказанное выше можно было бы повторить применительно к ошибкам аппроксимации $f'(x)$ по левой конечно-разностной формуле

$$\varphi_B(f, h) = \frac{f(x) - f(x-h)}{h}$$

с тем единственным отличием, что $\xi \in [x-h, x]$.

8.6.1.2. Аппроксимация по центральным разностям. Первую производную можно аппроксимировать и по центральной конечно-разностной формуле

$$\varphi_C(f, h) = \frac{f(x+h) - f(x-h)}{2h}, \tag{8.40}$$

где $h > 0$. В этом случае ошибка отбрасывания составит $^1/_6 h^2 |f'''(\mu)|$, где $\mu \in [x-h, x+h]$, а ошибка условий будет ограничена сверху величиной ε_A/h. Таким образом, суммарная погрешность не превзойдет $^1/_6 h^2 |f'''(\mu)| + \varepsilon_A/h$. Минимум этой величины достигается при h, равном

$$h_C = \sqrt[3]{\frac{3\varepsilon_A}{|f'''(\mu)|}}$$

Если выполнено (8.39) и для всех точек из $[x-h, x+h]$ справедливо отношение $O(|f|) = O(|f'''|)$, то получим

$$h_C \sim \sqrt[3]{\varepsilon_R}. \tag{8.41}$$

Важно подчеркнуть, что (8.41) реализуется только при сделанных предположениях.

8.6.1.3. Разность второго порядка. Для аппроксимации f'' обычно используют следующую формулу:

$$\Phi(f, h) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}. \tag{8.42}$$

Ошибка отбрасывания при этом равна $^1/_{12} h^2 f^{(4)}(\eta)$, где $\eta \in [x-h, x+h]$, а ошибка условий, возникающая в связи с неточностью вычисленных значений $f(x)$, $f(x+h)$ и $f(x-h)$, не превзойдет величины $4\varepsilon_A/h^2$, где ε_A — оценка сверху погрешностей подсчета f в точках x, $x-h$ и $x+h$. Таким образом, суммарная ошибка приближения $f''(x)$ вычисляемым значением Φ будет не больше, чем $4\varepsilon_A/h^2 + (h^2/12)|f^{(4)}(\eta)|$. Минимум этой суммы достигается при h, равном

$$h_\Phi = 2\sqrt[4]{\frac{3\varepsilon_A}{|f^{(4)}(\eta)|}}. \tag{8.43}$$

Если f удовлетворяет (8.39) и $O(|f|) = O(|f^{(4)}|)$ для всех точек из $[x-h, x+h]$, то $h_\Phi \sim \sqrt[4]{\varepsilon_R}$.

8.6.2. ПРОЦЕДУРА АВТОМАТИЧЕСКОГО ОЦЕНИВАНИЯ КОНЕЧНО-РАЗНОСТНЫХ ИНТЕРВАЛОВ

В этом разделе описан алгоритм вычисления «хороших» конечно-разностных интервалов, предназначенный для оптимизационных методов с оцениванием f' по правой конечно-разностной формуле. Речь пойдет не об отыскании максимально точного приближения производной при единственном значении аргумента, а о том, как получить «разумный» интервал, обеспечивающий удовлетворительные оценки производных в разных точках, получающихся по ходу минимизации. Предлагаемый алгоритм можно применять в n-мерном случае, независимо определяя интервалы для разных переменных.

8.6.2.1. Исходные соображения. В основе алгоритма лежит оценка (6.37), но поскольку f'' и ξ неизвестны, предполагается аппроксимация $f''(\xi)$ с использованием формулы (8.42). Это в свою очередь означает, что должен быть найден интервал h_Φ, позволяющий получить разумную оценку Φ. Затем в качестве искомого приближения «оптимального» h для правой конечно-разностной формулы берется

$$h_F = 2\sqrt{\frac{\varepsilon_A}{|\Phi|}}. \tag{8.44}$$

Такой способ выбора h_F оправдан, если (i) между $f'(x)$ и $f''(\xi)$ нет большой разницы (т. е. вторая производная меняется вблизи x не слишком быстро) и (ii) Φ достаточно хорошо приближает $f''(x)$. Имея в виду назначение Φ, можно довольствоваться совпадением у Φ и $f''(x)$ лишь старшего разряда.

При поиске h_Φ учитывается, что ошибка отбрасывания в (8.42) ограничена сверху величиной, которая с ростом h, вообще говоря, возрастает, а ошибка условий при увеличении h имеет тенденцию убывать. Формально поиск выглядит как перебор пробных значений $\{h_i\}$. Качество соответствующих Φ определяется по вычисляемым оценкам относительных ошибок условий:

$$\hat{C}(\Phi) = \frac{4\varepsilon_A}{h^2|\Phi|}. \tag{8.45}$$

(Когда Φ оказывается нулем, вместо правой части (8.45) берется произвольное большое число.) Никаких попыток явно оценивать ошибки отбрасывания не делается.

Критерием приемлемости Φ считается принадлежность $\hat{C}(\Phi)$ отрезку $[0.001, 0.1]$. Зачем нужно ограничивать $\hat{C}(\Phi)$ сверху, понятно без пояснений. Что же касается требования, чтобы величина $\hat{C}(\Phi)$ была не меньше, чем 0.001, то его надо трактовать как неявный способ ограничить ошибку отбрасывания (напомним, что ошибки отбрасывания и условий, грубо говоря, связаны обратной пропорцией).

Если оценка $\hat{C}(\Phi)$, отвечающая очередному h_i, оказывается неудовлетворительной, следующий пробный интервал берется больше или меньше, чем h_i, в зависимости от того, какая граница нарушена. При $\hat{C}(\Phi)>0.1$ интервал увеличивается, а при $\hat{C}(\Phi)<0.001$ уменьшается. Выбрана простейшая схема, в которой h_i либо умножается, либо делится на 10. Можно было бы предложить и более хитроумные правила пересчета h_i, учитывающие конкретный вид зависимости (8.45). Однако подобные правила всегда опираются на какие-то дополнительные предположения (например, о постоянстве порядка Φ при изменениях h), а они могут не выполняться.

В двух ситуациях алгоритм не сумеет обеспечить подходящего значения $\hat{C}(\Phi)$. Во-первых, ему не удастся добиться неравенства $\hat{C}(\Phi)\leqslant 0.1$, если Φ при всех h будут почти нулевыми. Так случается, когда функция f почти постоянна, почти линейна или $f(y-x)$ нечетна по y. Во-вторых, не исключено, что из-за слишком больших Φ даже при очень близких к нулю h не будет выполняться неравенство $\hat{C}(\Phi)\geqslant 0.001$. Это возможно, когда x похожа на точку излома первой производной.

Для контроля пригодности результатов применения алгоритма предусмотрено сравнение оценок $f'(x)$ по правой и центральной конечно-разностным формулам с интегралами h_{i^+} и h_{i^-} соответственно. Если эти оценки получаются совсем разными, ни той ни другой доверять нельзя.

Начальный пробный интервал h_0 определяется в предположении, что модули значений x, $f(x)$, и $f''(x)$ связаны следующим образом:

$$O(|f''|)=O\left(\frac{\omega+|f(x)|}{(\eta+|x|)^2}\right), \tag{8.46}$$

где $\omega>0$ и $\eta>0$. Параметр ω введен в числитель правой части (8.46), чтобы не было необходимости отдельно рассматривать случай, когда модуль $|f|$ очень мал; из аналогичных соображений в знаменатель введен параметр η. В остальном вид правой части (8.46) согласован с характером зависимости второй производной от выбора масштабов функции и аргумента. Поскольку гипотеза о существовании связи (8.46) используется только при вычислении h_0, на значении окончательного интервала h_0 она практически не отражается; однако эффективность алгоритма при разных ω и η может быть разной. На основании своего опыта мы рекомендуем $\omega=1$ и $\eta=1$.

Подстановкой (8.46) в (8.38) получается интервал $\bar{h}=2(\eta+|x|)\sqrt{\varepsilon_M/(\omega+|f|)}$. Если (8.46) действительно имеет место, отвечающая $\bar{h}$ ошибка условий в φ_F будет близка к «оптимальной». В то же время формула (8.42) с $h=\bar{h}$ даст относительную ошибку $C(\Phi)$ порядка единицы (т. е. у Φ не будет ни одной правильной цифры). Исходя из этого предела остается брать $h_0=10\bar{h}$, что позволяет рассчитывать на $\bar{C}(\Phi)$ порядка 0.01

(поскольку $\hat{C}(\Phi)$ обратно пропорциональна h^2). Для хорошо отмасштабированной (в смысле разд. 8.6.1.1) функции f значение $C(\Phi)$, отвечающее h_0, попадет в требуемый диапазон, так что другие интервалы рассматривать не придется.

8.6.2.2. Формулировка алгоритма. Описанный ниже алгоритм FD вычисляет интервал h_F, который должен обеспечивать хорошее качество аппроксимации f' по правой конечно-разностной формуле, а также величины q (оценку $f'(x)$), Φ (оценку $f''(x)$) и E_F (верхнюю границу погрешности оценки q). Число итераций ограничивается параметром K. На тот случай, когда после просмотра K пробных интервалов подходящего h_F получено не будет, организуется хранение h_s — минимального из тех h_k, для которых оценки относительных ошибок условий в приближениях $f'(x)$ по правой и левой конечно-разностным формулам имеют приемлемые значения (не превосходят 0.1). Таким образом, при каждом h_k вычисляются

$$C(q_F) = \frac{2\varepsilon_A}{h|q_F|}, \quad \hat{C}(q_B) \equiv \frac{2\varepsilon_A}{h|q_B|}.$$

(Если q_F или q_B оказывается нулем, в качестве соответствующей $\hat{C}$ берется произвольное большое число.) Формально алгоритм определяется так:

Алгоритм FD (*автоматическое оценивание h_F, f' и f''*)

FD1. [Инициализация.] Зафиксировать ω, η и K. Вычислить $f(x)$ и $h_0 = 10\bar{h}$, где

$$\bar{h} = 2(\eta + |x|)\sqrt{\frac{\varepsilon_A}{\omega + |f|}}. \tag{8.47}$$

Присвоить $k \leftarrow 0$, $h_s \leftarrow -1$. Вычислить $f(x+h_0)$, $f(x-h_0)$, $q_F(h_0)$, $q_B(h_0)$, $\Phi(h_0)$, $\hat{C}(q_F)$, $\hat{C}(q_B)$, $\hat{C}(\Phi)$.

FD2. [Проверка пригодности начального интервала.] Если $\max\{\hat{C}(q_F), \hat{C}(q_B)\} \leqslant 0.1$, присвоить $h_s \leftarrow h_0$. Если $0{,}001 \leqslant \hat{C}(\Phi) \leqslant 0.1$, присвоить $h_\Phi \leftarrow h_0$ и перейти к шагу FD5. Если $\hat{C}(\Phi) < 0{,}001$, перейти к шагу FD4; иначе перейти к шагу FD3.

FD3. [Увеличение h.] Присвоить $k \leftarrow k+1$, $h_k \leftarrow 10h_{k-1}$. Вычислить соответствующие конечно-разностные оценки и связанные с ними относительные ошибки условий. Если $h_s < 0$ и $\max\{\hat{C}(q_F), \hat{C}(q_B)\} \leqslant 0.1$, присвоить $h_s \leftarrow h_k$. Если $C(\Phi) \leqslant 0.1$ присвоить $h_\Phi \leftarrow h_k$ и перейти к шагу FD5. Если $k = K$, перейти к шагу FD6; в противном случае, повторить шаг FD3.

FD4. [Уменьшение h.] Присвоить $k \leftarrow k+1$ и $h_k \leftarrow h_{k-1}/10$. Вычислить соответствующие конечно-разностные оценки и

связанные с ними относительные ошибки условий. Если $\hat{C}(\Phi) > 0.1$, присвоить $h_\Phi \leftarrow h_{k-1}$ и перейти к шагу FD5. Если $\max\{\hat{C}(\varphi_F),\ \hat{C}(\varphi_B)\} \leqslant 0.1$, присвоить $h_\Phi \leftarrow h_k$. Если $0.001 \leqslant \hat{C}(\Phi) \leqslant 0.1$, присвоить $h_\Phi \leftarrow h_k$ и перейти к шагу FD5. Если $k = K$, перейти к шагу FD6; в противном случае повторить шаг FD4.

FD5. [Расчет оценки оптимального интервала.] Определить h_F по формуле (8.44) и вычислить $\varphi = \varphi_F(h_F)$, $\varphi_c(h_\Phi)$,

$$E_F = \frac{h_F|\Phi|}{2} + \frac{2\varepsilon_A}{h_F} \tag{8.48}$$

и модуль $\bar{E} = |\varphi - \varphi_c(h_\Phi)|$. Если $\max\{E_F, \bar{E}\} \leqslant 0.5$, выдать сообщение об успешном завершении счета; иначе выдать сообщение об отказе.

FD6. [Разбор ситуации, когда искомого h_Φ получить не удалось.] Если $h_k < 0$ (т. е. для всех h_k выполнилось неравенство $\max\{\hat{C}(\varphi_F), \hat{C}(\varphi_B)\} > 0.1$), то похоже, что f почти постоянна, и тогда присвоить $h_F \leftarrow \bar{h}$ (8.47), $\varphi \leftarrow 0$, $\Phi \leftarrow 0$ и $E_F \leftarrow 0$. Если $\hat{C}(\Phi) > 0.1$ и $h_k > 0$, то похоже, что f нечетна или почти линейна, и тогда присвоить $h_I \leftarrow h_k$, $\varphi \leftarrow \varphi_F(h_I)$, $\Phi \leftarrow 0$, а E_F вычислить по формуле (8.48). Иначе похоже, что f' быстро растет с уменьшением h (поскольку $\hat{C}(\Phi) < 0.001$ для всех h_k), и тогда присвоить $h_F \leftarrow h_K$, $\varphi \leftarrow \varphi_F(h_I)$, $\Phi \leftarrow \Phi(h_I)$ а E_F вычислить по формуле (8.48). Во всех трех случаях выдать сообщение об отказе.

8.6.2.3. Численные примеры. Описанные ниже эксперименты с алгоритмом FD проводились на IBM 370, причем вычисления осуществлялись с одинарной точностью ($\varepsilon_A \approx .9375 \times 10^{-6}$). Параметры алгоритма во всех случаях были таковы: $K = 6$, $\omega = .1$, $\eta = 1$. Значения ε_A определялись процедурой, обсуждавшейся в разд. 8.5.2.3.

Пример 8.4. Когда алгоритм FD применялся к функции из примера 4.9 (разд. 4.6) при $x = 1$ и $\varepsilon_A = .4 \times 10^{-6}$, относительная ошибка $\hat{C}(\Phi)$, отвечающая интервалу h_6 ($.398 \times 10^{-1}$), составила $.416 \times 10^{-2}$, т. е. оказалась меньше установленной нижней границы. После деления h_6 на 10 значение $\hat{C}(\Phi)$ стало равным $.424 \times 10^{-1}$ и, соответственно, был зафиксирован интервал $h_\Phi = h_5$. Подстановка h_Φ в (8.42) дала $\Phi = .238294 \times 10^2$ и это — хорошая оценка $f''(x) = .242061 \times 10^2$. По Φ были вычислены $h_F = .819 \times 10^{-3}$ и $\varphi_F(h_F) = .955636 \times 10^1$. Относительная погрешность полученной оценки $\varphi_F(h_F)$ равна $.807 \times 10^{-5}$. Заметим, что выбранное значение h_F вполне согласуется с данными табл. 4а и что достигнута максимальная точность, на которую можно надеяться при использовании шестиразрядной арифметики.

Пример 8.5. Второй тестовой функцией послужила

$$f(x) = (x-100)^2 + 10^{-6}(x-300)^3.$$

Рассматривалась точка $x=0$, где $\varepsilon_A = .9\times10^{-2}$. Истинными значениями $f(x)$ и $f'(x)$ являются $.997300\times10^4$ и $.199820\times10^1$, так что связь (8.46) отсутствует. Поскольку начальный интервал h_0 оказался слишком малым, алгоритму пришлось дважды увеличить h прежде, чем получилась приемлемая относительная ошибка $\tilde{C}(\Phi)$. В результате были зафиксированы $\Phi = .199567\times10^1$, $h_F = .134$ и $\varphi_F = -.199632\times10^1$. В то же время, точное значение $f'(x)$ равно $-.199730\times10^1$. Таким образом, относительная погрешность оценки составила $.49\times10^{-3}$.

Пример 8.6. Как уже отмечалось в разд. 8.6.1.1, при близких к нулю $|f'|$ правая конечно-разностная формула дает плохую относительную точность приближения f', даже если h выбирается наилучшим образом. Чтобы проиллюстрировать это, мы применили алгоритм FD к $f(x) = x^4 + 3x^2 - 10x$ в точке $x = .99999$, где $\varepsilon_A = 7\times10^{-5}$ и $f'(x) = -.180244\times10^{-3}$. Требования к $\tilde{C}(\Phi)$ удовлетворились при $h = h_0$ и соответствующая оценка $\Phi = .180008\times10^{-3}$ оказалась вполне приемлемой ($f'(x) = .179999\times10^{-3}$). Отвечающее ей значение h_F равно $.125\times10^{-4}$. При этом $\varphi_F(h_F) = .913\times10^{-2}$, т. е. отличается от $f'(x)$ почти на два порядка (хотя h_F — хорошее приближение оптимального интервала). Центральная конечно-разностная формула с $h = h_0$ тоже определила неудовлетворительную оценку: $\varphi_C(h_\Phi) = -.357\times10^{-2}$. Поскольку $\varphi_F(h_F)$ и $\varphi_C(h_\Phi)$ оказались совершенно разными, алгоритм остановился с признаком отказа.

