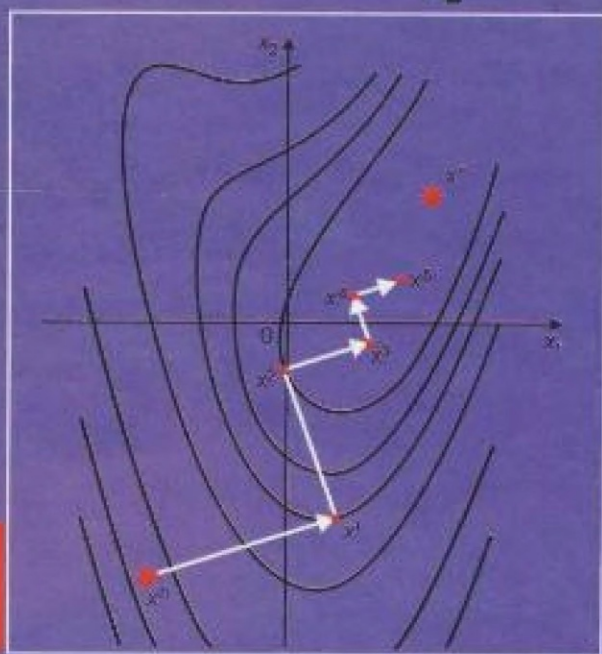


А.В.Аттетков
В.С.Зарубин
А.Н.Канатников

ВВЕДЕНИЕ В МЕТОДЫ ОПТИМИЗАЦИИ



**А.В.Аттетков
В.С.Зарубин
А.Н.Канатников**

ВВЕДЕНИЕ В МЕТОДЫ ОПТИМИЗАЦИИ



**МОСКВА
"ФИНАНСЫ И СТАТИСТИКА"**



**ИЗДАТЕЛЬСКИЙ ДОМ
"ИНФРА-М"**

2008

РЕЦЕНЗЕНТЫ:

А.В. Манжиров,

доктор физико-математических наук, профессор;

В.Ф. Формалев,

Заслуженный деятель науки РФ,

доктор физико-математических наук, профессор

Аттетков А.В.

А92 Введение в методы оптимизации: учеб. пособие / А.В. Аттетков, В.С. Зарубин, А.Н. Канатников. — М.: Финансы и статистика; ИНФРА-М, 2008. — 272 с.: ил.

ISBN 978-5-279-03251-8 (Финансы и статистика)

ISBN 978-5-16-003416-4 (ИНФРА-М)

Освещается одно из важнейших направлений математики — теория оптимизации. Рассмотрены теоретические, вычислительные и прикладные аспекты методов конечномерной оптимизации. Описаны алгоритмы численного решения задач безусловной минимизации функций одного и нескольких переменных, изложены методы условной оптимизации. Приведены примеры решения конкретных задач, дана наглядная интерпретация полученных результатов.

Для студентов, аспирантов и преподавателей технических, экономических и других вузов.

А $\frac{1602090000 - 004}{010(01) - 2008}$ 39 — 2007

УДК 519.863
ББК 22.18

ISBN 978-5-279-03251-8
ISBN 978-5-16-003416-4

© Аттетков А.В., Зарубин В.С.,
Канатников А.Н., 2008

© Издательство «Финансы
и статистика», 2008

ОГЛАВЛЕНИЕ

Предисловие	5
Список принятых обозначений	7
Введение	9
Глава 1.	
Задачи оптимизации	11
1.1. Основные понятия	11
1.2. Примеры задач оптимизации	12
1.3. Классы задач оптимизации	19
Вопросы для самопроверки	23
Глава 2.	
Методы одномерной минимизации	25
2.1. Предварительные замечания	25
2.2. Методы прямого поиска	27
2.3. Сравнение методов прямого поиска	34
2.4. Методы полиномиальной аппроксимации	37
Вопросы для самопроверки	43
Глава 3.	
Многомерная безусловная минимизация	44
3.1. Методы спуска	47
3.2. Метод градиентного спуска	50
3.3. Минимизация квадратичной функции	59
3.4. Метод сопряженных направлений	68
3.5. Метод Ньютона и его модификации	80
3.6. Квазиньютоновские методы	89
3.7. Методы прямого поиска	98
3.8. Методы случайного поиска	119
Вопросы для самопроверки	126

Глава 4.

Аналитические методы нелинейного программирования	128
4.1. Минимизация целевой функции на заданном множестве	128
4.2. Минимизация при ограничениях типа равенства	133
4.3. Общая задача нелинейного программирования	136
4.4. Седловая точка функции Лагранжа	143
4.5. Двойственная функция	145
Вопросы для самопроверки	149

Глава 5.

Численные методы нелинейного программирования	151
5.1. Метод условного градиента	151
5.2. Использование приведенного градиента	158
5.3. Проектирование точки на множество	168
5.4. Метод проекции точки на множество	172
5.5. Метод проекции антиградиента	178
5.6. Метод возможных направлений	199
5.7. Методы последовательной безусловной минимизации	212
Вопросы для самопроверки	221

Глава 6.

Методы линейного программирования	222
6.1. Виды задач линейного программирования	223
6.2. Графический метод решения задач линейного програм- мирования	227
6.3. Основы теории линейного программирования	230
6.4. Симплекс-метод	233
6.5. Построение начального допустимого базисного реше- ния	244
6.6. Двойственная задача линейного программирования .	250
Вопросы для самопроверки	258

Список рекомендуемой литературы	260
--	------------

Предметный указатель	266
---------------------------------------	------------

Предлагаемое пособие посвящено конечномерным задачам оптимизации. В излагаемом материале основное внимание уделено прикладным и вычислительным аспектам, в первую очередь анализу численных методов оптимизации и построению алгоритмов их реализации. В книге опущены доказательства многих теоретических положений, в частности теорем о сходимости методов. По этим вопросам есть обширная литература, и „математические тонкости“, связанные с оценками сходимости методов, можно изучить самостоятельно. В практической деятельности более важно понимание сути методов и алгоритмов их реализации, знание условий их применения. Большое количество примеров и разобранных задач, поясняемых графическими иллюстрациями и интерпретацией полученных результатов, позволяет использовать данное издание не только как учебное пособие, но и как методическое, предназначенное для проведения семинарских занятий и организации лабораторных работ.

В основу книги лег методический опыт авторов, которые в течение ряда лет в стенах МГТУ имени Н.Э. Баумана читают лекции по методам оптимизации, ведут семинарские и лабораторные занятия. Этот опыт нашел свое отражение в учебнике авторов, изданном ранее*. В предлагаемом издании методические наработки авторов получили дальнейшее развитие. Некоторые разделы переработаны. Так, пересмотрена глава по одномерной оптимизации, три главы по методам многомерной оптимизации сокращены и собраны в одну. Глава по оптимизации выпуклых функций не включена в пособие, поскольку этот материал имеет в основном теоретическое значение. Авторы ограничились изложением минимальных сведений по выпуклым функциям в первой главе. В то же время в книгу включен новый материал: методы случайного поиска в третьей главе, новая глава по методам линейного программирования.

* *Аттетков А.В.* Методы оптимизации: учебник для вузов / А.В. Аттетков, С.В. Галкин, В.С. Зарубин; под ред. В.С. Зарубина, А.П. Крищенко. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2001. – 440 с.

Содержание пособия относится к специальным разделам высшей математики, для работы с ним требуется хорошее знание базового курса. В частности, предполагается, что читатель умеет оперировать основными понятиями линейной алгебры, аналитической геометрии, теории матриц и математического анализа.

Мы признательны нашим рецензентам доктору физико-математических наук, профессору А.В. Манжирову и доктору физико-математических наук, профессору В.Ф. Формалеву за ряд замечаний, учтенных нами при подготовке пособия.

СПИСОК ПРИНЯТЫХ ОБОЗНАЧЕНИЙ

- ◀ и ▶ — начало и окончание доказательства
- # — окончание примера, замечания
- $\{a_1, a_2, \dots, a_n\}$ — конечное множество, состоящее из элементов a_1, a_2, \dots, a_n
- $\{x: P(x)\}$ — множество, состоящее из тех и только тех элементов x , которые обладают характеристическим свойством $P(x)$
- $k = \overline{1, N}$ — число k принимает последовательно все значения из множества натуральных чисел от 1 до N включительно
- \mathbb{N} — множество натуральных чисел
- \mathbb{Z} — множество целых чисел
- \mathbb{R} — множество действительных чисел
- \mathbb{R}_+ — множество всех положительных действительных чисел
- \mathbb{R}_* — множество всех неотрицательных действительных чисел
- \mathbb{R}^n — n -я декартова степень множества \mathbb{R} действительных чисел (также n -мерное евклидово арифметическое пространство)
- \mathbb{R}_+^n — положительный ортант, т.е. n -я декартова степень множества \mathbb{R}_+ положительных действительных чисел
- \mathbb{R}_*^n — неотрицательный ортант, т.е. n -я декартова степень множества \mathbb{R}_* неотрицательных действительных чисел
- AB и $|AB|$ — отрезок, соединяющий точки A и B , и его длина
- $x = (x_1 \dots x_n)^T$ — n -мерный арифметический вектор, т.е. элемент n -мерного евклидова арифметического пространства \mathbb{R}^n
- $|x|$ — длина (модуль, евклидова норма) вектора x
- $\sum_{k=1}^n a_k$ — сумма n слагаемых a_1, a_2, \dots, a_n
- $\prod_{m=1}^n a_m$ — произведение n сомножителей a_1, a_2, \dots, a_n
- $D(f)$ и $R(f)$ — область определения и область значений функции $f(x)$

- $\operatorname{sgn} x$ — функция знака числа действительного числа x , т.е. $\operatorname{sgn} x = 1$ при $x > 0$, $\operatorname{sgn} x = 0$ при $x = 0$ и $\operatorname{sgn} x = -1$ при $x < 0$
- $\exp(x)$ — экспоненциальная функция e^x
- $\operatorname{diam} X$ — диаметр ограниченного множества X
- ∂X — граница множества X
- $\sup_{x \in X} f(x)$ и $\inf_{x \in X} f(x)$ — точная верхняя грань и точная нижняя грань функции $f(x)$ на множестве X
- $\max_{x \in X} f(x)$ и $\min_{x \in X} f(x)$ — максимальное и минимальное значения функции $f(x)$ на множестве X
- $f(x+0) = \lim_{x \rightarrow a+0} f(x)$ и $f(x-0) = \lim_{x \rightarrow a+0} f(x)$ — правосторонний и левосторонний пределы функции $f(x)$ одного действительного переменного в точке a
- $\operatorname{grad} f(x)$ — градиент скалярной функции $f(x)$ многих переменных
- (a, b) — скалярное произведение векторов a и b
- A^T — матрица, транспонированная к матрице A
- $\operatorname{Rg} A$ — ранг матрицы A
- $\det A$ — определитель матрицы A
- $c(A)$ — число обусловленности матрицы A
- $\|A\|$ — евклидова норма матрицы A
- I — единичная матрица
- $f(x) \rightarrow \min, x \in \Omega,$ — задача минимизации функции $f(x)$ на множестве Ω
- $f(x) \rightarrow \inf, x \in \Omega,$ — задача нахождения точной нижней грани функции $f(x)$ на множестве Ω
- $a \geq b, b \leq a$ — каждая координата вектора $a \in \mathbb{R}^n$ не меньше соответствующей координаты вектора $b \in \mathbb{R}^n$
- w^k — антиградиент целевой функции в текущей точке x^{k-1} итерационной последовательности $\{x^k\}$
- u^k и β_k — направление спуска и шаг спуска на k -й итерации метода спуска
- $H(x)$ — матрица Гессе скалярной функции многих переменных

Бурное развитие информационных технологий сместило многие акценты в развитии науки. В математике повысилась роль численных методов решения прикладных задач. Эти методы в сочетании с мощностью современной вычислительной техники позволяют решать такие задачи, которые еще 50 лет назад не поддавались исследователям. Широкий класс задач, связанных с применением численных методов, — класс оптимизационных задач.

На практике часто возникает ситуация, когда из нескольких возможных вариантов поведения необходимо выбрать один вариант, в том или ином смысле наилучший. Такой выбор (принятие наилучшего решения) может быть осуществлен по-разному. Один из подходов заключается в количественной оценке каждого возможного варианта поведения (решения) и выборе среди них того, у которого оценка наилучшая (максимальная или минимальная). Так мы приходим к *задаче оптимизации*, которую можно сформулировать следующим образом. Есть некоторое множество возможных решений, называемых *альтернативами*. Каждой альтернативе можно дать некоторую количественную оценку на основе некоторого *критерия оптимальности*. Решение задачи оптимизации состоит в определении той альтернативы, для которой критерий оптимальности дает наибольшую (или наименьшую) количественную оценку.

Задачи оптимизации как задачи выбора наилучшей альтернативы на основе критерия оптимальности возникают в самых различных ситуациях. В этом ряду особенно важна задача проектирования технических устройств и технологических процессов, в которой какие-либо параметры устройства или процесса могут в определенной степени варьироваться, причем за счет изменения этих параметров можно повысить эксплуатационные характеристики устройства или экономичность проведения технологического процесса. Упомянем задачу распределения материальных и финансовых ресурсов, встречающуюся во многих областях экономики и управления. В ряде случаев закон развития процесса или явления можно формулировать как достижение минимума некоторого показателя (например, принцип минимума по-

тенциальной энергии в теоретической механике или принцип Ферма в оптике, согласно которому луч света следует по пути с наименьшим временем движения).