8.6.2.4. Оценивание конечно-разностного интервала в произвольной точке. В процедуре, представленной в разд. 8.6.2.2, для оценки одной компоненты градиента требуются по меньшей мере три вычисления функции. Поэтому применять ее на каждой итерации алгоритма поиска минимума с конечно-разностными приближениями производных слишком накладно. К счастью, необходимости в этом нет: во-первых, что самое главное, для многих встречающихся на практике функций, интервалы, генерируемые в разных точках, оказываются довольно похожими. Во-вторых, алгоритму минимизации нужен «разумный» уровень точности приближения градиента в целом; по отдельным компонентам относительные ошибки могут быть большими, т. е. не обязательно, чтобы на каждой итерации все конечно-разностные интервалы имели значения, близкие к оптимальным.

Основываясь на высказанных положениях, мы рекомендуем обращаться к процедуре из разд. 8.6.2.2 один раз, в «типичной» точке (как правило, в x_0), и сохранять полученные интервалы неизменными на всех последующих итерациях. Возьмем, к примеру,

функцию из примера 8.4. В $x_0 = 1$ алгоритм FD дает для нее $h_F = .819 \times 10^{-8}$. Если использовать этот интервал для оценки $f'(x)$ при $x_1 = 10$, то относительная погрешность φ_F составит $.409 \times 10^{-8}$ (значения φ_F и f' для $x = 10$ равны $.971 \times 10^6$ и $.970286 \times 10^6$). Пусть теперь алгоритм FD применяется в точке $x_0 = 10$ (с $\varepsilon_A = .5 \times 10^{-8}$). Тогда получится интервал $h_F = .161 \times 10^{-7}$ и если использовать его для оценки $f'(x)$ при $x = 1$, относительная погрешность φ_F окажется равной $.102 \times 10^{-2}$. Подобная картина типична, хотя, разумеется, нетрудно сконструировать такую f, чтобы «оптимальные» интервалы в разных точках были совершенно разными.

В заключение отметим, что в n-мерном случае наряду с набором конечно-разностных интервалов алгоритм FD выдаст набор оценок диагональных элементов матрицы Гессе в x_0. Для квазиньютоновского метода это — очень полезная дополнительная информация.

8.6.2.5. Конечно-разностные приближения при решении задач с ограничениями. Подход, изложенный в разд. 8.6.2.1, применим и для выбора интервалов при непосредственной аппроксимации спроектированных градиентов $Z(x)^T g(x)$ в методах поиска условного минимума: просто надо рассматривать приращения x не вдоль координатных осей, а вдоль векторов, являющихся столбцами матрицы $Z(x)$. Никаких осложнений из-за такой модификации не возникнет. Другое дело, что не ясно, как пересчитывать интервалы по ходу оптимизации — ведь в моменты смен рабочего списка матрица Z может полностью измениться. Многие вопросы касательно эффективной организации вычисления конечно-разностных интервалов при решении задач с ограничениями пока еще открыты.

Иногда считают, что трудности, связанные с потерей относительной точности аппроксимации производных вблизи решения, не возникнут, если при поиске условного оптимума оценивать полный градиент, а не спроектированный. Основанием служит отличие $g(x^*)$ от нуля. Можно, однако, показать, что конечно-разностное приближение градиента в точке x, близкой к x^*, будет иметь высокую точность *только в той своей составляющей, которая принадлежит ранг-пространству* $\bar{A}(x)^T$ и при умножении $g(x)$ на $Z(x)^T$ обнуляется. Для оптимизации же важна точность приближения составляющей $g(x)$ из нуль-пространства $\bar{A}(x)^T$, т. е. точность оценивания компонент спроектированного градиента (даже если он не строится явно).

Замечания и избранная библиография к разделу 8.6

Ссылки по поводу конечно-разностной аппроксимации производных даны в замечаниях к разд. 4.6.

8.7. ПОДРОБНЕЕ О МАСШТАБИРОВАНИИ

В разд. 7.5.1 уже обсуждалась одна из простых форм масштабирования оптимизационных задач, а именно диагональное преобразование переменных с целью добиться, чтобы в представляющей интерес области их значения стали величинами одного порядка. Ниже идеи масштабирования трактуются в более широком плане, причем речь пойдет как о задачах безусловной минимизации, так и о задачах с ограничениями. Все представленные далее приемы предполагают доступность производных.

8.7.1. МАСШТАБИРОВАНИЕ ЗАМЕНОЙ ПЕРЕМЕННЫХ

В этом разделе обсуждается масштабирование задач путем замены исходных переменных x преобразованными переменными y. С теоретической точки зрения задача, получающаяся в результате такого преобразования, эквивалентна исходной; однако из материала предыдущих разделов данной главы должно быть ясно, что о практической эквивалентности говорить не приходится.

Мы будем рассматривать только *линейные* замены вида

$$x = Ly, \tag{8.49}$$

где L — фиксированная невырожденная матрица. (О нелинейных заменах кратко говорилось в разд. 7.4.1.) Обозначим через $\mathcal{F}(y)$ суперпозицию F и (8.49), т. е.

$$\mathcal{F}(y) \equiv F(Ly). \tag{8.50}$$

Тогда производные $\mathcal{F}$ по y будут связаны с производными F по x следующим образом: $\nabla_y \mathcal{F} = L g(x)$, $\nabla_y^2 \mathcal{F} = L^T G(x) L$.

8.7.1.1. Инвариантность методов относительно замен переменных. Допустим, что некая задача сначала была сформулирована в переменных x, а затем был совершен переход к переменным y, связанным с x преобразованием (8.49). Пусть, далее, к исходной и преобразованной задачам применяется один и тот же метод, причем в качестве начальных приближений берутся соответственные точки, т. е. $x_0 = Ly_0$. Обозначим через $\{x_k\}$ и $\{y_k\}$ генерируемые последовательности. Если

$$x_k = Ly_k \tag{8.51}$$

для всех k и это равенство при определенных условиях гарантируется независимо от конкретной формулировки задачи и конкретной матрицы L, то говорят, что метод *инвариантен относительно линейных замен переменных*.

В отсутствие погрешностей машинной арифметики рассматриваемым свойством инвариантности обладали бы многие методы. Среди них ньютоновские с использованием точных матриц Гессе и ряд квазиньютоновских, оперирующих точными значениями градиен-

тов (и те и другие обеспечивали бы (8.51) при условии, что матрицы Гессе или ее приближения будут положительно определенными на всех итерациях). Однако при машинной реализации методов инвариантность безвозвратно теряется.

Первая причина — неинвариантность вычислений с конечной точностью по отношению к масштабам операндов. Например, ошибки округления при подсчете на машине сумм x_1+x_2 и $\alpha x_1+\beta x_2$ будут связаны простым образом только при $\alpha=\beta$.

Вторая причина — очень часто по соображениям численной устойчивости приходится делать так, чтобы «достаточно близкие к нулю» величины трактовались программой как нулевые. При этом невозможно организовать разделение этих величин на действительно несущественные и существенные, но получающиеся малыми в силу выбора неудачных масштабов. Как следствие, некоторые алгоритмы могут быть почти инвариантны к умножению переменных на большое число и совсем не инвариантны к умножению на малое.

8.7.1.2. Обусловленность матрицы Гессе. Важной характеристикой задачи минимизации без ограничений является число обусловленности матрицы Гессе ее целевой функции в точке оптимума. Если оно велико, отклики F на одинаковые по норме, но разные по направлению вариации x^* могут быть существенно разными. Эта одна из форм плохой отмасштабированности.

С вычислительной точки зрения большой разброс собственных чисел $G(x^*)$ нежелателен по нескольким причинам. В частности, как следует из разд. 8.2.2.1, чем хуже обусловлена $G(x^*)$, тем меньше точность численного решения, на которую можно рассчитывать. К тому же плохая обусловленность $G(x^*)$ отрицательно сказывается на характеристиках методов. Например, она может приводить к деградации (утрате сверхлинейной сходимости) алгоритмов, в которых направления спуска определяются решением систем линейных уравнений, включающих матрицы $G(x)$ (или их аппроксимации).

Один из способов улучшить обусловленность $G(x^*)$ — подобрать подходящую замену переменных. При известной (и положительно определенной $G(x^*)$) идеальным было бы преобразование (8.49) с

$$L=G(x^*)^{-1/2}, \tag{8.52}$$

после которого матрица Гессе, отвечающая решению, становится единичной. (Некоторые алгоритмы безусловной минимизации можно проинтерпретировать как итерационный поиск матрицы (8.52); в частности, к таковым относятся квазиньютоновские методы и методы сопряженных градиентов.) Однако, поскольку обычно мы не знаем $G(x^*)$, на практике приходится использовать другие L. Если вторые производные доступны и есть основания полагать, что в точках x_0 и x^* они будут различаться «не слишком сильно», то в качестве L для (8.49) берут $G(x_0)^{-1/2}$. Располагая лишь первыми производными, L строят по конечно-разностной аппроксимации $G(x_0)$.

Наконец, не имея возможности вычислять даже градиенты, иногда применяют диагональное масштабирование. Точнее говоря, выполняют преобразование с матрицей L, у которой на диагонали стоят квадратные корни (вычисляемых по значениям F) конечно-разностных оценок в x_0 соответствующих вторых производных (процедура построения таких оценок описана в разд. 8.6.2.2), а остальные элементы равны нулю. Еще раз подчеркнем, что все это имеет смысл, только если между матрицами Гессе в x_0 и в x^* нет больших различий.

Чтобы показать, что даже простой прием диагонального масштабирования может давать хорошие результаты, рассмотрим следующий пример.

Пример 8.7. Допустим, что требуется найти минимум функции двух переменных

$$F(x) = x_1^{\beta} x_2^{\delta},$$

где b и c — константы. Матрица Гессе $F(x)$ выглядит так:

$$G(x) = \frac{F}{x_1^2 x_2^2} \begin{pmatrix} b(b-1) x_2^2 & bc x_1 x_2 \\ bc x_1 x_2 & c(c-1) x_1^2 \end{pmatrix}. \qquad (8.53)$$

При этом справедливо разложение

$$G(x) = \frac{F}{x_1^2 x_2^2} LL^T,$$

где

$$L = \begin{pmatrix} \beta x_2 & 0 \\ \Gamma x_2 & \delta x_1 \end{pmatrix}$$

с $\beta^2 = b(b-1)$, $\Gamma = bc/\beta$ и $\delta^2 = c(1-b-c)/(b-1)$. Об обусловленности $G(x)$ можно судить (см. разд. 8.3.3.2) по отношению максимального по модулю диагонального элемента L к минимальному, т. е. в предположении, что $|\beta x_2| \geqslant |\delta x_1|$,

$$\operatorname{cond}(G) = \operatorname{cond}(L)^2 \sim \left(\frac{\beta x_2}{\delta x_1}\right)^2.$$

Отсюда видно, что если β и δ — величина одного порядка, то для достижения хорошей обусловленности $G(x^*)$ достаточно применить диагональное масштабирование, обеспечивающее примерное равенство оптимальных значений переменных x_1 и x_2 (к тому же выводу придем и при $|\beta x_2| \leqslant |\delta x_1|$).

8.7.1.3. Балансирование производных. В разд. 7.5.1 была рассмотрена одна из форм плохой отмасштабированности задачи безусловной минимизации, состоящая в большом разбросе характерных значений модулей переменных. Другая (иногда связанная с первой) форма — «несбалансированность» производных минимизируемой функции. Способы ее преодоления, предлагаемые ниже,

рекомендуется применять после того, как переменные отмасштабированы методом, описанным в разд. 7.5.1.

Масштабирование по первой производной. Ниже речь пойдет о ситуации, когда какие-то компоненты градиента «малы» относительно погрешности ε_A вычисления F. Чтобы проиллюстрировать ее, возьмем функцию семи переменных.

Пример 8.8.

$$F(x) = (x_1 - 100)^2 + (x_2 - 1000)^2 + (x_3 + x_2 - 1000)^2 + (x_2 - 1000)^2 - (x_1 + x_2 - 200)^2 + 10^{-6}(x_3 + 2x_2 + 3x_3 + 4x_4 + 5x_5 + 6x_6 + 7x_7 - 1900)^2.$$

Ее точное значение при $x = (0, 0, 400, 100, 0, 0, 0)^T$ равно $F(x) = .2219973 \times 10^7$. Если вычислять F на IBM 370 с одинарной точностью ($\varepsilon_M = 16^{-5} \approx .9537 \times 10^{-6}$), то величина ε_A в окрестности x будет иметь порядок $\varepsilon_M |F|$.

Плохая отмасштабированность F по отношению к первой переменной становится очевидной при подсчете F в $\bar{x} = x + he_1$, где шаг h определяется по формуле (8.47). Будь F хорошо отмасштабированной, такое возмущение привело бы к изменению примерно половины значащих разрядов (см. разд. 8.6.1). На самом же деле меняется (на единицу) лишь последняя цифра мантиссы.

Плохая отмасштабированность рассматриваемого типа особенно опасна, когда применяется конечно-разностная аппроксимация производных и, в *частности, когда используются «стандартные» конечно-разностные интервалы*. Коль скоро численная оценка производной строится по интервалу, на котором F почти постоянна, доверять ей нельзя: полученное значение может в основном определяться ошибкой условий.

Пример 8.9. Рассмотрим конечно-разностную аппроксимацию для функции из примера 8.8, когда вычисления выполняются на IBM 370 ($\varepsilon_M = 16^{-5} \approx .9537 \times 10^{-6}$) с одинарной точностью. Если взять «стандартный» интервал

$$h_1 = \bar{h} = 2\sqrt{\varepsilon_A / (1 + |F(x)|)}, \tag{8.54}$$

то правая конечно-разностная формула даст оценку $g_1(x)$, равную $-.512000 \times 10^6$. В то же время истинным значением $g_1(x)$ является $-.199730 \times 10^6$. Таким образом, из-за плохой отмасштабированности F «стандартный» способ выбора h приводит к совершенно неудовлетворительному результату.

Отметим, что тяжесть последствий обсуждаемого дефекта функции зависит от погрешностей, с которыми проводятся вычисления. Если, скажем, считать на IBM 370 с двойной точностью (когда $\varepsilon_M = 16^{-13} \approx .222 \times 10^{-15}$), то для той же функции

F при том же h у $F(x)$ и $F(\overline{x})$ будут различаться в 9 из 14 значащих шестнадцатеричных цифр. Правда, если использовать интервал $\overline{h}$, который формула (8.54) дает при соответствующем ε_A, то у численной оценки производной $g_i(x)$ верными окажутся лишь 5 (а не 8, как можно было бы ожидать при хорошо отмасштабированной функции) десятичных разрядов, т. е. и здесь плохая отмасштабированность опустима.

Большая ошибка аппроксимации производной, наблюдаемая в примере 8.9, объясняется тем, что $g_i(x)$ имеет порядок $\sqrt{\varepsilon_A(1+|F(x)|)}$. Тейлоровское разложение F вдоль e_j в окрестности x выглядит так:

$$F(x+h_je_j)-F(x)=h_jg_j(x)+\frac{1}{2}h_j^2G_{jj}(x)+O(h_j^3). \qquad (8.55)$$

Коль скоро линейная составляющая правой части этого равенства доминирует, изменение F будет хорошо оцениваться величиной $|h_jg_j(x)|$. Последняя же при h_i и $g_i(x)$ из примера 8.9 близка к ε_A, т. е. «не выходит за уровень шума».

При определенных условиях рассматриваемую неотмасштабированность производных удается устранить диагональным преобразованием переменных

$$x=Dy. \qquad (8.56)$$

В результате такого преобразования из F получается функция $\mathcal{F}(y)$, разложение которой вдоль e_j в окрестности точки $y=D^{-1}x$ будет выглядеть так:

$$\mathcal{F}(y+\gamma_je_j)-\mathcal{F}(y)=\gamma_jc_j^T\nabla_y\mathcal{F}(y)+\frac{1}{2}\gamma_j^2c_j^T\nabla_y^2\mathcal{F}(y)\,e_j+O(\gamma_j^3)=$$
$$=\gamma_jd_jg_j(x)+\frac{1}{2}\gamma_j^2d_j^2G_{jj}(x)+O(\gamma_j^3). \qquad (8.57)$$

Когда нелинейными по γ_j слагаемыми можно пренебречь, отсюда следует, что модуль приращения $\mathcal{F}$ в результате сдвига y_j на γ_j примерно равен $|\gamma_jd_jg_j|$.

При выборе d_j исходят из желания добиться, чтобы при шаге, равном значению «оптимального» конечно-разностного интервала h (8.54), изменялась «половина» значащих разрядов F. Это приводит к уравнению

$$2|d_jg_j|=1+|F(x)|$$

с решением

$$d_j=\frac{1+|F(x)|}{2|g_j(x)|}. \qquad (8.58)$$

Масштабирование на основе вторых производных. Рассмотрим случай, когда в правой части (8.57) можно пренебречь линейным по d_j слагаемым. В частности, при нулевой производной $g_j(x)$ изменение $\mathcal{F}$ при шаге γ_je_j будет равно

$$\mathcal{F}(y+\gamma_je_j)-\mathcal{F}(y)=\frac{1}{2}\gamma_j^2d_j^2G_{jj}(x)+O(\gamma_j^3). \qquad (8.59)$$

Если в качестве γ_j берется $\bar{h}$ из (8.54), то для хорошо отмасштабированной функции такое изменение должно иметь порядок ε_A. Исходя из данной установки (после отбрасывания в (8.59) остаточного члена) можно выписать для d_j уравнение, решением которого будет

$$d_j = \sqrt{\frac{\bar{h} \, |F(x)|}{2 \, |G_{jj}(x)|}}. \tag{8.60}$$

Этот масштабирующий множитель хорош и в ситуации, когда производная $g_j(x)$ отлична от нуля, но настолько мала по модулю, что при шаге h второе слагаемое справа в (8.57) доминирует.

Итак, решив отмасштабировать переменные по значениям производных, надо еще понять, какой из формул (8.58) или (8.60) воспользоваться. Неверный выбор может существенно осложнить последующую минимизацию. Если, например, взять формулу (8.58), хотя в действительности квадратичным членом гейлоровского разложения пренебрегать не следовало бы, то получаемое значение d_j может оказаться намного больше того, которое требуется на самом деле. Тогда функция $\tilde{F}$ будет «слишком нелинейной», и это затруднит работу любого алгоритма, ориентированного на линеаризацию.

Масштабирование, основанное на оценке оптимального конечно-разностного интервала. Проблема выбора между (8.58)—(8.60) отпадает, если при масштабировании опираться не на значения производных, а на численную оценку конечно-разностного интервала, минимизирующего сумму ошибок условий и отбрасывания при аппроксимации $g_j(x)$ (см. разд. 8.6).

Пусть $\hat{h}_j$ — вычисленная оценка «оптимального» интервала для аппроксимации первой производной целевой функции по j-й исходной переменной (см. разд. 4.6.1.3 и 8.6.2). Возмущение x_j на $\hat{h}_j$ эквивалентно возмущению y_j из (8.56) на $\gamma_j = \hat{h}_j/d_j$, причем для хорошо отмасштабированной функции оптимальный интервал равен $\bar{h}$ из (8.54). Таким образом, логично взять

$$d_j = \frac{\hat{h}_j}{2} \sqrt{\frac{\bar{h} \, |F(x)|}{\varepsilon_A}}. \tag{8.61}$$

Пример нормировки производных. Рассмотрим вычисление конечно-разностной оценки первой компоненты градиента функции из примера 8.5 после того, как первая переменная отмасштабирована в соответствии с (8.58) и (8.61). Формула (8.58) дает $d_1 = .56 \times 10^4$. Значение $\tilde{F}(y + \bar{h}e_1)$ оказывается равным $.221792 \times 10^7$; при шаге $\bar{h}$ меняются три из шести шестнадцатеричных цифр мантиссы, т. е. масштабирование приводит к желаемому результату. Однако оценка первой компоненты градиента, вычисленная по правой конечно-разностной формуле с интервалом $\bar{h}$, все же получается не слишком точной. Она равна $-.105062 \times 10^7$,

в то время как истинным значением является $-.110999\times10^{2}$. Ощутимое различие объясняется большой ошибкой отбрасывания — вторая производная при масштабировании увеличилась в 3.1×10^{5} раз.

Величина d_1, вычисленная в рассматриваемом случае по формуле (8.61), равна $.106\times10^{4}$. Соответствующим значением $\mathcal{F}(y+he_1)$ будет $.221956\times10^{2}$. Шаг h и при таком d_1 приводит к изменению трех из шести шестнадцатеричных цифр мантиссы. Правая конечно-разностная формула с интервалом h теперь дает оценку производной, равную $-.209920\times10^{4}$. Точным значением является $.211682\times10^{4}$. Как и можно было ожидать, относительная погрешность конечно-разностной аппроксимации при использовании (8.61) оказалась меньше, чем при использовании (8.58).

8.7.2. МАСШТАБИРОВАНИЕ ЗНАЧЕНИЙ ЦЕЛЕВОЙ ФУНКЦИИ

Иногда считают, что масштаб значений целевой функции не играет никакой роли. Теоретически это действительно так, поскольку ни умножение $F(x)$ на *положительную* константу, ни дополнение $F(x)$ постоянным слагаемым на решении не сказываются. (Заметим, что при умножении F на число пропорционально изменяется и все производные; сложение F с числом производных не меняет.) Однако отсюда еще не следует, что такие операции не отразятся на работе алгоритма поиска минимума.

Желательно, чтобы в окрестности искомой точки модули значений целевой функции были величинами порядка единицы или меньше. Этого всегда можно добиться, почленив F на подходящую константу. Например, если в исходном определении функция $F(x)$ имеет порядок 10^{6}, то в оптимизационной программе имеет смысл умножать ее и ее производные на 10^{-6}.

Плохо, когда целевая функция *всюду близка к нулю*. Как отмечалось в нескольких из предыдущих разделов, в правила останова реальных алгоритмов всегда включают не только тесты, оперирующие относительными величинами, но также альтернативный тест на близость к нулю некой абсолютной величины. Поэтому, хотя в теории умножение F, скажем, на 10^{-20} решения задачи не изменит, на практике оно приведет к тому, что алгоритм будет объявлять любую начальную точку оптимальной (из-за того что норма градиента везде исчезающе мала).

Когда F содержит большую постоянную составляющую, переменная часть F отражается в вычисляемых значениях F с пониженной точностью. Поэтому рекомендуется избавляться от постоянных слагаемых, и лучше, например, определить $F(x)$ как $x_1^2+x_2^2$, чем как $x_1^2+x_2^2+1000$ (даже из $x_1^2+x_2^2+1$ единицу стоит убрать).

8.7.3. МАСШТАБИРОВАНИЕ ОГРАНИЧЕНИЙ

Набор функций ограничений можно считать «хорошо отмасштабированным», если выполнены два требования. Во-первых, каждая из этих функций должна быть хорошо обусловлена по отношению к вариациям разных переменных. Во-вторых, функции должны быть *взаимно сбалансированы*, т. е. все ограничения в процессе поиска должны быть «одинаково весомыми».

8.7.3.1. Некоторые следствия масштабирования ограничений. Во время оптимизационных вычислений и интерпретации результатов отмасштабированность ограничений проявляется несколькими способами. Вспомним, например, что точности аналитических и вычисляемых оценок множителей Лагранжа сильно зависят от числа обусловленности матрицы Якоби для ограничений из рабочего списка (см. разд. 5.1.5). От этой обусловленности зависит и предельная точность численного решения в ранг-пространстве активного набора (см. разд. 3.2.2.2).

В методах активного набора нормировка ограничений проявляется также в моменты сокращений рабочего списка вблизи условно стационарных точек x_k. Вектор множителей Лагранжа λ_k и x_k определяются как решение совместной системы уравнений

$$\hat{A}_k^T \lambda_k = g_k.$$

Умножение ограничений на различные веса изменит строки $\hat{A}_k$ и соответственно значения λ_k. В частности, умножение j-го ограничения на ω_j приводит к делению на ω_j компоненты λ_k с j-м номером. При этом в большинстве алгоритмов из рабочего списка выводится то ограничение, которому отвечает максимальная по модулю оценка множителя. Значит, «взвешивание» ограничений может приводить к изменению последовательности точек, генерируемых алгоритмом.

Пример 8.10. Рассмотрим задачу с двумя переменными. Предположим, что скалярное произведение $\hat{a}_1^T x$ представляет собой концентрацию растворенного в воде кислорода (характерное значение этой величины — одна миллионная), а скалярное произведение $\hat{a}_2^T x$ — скорость течения (характерное значение — пять миль в час). Допустим, что они должны удовлетворять двусторонним неравенствам

$$0 \leqslant \hat{a}_1^T x \leqslant 8 \times 10^{-6},$$
$$3 \leqslant \hat{a}_2^T x \leqslant 10$$

и что 2×2-матрица $\hat{A}$ со строками $\hat{a}_1^T$ и $\hat{a}_2^T$ выглядит так:

$$\hat{A} = \begin{pmatrix} 10^{-6} & -10^{-6} \\ 1 & 1 \end{pmatrix}. \qquad (8.62)$$

Если при решении задачи будет получена точка, где оба левых неравенства обращаются в равенства и оба множителя Лагранжа отрицательны, то почти наверняка модуль первого множителя окажется намного больше модуля второго и на рабочего списка будет внедрено первое ограничение.

В приведенном примере верхнее неравенство следует умножать на 10^6. После такой нормировки матрица A преобразуется в

$$\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} \tag{8.63}$$

и становится очень хорошо обусловленной (столбцы (8.63) взаимно ортогональны).

В задаче, о которой шла речь, близость к нулю коэффициентов первой строки (8.62) была связана с существом постановки и нормировка означала переход к более естественным единицам измерения. Можно представить себе и иную ситуацию: допустим, что имеется та же самая матрица (8.62), но ее первая строка отвечает дополнительному ограничению, которое определялось как сумма нескольких других ограничений с существенно отличными от нуля коэффициентами. Тогда $\tilde{a}_1^j$ в основном будет отражать ошибки компенсации при агрегировании, и здесь умножение на большое число с целью придать искусственному ограничению «тот же вес», что и у «настоящих», смысла не имеет.

8.7.3.2. Методы масштабирования линейных ограничений. Для пользователя, который хочет отмасштабировать линейные ограничения, открыто несколько возможностей. В частности, можно отмасштабировать переменные каким-нибудь из методов разд. 8.7.1, а затем отбалансировать матрицу условий умножением строк на соответствующие веса (как в примере из разд. 8.7.3.1). Однако такая процедура подразумевает, что масштабы переменных, подходящие для целевой функции, подойдут и для ограничений. Поэтому, если решение в основном определяется ограничениями, применять ее следует с осторожностью.

Второй подход состоит в том, чтобы масштабировать *строки и столбцы матрицы условий*, не обращая внимания на целевую функцию. При решении задач линейного и квадратичного программирования чаще всего поступают именно так. В результате из исходной матрицы A получается D_1AD_2, где D_1 и D_2 — диагональные матрицы с положительными элементами на диагоналях. Ниже описан метод построения D_1 и D_2, широко применяемый для задач линейного программирования большой размерности (другие методы можно найти по ссылкам, данным в замечаниях). В основе лежит идея многократной нормировки каждой строки (столбца) по геометрическому среднему ее (его) максимального и минимального по модулю элементов. Формальное описание метода таково:

Алгоритм SC (*масштабирование линейных ограничений*)

SC1. [Подсчет максимального отношения модулей для двух элементов одного столбца.] Вычислить

$$\rho_0 = \max_i \max_{s,t} \{ |a_{ri}/a_{si}| \},$$

просматривая только те пары s, t, для которых $a_{si} \neq 0$.

SC2. [Нормировка строк.] При каждом i поделить i-ю строку матрицы A на $((\min_j |a_{ij}|)(\max_j |a_{ij}|))^{1/2}$, где минимум берется только по тем j, для которых $a_{ij} \neq 0$.

SC3. [Нормировка столбцов.] При каждом j поделить j-й столбец матрицы A на $((\min_i |a_{ij}|)(\max_i |a_{ij}|))^{1/2}$, где минимум берется только по тем i, для которых $a_{ij} \neq 0$.

SC4. [Подсчет максимального отношения модулей для двух элементов одного столбца.] Вычислить

$$\rho = \max_i \max_{s,t} \{ |a_{ri}/a_{si}| \},$$

просматривая только те пары s, t, для которых $a_{si} \neq 0$.

SC5. [Проверка соблюдения условий останова.] Если $|\rho - \rho_0| \geqslant \geqslant 10^{-1} |\rho_0|$, вернуться к шагу SC1. В противном случае — остановиться.

На шаге SC5 можно использовать и другие условия останова. Например, можно прерывать счет после выполнения определенного числа (скажем, пяти) итераций.

Если алгоритм SC применить к (8.62), то хорошо обусловленная матрица (8.63) будет получена на первой итерации.

8.7.3.3. Методы масштабирования нелинейных ограничений. Для оценки точности, с которой численное решение задачи с линейными ограничениями должно (и будет) удовлетворять тем из них, что окажутся активными, достаточно результатов из вычислительной линейной алгебры. При правильной нормировке соответствующие невязки обычно оказываются величинами порядка ε_M. Что же касается нелинейных ограничений, то для них приемлемые значения невязок могут быть обоснованно оценены лишь при условии, что известны достижимые точности (ε_A) подсчета функций $c_i(x)$. Не располагая такой информацией, стандартный алгоритм будет стремиться обеспечить такие невязки ограничений, которые подходили бы под «естественное» определение «малых». Однако даже малые невязки, как правило, окажутся существенно различными (например, $c_1(x^*) = 10^{-10}$, $c_2(x^*) = -10^{-6}$ и т. д.).

Считая все ограничения одинаково важными, желательно обеспечить, чтобы отклики их функций на единичные возмущения x были соразмеримыми. Весовые множители, позволяющие добиться этого, часто бывают подсказаны существом задачи. Скажем, в за-

даче из разд. 1.1 веса ограничений по объему и длине носового конуса надо подобрать так, чтобы правые части стали единичными.

Для нелинейных функций ограничений в принципе можно использовать методы масштабирования из разд. 8.7.1. К сожалению, это не просто, поскольку для разных функций потребуются разные преобразования переменных. Коль скоро задачу предполагается решать методом с использованием функции Лагранжа (см. разд. 6.4, 6.5 и 6.6), достаточно было бы отмасштабировать переменные относительно нее. Однако обычно сделать это не удается, поскольку нет хороших априорных оценок компонент λ^*.

Многие методы оптимизации при нелинейных ограничениях включают вычисления с матрицей Якоби для функций ограничений из рабочего списка. Ради повышения устойчивости таких методов при их реализации можно предусматривать предварительную «балансировку» матрицы Якоби по каждой итерации с помощью какой-нибудь процедуры типа алгоритма SC; впрочем, это работа разработчика, а не пользователя.

Еще один способ нормировки матрицы Якоби — делить каждый входящий в нее градиент на величину его нормы. Если последняя очень близка к нулю, здесь могут возникать осложнения, и тогда встает извечный вопрос о том, означает ли малость нормы градиента, что их можно пренебречь, или она есть следствие плохого масштабирования. Универсального способа разрешения этой дилеммы не существует. Практика показывает, что лучше придерживаться консервативной точки зрения и считать очень малые числа пренебрежимыми. Если такая стратегия подведет, неучтенные градиенты надо будет перенормировать.

Замечания и избранная библиография к разделу 8.7

Пример 8.1 предложен Муртафом и Сондерсом (1977). Схема масштабирования по геометрическим средним успешно применяется в линейном программировании много лет; см., например, Бенишу и др. (1977). В той форме, в которой она изложена здесь, ее применял Фоурер (1979). О других схемах масштабирования линейных ограничений можно прочесть у Хэмминга (1971), Фалкерсона и Вулфа (1962), Кёртиса и Райда (1972). Результаты численных экспериментов с несколькими схемами представлены Томлином (1975b).

ВОПРОСЫ И ОТВЕТЫ

Во время великих событий общие принципы не помогают.
Г. Гегель (1832)

Занимаясь разработкой и сопровождением оптимизационного математического обеспечения на протяжении многих лет, мы обратили внимание на то, что среди вопросов, которые возникают у пользователей, есть постоянно повторяющиеся. Ниже приведены 20 наиболее типичных вопросов и возможные ответы на них.

В1. Могу ли я рассчитывать на то, что машина нашла точку глобального минимума?

О1. Если ваша задача не обладает специальными свойствами (см. разд. 6.8), никаких гарантий относительно глобальности найденного минимума дать нельзя. К сожалению, существующие общие методы глобальной минимизации гарантируют успех только при сильных предположениях о задаче и требуют так много вычислений, что оказываются практичными лишь для случаев с очень малым числом переменных.

В2. Значения моей целевой функции определяются решениями итеративных подзадач, причем я могу контролировать точность этих решений. Оптимизирующая программа, которую я хочу использовать, предполагает, что целевая функция всегда будет вычисляться с полной машинной точностью. Можно ли избежать необходимости столь аккуратного решения подзадач?

О2. В большинстве стандартных оптимизационных программ машинная точность фигурирует как значение некоторого параметра. Это значение вы можете зафиксировать сами и, в частности, можете записать его на начальной стадии расчетов, ограничиваясь при этом грубыми решениями подзадач. После того как в результате определится соответствующее грубое приближение оптимума в исходной задаче, надо будет вернуться к правильному значению параметра и (аккуратно решая подзадачи) выполнить завершающий цикл итераций, чтобы уточнить это приближение.

В3. Я решил свою задачу безусловной минимизации квазиньютоновским алгоритмом без вычисления производных и теперь хочу увеличить число переменных. К каким изменениям числа итераций

и количества обращений к процедуре подсчета целевой функции это приведет?

О3. Поскольку квазиньютоновские методы не являются конечными, количество итераций, нужных для отыскания минимума функции общего вида с требуемой точностью, заранее оценить нельзя. Оно зависит от того, как удачно выбрано начальное приближение, а также от степени нелинейности и от обусловленности функции. Каждая итерация квазиньютоновского метода без вычисления производных требует по меньшей мере столько обращений к модулю подсчета функции, сколько компонент у ее градиента. Таким образом, трудоемкость одной итерации пропорциональна размерности. В некоторых случаях увеличение числа переменных очень слабо отражается на числе итераций. Так будет, например, когда переменными являются коэффициенты полиномиальной аппроксимации функции, а их число есть степень аппроксимирующего полинома.

В4. Я собираюсь решать задачу с нелинейными ограничениями-неравенствами, причем про некоторые из них я знаю, что в решении они будут активными. Следует ли определить эти ограничения как равенства, или лучше задать их алгоритму как неравенства?