Разнообразны задачи оптимизации и по своему внутреннему содержанию. Множество альтернатив может описываться одним числовым параметром (*одномерная оптимизация*) или несколькими параметрами (*многомерная оптимизация*). Есть класс задач оптимизации, в которых каждая альтернатива характеризуется бесконечным числом параметров (например, задача распределения заряда в твердом теле или задача распределения температуры в некоторой области пространства), хотя в этом случае корректнее говорить не о бесконечном числе параметров, а о функции. Выделяя этот класс задач, говорят о *бесконечномерной оптимизации*, противопоставляя ей *конечномерную оптимизацию*.

В задаче оптимизации может быть несколько критериев оптимальности. Так, в экономике развитие связано с получением наибольшей прибыли, но при принятии решений могут учитываться и другие факторы, например, отражающие экологическую чистоту производства, экономические риски, возникающие из-за противодействия конкурентов, и т.п. В этом случае возникает задача *многокритериальной оптимизации*.

В этом учебном курсе мы остановимся на задачах конечномерной оптимизации с одним критерием оптимальности.

ЗАДАЧИ ОПТИМИЗАЦИИ

1.1. Основные понятия

Если каждая альтернатива в задаче оптимизации характеризуется совокупностью из n числовых параметров, то естественно альтернативы ассоциировать с элементами n -мерного арифметического пространства \mathbb{R}^n . В этом случае множество всех альтернатив будет представлять собой некоторое подмножество $\Omega \subset \mathbb{R}^n$, а критерий оптимальности, который каждой альтернативе ставит в соответствие некоторую числовую оценку, — функцию многих переменных $f(x_1, x_2, \dots, x_n)$, определенную на множестве Ω .

Таким образом, с математической точки зрения задача конечно-мерной оптимизации заключается в определении наибольшего или наименьшего значения функции многих переменных $f(x_1, x_2, \dots, x_n)$ на заданном множестве Ω и точки $x^* \in \Omega$, в которой это значение достигается. Так сформулированную задачу называют *задачей математического программирования*, а функцию $f(x) = f(x_1, x_2, \dots, x_n)$ — *целевой функцией*. Переменные x_1, x_2, \dots, x_n — это *параметры оптимизации*, каждая точка $x = (x_1, x_2, \dots, x_n) \in \Omega$ — *допустимое решение*, а множество $\Omega \subset \mathbb{R}^n$ — множество допустимых решений, или *допустимое множество*. Точку $x^* \in \Omega$, в которой целевая функция принимает наименьшее значение, называют *оптимальным решением*. Задачу математического программирования будем записывать следующим образом:

$$f(x) \rightarrow \min, \quad x \in \Omega.$$

С математической точки зрения нет различий между поиском наибольшего значения функции и поиском наименьшего значения функции. Чтобы преобразовать одну задачу в другую, достаточно сменить знак целевой функции. Учитывая это, в дальнейшем ограничимся рассмотрением только одной из этих двух задач, а именно *задачей минимизации функции*, состоящей в поиске наименьшего значения целевой функции на допустимом множестве и точки, в которой это значение достигается.

Из курса математического анализа известно, что функция на заданном множестве может не достигать наименьшего значения. Причина в том, что она либо не ограничена снизу, либо ограничена снизу, но тем не менее не достигает точной нижней грани множества своих значений. И в том, и в другом случае задача минимизации не имеет решения, что говорит о некорректной ее постановке и о необходимости вносить изменения в математическую модель изучаемого объекта или явления. В некоторых случаях может решаться задача поиска точной нижней грани m_f функции $f(x)$ на допустимом множестве Ω и построения такой последовательности точек $\{x^k\} \subset \Omega$, для которой $f(x^k) \rightarrow m_f$ при $k \rightarrow \infty$. Подобную задачу записывают в виде

$$f(x) \rightarrow \inf, \quad x \in \Omega.$$

Задачу поиска точной нижней грани можно рассматривать как обобщение задачи минимизации. Методы решения такой задачи в целом те же, что и методы решения задачи минимизации. Поэтому мы на ней останавливаться не будем, считая, что исследуемая целевая функция на допустимом множестве достигает наименьшего значения.

1.2. Примеры задач оптимизации

Рассмотрим некоторые задачи оптимизации, возникающие в геометрии, алгебре и других разделах математики. Многие из подобных задач можно решить геометрическим или алгебраическим путем, а также с помощью методов исследования функции на экстремум, изучаемых в курсе математического анализа. Рассматриваемые задачи тесно связаны с историей развития методов оптимизации и позволяют наглядно продемонстрировать многообразие объектов оптимизации — тех устройств, процессов или явлений, при исследовании которых возникают задачи оптимизации. Кроме того, они наглядно показывают, как возникает проблема поиска оптимального решения и как такая проблема превращается в конкретную математическую задачу. Наконец, подходы к решению простейших задач оптимизации являются источниками важнейших идей, лежащих в основе современных методов решения задач оптимизации.

Пример 1.1. Рассмотрим задачу определения сторон прямоугольника, вписанного в окружность радиуса R и имеющего наибольшую площадь S (рис. 1.1).

Эта задача известна с глубокой древности, и ее нетрудно решить геометрическим путем. Диагонали вписанного в окружность прямоугольника являются диаметрами окружности и имеют фиксированную длину. Площадь S прямоугольника равна половине произведения его диагоналей на синус угла φ между ними, т.е. $S = 2R^2 \sin \varphi$. Ясно, что эта площадь будет наибольшей при $\sin \varphi = 1$, или при $\varphi = \pi/2$. В этом случае диагонали прямоугольника перпендикулярны, а сам прямоугольник представляет собой квадрат. Таким образом, среди всех прямоугольников, вписанных в окружность, наибольшую площадь имеет квадрат. Сторона вписанного квадрата равна $R\sqrt{2}$, а площадь — $2R^2$.

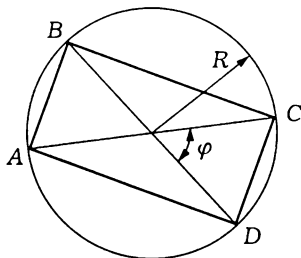


Рис. 1.1

Однако возможны и другие подходы к решению рассматриваемой задачи. В качестве параметров оптимизации выберем длины сторон прямоугольника a и b . Тогда площадь S через параметры оптимизации a и b будет выражаться формулой $S = ab$, а сами параметры a и b должны удовлетворять условиям $\sqrt{a^2 + b^2} = 2R$, $a > 0$, $b > 0$. Мы приходим к следующей задаче минимизации, в которой допустимое множество задано с помощью одного уравнения и двух неравенств:

$$\begin{cases} S = ab \rightarrow \max; \\ a^2 + b^2 = 4R^2, \quad a > 0, \quad b > 0. \end{cases}$$

Отметим, что рассматриваемая задача имеет естественное обобщение: среди всех четырехугольников, вписанных в заданную окружность, найти четырехугольник максимальной площади. Здесь расширено множество допустимых решений (рассматриваются произвольные четырехугольники, вписанные в окружность, а не только прямоугольники). Интуитивно можно понять, что оптимальное решение остается тем же — наибольшую площадь среди всех четырехугольников имеет квадрат. Однако формулировка этой задачи как задачи оптимизации и ее решение заметно сложнее.

Пример 1.2 (задача Евклида). Среди всех параллелограммов, вписанных в заданный треугольник ABC с высотой H и основанием b (рис. 1.2), найдем тот, который имеет наибольшую площадь.

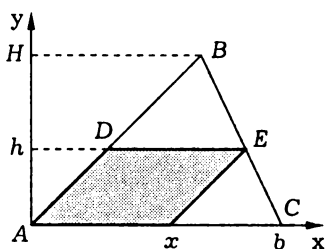


Рис. 1.2

В качестве параметров оптимизации в данном случае можно выбрать длину x стороны параллелограмма, лежащей на стороне AC треугольника, и высоту h параллелограмма, опущенную на эту сторону. Тогда площадь S параллелограмма будет определяться формулой $S = hx$. Чтобы параллелограмм был вписан в треугольник, параметры x и h должны быть связаны равенством $(H - h)/H = x/b$, где H — высота треугольника ABC , опущенная на сторону AC , b — длина стороны AC этого треугольника. Добавляя естественные ограничения $x > 0$, $h > 0$ на параметры оптимизации, приходим к следующей задаче минимизации:

$$\begin{cases} S = hx \rightarrow \max; \\ \frac{H - h}{H} = \frac{x}{b}, \quad h \geq 0, \quad x \geq 0. \end{cases}$$

Пример 1.3. Проведено n измерений d_1, d_2, \dots, d_n расстояния d до некоторого объекта разными наблюдателями. В качестве приближенного значения \tilde{d} этого расстояния выберем то, для которого среднее квадратичное отклонение

$$J(\tilde{d}) = \sqrt{\frac{1}{n} \sum_{i=1}^n (\tilde{d} - d_i)^2}$$

является наименьшим.

Эта задача имеет единственный параметр оптимизации — неизвестное значение \tilde{d} , подчиняющееся естественному условию $\tilde{d} \geq 0$. Задачу несложно решить методами дифференциального исчисления функций одного переменного. Приравнявая нулю производную $J'(\tilde{d})$, получаем линейное уравнение, из которого находим

$$\tilde{d} = \frac{1}{n} \sum_{i=1}^n d_i.$$

Таким образом, искомым приближенным значением оказалось среднее арифметическое результатов измерений. #

При разработке того или иного технического устройства обычно часть его параметров можно изменять в определенных пределах. Образуется некоторое множество вариантов создаваемого устройства. Среди этих вариантов желательно выбрать тот, который имеет наилучшие эксплуатационные характеристики. В результате возникает задача оптимизации, которую часто называют *задачей оптимального проектирования*.

Рассмотрим одну из простых задач оптимального проектирования, состоящую в выборе емкости такой формы, которая при заданной площади поверхности имеет наибольший объем или при заданном объеме — наименьшую площадь поверхности.

Пример 1.4. Пусть требуется спроектировать бак горючего в виде прямого кругового цилиндра заданного объема V , на изготовление которого будет затрачено наименьшее количество листовой стали.

В качестве параметров оптимизации выберем радиус R и высоту H цилиндра. Тогда затраты материала на изготовление бака будут определять суммарная площадь S его боковой поверхности и двух плоских днщ, вычисляемая по формуле $S = 2\pi R(H + R)$. Заданный объем V бака можно вычислить через параметры оптимизации по формуле $\pi R^2 H = V$. Таким образом, мы приходим к задаче минимизации

$$\begin{aligned} 2\pi R(H + R) &\rightarrow \min; \\ \pi R^2 H &= V, \quad R > 0, \quad H > 0. \end{aligned}$$

Чтобы решить поставленную задачу, можно с помощью уравнения, связывающего параметры оптимизации R и H , выразить один из них через другой, например, H через R :

$$H = \frac{V}{\pi R^2}.$$

Исключив из целевой функции параметр H , придем к задаче минимизации функции одного переменного:

$$S(R) = 2\frac{V}{R} + 2\pi R^2.$$

Вычислим производную $S'(R)$ и приравняем нулю:

$$S'(R) = -2\frac{V}{R^2} + 2\pi R = 0.$$

В результате получаем

$$R_* = \sqrt[3]{\frac{V}{2\pi}}.$$

Нетрудно убедиться в том, что R_* — точка локального минимума функции $S(R)$ и что функция $S(R)$ достигает наименьшего значения на интервале $(0, +\infty)$, так как она является непрерывной на этом интервале и бесконечно большой при $R \rightarrow +0$ и при $R \rightarrow +\infty$. Значит, точка R_* является точкой наименьшего значения функции $S(R)$. Зная R_* , находим

$$H_* = \frac{V}{\pi R_*^2} = \sqrt[3]{\frac{4V}{\pi}} = 2R_*.$$

Итак, цилиндрический бак горючего объема V будет иметь наименьшую площадь поверхности в том случае, когда его высота совпадает с диаметром его основания. #

При постановке задачи оптимального проектирования важно точно формулировать цель оптимизации. Например, задача, рассмотренная в примере 1.4, могла появиться как задача уменьшения количества листовой стали, расходуемой на изготовление партии баков горючего. Минимизация площади поверхности отвечает требованию сократить расход материала, но не учитывает всех обстоятельств раскроя материала. Задачу можно поставить по-иному и получить лучшее техническое решение. Для заготовки одного круглого днища площадью πR^2 необходимо использовать квадратный лист площадью $4R^2$, причем после раскроя оставшаяся часть листа, скорее всего, окажется в отходах. Поэтому в качестве целевой следует взять функцию $\tilde{S} = 2\pi RH + 8R^2$. Изменение целевой функции приведет к другому ответу:

$$\tilde{R}_* = \frac{1}{2} \sqrt[3]{V}, \quad \tilde{H}_* = \frac{8}{\pi} \tilde{R}_*.$$

Впрочем, уточнять цель оптимизации можно разными способами. При изготовлении партии баков на одном листе стали стандартного размера может разместиться несколько элементов одного изделия (боковых поверхностей и торцов). Возникает непростая задача оптимального раскроя листового материала.

При постановке задач оптимального проектирования также важно, чтобы математическая модель объекта оптимизации достаточно полно отражала именно те свойства объекта, улучшение которых является

целью оптимизации. Разработка такой модели обычно требует использования сведений из соответствующих инженерных дисциплин или областей техники.

Задачи оптимизации широко распространены в экономике и управлении. Многие из них основаны на сопоставлении расходуемых ресурсов и получаемого экономического результата. Этот подход часто называют методом „затраты — эффективность“. Его использование приводит к задачам оптимизации двух разновидностей: обеспечение максимальной эффективности при заданных затратах; определение минимальных затрат, позволяющих обеспечить эффективность не ниже заданной. Рассмотрим несколько примеров подобных задач.

Пример 1.5. Предположим, что предприятие может выпускать продукцию n наименований, для производства которой требуется m видов ресурсов (сырья, энергии, оборудования и т.п.), причем на изготовление одной единицы продукции j -го наименования, $j = \overline{1, n}$, предприятие затрачивает объем a_{ij} i -го вида ресурсов, $i = \overline{1, m}$. Реализация одной единицы продукции j -го наименования обеспечивает предприятию доход, равный d_j . Требуется так спланировать производство, чтобы, во-первых, объем выпуска продукции j -го наименования был не менее m_j , во-вторых, общий объем расходуемого ресурса i -го вида не превышал b_i , в-третьих, доход от деятельности предприятия был максимальным.

Нами сформулирована простейшая задача оптимального распределения ресурсов. Обозначим через x_j , $j = \overline{1, n}$, планируемый объем выпуска продукции j -го наименования. Тогда суммарный доход S будет равен

$$S(x_1, x_2, \dots, x_n) = \sum_{j=1}^n d_j x_j,$$

а расходуемые объемы ресурсов составляют $\sum_{j=1}^n a_{ij} x_j$. С учетом этого получаем следующую задачу:

$$S(x_1, x_2, \dots, x_n) = \sum_{j=1}^n d_j x_j \rightarrow \max;$$

$$\sum_{j=1}^n a_{ij} x_j \leq b_j, \quad j = \overline{1, n};$$

$$x_j \geq a_j, \quad j = \overline{1, n}.$$

Пример 1.6 (транспортная задача). Предположим, что фирма имеет n магазинов, в которые поступает товар, хранимый на m складах. Известна стоимость c_{ij} перевозки товара с i -го склада в j -й магазин, количество a_i единиц товара на i -м складе и заказанный объем b_j товара для доставки в j -й магазин. Требуется составить план перевозок товара со складов в магазины так, чтобы суммарная стоимость перевозок была минимальной.

Обозначим через x_{ij} количество товара, которое планируется перевезти с i -го склада в j -й магазин. Тогда стоимость перевозки этого товара составит $c_{ij}x_{ij}$, а общая стоимость S всех перевозок будет равна

$$S = \sum_{i=1}^m \sum_{j=1}^n c_{ij}x_{ij}.$$

Магазины должны быть обеспечены товаром в точном соответствии с заказом. Поэтому планируемые объемы перевозок x_{ij} должны удовлетворять условиям

$$\sum_{i=1}^m x_{ij} = b_j, \quad j = \overline{1, n}.$$

Однако с любого склада нельзя вывезти товара больше, чем там его находится. Следовательно, должны быть выполнены условия

$$\sum_{j=1}^n x_{ij} \leq a_i, \quad i = \overline{1, m}.$$

В результате приходим к задаче

$$\begin{aligned} \sum_{i=1}^m \sum_{j=1}^n c_{ij}x_{ij} &\rightarrow \min; \\ \sum_{i=1}^m x_{ij} &= b_j, \quad j = \overline{1, n}; \\ \sum_{j=1}^n x_{ij} &\leq a_i, \quad i = \overline{1, m}; \\ x_{ij} &\geq 0, \quad i = \overline{1, m}, \quad j = \overline{1, n}. \end{aligned}$$

Отметим, что сформулированная задача оптимизации может иметь решение только в том случае, когда сумма заказов всех магазинов не превышает суммарного запаса товара на всех складах, т.е.

$$\sum_{j=1}^n b_j \leq \sum_{i=1}^m a_i.$$

Пример 1.7 (задача о диете). Рассмотрим задачу построения оптимального рациона питания. Обозначим: n — число видов пищевых продуктов; m — число видов питательных веществ; a_{ij} — число единиц i -го питательного вещества в единице j -го продукта; b_i — ежегодная потребность в i -м питательном веществе; c_j — стоимость единицы j -го продукта. Выясним, сколько единиц каждого пищевого продукта нужно употребить за рассматриваемый период (в данном случае за год), чтобы обеспечить потребности в каждом питательном веществе и затратить на это минимальную сумму денег.

Пусть x_j — ежегодное потребление j -го пищевого продукта. Тогда построение рациона минимальной стоимости приводит к задаче оптимизации

$$\begin{aligned} S &= \sum_j^n c_j x_j \rightarrow \min; \\ \sum_j^n a_{ij} x_j &\geq b_i, \quad i = \overline{1, m}; \\ x_j &\geq 0, \quad j = \overline{1, n}. \end{aligned}$$

1.3. Классы задач оптимизации

Предлагаемая классификации задач оптимизации, как и любая другая, в значительной степени условна. В ее основу положены тип целевой функции и характер допустимого множества. Эти особенности задачи оптимизации оказывают решающее влияние на выбор методов ее решения.

Как отмечено выше, с математической точки зрения задача оптимизации представляет собой задачу минимизации целевой функции

$f_0(x)$, $x \in \mathbb{R}^n$, на допустимом множестве $\Omega \subset \mathbb{R}^n$ и записывается следующим образом:

$$f_0(x) \rightarrow \min, \quad x \in \Omega.$$

Эта запись предполагает, что функция $f_0(x)$ определена всюду на допустимом множестве Ω , т.е. область определения целевой функции включает в себя допустимое множество, хотя может и не совпадать с ним.

Если допустимое множество Ω совпадает с \mathbb{R}^n , то на изменение параметров оптимизации (координат точки в \mathbb{R}^n) нет никаких ограничений. В этом случае задачу минимизации называют *задачей безусловной минимизации*. В противном случае, когда $\Omega \neq \mathbb{R}^n$, имеем *задачу условной минимизации*.

Допустимое множество может описываться разными способами. В наиболее типичной ситуации (см. разд. 1.2) допустимое множество описывается системой уравнений и неравенств, связывающих возможные значения параметров оптимизации. При этом уравнения называют *ограничениями типа равенства*, а неравенства — *ограничениями типа неравенства*.

Если целевая функция линейная, то ее область определения совпадает с \mathbb{R}^n и она в \mathbb{R}^n не достигает экстремума. Поэтому в задаче оптимизации с линейной целевой функцией должны быть ограничения. В примерах 1.5–1.7 *ограничения линейные*, т.е. заданы уравнениями и неравенствами с помощью линейных функций. Задачу оптимизации, в которой и целевая функция, и ограничения являются линейными, называют *задачей линейного программирования*. Для решения подобной задачи не годятся стандартные методы исследования функций на экстремум, поскольку линейная целевая функция не имеет локальных экстремумов, а линейный характер ограничений не позволяет привлекать и технику исследования функций на условный экстремум.

Если в задаче минимизации либо целевая функция нелинейная, либо есть *нелинейное ограничение* (т.е. функция в левой части этого ограничения *нелинейная*), то эту задачу называют *задачей нелинейного программирования*.

Задача нелинейного программирования может быть сложной вследствие того, что допустимое множество, описываемое системой ограничений типа равенства и неравенства, может иметь очень сложную

структуру. На это указывает и отсутствие универсального метода решения, пригодного для любой задачи нелинейного программирования. В связи с этим в классе задач нелинейного программирования выделяют более узкие классы задач, допускающих специальные методы решения.

Если в задаче нелинейного программирования все ограничения (и типа равенства, и типа неравенства) линейные, а целевая функция *квадратичная*, т.е. представляет собой многочлен второй степени, то такую задачу называют *задачей квадратичного программирования*. Частный случай подобной задачи — безусловная минимизация квадратичной функции — часто используется для проверки качества используемых методов оптимизации.

Выделим частный случай задачи нелинейного программирования, близкий к задаче квадратичного программирования. Речь идет о задаче, в которой все ограничения линейные, а целевая функция является *дробно-линейной*, т.е. представляет собой отношение двух линейных функций. В этой задаче область определения целевой функции не покрывает \mathbb{R}^n , и, следовательно, задача должна включать ограничения.

В задачах оптимизации интерес представляют не только линейные функции, но и функции вида

$$f(x) = \sum_{j=1}^n \varphi_j(x_j), \quad x = (x_1, x_2, \dots, x_n),$$

равные сумме функций одного переменного, — *сепарабельные функции*. Задачу нелинейного программирования, в которой целевая функция и левые части ограничений являются сепарабельными функциями, называют *задачей сепарабельного программирования*.

В инженерных расчетах нередко встречаются функции вида

$$y(x) = \sum_{i=1}^m c_i p_i(x),$$

где $c_i > 0$, $i = \overline{1, m}$, а функции $p_i(x)$ — мономы

$$p_i(x) = \prod_{j=1}^n x_j^{a_{ij}}$$

с действительными степенями a_{ij} . Такие функции рассматриваются на *положительном ортанте*

$$\mathbb{R}_+^n = \{(x_1, x_2, \dots, x_n): x_j > 0, j = \overline{1, n}\}.$$

и называются *позиномами*.

Позиномы близки к многочленам (полиномам), но отличаются от последних тем, что показатели степени a_{ij} могут быть любыми действительными числами (у многочленов они являются целыми неотрицательными), а коэффициенты c_i всегда положительны (у многочленов коэффициенты — любые действительные числа).

Задачу нелинейного программирования, в которой целевая функция и левые части ограничений являются позиномами, относят к *задачам геометрического программирования*.

Среди задач нелинейного программирования выделяется достаточно широкий класс *задач выпуклого программирования*. В таких задачах допустимое множество Ω является выпуклым множеством, а целевая функция $f_0(x)$ — выпуклой функцией. Дадим определение того и другого.

Множество $X \subset \mathbb{R}^n$ называется *выпуклым*, если для любых точек x и y из X и любого действительного числа $\alpha \in (0, 1)$ выполняется условие $\alpha x + (1 - \alpha)y \in X$. Выпуклые множества обладают важным свойством: пересечение любого семейства таких множеств есть выпуклое множество.

Функция $f(x)$, определенная на выпуклом множестве X , называется *выпуклой функцией*^{*}, если для любых точек x и y из множества X и действительного числа $\alpha \in (0, 1)$ выполняется неравенство

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y).$$

Если это неравенство для любых $x, y \in X$ строгое, то и функцию $f(x)$ называют *строго выпуклой функцией*. Пример такой функции — квадратичная функция с положительно определенной квадратичной формой.

Отметим, что для выпуклой функции $f(x)$, определенной на множестве X , множество $\{x \in X: f(x) \leq c\}$ выпуклое (может быть, пустое). Отсюда, в частности, следует, что если функции $f_1(x)$,

^{*}В математическом анализе в этом случае говорят о функции, выпуклой вниз.

$f_2(x), \dots, f_k(x)$ определены на одном выпуклом множестве $X \subset \mathbb{R}^n$ и являются выпуклыми, то и функция

$$f(x) = \min \{f_1(x), f_2(x), \dots, f_k(x)\}$$

выпуклая.

Для выпуклой функции любая точка локального минимума есть точка наименьшего значения. Для выпуклой и дифференцируемой функции любая **стационарная точка** (точка с нулевым значением градиента) есть точка наименьшего значения. Строго выпуклая функция может иметь только одну точку наименьшего значения. **Сильно выпуклая функция**, т.е. функция, которая для любых точек $x, y \in X$ удовлетворяет условию

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y) - \alpha(1 - \alpha)\gamma|x - y|^2,$$

с некоторой постоянной $\gamma > 0$, всегда достигает наименьшего значения.

Особняком стоят задачи, в которых параметры оптимизации могут принимать дискретное, в частности конечное, множество значений. Такие задачи относят к **задачам дискретного программирования**. Среди них — **задачи целочисленного программирования**, в которых параметры оптимизации могут принимать лишь целочисленные значения.

Вопросы для самопроверки

1. Сформулируйте задачу математического программирования. Что в задаче математического программирования называют допустимым решением, а что — оптимальным решением?

2. Может ли в задаче нелинейного программирования целевая функция быть линейной?

3. Чем задача безусловной оптимизации отличается от задачи условной оптимизации?

4. Может ли задача линейного программирования быть в то же время задачей целочисленного программирования?

5. Что такое целевая функция? Какой вид целевая функция имеет в задаче выпуклого программирования?

6. Дайте пример задачи математического программирования, не имеющей решения. Можно ли утверждать, что задача математического программирования имеет решение, если допустимое множество замкнуто и ограничено?

7. Приведите пример задачи квадратичного программирования, не имеющей решения. Можно ли утверждать, что задача квадратичного программирования имеет решение, если допустимое множество замкнуто и ограничено?

8. Приведите пример задачи математического программирования, в которой стационарная точка не является решением задачи. Всегда ли решение задачи математического программирования является стационарной точкой целевой функции?

МЕТОДЫ ОДНОМЕРНОЙ МИНИМИЗАЦИИ

2.1. Предварительные замечания

Остановимся на задаче одномерной оптимизации. В этом случае целевая функция есть функция одного действительного переменного, а допустимое множество представляет собой некоторое подмножество числовой оси. Наиболее важным с практической точки зрения является случай, когда допустимое множество есть числовой промежуток, т.е. интервал, полуинтервал или отрезок.

Напомним, что функция, непрерывная на отрезке, достигает на этом отрезке наибольшего и наименьшего значения. Поэтому задача минимизации в случае, когда допустимое множество есть отрезок, а целевая функция непрерывна на этом отрезке, всегда имеет решение. Задача минимизации функции одного переменного может не иметь решения, если:

- допустимое множество есть отрезок, но на этом отрезке целевая функция имеет точки разрыва;
- допустимое множество есть интервал или полуинтервал.