О4. Это зависит от программы, которую вы собираетесь использовать. Коль скоро есть доступ к хорошей программе, которая генерирует только допустимые точки, и вы можете указать допустимое начальное приближение, то лучше к ней и обратиться, определив все ограничения как неравенства. Если же предполагается использовать метод, порождающий недопустимые приближения, то ограничения, о которых идет речь, имеет смысл специфицировать как равенства, упрощая тем самым проблему идентификации правильного активного набора.

В5. Я запрограммировал вычисление аналитических значений производных целевой функции, а выражения для вторых производных слишком сложны. Какой метод мне взять: дискретный ньютоновский или квазиньютоновский?

О5. Ответ на этот вопрос зависит от того, какое качество численного решения вам нужно и какова размерность задачи. Конечно-разностная аппроксимация матрицы Гессе, которой вы будете располагать, если возьмете дискретный ньютоновский метод, позволит оценить обусловленность и чувствительность оптимума (см. разд. 8.3.3.1); по квазиньютоновской аппроксимации этого сделать, вообще говоря, нельзя, поскольку она может сильно отличаться от истинной матрицы Гессе. К тому же существуют функции (имеющие седловые точки), для которых дискретный ньютоновский метод намного надежнее. Однако в общем случае он предполагает по меньшей мере n вычислений градиента на каждой итерации. Поэтому, если у матрицы Гессе нет специальной структуры заполненности (см. разд. 4.8.1), позволяющей строить ее конечно-разностную ап-

проксимацию экономнее, то при значительной размерности вектора переменных дискретный ньютоновский метод скорее всего потребует больше машинного времени.

В6. Я воспользовался библиотечной процедурой проверки правильности вычисления градиентов с применением конечных разностей. Она выдала сообщение о том, что одна из производных запрограммирована неверно, а я выяснил, что ошибки нет. Значит ли это, что ошибка есть в самой процедуре тестирования?

О6. Не обязательно. Ваша функция может быть столь плохо отмасштабированной, что ее вариации, вычисляемые тестирующей процедурой, оказываются на уровне погрешностей счета. Тогда желательно отмасштабировать ее каким-нибудь из способов разд. 8.7. Правда, если есть полная уверенность, что трудности носят сугубо локальный характер, можно и просто проигнорировать указание на ошибку.

В7. Мне надо оценить параметры некой модели, и я собираюсь воспользоваться программой метода наименьших квадратов. Исходные данные, по которым я «настраиваю» модель, надежны лишь в двух старших разрядах. Разумно ли потребовать такую же точность численного решения?

О7. Не всегда! Относительные ошибки вычисления целевой функции вашей задачи о наименьших квадратах скорее всего будут существенно меньше чем 10^{-2}, т. е. предельная точность численного решения будет намного выше. Когда запрашивается значительно более низкая точность, во-первых, есть риск получить совсем неверный ответ, а во-вторых, не исключено, что программа сама изменит значение соответствующего параметра (см. разд. 8.2.3.6).

В8. Получив от программы безусловной минимизации численное решение своей задачи, я обнаружил, что оптимальные значения некоторых переменных совершенно бессмысленны. Что мне делать?

О8. Раз вы можете говорить о бессмысленных значениях, значит, вы представляете себе диапазоны значений, имеющих смысл. В таком случае лучше установить подходящие границы и обратиться к программе минимизации при простых ограничениях на переменные (см. разд. 5.5.1).

В9. Для своей многомерной задачи на безусловный минимум я хочу применить метод с сопоставлением значений функции. Как я смогу убедиться, что он дал правильный ответ?

О9. Серьезный недостаток любого метода с сопоставлением значений функции заключается в том, что в общем случае невозможно гарантировать правильность найденного им решения. (Поскольку никаких производных не вычисляется, проверить, выполнены ли необходимые условия первого порядка, нельзя.) Ради собственного успокоения можно повторно применить алгоритм, задав другое на-

чальное приближение: если он даст прежнюю точку, доверие к ней естественно возрастет.

В10. Программа безусловной минимизации сообщила об успешном завершении счета, хотя в точке, которую она нашла, составляющие градиента довольно велики. Не означает ли это, что в программе есть ошибка?

О10. Если задача плохо отмасштабирована, градиент может быть большим даже для вполне подходящего приближенного решения. В критериях останова норма градиента обычно соотносится с модулем целевой функции (см. разд. 8.2.3).

В11. Значения моей целевой функции и ее градиента вычисляются очень громоздкой и сложной подпрограммой. Я пытался отыскать минимум с помощью квазиньютоновского алгоритма, но сходимость была крайне медленной и ощутимого уменьшения целевой функции получить не удалось. Чем это можно объяснить?

О11. Одна из наиболее вероятных причин — ошибка в модуле вычисления производных. Вам следует оттестировать этот модуль, используя технику, описанную в разд. 8.1.4.2. Вторая возможная причина — разрывы градиента. Эти разрывы часто удается выявить сравнением правых и левых конечно-разностных приближений его компонент (см. разд. 8.4.2.4).

В12. В моей задаче есть набор нелинейных ограничений-равенств, и я могу применить быстрый специальный алгоритм, который позволит выдерживать их, вычисляя значение одних («зависимых») переменных по значениям других («независимых») переменных. Стоит ли делать это?

О12. Если равенства, о которых идет речь, существенно нелинейны, то скорее всего не стоит. Во-первых, не ясно, как быть, когда для исключаемых (зависимых) переменных есть простые ограничения. Во-вторых, получится метод, который будет «отслеживать» криволинейную поверхность, что нередко оборачивается потерей скорости сходимости (см. разд. 6.3). Наконец, по своему опыту можем сказать, что машинное время, которое тратится на подсчет значений зависимых переменных в каждой пробной точке, обычно не компенсируется экономией, связанной с понижением размерности.

В13. Целевая функция моей задачи гладкая, но очень нелинейная. Я слышал, что ньютоновские и квазиньютоновские алгоритмы для таких функций работают плохо и что в подобных случаях имеет смысл использовать алгоритмы прямого поиска. Верно ли это?

О13. Абсолютно не верно! Высказанная вами точка зрения родилась в те времена, когда не было хороших реализаций ньютоновских и квазиньютоновских методов. Сейчас их сколько угодно, и применять для сложной гладкой задачи метод с сопоставлением значений функций — значит сделать худший выбор.

В14. По выдаче программы ньютоновского метода я вижу, что достигнута очень высокая скорость сходимости, и в то же время счет был прерван с признаком отказа, причем из-за того, что на очередной итерации процедура одномерного поиска не смогла обеспечить существенного убывания целевой функции. Как это понимать?

О14. Иногда ньютоновский алгоритм находит очень хорошее приближение, а условия нормального останова не соблюдаются (см. разд. 8.2.3). Если при этом полученная точка оказывается численно неулучшаемой, происходит аварийный останов в блоке выбора шага.

В15. Просматривая распечатку решения задачи с линейными ограничениями, я заметил, что несколько множителей Лагранжа получились почти нулевыми. Значит ли это, что соответствующие ограничения избыточны и что их можно снять?

О15. Не обязательно! (См. рис. 8с и 8f в гл. 8.)

В16. Я могу воспользоваться оптимизационной программой, в которой вычисления проводятся с двойной точностью, а в составленных мною подпрограммах подсчета значений функций задачи затребована одинарная точность. Возможны ли в связи с этим какие-нибудь неожиданности?

О16. Это зависит от того, как в оптимизационной программе будет задана точность подсчета функций. Если она представляется параметром, значения которого может определяться пользователем, то, зафиксировав это значение в соответствии с погрешностями вычислений в ваших подпрограммах, вы тем самым согласуете их с оптимизационной программой. Если же такого параметра нет, то возможно, что оптимизационная программа предполагает вычисление функций с двойной точностью. Тогда существует риск получить неверное численное решение.

В17. Я хочу подобрать параметры своей модели, минимизируя уклонение предсказываемых ею величин от данных наблюдений. Какую норму уклонений лучше использовать — квадратичную, норму l_1 или норму l_∞?

О17. Если никаких содержательных доводов в пользу той или иной нормы нет, мы рекомендуем квадратичную. При использовании двух других норм возникают негладкие задачи безусловной минимизации. Хотя для них разработаны специальные методы, минимизировать квадратичную норму все же намного проще (см. разд. 4.2.3 и 6.8.1).

В18. Мне кажется, что в решении моей задачи ее ограничения будут выполнены как неравенства. Следует ли мне воспользоваться программой безусловной минимизации или все же надо обратиться к программе минимизации при ограничениях?

О18. Ответ на этот вопрос зависит от нескольких факторов. Если запрограммировать функции ограничений (и, возможно, их

производные) не просто или качество доступных средств поиска безусловного минимума намного ниже, чем средств минимизации при ограничениях, то сначала стоит попробовать решить задачу без ограничений. Однако если целевая функция может принимать сколь угодно большие по модулю отрицательные значения, то снимать ограничения не следует (отметим, что присутствие простых ограничений обычно облегчает решение независимо от того, активны они в точке оптимума или нет).

В19. Моя целевая функция является гладкой в решении, но может иметь изломы и разрывы в других точках. Должен ли я использовать алгоритм негладкой минимизации?

О19. Как правило, сначала стоит попробовать найти решение алгоритмом, предназначенным для гладких функций. Надо, однако, помнить, что не все такие алгоритмы одинаково чувствительны к негладкости. Обычно лучше других в рассматриваемой ситуации работают ньютоновские методы, так как они моментально реагируют на скачкообразные изменения в матрице Гессе. Программы с конечно-разностной аппроксимацией производных здесь довольно ненадежны, поскольку при попадании точек разрывов на интервалы аппроксимации будут получаться бессмысленные оценки.

В20. Каковы общие характеристики задачи, существенные для выбора алгоритма решения?

О20. Список основных факторов, которые должны приниматься во внимание при выборе алгоритма, выглядит так:

(a) число переменных;

(b) наличие простых ограничений на переменные (см. разд. 5.5.1);

(c) гладкость функций задачи и их производных (см. разд. 4.2.1 и 6.8);

(d) порядок производных, подлежащих эффективному вычислению (см. разд. 8.1.1);

(e) доли ненулей в матрице Гессе целевой функции и в матрице Якоби функций ограничений (это надо учитывать при больших размерностях; см. разд. 4.8, 5.6 и 6.7);

(f) отношение числа линейных ограничений общего вида к числу переменных и ожидаемое количество активных в решении ограничений (см. разд.5.4 и 5.5.2);

(g) определенность целевой функции за пределами допустимой области и осмысленность соответствующих значений (см. разд. 6.2.1.1 и 8.1.1.3).

БИБЛИОГРАФИЯ

Абади

Abadie J. (1978). «The GRG method for nonlinear programming», in Design and Implementation of Optimization Software (H. J. Greenberg, ed.), pp. 335—362, Sijthoff and Noordhoff, Netherlands.

Абади и Гиго

Abadie J. and Guigou J. (1970). «Numerical experiments with the GRG method», in Integer and Nonlinear Programming (J. Abadie, ed.), pp. 529—536, North-Holland, Amsterdam.

Абади и Карпентье

Abadie J. and Carpentier J. (1965). Généralisation de la méthode du gradient réduit de Wolfe au cas de contraintes non-linéaires, Note HR6678, Électricité de France, Paris.

— (1969). «Generalization of the Wolfe reduced-gradient method to the case of nonlinear constraints», in Optimization (R. Fletcher, ed.), pp. 37—49, Academic Press, London and New York.

Авила и Конкус

Avila J. H. and Concus P. (1979). Update methods for highly structured systems of nonlinear equations, SIAM J. Numer. Anal. 16, pp. 260—269.

Авриель

Avriel M. (1976). Nonlinear Programming: Analysis and Methods, Prientice-Hall, Inc., Englewood Cliffs, New Jersey.

Авриель и Дембо

Avriel M. and Dembo R. S. (eds.) (1979). Engineering Optimization, Math. Prog. Study 11.

Авриель, Рейкарт и Уайлд

Avriel M., Rijckaert M. J. and Wilde D. J. (eds.) (1973). Optimization and Design, Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Аксельсон

Axelsson O. (1974). On preconditioning and convergence acceleration in sparse matrix problems, Report 74-10, CERN European Organization for Nuclear Research, Geneva.

Андерсен и Блюмфилд

Anderssen R. S. and Bloomfield P. (1974). Numerical differentiation procedures for non-exact data, Num. Math. 22, pp. 157—182.

Андерсон и Бьёрк

Anderson N. and Björk Å. (1973). A new high-order method of the regula falsi type for computing a root of an equation, Nordisk Tidskr. Informationsbehandling (BIT) 13, pp. 253—264.

Андерсон и Осборн

Anderson D. H. and Osborne M. R. (1977). Discrete, nonlinear approximations in polyhedral norms; a Levenberg-like algorithm, Num. Math. 28, pp. 157—170.

Апостол
Apostol T. M. (1957). Mathematical Analysis, Addison-Wesley, Massachusetts and London.

Армстронг и Годфри
Armstrong R. D. and Godfrey J. P. (1979). Two linear programming algorithms for the discrete l_1 norm problem, Mathematics of Computation 33, pp. 289—300.

Байс
Buys J. D. (1972). Dual Algorithms for Constrained Optimization Problems, Ph. D. Thesis, University of Leiden, Netherlands.

Байс и Гонин
Buys J. D. and Gonin R. (1977). The use of augmented Lagrangian functions for sensitivity analysis in nonlinear programming, Math. Prog. 12, pp. 281—284.

Баклн
Buckley A. G. (1975). An alternative implementation of Goldfarb's minimization algorithm, Math. Prog. 8, pp. 207—231.
— (1978). A combined conjugate-gradient quasi-Newton minimization algorithm, Math. Prog. 15, pp. 200—210.

Балинский и Лемарешаль
Balinski M. L. and Lemaréchal C. (eds.) (1978). Mathematical Programming in Use, Math. Prog. Study, 9.

Банч и Кауфман
Bunch J. R. and Kaufman L. C. (1977). Some stable methods for calculating inertia and solving symmetric linear equations. Mathematics of Computation 31, pp. 163—179.
— — (1980). A computational method for the indefinite quadratic programming problem, Linear Algebra and its Applics. 34, pp. 341—370.

Банч и Парлет
Bunch J. R. and Parlett B. N. (1971). Direct methods for solving symmetric indefinite systems of linear equations, SIAM, J. Numer. Anal. 8, pp. 639—655.

Бард
Bard Y. (1976). Nonlinear Parameter Estimation, Academic Press, London and New York.

Бард и Гринштадт
Bard Y. and Greenstadt J. L. (1969). «A modified Newton method for optimization with equality constraints», in Optimization (R. Fletcher, ed.), pp. 299—306, Academic Press, London and New York.

Бартелс
Bartels R. H. (1971). A stabilization of the simplex method, Num. Math. 16, pp. 414—434.
— (1980). A penalty linear programming method using reduced-gradient basis-exchange techniques, Linear Algebra and its Applics. 29, pp. 17—32.

Бартелс и Голуб
Bartels R. H. and Golub G. H. (1969). The simplex method of linear programming using the LU decomposition, Comm. ACM 12, pp. 266—268

Бартелс, Голуб и Сондерс
Bartels R. H., Golub G. H. and Saunders M. A. (1970). «Numerical techniques in mathematical programming», in Nonlinear Programming (J. B. Rosen, O. L. Mangasarian and K. Ritter, eds.), pp. 123—176, Academic Press, London and New York.

Бартелс и Конн
Bartels R. H. and Conn A. R. (1980). Linearly constrained discrete problems, ACM Trans. Math. Software 6, pp. 594—608.

Бас и Деккер
Bus J. C. P. and Dekker T. J. (1975). Two efficient algorithms with guaranteed convergence for finding a zero of a function, ACM Trans. Math. Software 1, pp. 330—345.

Бейкер и Вентнер
Baker T. E. and Ventker R. (1960). «Successive linear programming in refinery logistic models», presented at ORSA/TIMS Joint National Meeting, Colorado Springs, Colorado.

Бен-Израиль
Ben-Israel A. (1967). On iterative methods for solving nonlinear least-squares problems over convex sets, Israel J. of Maths. 5, pp. 211—224.

Бенишу, Готье, Анже и Рибьер
Benichou M., Gauthier J. M., Hentges G. and Ribière G. (1977). The efficient solution of large-scale linear programming problems—some algorithmic techniques and computational results, Math. Prog. 13, pp. 280—322.

Берд
Byrd R. H. (1976). Local convergence of the diagonalized method of multipliers, Ph. D. Thesis, Rice University, Texas.
— (1978). Local convergence of the diagonalized method of multipliers, J. Opt. Th. Applics. 26, pp. 485—500.

Бертсекас
Bertsekas D. P. (1975a). Necessary and sufficient conditions for a penalty function to be exact, Math. Prog. 9, pp. 87—99.
— (1975b). Combined primal-dual and penalty methods for constrained minimization, SIAM J. Control and Optimization 13, pp. 521—544.
— (1976a). Multiplier methods: a survey, Automatica 12, pp. 133—145.
— (1976b). On penalty and multiplier methods for constrained minimization, SIAM J. Control and Optimization 14, pp. 216—235.
— (1979). «Convergence analysis of augmented Lagrangian methods», presented at the IIASA Task Force Meeting on «Generalized Lagrangians in Systems and Economic Theory», IIASA, Laxenburg, Austria (proceedings to be published in 1981).

Бест, Брауэингер, Риттер и Робинсон
Best M. J., Bräuninger J., Ritter K. and Robinson S. M. (1981). A globaly and quadratically convergent algorithm for general nonlinear programming problems, Computing 26, pp. 141—163.

Бетс
Betts J. T. (1970). Solving the nonlinear least-square problem, J. Opt. Th. Applics 18, pp. 469—483.