Например, функция $f(x)$, (строго) убывающая на полуинтервале $[a, b)$, не достигает наименьшего значения на этом полуинтервале, хотя при этом она может быть непрерывной и ограниченной снизу. Та же функция, доопределенная в точке b значением $f(a)$, не достигает наименьшего значения и на отрезке $[a, b]$.

Функция может достигать наименьшего значения как в единственной точке, так и на некотором множестве точек, конечном, счетном или несчетном. Например, функция $f(x) = x^4$, рассматриваемая на отрезке $[-1, 1]$, достигает наименьшего значения 0 в единственной точке $x_* = 0$, в то время как функция $g(x) = 1 - x^4$ на том же отрезке достигает своего наименьшего значения 0 в двух точках: -1 и 1 . Функция $h(x) = x^2 \sin^2 \frac{1}{x}$, доопределенная в точке $x = 0$ значением 0, достигает

наименьшего значения 0 в счетном числе точек, а именно в точках $x_k = \frac{1}{k\pi}$, $k = \pm 1, \pm 2 \dots$, и в точке $x_0 = 0$.

В теории одномерной оптимизации выделен важный класс функций, которые на отрезке достигают наименьшего значения и притом в единственной точке. Функцию $f(x)$ называют **унимодальной функцией** на отрезке $[a, b]$, если существует такая точка $x_* \in [a, b]$, что функция $f(x)$ на отрезке $[a, x_*]$ убывает, а на отрезке $[x_*, b]$ возрастает. Точка x_* может быть внутренней точкой отрезка $[a, b]$ (т.е. $a < x_* < b$) или совпадать с одним из его концов. Унимодальная функция может иметь точки разрыва, но таких точек не более чем счетное множество и все они будут точками разрыва I рода. Ясно, что унимодальная функция $f(x)$ достигает наименьшего значения в рубежной точке x_* . Примеры графиков унимодальных функций приведены на рис. 2.1.

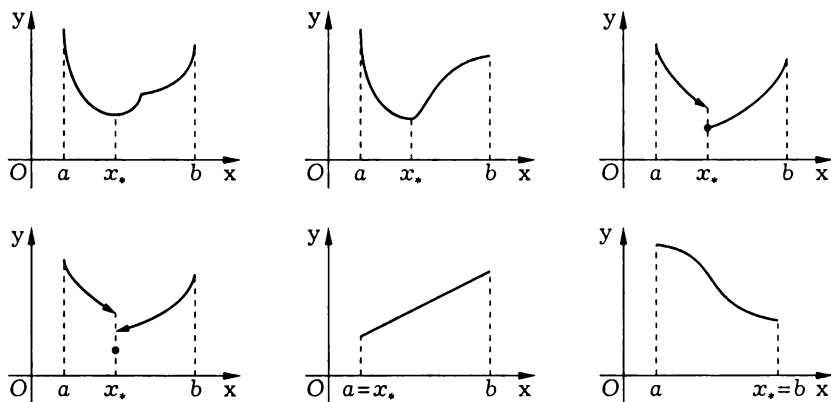


Рис. 2.1

Существуют разные методы решения задач одномерной оптимизации. Если функция дифференцируема на отрезке, то можно, используя необходимые и достаточные условия локального экстремума, определить все точки локального минимума, добавить к ним концы отрезка и простым перебором среди этих точек найти ту, в которой функция достигает наименьшего значения. Однако на практике нередки ситуации, когда либо функция не является дифференцируемой, либо неизвестно, является ли она дифференцируемой, либо функция дифференцируема, но трудно вычислить ее производные. В таких ситуациях применяют **методы прямого поиска**, характеризующиеся тем, что в них не

используются производные функции, а вычисляются лишь значения самой функции.

Другие методы можно классифицировать по порядку используемых производных: в методах первого порядка вычисляются производные первого порядка, но не второго и более высоких порядков, в методах второго порядка вычисляются производные не выше второго порядка и т.д.

2.2. Методы прямого поиска

Можно выделить две группы методов прямого поиска, обусловленные двумя принципиально различными подходами к решению задачи:

1) методы *пассивного поиска* — точки, в которых вычисляются значения функции для определения ее наименьшего значения, выбирают заранее, т.е. до вычисления значений функции в этих точках;

2) методы *последовательного поиска* — точки, в которых вычисляются значения функции, выбирают последовательно исходя из уже выбранных точек и значений функции в них.

Последовательный поиск в общем случае эффективнее пассивного. В этом можно убедиться, вспомнив детскую игру, в которой надо найти спрятанную вещь, задавая вопросы и получая на них ответы „да“ или „нет“. Задавая вопросы последовательно с учетом предыдущих ответов, можно найти спрятанную вещь быстрее, чем задав определенное количество заранее подготовленных вопросов. Однако методы пассивного поиска проще в реализации.

Эффективность метода прямого поиска в задаче одномерной оптимизации определяется количеством значений функции, которые нужно вычислить для достижения требуемой точности. Количество выполненных итераций характеризует не объем вычислительной работы, а лишь ее организацию (что и в каком порядке вычисляется).

Эффективность метода — один из критериев, учитываемых при выборе метода для решения конкретной задачи. Другой критерий — простота реализации метода. В простейших задачах оптимизации эффективность может быть менее важным критерием, чем простота реализации. Например, для поиска наименьшего значения элементарной функции вполне допустимо использовать какой-либо вариант пассивного поиска, проигрывая при этом в эффективности, но выигрывая в простоте метода. Ситуация меняется, если вычисление

значения функции является трудоемкой операцией, связанной, например, с решением системы дифференциальных уравнений. Тогда на первый план выходит эффективность метода, а простота реализации становится вторичным фактором.

В методах последовательного поиска обычно используют один из следующих двух подходов. Первый подход состоит в построении последовательности точек, сходящейся к искомой точке минимума целевой функции (это общий подход, характерный для задач многомерной оптимизации). Второй подход состоит в построении последовательности вложенных отрезков, каждый из которых содержит точку минимума целевой функции. Если последовательность длин отрезков стремится к нулю, то эти отрезки будут стягиваться к точке минимума. Этот подход возможен только в задачах одномерной оптимизации и имеет дополнительные преимущества: в качестве приближенного значения точки минимума можно взять произвольную точку очередного отрезка из последовательности вложенных отрезков, при этом длина отрезка будет определять точность приближения. В теории одномерной оптимизации каждый отрезок из последовательности вложенных отрезков, стягивающихся к точке минимума, называют *интервалом неопределенности*.

Второй подход к последовательному поиску реализуется в наиболее распространенных методах одномерной оптимизации. Основой построения последовательности вложенных отрезков, стягивающихся к точке минимума функции, является так называемая *процедура исключения отрезка*, вытекающая из следующего утверждения.

Теорема 2.1. Пусть функция $f(x)$, унимодальная на отрезке $[a, b]$, достигает минимума в точке $x_* \in [a, b]$, а точки c и d принадлежат отрезку $[a, b]$, причем $c < d$. Если $f(c) \leq f(d)$, то $x_* \in [a, d]$, а если $f(c) \geq f(d)$, то $x_* \in [c, b]$.

◀ Если $f(c) \leq f(d)$, то точки c и d не могут обе располагаться левее рубежной точки x_* унимодальной функции $f(x)$, так как в противном случае они попадают на отрезок $[a, x_*]$, на котором функция $f(x)$ убывает, а следовательно, будет выполняться неравенство $f(c) > f(d)$. Значит, по крайней мере точка d расположена правее точки x_* , т.е. $d > x_*$, а точка x_* оказывается на отрезке $[a, d]$.

Если $f(c) \geq f(d)$, теорема доказывается аналогичным образом. ►

Пусть функция $f(x)$ унимодальна на отрезке $[a, b]$ и выделен интервал неопределенности $[a_k, b_k]$, содержащий точку минимума x_* . Вы-

берем на интервале (a_k, b_k) произвольные точки c_k и d_k , $c_k < d_k$. Если $f(c_k) \leq f(d_k)$, то в соответствии с доказанной теоремой $x_* \in [a_k, d_k]$, и в качестве нового интервала неопределенности $[a_{k+1}, b_{k+1}]$ можно выбрать отрезок $[a_k, d_k]$. Если же $f(c_k) \geq f(d_k)$, то $x_* \in [c_k, b_k]$, и в качестве нового интервала неопределенности можно выбрать отрезок $[c_k, b_k]$. В результате исходный отрезок $[a_k, b_k]$, содержащий точку минимума, будет заменен меньшим отрезком $[a_{k+1}, b_{k+1}]$. Это и есть процедура исключения отрезка.

Отметим случай $f(c_k) = f(d_k)$. Нетрудно показать, что тогда $x_* \in [c_k, d_k]$, т.е. в качестве нового интервала неопределенности можно выбрать отрезок $[c_k, d_k]$, меньший, чем дает процедура исключения отрезка. Однако в распространенных методах прямого поиска такое уточнение процедуры исключения отрезка не дает заметных преимуществ. Равенство значений функции трактуют либо как частный случай неравенства „меньше или равно“, либо как частный случай противоположного неравенства.

Процедура исключения отрезка позволяет строить последовательность вложенных отрезков, каждый из которых содержит искомую точку минимума функции. Конкретные методы последовательного поиска, связанные с построением последовательности вложенных отрезков, различаются способом выбора точек c_k и d_k .

Какой из отрезков $[a_k, d_k]$ и $[c_k, b_k]$ будет выбран на очередном шаге построения последовательности интервалов неопределенности, заранее неизвестно: это зависит от поведения минимизируемой функции. Поэтому целесообразно выбирать точки c_k и d_k на интервале (a_k, b_k) симметричным образом, т.е. так, чтобы отрезки $[a_k, d_k]$ и $[c_k, b_k]$ имели одинаковую длину. В этом случае $d_k - a_k = b_k - c_k$, что равносильно равенству $a_k + b_k = c_k + d_k$. Методы прямого поиска, основанные на принципе симметрии точек c_k и d_k , называют **симметричными методами прямого поиска**. Рассмотрим два таких метода.

Метод дихотомии. В последовательности вложенных отрезков, построенной с помощью процедуры исключения отрезка с симметричным выбором внутренних точек, длина очередного отрезка превышает половину длины предыдущего. Напрашивается такой способ выбора точек, при котором точки c_k и d_k на очередном отрезке $[a_k, b_k]$ расположены максимально близко к середине этого отрезка. Этот способ положен в основу одного из распространенных методов последовательного поиска, называемого **методом дихотомии** (или методом деления пополам).

В методе дихотомии в качестве параметра используют достаточно малое число $\Delta > 0$ (это число должно быть меньше точности вычисления точки минимума). На k -м шаге вычислений на отрезке $[a_k, b_k]$ выбирают точки $c_k = (a_k + b_k)/2 - \Delta$ и $d_k = (a_k + b_k)/2 + \Delta$, отстоящие от середины $(a_k + b_k)/2$ отрезка $[a_k, b_k]$ на расстояние Δ (рис. 2.2).

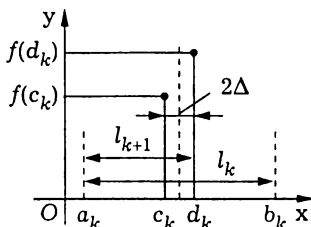


Рис. 2.2

Полагая $l_k = b_k - a_k$, $k \in \mathbb{Z}$, из очевидных геометрических соображений получаем формулу $2l_{k+1} - 2\Delta = l_k$, $k \in \mathbb{Z}$, связывающую длины двух последовательных интервалов неопределенности. Из этой формулы находим $l_{k+1} - 2\Delta = (l_k - 2\Delta)/2$, откуда

$$l_k - 2\Delta = \frac{l_1 - 2\Delta}{2^{k-1}}. \quad (2.1)$$

Из равенства (2.1) выводим следующую формулу длины интервала неопределенности, получаемого на k -м шаге метода дихотомии:

$$l_k = \frac{l_1 - 2\Delta}{2^{k-1}} + 2\Delta. \quad (2.2)$$

Из равенства (2.2) следует, что $l_k > 2\Delta$, причем $l_k \rightarrow 2\Delta$ при $k \rightarrow \infty$. Рассматривая величину l_k как удвоенную точность приближенного решения задачи минимизации, полученную на k -м шаге (при выборе в качестве приближения середины интервала неопределенности), делаем вывод, что параметр Δ дает оценку снизу погрешности приближения в методе дихотомии. В этой ситуации естественно желание выбирать как можно меньшим параметр Δ . Однако на самом деле его нельзя уменьшать неограниченно. Дело в том, что значения функции, как правило, вычисляются приближенно. При очень малом параметре Δ точки c_k и d_k на отрезке $[a_k, b_k]$ будут расположены настолько близко друг к другу, что не удастся достоверно выяснить,

какое из значений функции является большим: различие в значениях функции будет покрываться погрешностью вычислений.

Метод золотого сечения. Метод дихотомии, как видно из его описания, обеспечивает наименьшее количество итераций, необходимых для достижения заданной точности. При этом на каждой итерации метода вычисляются два значения функции (в точках c_k и d_k). Однако эффективность метода прямого поиска характеризуется не количеством итераций, а количеством вычисленных значений функции. Существуют методы, в которых на каждой итерации вычисляется только одно значение функции. Количество итераций в них больше, чем в методе дихотомии, но общее количество вычисленных значений функции может быть меньше.

Можно заметить, что процедура исключения отрезка, примененная к отрезку $[a_k, b_k]$, дает новый отрезок $[a_{k+1}, b_{k+1}]$, который содержит внутри себя одну из точек c_k, d_k (вторая является одним из концов нового отрезка). В связи с этим возникает идея использовать эту точку повторно, с тем чтобы сократить количество вычисляемых значений целевой функции. В результате возникает следующая схема вычислений.

На начальном отрезке $[a_1, b_1]$ выбираем точку x_1 и симметричную ей точку $y_1 = a_1 + b_1 - x_1$. Полагаем $c_1 = \min\{x_1, y_1\}$ и $d_1 = \max\{x_1, y_1\}$. Проводим процедуру исключения отрезка в соответствии с выбранными точками c_1 и d_1 и определяем новый отрезок $[a_2, b_2]$. Если будет выбран отрезок $[a_1, d_1]$, то внутри него будет располагаться точка c_1 , которую используем в качестве точки x_2 . Если же будет выбран отрезок $[c_1, b_1]$, то в качестве x_2 используем точку d_1 . После выбора точки x_2 переходим ко второму шагу вычислений и проводим процедуру исключения отрезка $[a_2, b_2]$.

В описанной схеме на каждом шаге необходимо вычислить всего лишь одно значение функции. Но при этом последовательность длин вложенных отрезков устроена сложнее, чем в методе дихотомии. В общем случае трудно оценить, как быстро эта последовательность стремится к нулю. Кроме того, возможна ситуация, когда на очередном k -м шаге точка, сохраненная с предыдущего шага, окажется в середине отрезка $[a_k, b_k]$. В такой ситуации не удастся выбрать вторую точку для проведения процедуры исключения отрезка и необходимо выбирать новые точки.

Описанных выше проблем можно избежать, если определенным образом выбрать начальную точку x_1 . Более того, начальную точку x_1

можно выбрать так, что в последовательности $\{l_k\}$ длин вложенных отрезков отношение l_{k+1}/l_k будет оставаться постоянным. Оказывается, что при таком выборе точка x_1 будет осуществлять *золотое сечение отрезка* $[a_1, b_1]$, т.е. такое его деление на две неравные части, при котором отношение длины всего отрезка к длине его большей части равно отношению длины большей части к длине меньшей.

В самом деле, имеем $l_1 = b_1 - a_1$. Пусть для определенности точка x_1 расположена правее середины отрезка $[a_1, b_1]$, т.е. $d_1 = x_1$. В результате выполнения первого шага будет выбран новый отрезок $[a_2, b_2]$, по длине равный $l_2 = x_1 - a_1$ (рис. 2.3). Пусть, например, выбран отрезок $[a_2, b_2] = [a_1, x_1]$. Тогда $x_2 = c_1 = y_1$. Выполнение второго шага будет зависеть от того, как точка $x_2 = y_1$ расположена по отношению к середине отрезка $[a_1, x_1]$. Здесь возможны два случая. В первом случае точка y_1 находится правее середины отрезка $[a_1, x_1]$, т.е. $y_1 - a_1 > x_1 - y_1$, и на втором шаге будет выбран отрезок длины $l_3 = y_1 - a_1$. Во втором случае точка y_1 находится левее середины отрезка $[a_1, x_1]$, т.е. $y_1 - a_1 < x_1 - y_1$, и на втором шаге будет выбран отрезок длины $l_3 = x_1 - y_1$. Рассмотрим каждый из случаев подробнее.

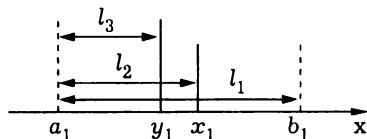


Рис. 2.3

В первом случае равенство $l_3/l_2 = l_2/l_1$ принимает вид

$$\frac{y_1 - a_1}{x_1 - a_1} = \frac{x_1 - a_1}{b_1 - a_1}.$$

Исключив из этого равенства y_1 с помощью уравнения $x_1 + y_1 = a_1 + b_1$, получим

$$\frac{b_1 - x_1}{x_1 - a_1} = \frac{x_1 - a_1}{b_1 - a_1},$$

откуда, вводя обозначение $\tau = \frac{x_1 - a_1}{b_1 - a_1}$, приходим к уравнению

$$\frac{1 - \tau}{\tau} = \tau, \quad (2.3)$$

соответствующему золотому сечению отрезка длины 1 (величина τ составляет большую часть этого сечения, а величина $1 - \tau$ — меньшую часть). Уравнение (2.3) сводится к квадратному уравнению $\tau^2 + \tau - 1 = 0$, имеющему единственное положительное решение

$$\tau = \frac{\sqrt{5} - 1}{2} \approx 0,618034.$$

Это решение мы будем называть *коэффициентом золотого сечения**. Отметим, что если x_1 осуществляет золотое сечение отрезка $[a_1, b_1]$, т.е. $x_1 - a_1 = \tau(b_1 - a_1)$, то

$$x_1 - a_1 < \frac{2}{3}(b_1 - a_1), \quad y_1 - a_1 = b_1 - a_1 - x_1 - a_1 > \frac{1}{3}(b_1 - a_1)$$

и

$$x_1 - y_1 = 2(x_1 - a_1) - (b_1 - a_1) < \frac{1}{3}(b_1 - a_1) < y_1 - a_1,$$

т.е. золотое сечение соответствует первому случаю при выполнении второго шага.

Во втором случае равенство $l_3/l_2 = l_2/l_1$ принимает вид

$$\frac{x_1 - y_1}{x_1 - a_1} = \frac{x_1 - a_1}{b_1 - a_1},$$

что после исключения y_1 приводит к равенству

$$\frac{2(x_1 - a_1) - (b_1 - a_1)}{x_1 - a_1} = \frac{x_1 - a_1}{b_1 - a_1},$$

или

$$\frac{2\tau - 1}{\tau} = \tau.$$

Последнее уравнение имеет единственное решение $\tau = 1$, которое не может быть использовано для выбора точки x_1 : необходимо значение τ из интервала $(0, 1)$.

Итак, если точка x_1 осуществляет золотое сечение начального отрезка $[a_1, b_1]$, то одна из точек x_1 и y_1 окажется внутри следующего отрезка $[a_2, b_2]$, причем эта точка (точка x_2) будет осуществлять золотое сечение отрезка $[a_2, b_2]$. Повторяя рассуждения, заключаем, что на каждом k -м шаге точка x_k будет осуществлять золотое сечение отрезка $[a_k, b_k]$.

*В литературе наряду с τ используют и обратную величину $1/\tau = \frac{\sqrt{5}+1}{2} \approx 1,618034$, называемую *отношением золотого сечения*.

Метод последовательного поиска, в котором на k -м шаге каждая из точек c_k, d_k осуществляет золотое сечение отрезка $[a_k, b_k]$, называют **методом золотого сечения**. В этом методе отношение очередного интервала неопределенности к предыдущему равно $\tau = \frac{\sqrt{5}-1}{2}$. Для длин l_k интервалов неопределенности выполняются соотношения

$$l_k = \tau^k l_1. \quad (2.4)$$

На каждом шаге метода, кроме первого, вычисляется одно значение функции (на первом шаге требуется вычислить два значения функции в точках x_1 и y_1).

2.3. Сравнение методов прямого поиска

Напомним, что эффективность методов прямого поиска оценивают по количеству значений целевой функции, которые нужно вычислить для достижения заданной точности. Сравним простейший метод пассивного поиска, метод дихотомии и метод золотого сечения по данному показателю. Для этого будем определять точность найденного приближения в зависимости от количества вычисленных значений функции, предполагая, что отрезок $[a, b]$ минимизации имеет единичную длину, т.е. $l_1 = b - a = 1$.

Ни один из методов прямого поиска не дает гарантированного результата для любой функции, определенной на отрезке минимизации. Действительно, в каждом методе прямого поиска результат получают, сравнивая значения функции в некотором конечном числе точек. Несложно придумать функцию, даже дифференцируемую, имеющую на заданном конечном множестве точек заданные значения и достигающую наименьшего значения в заданной точке. Для такой функции рассматриваемый метод прямого поиска даст неверный результат.

О точности и эффективности того или иного метода можно говорить в рамках какого-либо класса функций. Для метода дихотомии и метода золотого сечения можно ограничиться классом унимодальных функций, поскольку именно для таких функций можно утверждать, что каждый отрезок в последовательности вложенных отрезков, построенных любым из этих методов, содержит точку минимума функции. Процедура исключения отрезка позволяет для унимодальных функций оценить и точность простейшего пассивного поиска, при котором

начальный отрезок минимизации $[a, b]$ делится на N равных частей точками $x_k = a + k\delta_N$, $k = \overline{0, N}$, где $\delta_N = \frac{b-a}{N}$, и среди этих точек ищется точка x_{k^*} с наименьшим значением. Действительно, из неравенства $f(x_{k^*}) \leq f(x_{k^*} + \delta_N)$ в соответствии с процедурой исключения отрезка заключаем, что точка минимума x_* находится на отрезке $[a, x_{k^*} + \delta_N]$, а из неравенства $f(x_{k^*}) \leq f(x_{k^*} - \delta_N)$ делаем вывод, что $x_* \in [x_{k^*} - \delta_N, b]$. Таким образом,

$$x_{k^*} - \delta_N \leq x_* \leq x_{k^*} + \delta_N$$

и точка x_{k^*} является приближением искомой точки минимума x_* с точностью δ_N .

Итак, в простейшем методе пассивного поиска $N + 1$ вычисленное значение функции обеспечивает точность

$$\delta_N^p = \frac{b-a}{N} = \frac{1}{N}.$$

В методе дихотомии за k шагов вычисляют $2k$ значений функции и получают приближение с точностью l_k , определяемой формулой (2.1). Следовательно, точность δ_N^d метода дихотомии в зависимости от количества вычисляемых значений функции определяется формулой

$$l_N^d = \frac{1-2\Delta}{2^{N/2}} + 2\Delta \approx \frac{1}{2^{N/2}}. \quad (2.5)$$

В методе золотого сечения за k шагов вычисляют $k + 1$ значение функции и получают приближение с точностью l_k , определяемой формулой (2.4). Из этой формулы вытекает, что точность δ_N^g этого метода зависит от количества вычисляемых значений функции следующим образом:

$$l_N^g = \tau^{N-1}. \quad (2.6)$$

Из приведенных формул видно, что метод золотого сечения эффективнее метода дихотомии, так как $\delta_N^g = o(\delta_N^d)$ при $N \rightarrow \infty$, а метод дихотомии эффективнее простейшего метода пассивного поиска, поскольку $\delta_N^p = o(\delta_N^d)$ при $N \rightarrow \infty$. В методах дихотомии и золотого сечения погрешность приближения при возрастании N убывает в геометрической прогрессии, причем знаменатель геометрической прогрессии в методе золотого сечения $\tau = 0,618034$ меньше знаменателя геометрической прогрессии в методе дихотомии $1/\sqrt{2} \approx 0,707107$. В

простейшем методе пассивного поиска погрешность при росте количества значений функции N убывает заметно медленнее, обратно пропорционально N .

Пример 2.1. Для функции $f(x) = 100(x - 0,24)^2$ на отрезке $[0, 1]$ найдем точку минимума x_* с точностью $\varepsilon_* = 0,1$, используя методы дихотомии и золотого сечения.

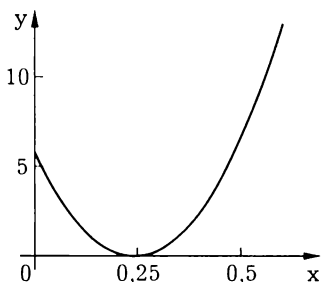


Рис. 2.4

Функция $f(x)$ на отрезке $[0, 1]$ унимодальна. График этой функции показан на рис. 2.4.

Для метода дихотомии выберем параметр $\Delta = 10^{-3}$. Результаты вычислений по этому методу представлены в табл. 2.1, а по методу золотого сечения — в табл. 2.2. Для достижения требуемой точности в методе дихотомии потребовалось вычислить восемь значений минимизируемой функции,

в то время как в методе золотого сечения оказалось достаточно шести значений функции. Отметим, что в простейшем методе пассивного поиска для достижения нужной точности требуется 10 значений функции.

Таблица 2.1

k	Интервал неопределенности	c_k	$f(c_k)$	d_k	$f(d_k)$
1	(0,000, 1,000)	0,499	6,7081	0,501	6,8121
2	(0,000, 0,501)	0,250	0,0090	0,252	0,0132
3	(0,000, 0,252)	0,125	1,3283	0,127	1,2826
4	(0,125, 0,252)	0,187	0,2796	0,189	0,2588
5	(0,187, 0,252)				

Таблица 2.2

k	Интервал неопределенности	c_k	$f(c_k)$	d_k	$f(d_k)$
1	(0,000, 1,000)	0,382	2,0154	0,618	14,2910
2	(0,000, 0,618)	0,236	0,0015	0,382	2,0154
3	(0,000, 0,382)	0,146	0,8855	0,236	0,0015
4	(0,146, 0,382)	0,236	0,0015	0,292	0,2683
5	(0,146, 0,292)	0,202	0,1473	0,236	0,0015
6	(0,202, 0,292)				

2.4. Методы полиномиальной аппроксимации

Наименьшее значение функции одного переменного можно искать, заменяя функцию приближенно (аппроксимируя) некоторым многочленом. В этом заключается суть совокупности методов, называемых *методами полиномиальной аппроксимации*. Различаются эти методы степенью аппроксимирующего многочлена и способами вычисления его коэффициентов. При построении аппроксимирующего многочлена могут использоваться не только значения самой функции, но и ее производных.