Биггс
Biggs M. C. (1972). «Constrained minimization using recursive equality quadratic programming», in Numerical Methods for Non-Linear Optimization (F. A. Lootsma), pp. 411—428, Academic Press, London and New York.
— (1974). The Development of a Class of Constrained Optimization Algorithms and Their Application to the Problem of Electric Power Scheduling, Ph. D. Thesis, University of London.
— (1975). «Constrained minimization using recursive quadratic programming: some alternative subproblem formulations», in Towards Global Optimization (L. C. W. Dixon and G. P. Szegö, eds.), pp. 341—349, North-Holland, Amsterdam.

Бил
Beale E. M. L. (1959). On quadratic programming, Naval Res. Logistics Quarterly 6, pp. 227—243.
— (1967a). «An introduction to Beale's method of quadratic programming», in Non-

linear Programming (J. Abadie, ed.), pp. 143—153, North-Holland, Amsterdam.
— (1967b). «Numerical methods», in Nonlinear Programming (J. Abadie, ed.), pp. 132—205, North-Holland, Amsterdam.
— (1972). «A derivation of conjugate gradients», in Numerical Methods for Nonlinear Optimization (F. A. Lootsma, ed.), pp. 39—43, Academic Press London and New York.
— (1974). «A conjugate-gradient method of approximation programming», in Optimization Methods for Resource Allocation (R. W. Cottle and J. Krarup, eds.), pp. 251—277, English Universities Press.
— (1975). The current algorithmic scope of mathematical programming systems, Math. Prog. Study 4, pp. 1—11.
— (1977). «Integer Programming», in The State of the Art in Numerical Analysis (D. Jacobs, ed.), pp. 409—448, Academic Press, London and New York.
— (1978). «Nonlinear programming: is there a general mathematical programming system», in Design and Implementation of Optimization Software (H. J. Greenberg, ed.), pp. 259—279, Sijthoff and Noordhoff, Netherlands.

Блэнд
Bland R. G. (1977). New finite pivoting rules for the simplex method, Math. of Oper. Res. 2, pp. 103—107.

Боггс
Boggs P. T. (1975). The solution of nonlinear operator equations by A-stable integration techniques, SIAM J. Numer. Anal. 8, pp. 767—785.

Боггс и Толле
Boggs P. T. and Tolle J. W. (1980). Augmented Lagrangians which are quadratic in the multiplier, J. Opt. Th. Applics. 31, pp. 17—26.

Брайсон и Келлам
Braylor R. K. and Cullum J. (1977). «Optimization with the parameters constrained to a box», in Proceedings of the IMACS International Symposium on Simulation Software and Numerical Methods for Differential Equations, IMACS.
— (1979). An algorithm for minimizing a differentiable function subject to box constraints and errors, J. Opt. Th. Applic. 29, pp. 521—658.

Брэкен и Мак-Кормик
Bracken J. and McCormick G. P. (1968). Selected Applications of Nonlinear Programming, John Wiley and Sons, New York and Toronto.

Брент
Brent R. P. (1973a). Algorithms for Minimization without Derivatives, Prentice-Hall, Inc., Englewood Cliffs, New Jersey.
— (1973b). Some efficient algorithms for solving systems of nonlinear equations, SIAM J. Numer. Anal. 10, pp. 327—344.

Бродли
Brodlie K. W. (1977a). «Unconstrained optimization», in The State of the Art in Numerical Analysis (D. Jacobs, ed.), pp. 229—268, Academic Press, London and New York.
— (1977b). An assessment of two approaches to variable metric methods, Math. Prog. 12, pp. 344—355.

Бройден
Broyden C. G. (1965). A class of methods for solving nonlinear simultaneous equations, Mathematics of Computation 19, pp. 577—593.
— (1967). Quasi-Newton methods and their application to function minimization, Mathematics of Computation 21, pp. 368—381.
— (1970). The convergence of a class of double rank minimization algorithms, J. Inst. Maths. Applics. 6, pp. 76—90.

Бройден, Деннис и Море
Broyden C. G., Dennis J. E., Jr. and Moré J. J. (1973). On the local and superlinear convergence of quasi-Newton methods, J. Inst. Maths. Applics. 12, pp. 223—245.

Бузингер и Голуб
Businger P. and Golub G. H. (1965). Linear least-squares solutions by Householder transformations, Num. Math. 7, pp. 269—276.

Барродейл и Робертс
Barrodale I. and Roberts F. D. K. (1973). An improved algorithm for discrete l_1 linear approximation, SIAM J. Numer. Anal. 10, pp. 839—848.

Батчелор и Бил
Batchelor A. S. J. and Beale E. M. L. (1976). «A revised method of conjugate-gradient approximation programming», presented at the Ninth International Symposium on Mathematical Programming, Budapest.

Ван-дер-Хук
Van der Hoek G. (1979). Asymptotic properties of reduction methods applying linearly equality constrained reduced problems, Report 7933, Econometric Institute, Erasmus University, Rotterdam.

Ведин
Wedin P. A. (1974). On the Gauss—Newton method for the nonlinear least-squares problem, Report 23, Swedish Institute for Applied Mathematics (ITM), Stockholm.

Вулф
Wolfe P. (1959). The simplex method for quadratic programming, Econometrica 27, pp. 382—398.
— (1962). The reduced-gradient method, unpublished manuscript, the RAND Corporation.
— (1963a). A technique for resolving degeneracy in linear programming, SIAM J. Appl. Math. 11, pp. 205—211.
— (1963b). «Methods of nonlinear programming», in Recent Advances in Mathematical Programming (J. Abadie, ed.), pp. 67—86, North-Holland, Amsterdam.
— (1966). On the convergence of gradient methods under constraints, IBM Research report, Zurich Laboratory.
— (1967). «Methods of nonlinear programming», in Nonlinear programming (J. Abadie, ed.), pp. 97—131, North-Holland, Amsterdam.
— (1969). Convergence conditions for ascent methods, SIAM Review 11, pp. 226—235.
— (1976). Checking the calculation of gradients, Report RC 6007, IBM Yorktown Heights Research Center (May 1976).
— (1980a). A bibliography for the ellipsoid algorithm, Report RC 8237, IBM Yorktown Heights Research Center (April 1980).
— (1980b). The ellipsoid algorithm, in Optima, 1 (Newsletter of the Mathematical Programming Society, June 1980).

Гарсия Паломарес и Мангасарьян
Garcia Palomares U. M. and Mangasarian O. L. (1976). Superlinearly convergent quasi-Newton algorithms for nonlinearly constrained optimization problems, Math. Prog. 11, pp. 1—13.

Гач и Ловас
Gács P. and Lovász L. (1981). Khachiyan's algorithm for linear programming, Math. Prog. Study 14, pp. 61—68.

Ге и Томас
Gue R. L. and Thomas M. E. (1968). Mathematical Methods in Operations Research, The Macmillan Company, New York.

Гилл
Gill P. E. (1975). Numerical Methods for Large-Scale Linearly Constrained Optimization Problems, Ph. D. Thesis, University of London.

Гилл, Голуб, Мюррей и Сондерс
Gill P. E., Golub G. H., Murray W. and Saunders M. A. (1974). Methods for modifying matrix factorizations, Mathematics of Computation 28, pp. 505—535.

Гилл и Мюррей
Gill P. E. and Murray W. (1972). Quasi-Newton methods for unconstrained optimization, J. Inst. Maths. Applics. 9, pp. 91—108.
— — (1973a). The numerical solution of a problem in the calculus of variations, in Recent Mathematical Developments in Control (D. J. Bell, ed.), pp. 97—122, Academic Press, London and New York.
— — (1973b). Quasi-Newton methods for linearly constrained optimization, Report NAC 32, National Physical Laboratory, England.
— — (1973c). A numerically stable form of the simplex method, Linear Algebra and its Applics. 7, pp. 99—138.
— — (1974a). Newton-type methods for unconstrained and linearly constrained optimization, Math. Prog. 28, pp. 311—350.
— — (1974b). Numerical Methods for Constrained Optimization, Academic Press, London and New York. (Имеется перевод: Ф. Гилл и У. Мюррей. Численные методы условной оптимизации.— М.: Мир, 1977.)
— — (1974c). «Newton-type methods for linearly constrained optimization», in Numerical Methods for Constrained Optimization (P. E. Gill and W. Murray, eds.), pp. 29—66, Academic Press, London and New York.
— — (1974d). «Quasi-Newton methods for linearly constrained optimization», in Numerical Methods for Constrained Optimization (P. E. Gill and W. Murray, eds.), pp. 67—92, Academic Press, London and New York.
— — (1974e). Safeguarded steplength algorithms for optimization using descent methods, Report NAC 37. National Physical Laboratory, England.
— — (1976a). «Nonlinear least squares and nonlinearly constrained optimization», in Numerical Analysis, Dundee 1975 (G. A. Watson, ed.), pp. 135—147, Springer-Verlag Lecture Notes in Mathematics 506, Berlin, Heidelberg and New York.
— — (1976b). Minimization subject to bounds on the variables, Report NAC 71, National Physical Laboratory, England.
— — (1977a). «Linearly constrained problems including linear and quadratic programming», in The State of the Art in Numerical Analysis (D. Jacobs, ed.), pp. 313—363, Academic Press, London and New York.
— — (1977b). The computation of Lagrange multiplier estimates for constrained minimization, Report NAC 77, National Physical Laboratory, England.
— — (1978a). Algorithms for the solution of the nonlinear least-squares problem, SIAM J. Numer. Anal. 15, pp. 977—992.
— — (1978b). Numerically stable methods for quadratic programming, Math. Prog. 14, pp. 349—372.
— — (1978c). The design and implementation of software for unconstrained optimization, in Design and Implementation of Optimization Software (H. Greenberg, ed.), pp. 221—234, Sijthoff and Noordhoff, Netherlands.
— — (1979a). Conjugate-gradient methods for large-scale nonlinear optimization, Report SOL 79-15, Department of Operations Research, Stanford University, California.
— — (1979b). The computation of Lagrange multiplier estimates for constrained minimization, Math. Prog. 17, pp. 32—60.
— — (1979c). «Performance evaluation for optimization software», in Performance Evaluation of Numerical Software (L. D. Fosdick, ed.), pp. 221—234, North-Holland, Amsterdam.

Гилл, Мюррей и Нэш
Gill P. E., Murray W. and Nash S. G. (1991). Newton-type minimization methods

using the linear conjugate-gradient method, Report (to appear), Department of Operations Research, Stanford University, California.

Гилл, Мюррей, Пикен и Райт
Gill P. E., Murray W., Picken S. M. and Wright M. H. (1979). The design and structure of a Fortran program library for optimization, ACM Trans. Math. Software 5, pp. 259—283.

Гилл, Мюррей и Сондерс
Gill P. E., Murray W. and Saunders M. A. (1975). Methods for computing and modifying the LDV factors of a matrix, Mathematics of Computation 29, pp. 1051—1077.

Гилл, Мюррей, Сондерс и Райт
Gill P. E., Murray W., Saunders M. A. and Wright M. H. (1979). Two step-length algorithms for numerical optimization, Report SOL 79-25, Department of Operations Research, Stanford University, California.
— — — — (1980). «A projected Lagrangian method for problems with both linear and nonlinear constraints», presented at the SIAM 1980 Fall Meeting, Houston, Texas.
— — — — (1981a). «A numerical investigation of ellipsoid algorithms for large-scale linear programming», in Large-Scale Linear Programming (Volume 1) (G. B. Dantzig, M. A. Dempster and M. J. Kallio, eds.), pp. 487—509, IIASA Collaborative Proceedings Series, CP-81-S1, IIASA, Laxenburg, Austria.
— — — — (1981b). QP-based methods for large-scale nonlinearly constrained optimization, Report SOL 81-4, Department of Operations Research, Stanford University, California. To appear in Nonlinear Programming 4, (O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds.), Academic Press, London and New York.

Гилл
Glad S. T. (1979). Properties of updating methods for the multipliers in augmented Lagrangians, J. Opt. Th. Applics 28, pp. 135—156.

Гилл и Полак
Glad S. T. and Polak E. (1979). A multiplier method with automatic limitation of penalty growth, Math. Prog. 17, pp. 140—155.

Голуб и Перейра
Golub G. H. and Pereyra V. (1973). The differentiation of pseudo-inverses and nonlinear least-squares problems whose variables separate, SIAM J. Numer. Anal. 10, pp. 413—432.

Голуб и Райнш
Golub G. H. and Reinsch C. (1971). «Singular value decomposition and least-squares solutions», in Handbook for Automatic Computation, Vol. II (J. H. Wilkinson and C. Reinsch, eds.), pp. 134—151, Springer-Verlag, Berlin, Heidelberg and New York.

Гольдфарб
Goldfarb D. (1969). Extension of Davidon's variable metric method to maximization under linear inequality and equality constraints, SIAM J. Appl. Math. 17, pp. 739—764.
— (1970). A family of variable metric methods derived by variational means, Mathematics of Computation 24, pp. 23—26.
— (1980). Curvilinear path steplength algorithms for minimization which use directions of negative curvature, Math. Prog. 18, pp. 31—40.

Гольдфарб и Рейд
Goldfarb D. and Reid J. K. (1977). A practicable steepest-edge simplex algorithm, Math. Prog. 12, pp. 361—371.

Гольдфелд, Квандт и Тротер
Goldfeld S. M., Quandt R. E. and Trotter H. F. (1966), Maximization by quadratic hill-climbing, Econometrica 34, pp. 541—551.

Гольдштейн и Прайс
Goldstein A. and Price J. (1967). An effective algorithm for minimization, Numer. Math. 10, pp. 184—189.

Гриффин
Griffin J. L. (1980). Convergence results in a class of variable metric subgradient methods, Working Paper 80-08, Faculty of Management, McGill University, Montreal, Canada. To appear in Nonlinear Programming 4 (O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds.), Academic Press, London and New York.

Гринберг
Greenberg H. J. (1978a). «A tutorial on matricial packing», in Design and Implementation of Optimization Software (H. J. Greenberg, ed.), pp. 109—142, Sijthoff and Noordhoff, Netherlands.
— (1978b). «Pivot selection tactics», in Design and Implementation of Optimization Software (H. J. Greenberg, ed.), pp. 143—174, Sijthoff and Noordhoff, Netherlands.

Гринберг и Кэлан
Greenberg H. J. and Kalan J. E. (1975). An exact update for Harris' TREAD, Math. Prog. Study 4, pp. 26—29.

Гринштадт
Greenstadt J. L. (1967). On the relative efficiencies of gradient methods, Mathematics of Computation 21, pp. 360—367.
— (1970). Variations on variable-metric methods, Mathematics of Computation 24, pp. 1—22.
— (1972). A quasi-Newton method with no derivatives, Mathematics of Computation 26, pp. 145—166.

Гриффин и Стюарт
Griffith R. E. and Stewart R. A. (1981). A nonlinear programming technique for the optimization of continuous processing systems, Management Science, 7, pp. 379—392.

Грэхем
Graham S. R. (1976). A matrix factorization and its application to unconstrained minimization, Project thesis for BSc. (Hons) in Mathematics for Business, Middlesex Polytechnic, Enfield, England.

Гэй
Gay D. M. (1979a). On robust and generalized linear regression problems, Report 200, Mathematics Research Center, University of Wisconsin, Madison, Wisconsin.
— (1979b). Computing optimal locally constrained steps, Report 2013, Mathematics Research Center, University of Wisconsin, Madison, Wisconsin.

Гэй и Шнабель
Gay D. M. and Schnabel R. B. (1978). «Solving systems of nonlinear equations by Broyden's method with projected updates», in Nonlinear Programming 3 (O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds.), pp. 245—281, Academic Press, London and New York.

Дальквист и Бьорк
Dahlquist G. and Björck A. (1974). Numerical Methods, Prentice-Hall Inc., Englewood Cliffs, New Jersey.

Данциг
Dantzig G. B. (1963). Linear Programming and Extensions, Princeton University Press, Princeton New Jersey. (Имеется перевод: Дж. Данциг. Линейное программирование, его обобщения и применения.— М.: Прогресс, 1966.)

Данциг, Орден и Вулф
Dantzig G. B., Orden A. and Wolfe P. (1955). Generalized simplex method for minimizing a linear form under linear inequality restraints, Pacific J. Math. 5, pp. 183—195.

Данциг, Демпстер и Каллио
Dantzig G. B., Dempster M. A. H. and Kallio M. J. (eds.) (1981). Large-Scale Linear Programming (Volume 1), IIASA Collaborative Proceedings Series, CP 81-51, IIASA, Laxenburg, Austria.

Дафф и Рейд
Duff I. S. and Reid J. K. (1978). An implementation of Tarjan's algorithm for the block triangularization of a matrix, ACM Trans. Math. Software 4, pp. 137—147.

Даффин, Петерсон и Зенер
Duffin R. J., Peterson E. L. and Zener C. (1967). Geometric Programming—Theory and Applications, John Wiley and Sons, New York and Toronto.

Деккер
Dekker T. J. (1969). «Finding a zero by means of successive linear interpolation», in Constructive Aspects of the Fundamental Theorem of Algebra (B. Dejon and P. Henrici, eds.), pp. 37—48, Wiley Interscience, London.

Дембо
Dembo R. S. (1978). Current state of the art of algorithms and computer software for geometric programming, J. Opt. Th. Applics. 26, pp. 149—184.

Дембо, Айзенштат и Штайхаут
Dembo R. S., Eisenstat S. C. and Steihaug T. (1980). Inexact Newton methods, Working Paper # 47, School of Organization and Management, Yale University.