Метод квадратичной аппроксимации. В этом методе аппроксимирующий многочлен второй степени определяется по трем известным значениям f_1, f_2, f_3 минимизируемой функции в трех точках x_1, x_2, x_3 .

Чтобы определить коэффициенты a, b, c многочлена второй степени $ax^2 + bx + c$, который в трех различных точках x_1, x_2, x_3 принимает заданные значения f_1, f_2, f_3 , нужно найти решение системы линейных уравнений

$$ax_i^2 + bx_i + c = f_i, \quad i = 1, 2, 3.$$

Определитель этой системы

$$\begin{vmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ x_3^2 & x_3 & 1 \end{vmatrix} = (x_1 - x_2)(x_1 - x_3)(x_2 - x_3)$$

представляет собой *определитель Вандермонда** и отличен от нуля, когда x_1, x_2, x_3 попарно различны. В этом случае система линейных уравнений имеет решение, и притом единственное. Его можно записать в виде

$$\begin{aligned} a &= \frac{f_1}{(x_1 - x_2)(x_1 - x_3)} + \frac{f_2}{(x_2 - x_1)(x_2 - x_3)} + \frac{f_3}{(x_3 - x_1)(x_3 - x_2)}; \\ b &= \frac{f_1(x_2 + x_3)}{(x_1 - x_2)(x_3 - x_1)} + \frac{f_2(x_3 + x_1)}{(x_2 - x_1)(x_3 - x_2)} + \frac{f_3(x_1 + x_2)}{(x_3 - x_1)(x_2 - x_3)}; \\ c &= \frac{f_1 x_2 x_3}{(x_1 - x_2)(x_1 - x_3)} + \frac{f_2 x_3 x_1}{(x_2 - x_1)(x_2 - x_3)} + \frac{f_3 x_1 x_2}{(x_3 - x_1)(x_3 - x_2)}. \end{aligned}$$

*Вандермонд А.Т. (1735–1796) — французский математик.

Подставив найденные выражения для коэффициентов a и b в необходимое условие $y' = 2ax + b = 0$ экстремума функции, получим единственную стационарную точку аппроксимирующего многочлена

$$\tilde{x}_* = -\frac{b}{2a} = \frac{1}{2} \frac{f_1 r_{23} + f_2 r_{31} + f_3 r_{12}}{f_1 s_{23} + f_2 s_{31} + f_3 s_{12}}, \quad (2.7)$$

где $r_{ij} = x_i^2 - x_j^2$ и $s_{ij} = x_i - x_j$, $i, j = 1, 2, 3$. Так как $y'' = a = \text{const}$, то в точке \tilde{x}_* при $a > 0$ имеем минимум аппроксимирующего многочлена $y(x)$, а при $a < 0$ — максимум.

Существуют различные модификации метода квадратичной аппроксимации, различающиеся способом выбора на каждом k -м шаге трех точек $x_1^{(k)}$, $x_2^{(k)}$, $x_3^{(k)}$ для аппроксимации целевой функции. В простейшем варианте, найдя точку минимума $x_*^{(k)}$ аппроксимирующего многочлена на k -м шаге, на следующем шаге используют эту точку и две точки предыдущего шага (например, ближайшие к точке $x_*^{(k)}$). Можно также к точке $x_*^{(k)}$ добавлять ближайшую точку предыдущего шага и новую точку, симметричную добавленной точке относительно точки $x_*^{(k)}$. К недостаткам таких модификаций метода квадратичной аппроксимации следует отнести трудности с определением точности полученного приближения, поскольку они не приводят к построению последовательности вложенных интервалов неопределенности. Процесс вычислений останавливают, если значения $f(x_*^{(k)})$ и $f(x_*^{(k+1)})$, полученные на двух шагах, окажутся близки.

Для функции, унимодальной на отрезке минимизации, метод квадратичной аппроксимации можно сочетать с процедурой исключения отрезка. На начальном отрезке $[a_1, b_1]$ полагаем $x_1^{(1)} = a$, $x_3^{(1)} = b$, а точку $x_2^{(1)}$ в интервале $[a, b]$ выберем произвольно. По формуле (2.7) находим точку минимума $x_*^{(1)}$ аппроксимирующего многочлена. Определив $c_1 = \min \{x_2^{(1)}, x_*^{(1)}\}$, $d_1 = \max \{x_2^{(1)}, x_*^{(1)}\}$, проводим процедуру исключения отрезка, получая новый интервал неопределенности $[a_2, b_2]$. В этот интервал попадает одна из точек $x_2^{(1)}$ и $x_*^{(1)}$. Ее обозначаем через $x_2^{(2)}$ и переходим к следующему шагу.

В описанном методе предполагается, что на k -м шаге найденная точка минимума $x_*^{(k)}$ аппроксимирующего многочлена попадает в интервал неопределенности $[a_k, b_k]$. Однако в общем случае это не так.

Например, функция $f(x) = x^3$ является унимодальной на отрезке $[1, 3]$. Выбрав $x_1 = 1$, $x_2 = 2$, $x_3 = 3$, по формуле (2.7) получим $\tilde{x}_* = \frac{11}{12} < 1$, т.е. точка \tilde{x}_* оказалась вне интервала $[1, 3]$.

При реализации приведенного метода необходимо на каждом шаге проверять условие $x_*^{(k)} \in [a_k, b_k]$. Если это условие оказалось невыполненным, то вместо точки $x_*^{(k)}$ можно взять произвольную точку отрезка $[a_k, b_k]$ (как на первом шаге).

Пример 2.2. Найдём методом квадратичной аппроксимации с процедурой исключения отрезка минимум функции $f(x) = x^2 + 16/x$ на отрезке $[1, 4]$.

На указанном отрезке функция $f(x)$ унимодальна. График этой функции приведен на рис. 2.5, расчеты — в табл. 2.3.

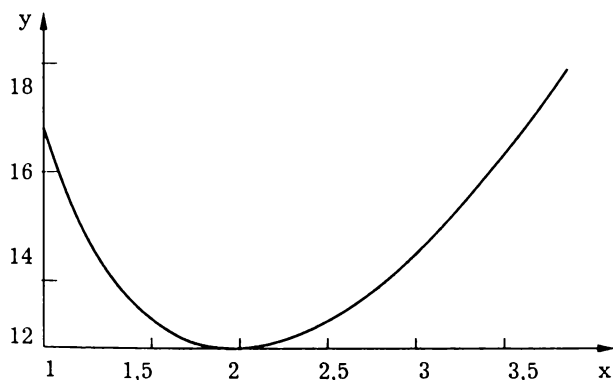


Рис. 2.5

Таблица 2.3

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$f_1^{(k)}$	$f_2^{(k)}$	$f_3^{(k)}$	$x_*^{(k)}$	$f(x_*^{(k)})$
1	1,000	3,000	4,000	17,000	14,333	20,000	2,286	12,224
2	1,000	2,286	3,000	17,000	12,224	14,333	2,200	12,113
3	1,000	2,200	2,286	17,000	12,113	12,224	2,087	12,022
4	1,000	2,087	2,200	17,000	12,022	12,113	2,054	12,009
5	1,000	2,054	2,087	17,000	12,009	12,022	2,027	12,002
6	1,000	2,027	2,054	17,000	12,002	12,009	2,016	12,001
7	1,000	2,016	2,027	17,000	12,001	12,002	2,009	12,000
8	1,000	2,009	2,016	17,000	12,000	12,001	2,005	12,000

Проведенные расчеты показывают, что в методе квадратичной интерполяции, даже если мы получим последовательность вложенных интервалов неопределенности, последовательность длин этих интервалов может не стремиться к нулю. В рассматриваемом случае последовательность длин интервалов неопределенности стремится к единице, и мы не можем адекватно оценить точность полученного решения. Отметим, что точное значение точки минимума функции нетрудно найти, используя необходимое условие локального экстремума. В результате получим $x_* = 2$ и $f(x_*) = 12$. #

Метод квадратичной аппроксимации хорошо „работает“, если точки, в которых вычисляют значения функции, расположены близко к точке минимума функции, а сама функция хорошо аппроксимируется многочленом. Поэтому метод квадратичной интерполяции целесообразно применять для уточнения приближенного решения, найденного другим методом, например, методом золотого сечения.

Эффективность метода квадратичной аппроксимации зависит от поведения функции в окрестности точки минимума. Если целевая функция дифференцируема, то ее производная в окрестности точки минимума мала, а поведение функции в соответствии с формулой Тейлора определяется квадратичным слагаемым. Это обстоятельство и положено в основу метода квадратичной аппроксимации.

Метод кубической аппроксимации. В этом методе целевая функция аппроксимируется многочленом третьей степени, причем для построения аппроксимирующего многочлена используются значения не только самой целевой функции, но и ее производной. Рассмотрим вариант этого метода, основанный на *кубическом интерполяционном многочлене Эрмита**.

Запишем произвольный многочлен третьей степени в следующем виде:

$$H(x) = a_0 + a_1(x - x_1) + a_2(x - x_1)(x - x_2) + a_3(x - x_1)^2(x - x_2).$$

Полагая, что $H(x_1) = f_1$, $H(x_2) = f_2$, находим $a_0 = f_1$, $a_0 + a_1(x_2 - x_1) = f_1$. Отсюда $a_1 = (f_2 - f_1)/(x_2 - x_1)$.

Продифференцируем $H(x)$:

$$H'(x) = a_1 + a_2(x - x_1) + a_2(x - x_2) + a_3(x - x_1)^2 + 2a_3(x - x_1)(x - x_2).$$

*Эрмит Ш. (1822–1901) — французский математик.

Полагая, что $H'(x_1) = f'_1$, $H'(x_2) = f'_2$, получаем два уравнения:

$$a_1 + a_2(x_1 - x_2) = f'_1; \quad a_1 + a_2(x_2 - x_1) + a_3(x_2 - x_1)^2 = f'_2.$$

Решая эти уравнения, приходим к следующим формулам для коэффициентов a_0, a_1, a_2, a_3 интерполяционного многочлена:

$$\begin{cases} a_0 = f_1; \\ a_1 = \frac{f_2 - f_1}{x_2 - x_1} = k; \\ a_2 = \frac{f_2 - f_1}{(x_2 - x_1)^2} - \frac{f'_1}{x_2 - x_1} = \frac{k - f'_1}{x_2 - x_1}; \\ a_3 = \frac{f'_1 + f'_2}{(x_2 - x_1)^2} - 2\frac{f_2 - f_1}{(x_2 - x_1)^3} = \frac{f'_1 + f'_2 - 2k}{(x_2 - x_1)^2}, \end{cases} \quad (2.8)$$

где $k = (f_2 - f_1)/(x_2 - x_1)$.

В качестве приближения к точке минимума целевой функции выбираем точку минимума многочлена $H(x)$. Эта точка может быть найдена как решение уравнения $H'(x) = 0$. Учитывая значения коэффициентов интерполяционного многочлена Эрмита, получаем

$$H'(x) = f'_1(x - x_1)^2 + f'_2(x - x_2)^2 + 2(f'_1 + f'_2 - 3k)(x - x_1)(x - x_2). \quad (2.9)$$

Из двух корней квадратного уравнения необходимо выбрать меньший, если коэффициент $f'_1 + f'_2 - 2k$ при x^2 положителен, и больший, если этот коэффициент отрицателен.

Пусть на отрезке $[x_1, x_2]$ целевая функция унимодальна, причем $f'_1 < 0$, $f'_2 > 0$. Тогда многочлен Эрмита $H(x)$ на интервале (x_1, x_2) имеет единственную точку локального минимума x_3 . Если $f'(x_3) = 0$, то x_3 будет стационарной точкой целевой функции $f(x)$, которая в силу условия унимодальности будет точкой ее локального минимума. Если $f'(x_3) < 0$, то выбираем отрезок $[x'_1, x'_2] = [x_3, x_2]$, на котором $f'(x'_1) < 0$, $f'(x'_2) > 0$. Если $f'(x_3) > 0$, то выбираем отрезок $[x'_1, x'_2] = [x_1, x_3]$, на котором $f'(x'_1) < 0$, $f'(x'_2) > 0$. На выбранном отрезке снова можно аппроксимировать целевую функцию многочленом Эрмита, уточняя положение точки минимума. В результате получаем итерационную процедуру, приводящую к построению

последовательности интервалов неопределенности, стягивающихся к точке локального минимума целевой функции.

Рассмотрим порядок вычисления точки x_3 локального многочлена на очередном интервале неопределенности. В уравнении $H'(x) = 0$, записанном в виде (2.9), вводим новое переменное $\mu = \frac{x - x_1}{x_2 - x_1}$. Получаем уравнение

$$f'_2 \mu^2 - 2(f'_1 + f'_2 - 3k)\mu + f'_1 = 0,$$

имеющее корни разных знаков, причем положительный корень соответствует корню уравнения $H'(x) = 0$, попадающему в интервал (x_1, x_2) . Этот корень выражается формулой

$$\mu = \frac{\gamma + \sqrt{\gamma^2 - f'_1 f'_2}}{f'_2},$$

где $\gamma = f'_1 + f'_2 - 3k$. Используя значение μ , вычисляем $x_3 = \frac{x_2 \mu + x_1}{\mu + 1}$.

Пример 2.3. Для функции $f(x) = x^2 + 16/x$ из примера 2.2 рассмотрим процесс поиска точки минимума x_* на отрезке $[1, 4]$ с помощью метода кубической аппроксимации.