Дембо и Штайхаут
Dembo R. S. and Steihaug T. (1980). Truncated-Newton algorithms for large-scale unconstrained optimization, Working Paper # 48, School of Organization and Management, Yale University.

Деннис
Dennis J. E., Jr. (1973). «Some computational techniques for the nonlinear least-squares problem», in Numerical Solution of Systems of Nonlinear Algebraic Equations (G. D. Byrne and C. A. Hall, eds.), pp. 157—183, Academic Press, London and New York.
— (1977). «Nonlinear Least-Squares», in The State of the Art in Numerical Analysis (D. Jacobs, ed.), pp. 269—312, Academic Press, London and New York.

Деннис и Море
Dennis J. E., Jr. and Moré J. J. (1974). A characterization of superlinear convergence and its application to quasi-Newton methods, Mathematics of Computation 28, pp. 549—560.
— — (1977). Quasi-Newton methods, motivation and theory, SIAM Review 19, pp. 46—89.

Деннис, Гей и Уэлш
Dennis J. E., Jr., Gay D. M. and Welsch R. E. (1977). An adaptive nonlinear least-squares algorithm, Report TR 77-321, Department of Computer Science, Cornell University.

Деннис и Шнабель
Dennis J. E., Jr. and Schnabel R. E. (1979). Least change secant updates for quasi-Newton methods, SIAM Review 21, pp. 443—469.
— — (1980). A new derivation of symmetric positive definite secant updates, Report CU-CS-185-80, Department of Mathematical Sciences, Rice University.

Джан
Djang A. (1980). Algorithmic Equivalence in Quadratic Programming, Ph. D. Thesis, Stanford University, California.

Джейн, Лэсдон и Сондерс
Jain A., Lasdon L. S. and Saunders M. A. (1976). «An in-core nonlinear mathematical programming system for large nonlinear programs», presented at ORSA/TIMS Joint National Meeting, Miami, Florida.

Джонсон
Johnson E. L. (1978). «Some considerations in using branch-and-bound codes», in Design and Implementation of Optimization Software (H. J. Greenberg, ed.), pp. 241—248, Sijthoff and Noordhoff, Netherlands.

Джонсон и Пауэлл
Johnson E. L. and Powell S. (1978). «Integer programming codes», in Design and Implementation of Optimization Software (H. J. Greenberg, ed.), pp. 225—240, Sijthoff and Noordhoff, Netherlands.

Диксон
Dixon L. C. W. (1972a). Quasi-Newton algorithms generate identical points, Math. Prog. 2, pp. 383—387.
— (1972b). Quasi-Newton algorithms generate identical points. II. The proof of four new theorems, Math. Prog. 3, pp. 345—358.
— (1975). Conjugate-gradient algorithms: quadratic termination without linear searches, J. Inst. Maths. Applics. 15, pp. 9—18.

Донгарра, Банч, Молер, Стюарт
Dongarra J. J., Bunch J. R., Moler C. B. and Stewart G. W. (1979). LINPACK Users Guide, SIAM Publications, Philadelphia.

Давидон
Davidon W. C. (1959). Variable metric methods for minimization, A. E. C. Res. and Develop. Report ANL-5990, Argonne National Laboratory, Argonne, Illinois.
— (1975). Optimally conditioned optimization algorithms without line searches, Math. Prog. 9, pp. 1—30.
— (1979). Conic approximations and collinear scalings for optimizers, SIAM J. Numer. Anal. 17, pp. 268—281.

Дэвис и Рабиновиц
Davis P. J. and Rabinowitz P. (1967). Numerical Integration, Blaisdell, London.

Дэниел, Грэг, Кауфман и Стюарт
Daniel J. W., Gragg W. B., Kaufman L. C. and Stewart G. W. (1976). Reorthogonalization and stable algorithms for updating the Gram—Schmidt QR factorization, Mathematics of Computation 30, pp. 772—795.

Дюмонте и Вивьес
Dumontet J. and Vignes J. (1977). Détermination du pas optimal dans le calcul des dérivées sur ordinateur, Revue française d'automatique, d'information et de recherche opérationelle, Analyse numérique (RAIRO) 11, pp. 13—25.

Зангвилл
Zangwill W. I. (1965). Nonlinear programming by sequential unconstrained maximization, Working Paper 131, Center for Research in Management Science, University of California, Berkeley.
— (1967a). Nonlinear programming via penalty functions, Management Science 13, pp. 344—358.
— (1967b). Algorithm for the Chebyshev problem, Management Science 14, pp. 58—78.

Зойтендейк
Zoutendijk G. (1970). «Nonlinear programming, computational methods», in Inte-

ger and Nonlinear Programming (J. Abadie, ed.), pp. 37—63, North-Holland, Amsterdam.

Зуховицкий С. И., Полак Р. А. и Примак М. Е. (1963). Алгоритм для решения задач выпуклого чебышевского приближения.— Докл. АН СССР, 1963, т. 153, № 5, с. 991—994.

Канторович Л. В. и Акилов Г. П. (1959). Функциональный анализ в нормированных пространствах.— М.: Физматгиз, 1959.

Караламбус
Charalambous C. (1978). A lower bound for the controlling parameter of the exact penalty function, Math. Prog. 15, pp. 278—290.

Караламбус и Конн
Charalambous C. and Conn A. R. (1978). An efficient method to solve the minimax problem directly, SIAM J. Numer. Anal. 15, pp. 162—187.

Кауфман и Перейра
Kaufman L. C. and Pereyra V. (1978). A method for separable nonlinear least-squares problems with separable nonlinear equality constraints, SIAM J. Numer. Anal. 15, pp. 12—20.

Кахан
Kahan W. (1973). The implementation of algorithms: Part I, Technical Report 20, Department of Computer Science, University of California, Berkeley.

Кели
Kelley J. E. (1960). The cutting plane method for solving convex programs, J. Soc. Indust. Appl. Math. 8, pp. 703—712.

Кертис, Пауэлл и Рейд
Curtis A. R., Powell M. J. D. and Reid J. K. (1974). On the estimation of sparse Jacobian matrices, J. Inst. Maths. Applics. 13, pp. 117—119.

Кертис и Рейд
Curtis A. R. and Reid J. K. (1972). On the automatic scaling of matrices for Gaussian elimination, J. Inst. Maths. Applics. 10, pp. 118—124.
— — (1974). The choice of step lengths when using differences to approximate Jacobian matrices, J. Inst. Maths. Applics. 13, pp. 121—126.

Клайн, Мелер, Стюарт и Уилкинсон
Cline A. K., Moler C. B., Stewart G. W. and Wilkinson J. H. (1979). An estimate for the condition number of a matrix, SIAM J. Numer. Anal. 16, pp. 368—375.

Кнут
Knuth D. E. (1979). TEX and METAFONT, New Directions in Typesetting, American Mathematical Society and Digital Press, Bedford, Massachusetts.

Коко
Cox M. G. (1977). A survey of numerical methods for data and function approximation, in: The State of the Art in Numerical Analysis (D. Jacobs, ed.), pp. 627—668, Academic Press, London and New York.

Колвилл
Colville A. R. (1968). A comparative study on nonlinear programming codes, Report No. 320-2949, IBM New York Scientific Center.

Колеман
Coleman T. F. (1979). A Superlinear Penalty Function Method to Solve the Nonlinear Programming Problem, Ph. D. Thesis, University of Waterloo, Ontario, Canada.

Колеман и Конн
Coleman T. F. and Conn A. R. (1980a). Second-order conditions for an exact penalty function, Math. Prog. 19, pp. 178—185.

— — (1980b). Nonlinear programming via an exact penalty function: asymptotic analysis, Report CS-80-30, Department of Computer Science, University of Waterloo, Ontario, Canada.
— — (1980c). Nonlinear programming via an exact penalty function: global analysis, Report CS-80-31, Department of Computer Science, University of Waterloo, Ontario, Canada.

Коулман и Мор
Coleman T. F. and Moré J. J. (1980), «Estimating large sparse Jacobian matrices and Hessians», presented at the SIAM 1980 Fall Meeting, Houston, November 1980.

Конн, Голаб и О'Лири
Conn N. P., Golub G. H. and O'Leary D. P. (1976), «A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations», in Sparse Matrix Computations (J. R. Bunch and D. J. Rose, eds.), pp. 309—332, Academic Press, London and New York.

Конн
Conn A. R. (1973). Constrained optimization using a non-differentiable penalty function, SIAM J. Numer. Anal. 10, pp. 760—779.
— (1976). Linear programming via a non-differentiable penalty function, SIAM J. Numer. Anal. 13, pp. 145—154.
— (1979). An efficient second-order method to solve the (constrained) minimax problem, Report CORR-79-5, University of Waterloo, Canada.

Конн и Питтуковский
Conn A. R. and Pietrzykowski T. (1977). A penalty function method converging directly to a constrained optimum, SIAM J. Numer. Anal. 14, pp. 348—375.

Конн и Сангиер
Conn A. R. and Sanchar J. W. (1976). Quadratic programming via a non-differentiable penalty function, Report 76/15, Department of Combinatorics and Optimization, University of Waterloo, Canada.

Корн
Korn B. W. (1973). «Rate of convergence of the method of multipliers with inexact minimizations», in Nonlinear Programming 2 (O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds.), pp. 193—214, Academic Press, London and New York.

Корт и Бертекас
Kort B. W. and Bertsekas D. P. (1976). Combined primal dual and penalty methods for convex programming, SIAM J. Control and Optimization 14, pp. 268—294.

Коттл
Cottle R. W. (1974). Manifestations of the Schur complement, Linear Algebra and its Applics. 8, pp. 189—211.

Коши
Cauchy A. (1847). Méthode Générale pour la Résolution des Systèmes d'Équations Simultanées, Comp. Rend. Acad. Sci. Paris, pp. 536—538.

Кун
Kuhn H. W. (1976). «Nonlinear programming: a historical view», in SIAM—AMS Proceedings, Volume IX, Mathematical Programming (R. C. Cottle and C. E. Lemke, eds.), pp. 1—26, American Mathematical Society, Providence, Rhode Island.

Кун и Таккер
Kuhn H. W. and Tucker A. W. (1951). «Nonlinear Programming», in Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability (J. Neyman, ed.), pp. 481—492, Berkeley, University of California Press.

Курант
Courant R. (1936). Differential and Integral Calculus (two volumes), Blackie, London and Glasgow.
— (1943). Variational methods for the solution of problems of equilibrium and vibrations, Bull. Amer. Math. Soc. 49, pp. 1—23.

Кеннеди и Дакс
Kandel S. and Dax A. (1979). A modified Newton's method for unconstrained minimization, SIAM J. Numer. Anal. 16, pp. 324—331.

Кэрролл
Carrol C. W. (1958). An Operations Research Approach to the Economic Optimization of a Kraft Pulping Process, Ph. D. Thesis, Institute of Paper Chemistry, Appleton, Wisconsin.
— (1961). The created response surface technique for optimizing nonlinear restrained systems, Operations Research 9, pp. 169—184.

Лайнесс
Lyness J. N. (1976). «An interface problem in numerical software», Proceedings of the 6th Manitoba Conference on Numerical Mathematics, pp. 251—263.
— (1977a). «Has numerical differentiation a future?» Proceedings of the 7th Manitoba Conference on Numerical Mathematics, pp. 107—129.
— (1977b). «Quid, quo, quadrature?» in The State of the Art in Numerical Analysis (D. Jacobs, ed.), pp. 535—560, Academic Press, London and New York.

Лайнесс и Молер
Lyness J. N. and Moler C. B. (1967). Numerical differentiation of analytic functions, SIAM J. Numer. Anal. 4, pp. 202—210.

Лайнесс и Сэнде
Lyness J. N. and Sande G. (1971). ENTCAF and ENTCRE: Evaluation of normalized Taylor coefficients of an analytic function, Comm. ACM 14, pp. 669—675.

Левенберг
Levenberg K. (1944). A method for the solution of certain problems in least squares, Quart. Appl. Math. 2, pp. 154—168.

Лемарешаль
Lemarechal C. (1975). An extension of Davidon methods to non-differentiable problems, Math. Prog. Study 3, pp. 95—109.

Лемке
Lemke C. E. (1965). Bimatrix equilibrium points and mathematical programming, Management Science 11, pp. 681—689.

Леонард
Lenard M. L. (1979). A computational study of active set strategies in nonlinear programming with linear constraints, Math. Prog. 16, pp. 81—97.

Лиенбергер
Luenberger D. G. (1973). Introduction to Linear and Nonlinear Programming, Addison-Wesley, Menlo Park, California.
— (1974). A combined penalty function and gradient projection method for nonlinear programming, J. Opt. Th. Applics. 14, pp. 477—495.

Лилл
Lill S. A. (1972). «Generalization of an exact method for solving equality constrained problems to deal with inequality constraints», in Numerical Methods for Non-Linear Optimization (F. A. Lootsma, ed.), pp. 383—394, Academic Press, London and New York.

Лоулер
Lawler E. L. (1980). The great mathematical sputnik of 1979, University of California, Berkeley, California (February 1980).

Лоусон и Хенсон
Lawson C. L. and Hanson R. J. (1974). Solving Least-Squares Problems, Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Лутсма
Loetsma F. A. (1969). Hessian matrices of penalty functions for solving constrained optimization problems, Philips Res. Repts **24**, pp. 322—331.
— (1970). Boundary properties of penalty functions for constrained optimization problems, Philips Res. Repts 3.
— (1972). «A survey of methods for solving constrained optimization problems via unconstrained minimizations», in Numerical Methods for Non-Linear Optimization (F. A. Loutsma, ed.), pp. 313—347, Academic Press, London and New York

Лесфон и Уорен
Lasdon L. S. and Waren A. D. (1978). «Generalized reduced gradient software for linearly and nonlinearly constrained problems», in Design and Implementation of Optimization Software (H. J. Greenberg, ed.), pp. 335—362, Sijthoff and Noordhoff, Netherlands.

Лесфон, Уорен, Джейн и Ратнер
Lasdon L. S., Waren A. D., Jain A. and Ratner M. (1978). Design and testing of a GRG code for nonlinear optimization, ACM Trans. Math. Software, **4**, pp. 34—50.

Лесфон, Фокс и Ратнер
Lasdon L. S., Fox R. L. and Ratner M. (1973). An efficient one-dimensional search procedure for barrier functions, Math. Prog. **4**, pp. 279—296.

Мадсен
Madsen K. (1975). An algorithm for the minimax solution of overdetermined systems of linear equations, J. Inst. Maths. Applics. **16**, pp. 321—328.

Мак-Кормик
McCormick G. P. (1969). Anti-zygzagging by bending, Management Science **15**, pp. 315—320.
— (1970a). The variable reduction method for nonlinear programming, Management Science **17**, pp. 146—160.
— (1970b). «A second-order method for the linearly constrained nonlinear programming problems», in Nonlinear Programming (J. B. Rosen, O. L. Mangasarian and K. Ritter, eds.), pp. 207—243, Academic Press, London and New York.
— (1977). A modification of Armijo's step-size rule for negative curvature, Math. Prog. **13**, pp. 11—115.

Мак-Лин и Уотсон
McLean R. A. and Watson G. A. (1979). Numerical methods for nonlinear discrete l_1 approximation problems, proceedings of the Oberwolfach Conference on Approximation Theory (to appear).

Мангасарьян
Mangasarian O. L. (1969). Nonlinear Programming, McGraw-Hill Book Co., New York.
— (1975). Unconstrained Lagrangians in nonlinear programming, SIAM J. Control and Optimization **13**, pp. 772—791.

Маратос
Maratos N. (1978). Exact Penalty Function Algorithms for Finite-Dimensional and Control Optimization Problems, Ph. D. Thesis, University of London.

Марвил
Marwil E. (1978). Exploiting Sparsity in Newton-Type Methods, Ph. D. Thesis, Cornell University, Ithaca, New York.

Маркардт
Marquardt D. (1963). An algorithm for least-squares estimation of nonlinear parameters, SIAM J. Appl. Math. 11, pp. 431—441.

Марковиц
Markowitz H. M. (1957). The elimination form of the inverse and its applications to linear programming, Management Science 3, pp. 255—269.

Марстен
Marsten R. E. (1978). XMP: A structured library of subroutines for experimental mathematical programming, Report 35), Department of Management Information Systems, University of Arizona, Tucson, Arizona.

Марстен и Шанно
Marsten R. E. and Shanno D. F. (1979). Conjugate-gradient methods for linearly constrained nonlinear programming, Report 79-13, Department of Management Information Systems, University of Arizona, Tucson, Arizona.

Миеле, Крэгг, Айер и Леви
Miele A., Cragg E. E., Iyer R. R. and Levy A. V. (1971). Use of the augmented penalty function in mathematical programming, Part I, J. Opt. Th. Applics. 8, pp. 115—130.

Миеле, Крэгг и Леви
Miele A., Cragg E. E. and Levy A. V. (1971). Use of the augmented penalty function in mathematical programming, Part II, J. Opt. Th. Applics. 8, pp. 131—153.

Миллер
Miller C. E. (1963), «The simplex method for local separable programming», in Recent Advances in Mathematical Programming (R. L. Graves and P. Wolfe, eds.), pp. 89—100, McGraw-Hill Book Co., New York.

Мифлин
Mifflin R. (1975). A superlinearly convergent algorithm for minimization without evaluating derivatives, Math. Prog. 9, pp. 100—117.
— (1977). Semismooth and semiconvex functions in constrained optimization, SIAM J. Control and Optimization 15, pp. 959—972.

Море
Moré J. J. (1977), «The Levenberg—Marquardt algorithm: implementation and theory», in Numerical Analysis (G. A. Watson, ed.), pp. 105—116, Lecture Notes in Mathematics 630, Springer-Verlag, Berlin, Heidelberg and New York.
— (1979a). On the design of optimization software, Report DAMTP 79-NA 8, University of Cambridge.
— (1979b), «Implementation and testing of optimization software», in Performance Evaluation of Numerical Software (L. D. Fosdick, ed.), pp. 253—265, North-Holland, Amsterdam.

Море и Соренсен
Moré J. J. and Sorensen D. C. (1979). On the use of directions of negative curvature in a modified Newton method, Math. Prog. 16, pp. 1—20.

Муртаф
Murtagh B. A. (1981). Advanced Linear Programming, McGraw-Hill Book Co., New York. (Имеется перевод: Б. Муртаф. Современное линейное программирование. — М.: Мир, 1984.)

Муртаф и Сарджент
Murtagh B. A. and Sargent R. H. W. (1969). «A constrained minimization method with quadratic convergence», in Optimization (R. Fletcher, ed.), pp. 215—246, Academic Press, London and New York.

Муртаф и Сондерс
Murtagh B. A. and Saunders M. A. (1977). MINOS User's Guide, Report SOL 77-9, Department of Operations Research, Stanford University, California.

— — (1978). Large-scale linearly constrained optimization, Math. Prog. 14, pp. 41—72.
— — (1980). A projected Lagrangian algorithm and its implementation for sparse nonlinear constraints, Report SOL 80-1R, Department of Operations Research, Stanford University, California, to appear in Math. Prog. Study on Constrained Optimization.

Мэйн и Маратос
Mayne D. Q. and Maratos N. (1979). A first-order exact penalty function algorithm for equality constrained optimization problems, Math. Prog. 16, pp. 303—324.

Мэйн и Полак
Mayne D. Q. and Polak E. (1976). Feasible direction algorithms for optimization problems with equality and inequality constraints, Math. Prog. 11, pp. 67—80.

Мюррей
Murray W. (1967). «Ill-conditioning in barrier and penalty functions arising in constrained nonlinear programming», presented at the Princeton Mathematical Programming Symposium, August 14—18, 1967.
— (1969a). Constrained Optimization, Ph. D. Thesis, University of London.
— (1969b). «An algorithm for constrained minimization», in Optimization (R. Fletcher, ed.), pp. 247—258, Academic Press, London and New York.
— (1971a). An algorithm for finding a local minimum of an indefinite quadratic program, Report NAC 1, National Physical Laboratory, England.
— (1971b). Analytical expressions for the eigenvalues and eigenvectors of the Hessian matrices of barrier and penalty functions, J. Opt. Th. Applics. 7, pp. 189—196.
— (1972a). «Second derivative methods», in Numerical Methods for Unconstrained Optimization (W. Murray, ed.), pp. 57—71, Academic Press, London and New York.
— (1972b). «Failure, the causes and cures», in Numerical Methods for Unconstrained Optimization (W. Murray, ed.), pp. 107—122, Academic Press, London and New York.
— (1976). «Constrained Optimization», in Optimization In Action (L. C. W. Dixon, ed.), pp. 217—251, Academic Press, London and New York.

Мюррей и Овертон
Murray W. and Overton M. L. (1980a). A projected Lagrangian algorithm for nonlinear minimax optimization, SIAM J. Sci. Stat. Comput. 1, pp. 345—370.
— — (1980b). A projected Lagrangian algorithm for nonlinear l_1 optimization, Report SOL 80-4, Department of Operations Research, Stanford University, California.

Мюррей и Райт
Murray W. and Wright M. H. (1976). Efficient linear search algorithms for the logarithmic barrier function, Report SOL 76-18, Department of Operations Research, Stanford University, California.
— — (1978). Projected Lagrangian methods based on the trajectories of penalty and barrier functions, Report SOL 78-23, Department of Operations Research, Stanford University, California.
— — (1980). Computation of the search direction in constrained optimization algorithms, Report SOL 80-2, Department of Operations Research, Stanford University, to appear in Math. Prog. Study on Constrained Optimization.

Назарет
Nazareth L. (1977). A conjugate-direction algorithm without line searches, J. Opt. Th. Applics. 23, pp. 373—388.
— (1979). A relationship between the BFGS and conjugate-gradient algorithms and its implications for new algorithms, SIAM J. Numer. Anal. 16, pp. 794—800.

Назарет и Носедал
Nazareth L. and Nocedal J. (1978). A study of conjugate-gradient methods, Report SOL 78-29, Department of Operations Research, Stanford University, California.

Нелдер и Мид
Nelder J. A. and Mead R. (1965). A simplex method for function minimization, Computer Journal 7, pp. 308—313.

Немировский А. С. и Юдин Д. Б. (1979). Эффективные методы решения задач выпуклого программирования большой размерности.— Экономика и математические методы, 1979, № 2, с. 135—152.

Носедаль
Nocedal J. (1980). Updating quasi-Newton matrices with limited storage, Mathematics of Computation 35, pp. 773—782.

Оливер
Oliver J. (1980). An algorithm for numerical differentiation of a function of one real variable, J. Comp. Appl. Math. 6, pp. 145—160.

Оливер и Реффхед
Oliver J. and Ruffhead A. (1975). The selection of interpolation points in numerical differentiation, Nordisk Tidskr. Informationbehandling (BIT) 15, pp. 283—295.

О'Лири
O'Leary D. P. (1980a). A discrete Newton algorithm for minimizing a function of many variables, Report 910, Computer Science Center, University of Maryland, College Park, Maryland
— (1980b). Estimating matrix condition numbers, SIAM J. Sci. Stat. Comput. 1, pp. 205—209.

Орен
Oren S. S. (1974a). Self-scaling variable metric (SSVM) algorithms, Part II: implementation and experiments, Management Science 20, pp. 863—874.
— (1974b). On the selection of parameters in self-scaling variable metric algorithms, Math. Prog. 7, pp. 351—367.

Орен и Люенбергер
Oren S. S. and Luenberger D. G. (1974). Self-scaling variable metric (SSVM) algorithms, Part I: criteria and sufficient conditions for scaling a class of algorithms, Management Science 20, pp. 845—862.

Орен и Спедикато
Oren S. S. and Spedicato E. (1976). Optimal conditioning of self-scaling and variable metric algorithms, Math. Prog. 10, pp. 70—90.

Ортега и Рейнболдт
Ortega J. M. and Rheinboldt W. C. (1970). Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, London and New York. (Имеется перевод: Дж. Ортега, В. Рейнболдт. Итерационные методы решения нелинейных систем уравнений со многими неизвестными.— М.: Мир, 1975.)

Орчард-Хейз
Orchard-Hays W. (1968). Advanced Linear Programming Computing Techniques, McGraw-Hill, New York.
— (1978a). «History of mathematical programming systems», in Design and Implementation of Optimization Software (H. J. Greenberg, ed.), pp. 1—26, Sijthoff and Noordhoff, Netherlands.
— (1978b). «Scope of mathematical programming software», in Design and Implementation of Optimization Software (H. J. Greenberg, ed.), pp. 27—40, Sijthoff and Noordhoff, Netherlands.
— (1978c). «Anatomy of a mathematical system», in Design and Implementation of Optimization Software (H. J. Greenberg, ed.), pp. 41—102, Sijthoff and Noordhoff, Netherlands.

Осборн и Райан
Osborne M. R. and Ryan D. M. (1972). «A hybrid algorithm for nonlinear programming», in Numerical Methods for Non-Linear Optimization (F. A. Lootsma, ed.), pp. 395—410, Academic Press, London and New York.

Осборн и Уотсон
Osborne M. R. and Watson G. A. (1969). An algorithm for minimax approximation in the nonlinear case, Computer Journal 12, pp. 63—68.
— — (1971). An algorithm for discrete nonlinear L_1 approximation, Computer Journal 10, pp. 172—177.

Осеи
Aasen J. O. (1971). On the reduction of a symmetric matrix to tridiagonal form, Nordisk Tidskr. Informationsbehandling (BIT) 11, pp. 233—242.

Паркинсон и Хатчинсон
Parkinson J. M. and Hutchinson D. (1972). «An investigation into the efficiency of variants of the simplex method», in Numerical Methods for Non-Linear Optimization (F. A. Lootsma, ed.), pp. 115—135, Academic Press, London and New York.

Пауэлл
Powell M. J. D. (1964). An efficient method for finding the minimum of a function of several variables without calculating derivatives, Computer Journal 7, pp. 155—162.
— (1969). «A method for nonlinear constraints in minimization problems», in Optimization (R. Fletcher, ed.), pp. 283—298, Academic Press, London and New York.
— (1970a). «A new algorithm for unconstrained optimization», in Nonlinear Programming (J. B. Rosen, O. L. Mangasarian and K. Ritter, eds.), pp. 31—65, Academic Press, London and New York.
— (1970b). «A hybrid method for nonlinear equations», in Numerical Methods for Nonlinear Algebraic Equations (P. Rabinowitz, ed.), pp. 87—114, Gordon and Breach, London.
— (1971). On the convergence of the variable metric algorithm, J. Inst. Maths, Applics 7, pp. 21—36.
— (1972). «Problems relating to unconstrained optimization», in Numerical Methods for Unconstrained Optimization (W. Murray, ed.), pp. 29—55, Academic Press, London and New York.
— (1974). «Introduction to constrained optimization», in Numerical Methods for Constrained Optimization (P. E. Gill and W. Murray, eds.), pp. 1—28, Academic Press, London and New York.
— (1975). «Convergence properties of a class of minimization algorithms», in Nonlinear Programming 2 (O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds.), pp. 1—27, Academic Press, London and New York.
— (1976a). «A view of unconstrained optimization», in Optimization in Action (L. C. W. Dixon, ed.), pp. 117—152, Academic Press, London and New York.
— (1976b). Some convergence properties of the conjugate-gradient method, Math. Prog. 11, pp. 42—49.
— (1976c). «Some global convergence properties of a variable metric algorithm without exact line searches», in SIAM—AMS Proceedings, Volume IX, Mathematical Programming (R. C. Cottle and C. E. Lemke, eds.), pp. 53—72, American Mathematical Society, Providence, Rhode Island.
— (1977a). Restart procedures for the conjugate-gradient method, Math. Prog. 12, pp. 241—254.
— (1977b). A fast algorithm for nonlinearly constrained optimization calculations, Report DAMTP 77/NA 2, University of Cambridge, England.
— (1977c). «Numerical methods for fitting functions of two variables», in The State of the Art in Numerical Analysis (D. Jacobs, ed.), pp. 563—604, Academic Press, London and New York.
— (1978). «The convergence of variable metric methods for nonlinearly constrained optimization calculations», in Nonlinear Programming 3 (O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds.), pp. 27—63, Academic Press, London and New York.

— (1980). «An upper triangular matrix method for quadratic programming», presented at the symposium: Nonlinear Programming 4, Madison, Wisconsin, July 1980.
— (1981). A note on quasi-Newton formulae for sparse second derivative matrices, Math. Prog. 20, pp. 144—151.

Пауэлл и Тойнз
Powell M. J. D. and Toint P. L. (1979). On the estimation of sparse Hessian matrices, SIAM J. Numer. Anal. 16, pp. 1060—1074.

Перолд
Perold A. F. (1981a). «Exploiting degeneracy in the simplex method», in Large-Scale Linear Programming (Volume 1) (G. B. Dantzig, M. A. H. Dempster and M. J. Kallio, eds.), pp. 55—66, IIASA Collaborative Proceedings Series, CP-81-51, IIASA, Laxenburg, Austria.
— (1981b). «A degeneracy-exploiting *LU* factorization for the simplex method», in Large-Scale Linear Programming (Volume 1) (G. B. Dantzig, M. A. H. Dempster and M. J. Kallio, eds.), pp. 67—80, IIASA Collaborative Proceedings Series, CP-81-51, IIASA, Laxenburg, Austria.

Перри
Perry A. (1977). A class of conjugate-gradient algorithms with a two-step variable-metric memory, Discussion Paper 269, Center for Mathematical Studies in Economics and Management Science, Northwestern University.

Петерс и Уилкинсон
Peters G. and Wilkinson J. H. (1970). The least-squares problem and pseudo-inverses, Computer Journal 13, pp. 309—316.

Петерсон
Peterson E. L. (1976). Geometric programming, SIAM Review 18, pp. 1—51.

Петницкевский
Pietrzykowski T. (1962). «Application of the steepest-ascent method to concave programming», in Proceedings of the IFIPS Congress, Munich, 1962, pp. 185—189, North-Holland, Amsterdam.
— (1969). An exact potential method for constrained maxima, SIAM J. Numer. Anal. 6, pp. 299—304.

Полиа
Polya G. (1913). Sur un algorithme toujours convergent pour obtenir les polynomes de meilleure approximation de Tchebycheff pour une fonction continue quelconque, Comptes Rendus Hébdomadaires, Séances de l'Académie des Sciences, Paris.

Пэйг
Paige C. C. (1980). Error analysis of some techniques for updating orthogonal decompositions, Mathematics of Computation 34, pp. 465—471.

Райан
Ryan D. M. (1971). Transformation Methods in Nonlinear Programming, Ph. D. Thesis, Australian National University.
— (1974). «Penalty and barrier functions», in Numerical Methods for Constrained Optimization (P. E. Gill and W. Murray, eds.), pp. 175—190, Academic Press, London and New York.

Райд
Reid J. K. (1975). A sparsity-exploiting variant of the Bartels—Golub decomposition for linear programming bases, Report CSS 20, Atomic Energy Research Establishment, Harwell, England.
— (1976). Fortran subroutines for handling sparse linear programming bases, Report R8269, Atomic Energy Research Establishment, Harwell, England.

Райт
Wright M. H. (1976). Numerical Methods for Nonlinearly Constrained Optimization, Ph. D. Thesis, Stanford University, California.

Робинсон
Robinson S. M. (1972). A quadratically convergent algorithm for general nonlinear programming problems, Math. Prog. 3, pp. 145—156.
— (1974). Perturbed Kuhn—Tucker point and rates of convergence for a class of nonlinear programming algorithms, Math. Prog. 7, pp. 1—16.

Розен
Rosen J. B. (1960). The gradient projection method for nonlinear programming, Part I—linear constraints, SIAM J. Appl. Math. 8, pp. 181—217.
— (1961). The gradient projection method for nonlinear programming, Part II—nonlinear constraints, SIAM J. Appl. Math. 9, pp. 514—532.
— (1978). «Two-phase algorithm for nonlinear constraint problems», in: Nonlinear Programming 3 (O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds.), pp. 97—124. Academic Press, London and New York.

Розен и Кройцер
Rosen J. B. and Kreuser J. (1972). «A gradient projection algorithm for nonlinear constraints», in: Numerical Methods for Non-Linear Optimization (F. A. Lootsma, ed.), pp. 297—300. Academic Press, London and New York.

Розенброк
Rosenbrock H. H. (1960). An automatic method for finding the greatest or least value of a function, Computer Journal 3, pp. 175—184.

Рокафеллар
Rockafellar R. T. (1970). Convex Analysis, Princeton University Press, Princeton New Jersey. (Имеется перевод: Р. Рокафеллар. Выпуклый анализ.— М., Мир, 1973.)
— (1973a). A dual approach to solving nonlinear programming problems by unconstrained optimization, Math. Prog. 5, pp. 354—373.
— (1973b). The multiplier method of Hestenes and Powell applied to convex programming, J. Opt. Th. Applics. 12, pp. 555—562.
— (1974). Augmented Lagrange multiplier functions and duality in nonconvex programming, SIAM J. Control and Optimization 12, pp. 268—285.

Руководство-справочник по ФОРТРАН-библиотеке
NAG Fortran Library Reference Manual (Mark 8) (1981). Numerical Algorithms Group Limited, Oxford, England.

Руководство-справочник по оптимизационному математическому обеспечению
Numerical Optimization Software Library Reference Manual (1978). Division of Numerical Analysis and Computing, National Physical Laboratory, England.

Рухе
Ruhe A. (1979). Accelerated Gauss—Newton algorithms for nonlinear least-squares problems, Nordisk Tidskr. Informations behandling (BIT) 19, pp. 356—367.

Рухе и Ведин
Ruhe A. and Wedin P. Å. (1980). Algorithms for separable nonlinear least-squares problems, SIAM Review 22, pp. 318—337.

Рамзин и Ведин
Ramsin H. and Wedin P. Å. (1977). A comparison of some algorithms for the nonlinear least-squares problem, Nordisk Tidskr. Informationsbehandling (BIT) 17, pp. 72—90.

Сарджент
Sargent R. W. (1974). «Reduced gradient and projection methods for nonlinear programming», in: Numerical Methods for Constrained Optimization (P. E. Gill and W. Murray, eds.), pp. 149—174, Academic Press, London and New York.

Сарджент и Гаминибандара
Sargent R. W. and Gaminibandara K. (1976). «Optimal design of plate distillation columns», in Optimization In Action (L. C. W. Dixon, ed.), pp. 267—314. Academic Press, London and New York.

Сарджент и Муртаф
Sargent R. W. and Murtagh B. A. (1973). Projection methods for nonlinear programming, Math. Prog. 4, pp. 245—268.

Свон
Swann W. H. (1972). «Direct search methods», In Numerical Methods for Unconstrained Optimization (W. Murray, ed.), pp. 13—28, Academic Press, London and New York.
— (1974). «Constrained optimization by direct search» in Numerical Methods for Constrained Optimization (P. E. Gill and W. Murray, eds.), pp. 191—217, Academic Press, London and New York.

Свифт
Swift F. S. (1981). Elimination of bounds in optimization problems by transforming variables, Math. Prog. 20, pp. 110—121.