Имеем $f_1 = f(1) = 17$; $f_2 = f(4) = 20$. По формуле $f'(x) = 2x - 16/x^2$ получаем $f'_1 = f'(1) = -14$; $f'_2 = f'(4) = 7$. Отсюда видно, что с помощью метода кубической интерполяции можно построить последовательность интервалов неопределенности, стягивающихся к точке минимума целевой функции.

Для отрезка $[x_1, x_2] = [1, 4]$ находим $k = 1$; $\gamma = -10$; $\mu \approx 0,5816$; $x_3 \approx 2,1032$. Так как $f'(x_3) \approx 0,5893 > 0$, то в качестве нового интервала неопределенности выбираем $[x_1^{(1)}, x_2^{(1)}]$, где $x_1^{(1)} = x_1 = 1$ и $x_2^{(1)} = x_3 \approx 2,1032$.

Для выбранного отрезка получаем $f_1 = 17$; $f_2 \approx 12,0309$; $f'_1 = -14$; $f'_2 \approx 0,5893$. Выполнив вычисления, находим $x_3^{(1)} \approx 1,9209$. Поскольку $f'(x_3^{(1)}) \approx -0,4946 < 0$, в качестве нового интервала неопределенности выбираем $[x_1^{(2)}, x_2^{(2)}]$, где $x_1^{(2)} = x_3^{(1)} \approx 1,9209$; $x_2^{(2)} = x_2^{(1)} \approx 2,1032$.

На третьем шаге имеем $f_1 \approx 12,0193$; $f_2 \approx 12,0309$; $f'_1 \approx -0,4946$; $f'_2 \approx 0,5893$. Выполнив вычисления, получаем $x_3^{(2)} \approx 2,00003$, что отличается от точного значения $x_* = 2$ лишь в пятом знаке после запятой.

Однако отметим при этом, что очередной интервал неопределенности $[x_1^{(3)}, x_2^{(3)}]$, где $x_1^{(3)} = x_1^{(2)} \approx 1,9209$; $x_2^{(3)} = x_3^{(2)} \approx 2,00003$, имеет длину $x_2^{(3)} - x_1^{(3)} \approx 0,079$, которая гарантирует точность лишь в пределах $0,079/2 \approx 0,04$.

Вопросы для самопроверки

1. Имеет ли функция $f(x) = xe^{-x}$ экстремум в интервале $(0, 3)$? Если имеет, то в какой точке? Имеет ли она минимум в том же интервале, минимум на отрезке $[0, 3]$, и если да, то в какой точке?

2. Являются ли унимодальными: а) функция $f(x) = x^2 - 2x - 1$ на отрезках $[0, 2]$ и $[1,5, 2]$; б) функция $f(x) = ||x-1| - 1|$ на отрезках $[-3, 3]$ и $[0, 2]$?

3. Объясните, в чем различие между пассивным и последовательным поиском.

4. Что называют интервалом неопределенности в задачах одномерной оптимизации?

5. Какое отношение имеет метод золотого сечения к золотому сечению?

6. В чем состоит метод дихотомии?

7. Опишите метод квадратичной аппроксимации. Какие трудности возникают при его реализации?

8. Каким образом сравнивают эффективность методов прямого поиска?

9. Ищется минимум функции $f(x) = 1 - xe^{-x^2}$ на отрезке $[0, 1]$. Определите количество значений функции, требуемых для достижения точности 0,01, если используется: а) метод дихотомии; б) метод золотого сечения. Как изменится количество вычисленных значений функции, если: а) целевую функцию $f(x)$ заменить функцией $g(x) = 1 - x^2e^{-x}$; б) отрезок $[0, 1]$ заменить отрезком $[-1, 2]$?

МНОГОМЕРНАЯ БЕЗУСЛОВНАЯ МИНИМИЗАЦИЯ

Задачи многомерной оптимизации, как правило, значительно сложнее, чем задачи одномерной оптимизации. С ростом числа переменных возрастают объемы вычислений. Но дело не только и не столько в объеме вычислений: поведение функции многих переменных более сложное, чем поведение функции одного переменного. Кроме того, допустимое множество в многомерном случае может иметь сложную структуру. Эти два фактора затрудняют предварительный анализ задачи оптимизации.

В задачах безусловной оптимизации допустимое множество совпадает со всем линейным арифметическим пространством соответствующей размерности, а основным фактором, определяющим сложность решаемой задачи, становится поведение целевой функции. Большие различия в поведении функций многих переменных порождают и большое количество методов безусловной оптимизации. Общего метода поиска наименьшего значения функции многих переменных, который можно было бы использовать во всех случаях, нет. Приступая к решению задачи безусловной оптимизации, нужно провести предварительный анализ целевой функции и на его основании выбрать один из численных методов, который в заданных условиях может обеспечить решение задачи.

Задачу безусловной минимизации можно записать в виде

$$f(x) \rightarrow \min, \quad x \in \mathbb{R}^n, \quad (3.1)$$

где $f(x)$ — целевая функция, в данном случае некоторая функция многих переменных, определенная в \mathbb{R}^n . Эта задача имеет решение, если функция $f(x)$ имеет хотя бы одну **точку минимума**, т.е. точку, в которой она достигает наименьшего значения. Точка минимума может быть не единственной.

Методы безусловной оптимизации можно разделить на три больших класса в зависимости от используемой информации о целевой функции.

1. *Методы нулевого порядка*, или *прямого поиска*, в которых используются значения целевой функции, но не значения ее частных производных.

2. *Методы первого порядка*, в которых наряду со значениями целевой функции используются и значения ее частных производных первого порядка.

3. *Методы второго порядка*, в которых используются значения самой целевой функции и ее частных производных до второго порядка включительно.

Методы первого и второго порядков хорошо разработаны. Большинство из них теоретически обоснованы, получены условия сходимости и оценки скорости сходимости для определенных классов целевых функций. Возможности методов прямого поиска менее изучены, как правило, они носят эвристический характер и не имеют теоретического обоснования. В то же время они просты в реализации, что предопределяет их широкое применение.

Общей чертой многих численных методов решения задач оптимизации является построение такой последовательности $\{x^k\}$ точек в \mathbb{R}^n , что значения $f_k = f(x^k)$, $k \in \mathbb{N}$, целевой функции в точках последовательности удовлетворяют неравенствам $f_k \leq f_{k-1}$, $k \in \mathbb{N}$. Подобная последовательность может строиться по-разному, но, как правило, ее строят, вычисляя очередную точку x^k путем выполнения некоторой последовательности операций, в которой используются ранее вычисленные точки x^i , $i < k$. Первая точка последовательности (начальная точка), а иногда и несколько первых точек (в зависимости от алгоритма) могут выбираться произвольно.

Упомянутая последовательность операций составляет одну *итерацию* в численном методе, а сам численный метод называют итерационным. Последовательность $\{x^k\}$ часто называют *итерационной последовательностью*, отражая в названии итерационный характер ее построения. Однако в задачах оптимизации более существенным является убывание значений функции в точках последовательности $\{x^k\}$, т.е. выполнение условия $f_k \leq f_{k-1}$, $k \in \mathbb{N}$. Если это условие выполнено, то последовательность $\{x^k\}$ будем называть *релаксационной последовательностью*. Численные методы оптимизации, основанные на построении такой последовательности, относят к классу *методов спуска*. Это название можно связать с тем, что, например, при минимизации функции $f(x_1, x_2)$ двух переменных уменьшение ее значения при переходе от точки (x_1^{k-1}, x_2^{k-1}) к точке (x_1^k, x_2^k) означает

спуск с линии уровня $f(x_1, x_2) = f_{k-1}$ на линию уровня $f(x_1, x_2) = f_k < f_{k-1}$.

Строя релаксационную последовательность с помощью того или иного метода оптимизации, мы рассчитываем на то, что получится последовательность, сходящаяся к точке минимума целевой функции. Если это так, то в качестве приближенного решения задачи оптимизации можно выбрать один из членов релаксационной последовательности с достаточно большим номером (отдельный вопрос — как определить такой номер; ответить на него можно, исследовав скорость сходимости последовательности). Однако ни один из численных методов, используемых на практике, не гарантирует того, что релаксационная последовательность, построенная с помощью этого метода, будет сходить к точке минимума целевой функции. В реальности может получиться релаксационная последовательность, которая либо вообще не сходится, либо сходится к точке, не являющейся точкой минимума минимизируемой функции.

Многие методы построения релаксационной последовательности рассчитаны на то, что эта последовательность сходится к стационарной точке функции, т.е. точке, в которой градиент функции обращается в нуль. Контролировать сходимость последовательности к стационарной точке проще, чем к точке минимума, так как для этого можно проверять лишь абсолютную величину градиента функции в точках последовательности. В самом деле, если последовательность $\{x^k\}$ сходится, а последовательность $\{\text{grad } f(x^k)\}$ градиентов непрерывно дифференцируемой функции стремится к нулю, то пределом последовательности $\{x^k\}$ является стационарная точка функции. Отметим, что для выпуклой функции любая стационарная точка является точкой минимума, но в общем случае это, конечно, неверно.

Последовательность $\{f_k\}$ значений функции в точках релаксационной последовательности убывающая, а потому сходится к конечному или бесконечному пределу. Конечность предела последовательности $\{f_k\}$ можно гарантировать, если, например, целевая функция $f(x)$ ограничена снизу. Но даже если последовательность $\{f_k\}$ имеет конечный предел, нельзя утверждать, что этот предел совпадает с наименьшим значением функции.

Пусть для заданной целевой функции $f(x)$ рассматриваемый численный метод приводит к релаксационной последовательности $\{x^k\}$, для которой последовательность значений $f_k = f(x^k)$, $k \in \mathbb{N}$, сходится к наименьшему значению функции f_* . Тогда говорят о *слабой сходимости* применяемого метода. При слабой сходимости релаксационная

последовательность может и не сходиться. Если рассматриваемый численный метод дает релаксационную последовательность, сходящуюся к какой-либо точке минимума целевой функции, то говорят о *сильной сходимости* этого метода (или алгоритма).

Описанные обстоятельства указывают на трудности, возникающие в решении задачи безусловной оптимизации. Выбрав тот или иной численный метод, необходимо убедиться в сходимости релаксационной последовательности, построенной с помощью этого метода, и проанализировать скорость сходимости этой последовательности (или хотя бы определиться с тем, как выбирать член релаксационной последовательности, обеспечивающий заданную точность решения задачи). В свою очередь теоретическое обоснование конкретного численного метода включает в себя выделение класса целевых функций, для которых метод дает релаксационную последовательность, сходящуюся к точке минимума, и оценки скорости сходимости релаксационной последовательности.

Построение релаксационной последовательности в методах спуска обычно прекращают, если на очередной k -й итерации выполнено одно или оба неравенства

$$|x^k - x^{k-1}| < \varepsilon_1, \quad |f(x^k) - f(x^{k-1})| < \varepsilon_2, \quad (3.2)$$

где ε_1 и ε_2 — заданные достаточно малые положительные числа, называемые *параметрами точности поиска*. В методах первого и второго порядка в качестве условия прекращения вычислений используют также неравенство

$$|\text{grad } f(x^{k-1})| < \varepsilon_3. \quad (3.3)$$

Все три условия называют *условиями останова*. После прекращения вычислений в качестве приближенного решения задачи минимизации выбирают последний найденный член релаксационной последовательности.

3.1. Методы спуска

Процесс построения релаксационной последовательности $\{x^k\}$ в методах спуска обычно описывают рекуррентным соотношением

$$x^k = x^{k-1} + \beta_k u^k, \quad k \in \mathbb{N}, \quad (3.4)$$

где $u^k \in \mathbb{R}^n$ — единичный вектор, определяющий *направление спуска* на k -й итерации; $\beta_k \geq 0$ — *шаг спуска*, т.е. расстояние от точки x^{k-1} до новой точки x^k . Построение последовательности $\{x^k\}$ начинается с выбора начальной точки x^0 , входящей в правую часть равенства (3.4) при $k = 1$.

Методы спуска различаются способами выбора направления и шага спуска. Будем говорить, что вектор u задает направление спуска в точке x для функции $f(x)$, если существует такое число $\delta > 0$, что $f(x + \beta u) < f(x)$ при $0 < \beta < \delta$, т.е. достаточно малое перемещение из точки x в заданном направлении приводит к уменьшению значения функции. Для функции $f(x)$, дифференцируемой в точке x , вектор u задает направление спуска тогда и только тогда, когда производная по направлению этого вектора отрицательна, т.е. когда он удовлетворяет условию

$$(\text{grad } f(x), u) < 0. \quad (3.5)$$

Геометрически условие (3.5) означает, что вектор u в точке x образует тупой угол с градиентом функции $f(x)$.

Шаг спуска β_k при заданном направлении спуска u^k удобно выбирать так, чтобы выполнялось неравенство

$$f(x^{k-1} + \beta_k u^k) \leq (1 - \lambda_k) f(x^{k-1}) + \lambda_k \min_{\beta \in \mathbb{R}_+} f(x^{k-1} + \beta u^k), \quad (3.6)$$

где $\lambda_k \in [0, 1]$. Для этого используют различные эвристические приемы. Выбор значения β_k , при котором выполняется условие (3.6), обеспечивает выполнение неравенства $f(x^{k-1} + \beta_k u^k) \leq f(x^{k-1})$, так что последовательность $\{x^k\}$, построенная в соответствии с соотношением (3.4), будет релаксационной.

При $\lambda_k = 0$ неравенство (3.6) переходит в неравенство

$$f(x^{k-1} + \beta_k u^k) \leq f(x^{k-1}), \quad (3.7)$$

т.е. в условие релаксационности последовательности $\{x^k\}$.