Смит, Бейл, Гарбоу, Икебе, Клема и Молер
Smith B. T., Boyle J. M., Garbow B. S., Ikebe Y., Klema V. C. and Moler C. B. (1974). Matrix Eigensystem Routines—EISPACK Guide, Lecture Notes in Computer Science 6, Springer-Verlag, Berlin, Heidelberg and New York.

Сондерс
Saunders M. A. (1976). «A fast, stable implementation of the simplex method using Bartels—Golub updating», in Sparse Matrix Computations (J. R. Bunch and D. J. Rose, eds.), pp. 213—226, Academic Press, New York.
— (1980). Private communication.

Соренсен
Sorensen D. (1980a). Newton's method with a model trust region modification, Report ANL-80-106, Argonne National Laboratory, Argonne, Illinois.
— (1980b). The Q-superlinear convergence of a collinear scaling algorithm for unconstrained minimization, SIAM J. Numer. Anal. 17, pp. 84—114.

Спедикато
Spedicato E. (1975). «On condition numbers of matrices in rank two minimization algorithms», in Towards Global Optimization (L. C. W. Dixon and G. P. Szegö, eds.), pp. 196—210. North-Holland, Amsterdam.

Спендли, Хекст и Химсворт
Spendley W., Hext G. R. and Himsworth F. R. (1962). Sequential application of simplex designs in optimization and evolutionary design, Technometrics 4, pp. 441—461

Стилмен и Виларсен
Stylemen R. S. and Wimarsky N. D. (1979). Adaptive numerical differentiation, Mathematics of Computation 33, pp. 1257—1264.

Стюр
Stoer J. (1971). On the numerical solution of constrained least-squares problems, SIAM J. Numer. Anal. 8, pp. 382—411.
— (1975). On the convergence rate of imperfect minimization algorithms in Broyden's β-class, Math. Prog. 9, pp. 313—335.
— (1977). On the relation between quadratic termination and convergence properties of minimization algorithms, Part I, theory, Num. Math. 28, pp. 343—366.

Стренг
Strang G. (1976). Linear Algebra and its Applications, Academic Press, London and New York. (Имеется перевод: Г. Стренг. Линейная алгебра и ее применения. — М.: Мир, 1980.)

Стюарт
Stewart G. W. (1967). A modification of Davidon's method to accept difference approximations of derivatives, J. ACM 14, pp. 72—83.
— (1973). Introduction to Matrix Computations, Academic Press, London and New York.

Сьярле, Шульц и Варга
Ciarlet P. G., Schultz M. H. and Varga R. S. (1967). Nonlinear boundary value problems I, Num. Math. 9, pp. 394—430.

Тапа
Thapa M. N. (1979). A note on sparse quasi-Newton methods, Report SOL 79-18, Department of Operations Research, Stanford University, California.
— (1980). Optimization of Unconstrained Functions with Sparse Hessian Matrices, Ph. D. Thesis, Stanford University, California.

Тапиа
Tapia R. A. (1974a). Newton's method for problems with equality constraints, SIAM J. Numer. Anal. 11, pp. 174—196.
— (1974b). Newton's method for optimization problems with equality constraints, SIAM J. Numer. Anal. 11, pp. 874—886.
— (1977). Diagonalized multiplier methods and quasi-Newton methods for constrained optimization, J. Opt. Th. Applics. 22, pp. 135—194.
— (1978). «Quasi-Newton methods for equality constrained optimization: equivalence of existing methods and a new implementations», in Nonlinear Programming 3 (O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds.), pp. 125—164, Academic Press, London and New York.

Тарьян
Tarjan R. (1972). Depth-first search and linear graph algorithms, SIAM J. Comput. 1, pp. 146—160.

Тойнт
Toint P. L. (1977). On sparse and symmetric matrix updating subject to a linear equation, Mathematics of Computation 31, pp. 954—961.
— (1978). Some numerical results using a sparse matrix updating formula in unconstrained optimization, Mathematics of Computation 32, pp. 839—851.
— (1979). On the superlinear convergence of an algorithm for solving a sparse minimization problem, SIAM J. Numer. Anal. 16, pp. 1036—1045.

Томлин
Tomlin J. A. (1975a). An accuracy test for updating triangular factors, Math. Prog. Study 4, pp. 142—145.
— (1975b). On scaling linear programming problems, Math. Prog. Study 4, pp. 146—166.
— (1976). Robust implementation of Lemke's method for the linear complementarity problem, Report SOL 76-24, Department of Operations Research, Stanford University.

Топкис и Вайнот
Topkis D. M. and Veinott A. F., Jr. (1967). On the convergence of some feasible direction algorithms for nonlinear programming, SIAM J. Control 5, pp. 268—279.

Уайлд
Wilde D. J. (1978). Globally Optimal Design, John Wiley and Sons, New York and Toronto.

Уилкинсон
Wilkinson J. H. (1963). Rounding Errors in Algebraic Processes, Notes on Applied Sciences 32, Her Majesty's Stationery Office, London; Prentice-Hall, Inc. [also published by Englewood Cliffs, New Jersey].
— (1965). The Algebraic Eigenvalue Problem, Oxford University Press.

Уилкинсон и Райнш
Wilkinson J. H. and Reinsch C. (1971). Handbook for Automatic Computation, Vol. II, Springer-Verlag, Berlin, Heidelberg and New York.

Уилсон
Wilson R. B. (1963). A Simplicial Algorithm for Concave Programming, Ph. D. Thesis, Harvard University.

Уотсон
Watson G. A. (1979). The minimax solution of an overdetermined system of nonlinear equations, J. Inst. Maths. Applics. 23, pp. 167—180.

Фалкерсон и Вулф
Falkerson D. R. and Wolfe P. (1962). An algorithm for scaling matrices, SIAM Review 4, pp. 142—146.

Фиакко
Fiacco A. V. (1976). Sensitivity analysis for mathematical programming using penalty functions, Math. Prog. 10, pp. 287—311.

Фиакко и Мак-Кормик
Fiacco A. V. and McCormick G. P. (1968). Nonlinear Programming: Sequential Unconstrained Minimization Techniques, John Wiley and Sons, New York and Toronto. (Имеется перевод: А. Фиакко, Г. П. Мак-Кормик. Нелинейное программирование. Методы последовательной безусловной минимизации.— М.: Мир, 1972.)

Флетчер
Fletcher R. (1958). Generalized inverse methods for the best least-squares solution of systems of nonlinear equations, Computer Journal 10, pp. 392—399.
— (1970a). A new approach to variable metric algorithms, Computer Journal 13, pp. 317—322.
— (1970b). «A class of methods for nonlinear programming with termination and convergence properties», in Integer and Nonlinear Programming (J. Abadie, ed.), pp. 157—175, North-Holland, Amsterdam.
— (1971a). A modified Marquardt subroutine for nonlinear least squares, Report R6799, Atomic Energy Research Establishment, England.
— (1971b). A general quadratic programming algorithm, J. Inst. Maths. Applics. 7, pp. 76—91.
— (1972a). An algorithm for solving linearly constrained optimization problems, Math. Prog. 2, pp. 133—165.
— (1972b). «Minimizing general functions subject to linear constraints», in Numerical Methods for Non-Linear Optimization (F. A. Lootsma, ed.), pp. 279—296, Academic Press, London and New York.
— (1972c). Methods for the solution of optimization problems, Comput. Phys. Comm. 3, pp. 159—172.
— (1973). An exact penalty function for nonlinear programming with inequalities, Math. Prog. 5, pp. 129—150.
— (1974). «Methods related to Lagrangian functions», in Numerical Methods for Constrained Optimization (P. E. Gill and W. Murray, eds.), pp. 219—240, Academic Press, London and New York.
— (1976). Factorizing symmetric indefinite matrices, Linear Algebra and its Applics. 14, pp. 257—272.
— (1977). «Methods for solving nonlinearly constrained optimization problems», in The State of the Art in Numerical Analysis (D. Jacobs, ed.), pp. 365—446, Academic Press, London and New York.
— (1980). Practical Methods of Optimization, Volume 1, Unconstrained Optimization, John Wiley and Sons, New York and Toronto.

Флетчер и Джексон
Fletcher R. and Jackson M. P. (1974). Minimization of a quadratic function of many variables subject only to upper and lower bounds, J. Inst. Maths. Applics. 14, pp. 159—174.

Флетчер и Лилл
Fletcher R. and Lill S. A. (1970), «A class of methods for non-linear programming: II. computational experience», in Nonlinear Programming (J. B. Rosen, O. L. Mangasarian and K. Ritter, eds.), pp. 67—92, Academic Press, London and New York

Флетчер и Мак-Кавин
Fletcher R. and McCann A. P. (1969). Acceleration techniques for nonlinear programming, in Optimization (R. Fletcher, ed.), pp. 37—49, Academic Press, London and New York.

Флетчер и Ривс
Fletcher R. and Reeves C. M. (1964). Function minimization by conjugate gradients, Computer Journal 7, pp. 149—164.

Флетчер и Пауэлл
Fletcher R. and Powell M. J. D. (1963). A rapidly convergent descent method for minimization, Computer Journal 6, pp. 163—158.
— — (1974). On the modification of LDL^T factorizations, Mathematics of Computation 28, pp. 1067—1087.

Флетчер и Фримен
Fletcher R. and Freeman T. L. (1977). A modified Newton method for minimization, J. Opt. Th. Applics. 23, pp. 357—372.

Форрест и Томлин
Forrest J. J. H. and Tomlin J. A. (1972). Updating triangular factors of the basis to maintain sparsity in the product form simplex method, Math. Prog. 2, pp. 263—278.

Форсайт и Молер
Forsythe G. E. and Moler C. B. (1967). Computer Solution of Linear Algebraic Systems, Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Фоурер
Fourer R. (1979). Sparse Gaussian elimination of staircase linear systems, Report SOL 79-17, Department of Operations Research, Stanford University, California.

Фриш
Frisch K. R. (1955). The logarithmic potential method of convex programming, Memorandum of May 13, 1955, University Institute of Economics, Oslo, Norway.

Хаархофф и Байс
Haarhoff P. C. and Buys J. D. (1970). A new method for the optimization of a nonlinear function subject to nonlinear constraints, Computer Journal 13, pp. 178—184.

Хайес
Hayes J. G. (1970). Numerical Approximation to Functions and Data, Academic Press, London and New York.

Хан
Han S.-P. (1970). Superlinearly convergent variable metric algorithms for general nonlinear programming problems, Math. Prog. 11, pp. 263—282.
— (1977a). Dual variable metric algorithms for constrained optimization, SIAM J. Control and Optimization 15, pp. 546—565.
— (1977b). A globally convergent method for nonlinear programming, J. Opt. Th. Applics. 22, pp. 297—310.
— (1978a). Superlinear convergence of a minimax method, Computer Science Department, Cornell University, Ithaca, New York.

— (1978b). On the validity of a nonlinear programming method for solving minimax problems, Report 1891, Mathematics Research Center, University of Wisconsin, Madison, Wisconsin.

Хан и Мангасарьян
Han S.-P. and Mangasarian O. L. (1979). Exact penalty function in nonlinear programming, Math. Prog. 17, pp. 251—269.

Харрис
Harris P. M. J. (1973). Pivot selection methods of the Devex LP code, Math. Prog. 5, pp. 1—28. [Reprinted in Math. Prog. Study 4 (1975), pp. 30—57.]

Хартли
Hartley H. O. (1961). Nonlinear programming by the simplex method, Econometrica 29, pp. 223—237.

Хачиян Л. Г. (1979). Полиномиальный алгоритм в линейном программировании.— Докл. АН СССР, 1979, т. 244, с. 1093—1096.

Хобден
Hebden M. D. (1973). An algorithm for minimization using exact second derivatives, Report TP515, Atomic Energy Research Establishment, Harwell, England.

Хеллерман и Рарик
Hellerman E. and Rarick D. (1971). Reinversion with the preassigned pivot procedure, Math. Prog. 1, pp. 195—216.
— — (1972). «The partitioned preassigned pivot procedure (P^4)», in Sparse Matrices and their Applications (D. J. Rose and R. A. Willoughby, eds.), pp. 67—76, Plenum Press, New York.

Хестенс
Hestenes M. R. (1940). Sufficient conditions for the isoperimetric problem of Bolza in the calculus of variations, Trans. Amer. Math. Soc. 60, pp. 93—118.
— (1947). An alternative sufficiency proof for the normal problem of Bolza, Trans. Amer. Math. Soc. 61, pp. 256—264.
— (1969). Multiplier and gradient methods, J. Opt. Th. Applics. 4, pp. 303—320.
— (1979). «Historical overview of generalized Lagrangians and augmentability», presented at the IIASA Task Force Meeting on «Generalized Lagrangians in Systems and Economic Theory», IIASA, Laxenburg, Austria (proceedings to be published in 1981).
— (1980a). Conjugate-Direction Methods in Optimization, Springer-Verlag, Berlin, Heidelberg and New York.
— (1980b). Augmentability in optimization theory, J. Opt. Th. Applics. 32, pp. 427—440.

Хестенс и Штифель
Hestenes M. R. and Stiefel E. (1952). Methods of conjugate gradients for solving linear systems, J. Res. Nat. Bur. Standards 49, pp. 409—436.

Хит
Heath M. T. (1978). Numerical Algorithms for Nonlinearly Constrained Optimization, Ph. D. Thesis, Stanford University, California.

Хоув
Howe S. (1973). New conditions for exactness of a simple penalty function, SIAM J. Control 11, pp. 378—381.

Хемминг
Hamming R. W. (1962). Numerical Methods for Scientists and Engineers, McGraw-Hill Book Co., New York.
— (1971). Introduction to Applied Numerical Analysis, McGraw-Hill Book Co., New York.

— (1973). Numerical Methods for Scientists and Engineers (2nd Edition), McGraw-Hill Book Co., New York.

Чарнес
Charnes A. (1952). Optimality and degeneracy in linear programming, Econometrica 20, pp. 160—170.

Чарнес, Купер и Фергюсон
Charnes A., Cooper W. W. and Ferguson R. (1955). Optimal estimation of executive compensation by linear programming, Management Science 2, pp. 138—151.

Чемберлен
Chamberlain R. M. (1979). Some examples of cycling in variable metric methods for constrained minimization, Math. Prog. 16, pp. 378—383.

Чемберлен, Лемарешаль, Педерсон и Пауэлл
Chamberlain R. M., Lemaréchal C., Pederson H. C. and Powell M. J. D. (1980), The watchdog technique for forcing convergence in algorithms for constrained optimization, Report DAMTP 80/NA 1, University of Cambridge.

Шанно
Shanno D. F. (1970). Conditioning of quasi-Newton methods for function minimization, Mathematics of Computation 24, pp. 647—657.
— (1978). Conjugate-gradient methods with inexact searches, Math. of Oper. Res. 3, pp. 244—256.
— (1980). On variable metric methods for sparse Hessians, Mathematics of Computation 34, pp. 499—514.

Шанно и Фуа
Shanno D. F. and Phua K. H. (1976). Algorithm 500—Minimization of unconstrained multivariate functions, ACM Trans. Math. Software 2, pp. 87—94.

Шиттковски
Schittkowski K. (1980). Nonlinear Programming Codes, Springer-Verlag Lecture Notes in Economics and Mathematical Systems, Volume 183, Berlin, Heidelberg and New York.

Шиттковски и Стоер
Schittkowski K. and Stoer J. (1979). A factorization method for the solution of constrained linear least-squares problems allowing data changes, Num. Math. 31, pp. 431—463.

Шор Н. З. (1970). О скорости сходимости метода обобщенного градиентного спуска с растяжением пространства.— Кибернетика, 1970, № 2, с. 80—85.

Шор Н. З. (1977). Метод отсечения с растяжением пространства для решения задач выпуклого программирования.— Кибернетика, 1977, № 1, с. 94—95.

Шор Н. З., Гершович В. И. (1979). Об одном семействе алгоритмов для решения задач выпуклого программирования.— Кибернетика, 1979, № 4, с. 62—67.

Штифель
Stiefel E. (1960). Note on Jordan elimination, linear programming and Tschebyscheff approximation, Num.—Math. 2, pp. 1—17.

Шуберт
Schubert L. K. (1970). Modification of a quasi-Newton method for nonlinear equations with a sparse Jacobian, Mathematics of Computation 24, pp. 27—30.

Эблоу и Брайхем
Ablow C. M. and Brigham G. (1955). An analog solution of programming problems, Operations Research 3, pp. 388—394.

Эванс, Гулд и Толле
Evans J. P., Gould F. J. and Tolle J. W. (1973). Exact penalty functions in nonlinear programming, Math. Prog. 4, pp. 72—97.

Эккер
Ecker J. G. (1980). Geometric programming: methods, computations and applications, SIAM Review 22, pp. 338–362.

Эль-Аттар, Видьясагар и Дутта
El-Attar R. A., Vidyasagar M. and Dutta S. R. K. (1979). An algorithm for l_1-norm minimization with application to nonlinear l_1 approximation, SIAM J. Numer. Anal. 16, pp. 70—86.

Эскудеро
Escudero L. (1980). A projected Lagrangian method for nonlinear programming, Report G320-3410, IBM Palo Alto Scientific Center.

Эспвалл и Стоун
Aspvall B. and Stone R. E. (1980). Khachiyan's linear programming algorithm, Journal of Algorithms 1, 1—13.

Эрвет и Нэш
Jarratt P. and Nash D. (1965). The use of rational functions in the iterative solution of equations on a computer, Computer Journal 8, pp. 62—65.

ОГЛАВЛЕНИЕ