При $\lambda_k = 1$ неравенство (3.6) переходит в равенство, а значение β_k^* , обеспечивающее выполнение этого равенства, соответствует выбору точки минимума функции $f(x)$ при изменении x вдоль луча, исходящего из точки x^{k-1} в направлении вектора u^k , т.е. на множестве

$$\{x \in \mathbb{R}^n: x = x^{k-1} + \beta u^k, \beta \in \mathbb{R}_+\}.$$

Для определения β_k^* необходимо решить задачу одномерной минимизации. Значение β_k^* может определяться неоднозначно, и в этом случае можно выбрать любое из возможных значений, желательно наименьшее.

Рассмотрим промежуточный случай $\lambda_k \in (0, 1)$. Из неравенства

$$f(x^{k-1}) - f(x^k) \geq \lambda_k (f(x^{k-1}) - f(x^{k-1} + \beta_k^* u^k)), \quad (3.8)$$

равносильного неравенству (3.6), заключаем, что параметр λ_k характеризует наименьшую долю из максимально возможного уменьшения функции в направлении u^k , которую должна обеспечивать релаксационная последовательность. Введение параметра λ_k позволяет избежать полного решения задачи одномерной оптимизации (как в случае $\lambda_k = 1$), ограничиваясь некоторой упрощенной процедурой.

Направление спуска в точке x для функции $f(x)$ можно определить с помощью **антиградиента** этой функции, т.е. вектора

$$w(x) = -\operatorname{grad} f(x).$$

В то время как градиент определяет направление наибольшего возрастания функции, антиградиент, наоборот, задает направление наибольшего убывания этой функции. Далее при построении релаксационных последовательностей на k -й итерации используем обозначения $w^k = -\operatorname{grad} f(x^{k-1})$ для антиградиента в точке x^{k-1} и

$$\alpha_k = \frac{(w^k, u^k)}{|w^k|} = -\frac{(\operatorname{grad} f(x^{k-1}), u^k)}{|\operatorname{grad} f(x^{k-1})|} \quad (3.9)$$

для косинуса угла между направлениями спуска и антиградиента. Отметим, что этот угол должен быть острым, если вектор u^k задает направление спуска. При $\alpha_k = 1$ единичный вектор u^k сонаправлен антиградиенту w^k и может быть получен нормировкой последнего, т.е. $u^k = \frac{w^k}{|w^k|}$. Такой выбор вектора u^k на каждой итерации характерен для **метода градиентного спуска** (хотя его точнее надо было бы назвать методом антиградиентного спуска). Если при этом на каждой итерации $\lambda_k = 1$ (т.е. спуск осуществляется в точку минимума функции в заданном направлении спуска), то метод называют **методом наискорейшего спуска**.

3.2. Метод градиентного спуска

Для метода градиентного спуска основное рекуррентное соотношение (3.4) можно записать в виде

$$\mathbf{x}^k = \mathbf{x}^{k-1} + \beta_k \frac{\mathbf{w}^k}{|\mathbf{w}^k|}, \quad k \in \mathbb{N}, \quad (3.10)$$

где $\mathbf{x}^k, k = 0, 1, \dots$, — элементы релаксационной последовательности; $\beta_k > 0$ — шаг спуска на k -й итерации; $\mathbf{u}^k = \frac{\mathbf{w}^k}{|\mathbf{w}^k|}$ — единичный вектор, который задает направление спуска, определяемое антиградиентом $\mathbf{w}^k = -\text{grad } f(\mathbf{x}^{k-1})$ в точке \mathbf{x}^{k-1} .

Существуют разные способы выбора шага спуска в методе градиентного спуска. Рассмотрим один из вариантов метода, в котором шаг спуска пропорционален длине вектора антиградиента, т.е. $\beta_k = \kappa |\mathbf{w}^k|$, $k \in \mathbb{N}$, $\kappa = \text{const} > 0$. В этом варианте рекуррентное соотношение (3.10) упрощается*:

$$\mathbf{x}^k = \mathbf{x}^{k-1} + \kappa \mathbf{w}^k, \quad k \in \mathbb{N}. \quad (3.11)$$

Один из недостатков метода градиентного спуска с рекуррентным соотношением (3.11) состоит в том, что в окрестности стационарной точки $\tilde{\mathbf{x}}$ дифференцируемой функции $f(\mathbf{x})$ очередной шаг спуска может оказаться чрезмерно большим, превышая расстояние от текущей точки до стационарной, т.е. метод может „проскакать“ точку минимума. Более того, этот шаг может быть настолько большим, что приведет к неравенству $f(\mathbf{x}^k) > f(\mathbf{x}^{k-1})$ и последовательность $\{\mathbf{x}^k\}$ перестает быть релаксационной. С этим можно бороться, уменьшая значение параметра κ , но при этом возникает вторая проблема: по мере приближения к стационарной точке шаг спуска $\kappa |\text{grad } f(\mathbf{x}^{k-1})|$ уменьшается, а это сильно замедляет сходимость релаксационной последовательности.

Описанные проблемы приводят к задаче оценки возможной величины κ , которая, с одной стороны, позволяет избегать „проскакивания“ стационарной точки, а с другой — обеспечивает приемлемую

*Параметр κ во многих случаях более удобен, чем шаг спуска, и его часто также называют шагом спуска.

скорость сходимости последовательности приближений к стационарной точке. Такие оценки можно получить в рамках некоторых классов целевых функций, удовлетворяющих некоторым дополнительным условиям.

Теорема 3.1. Пусть функция $f(x)$ ограничена снизу и дифференцируема в \mathbb{R}^n , а ее градиент удовлетворяет условию Липшица, т.е. для любых $x, y \in \mathbb{R}^n$

$$|\operatorname{grad} f(x) - \operatorname{grad} f(y)| \leq L|x - y|, \quad (3.12)$$

где $L > 0$ — некоторая константа. Тогда последовательность $\{x^k\}$, определяемая рекуррентным соотношением (3.11) с $\kappa \in (0, 2/L)$, является релаксационной. При этом справедлива оценка

$$f(x^k) \leq f(x^{k-1}) - \kappa \left(1 - \frac{\kappa L}{2}\right) |\operatorname{grad} f(x^{k-1})|^2 \quad (3.13)$$

и $|\operatorname{grad} f(x^k)| \rightarrow 0$ при $k \rightarrow \infty$. #

Если целевая функция удовлетворяет условиям теоремы 3.1, то при $\kappa = 1/L$ не только получается релаксационная последовательность, но и не происходит „проскакивания“ стационарной точки. Действительно, из неравенства (3.12) вытекает, что

$$|w^k| = |\operatorname{grad} f(x^{k-1})| \leq L|x^{k-1} - \tilde{x}|, \quad (3.14)$$

где \tilde{x} — стационарная точка. Следовательно, при $\kappa \leq 1/L$

$$|x^k - x^{k-1}| = \kappa |w^k| \leq |x^{k-1} - \tilde{x}|,$$

т.е. очередной шаг спуска не превышает расстояния от текущей точки до стационарной точки.

Пусть функция $f(x)$ ограничена снизу, а последовательность $\{x^k\}$ выбрана так, что для некоторого числа $\omega_0 > 0$ выполняются неравенства

$$f(x^{k-1}) - f(x^k) \geq \omega_0 |w^k|^2, \quad k \in \mathbb{N}. \quad (3.15)$$

Тогда, суммируя неравенства (3.15) для $k = \overline{1, m}$, получаем:

$$f(x^0) - f(x^m) \geq \omega_0 \sum_{k=1}^m |\operatorname{grad} f(x^{k-1})|^2.$$

Таким образом, знакоположительный ряд $\sum_{k=1}^{\infty} |\text{grad } f(x^{k-1})|^2$ сходится, а в силу необходимого признака сходимости его общий член стремится к нулю, т.е. $|\text{grad } f(x^k)| \rightarrow 0$ при $k \rightarrow \infty$.

Итак, если при построении последовательности $\{x^k\}$ за счет подходящего выбора шага спуска на каждой итерации удастся обеспечить выполнение условия (3.15), то соответствующая последовательность градиентов будет стремиться к нулю. Это дает некоторые гарантии, что итерационная последовательность сходится к стационарной точке. Отметим, что условия (3.15) выполняются в том случае, когда целевая функция $f(x)$ удовлетворяет условиям теоремы 3.1, а последовательность $\{x^k\}$ построена в соответствии с рекуррентным соотношением (3.11), в котором $\kappa \in (0, 2/L)$. В самом деле, неравенство (3.13) можно записать в виде

$$f(x^{k-1}) - f(x^k) \geq \kappa \left(1 - \frac{\kappa L}{2}\right) |w^k|^2,$$

что равносильно неравенству (3.15) с $\omega_0 = \kappa \left(1 - \frac{\kappa L}{2}\right)$.

Замечание 3.1. Из теоремы 3.1 не следует сходимость метода градиентного спуска, основанного на рекуррентном соотношении (3.11). Например, функция $f(x) = 1/(1 + |x|^2)$ удовлетворяет условиям теоремы, но при любом выборе начальной точки $x^0 \in \mathbb{R}^n$ имеем $|x^k| \rightarrow \infty$ при $k \rightarrow \infty$.

Приведенный пример показывает, что функция вообще может не достигать наименьшего значения. Однако если к условиям теоремы 3.1 добавить требование ограниченности множества $X_0 = \{x \in \mathbb{R}^n: f(x) \leq f(x^0)\}$, то функция $f(x)$ будет достигать наименьшего значения. Можно показать, что при этом последовательность $\{x^k\}$ будет сходиться к какой-либо стационарной точке функции. #

Условие (3.12) вполне естественно и выполняется для большинства элементарных функций: например, оно выполняется для всех дважды непрерывно дифференцируемых функций на заданном замкнутом ограниченном множестве. Однако в прикладных задачах найти конкретное значение константы L не всегда удается. Если это так, то реализовать метод градиентного спуска с рекуррентным соотношением (3.11) уже нельзя. Необходимо модифицировать рекуррентное

соотношение, заменяя его соотношением

$$\mathbf{x}^k = \mathbf{x}^{k-1} + \varkappa_k \mathbf{w}^k, \quad k \in \mathbb{N}, \quad (3.16)$$

в котором значения $\varkappa_k > 0$ меняются от итерации к итерации.

Существуют разные способы выбора на каждой итерации параметра \varkappa_k , определяющего шаг спуска. Ограничимся рассмотрением двух из них.

Если функция $f(\mathbf{x})$ непрерывно дифференцируема в \mathbb{R}^n , то функция $\varphi_k(\varkappa) = f(\mathbf{x}^{k-1} + \varkappa \mathbf{w}^k)$ непрерывно дифференцируема на числовой оси и при достаточно малых значениях \varkappa имеет отрицательную производную, поскольку при $\varkappa = 0$ имеем

$$\varphi'_k(0) = (\text{grad } f(\mathbf{x}), \mathbf{w}^k) = -|\mathbf{w}^k|^2 < 0.$$

Следовательно, функция $\varphi_k(\varkappa)$ убывает в окрестности $\varkappa = 0$. Она убывает при $\varkappa > 0$ по крайней мере до значения $\tilde{\varkappa}_k$, при котором ее производная обращается в нуль. Выбор в соотношении (3.16) значения $\varkappa_k = \tilde{\varkappa}_k$ приводит к варианту градиентного спуска, называемому *исчерпывающим спуском* (движение в направлении антиградиента \mathbf{w}^k происходит до тех пор, пока не будут исчерпаны „подряд идущие“ отрицательные значения производной функции $f(\mathbf{x})$ по направлению этого вектора).

Если целевая функция $f(\mathbf{x})$ удовлетворяет условиям теоремы 3.1, то итерационная последовательность, полученная по методу исчерпывающего спуска, удовлетворяет условиям (3.15). Действительно, поскольку

$$\varphi'_k(\varkappa) = (\text{grad } f(\mathbf{x}^{k-1} + \varkappa \mathbf{w}^k), \mathbf{w}^k),$$

то в точке $\mathbf{x}^k = \mathbf{x}^{k-1} + \tilde{\varkappa}_k \mathbf{w}^k$, найденной по методу наискорейшего спуска, имеем:

$$(\mathbf{w}^{k+1}, \mathbf{w}^k) = -(\text{grad } f(\mathbf{x}^k), \mathbf{w}^k) = -\varphi'_k(\tilde{\varkappa}_k) = 0. \quad (3.17)$$

В соответствии с неравенством Коши — Буняковского и неравенством (3.12) заключаем, что

$$\begin{aligned} |\mathbf{w}^k|^2 &= (\mathbf{w}^k, \mathbf{w}^k - \mathbf{w}^{k+1}) \leq |\mathbf{w}^k| |\mathbf{w}^k - \mathbf{w}^{k+1}| = \\ &= |\mathbf{w}^k| |\text{grad } f(\mathbf{x}^{k+1}) - \text{grad } f(\mathbf{x}^k)| \leq \\ &\leq L |\mathbf{w}^k| |\mathbf{x}^{k+1} - \mathbf{x}^k| = L \tilde{\varkappa}_k |\mathbf{w}^k|^2, \end{aligned}$$

откуда $\tilde{\kappa}_k \geq 1/L$. Следовательно,

$$f(x^{k-1}) - f(x^k) \geq f(x^{k-1}) - f(\tilde{x}^k) \geq \frac{1}{2L} |w^k|^2,$$

где $\tilde{x}^k = x^{k-1} + \frac{1}{L} w^k$ — точка, соответствующая значению $\kappa = 1/L$.

Замечание 3.2. Очередная точка, найденная по методу исчерпывающего спуска, может не совпадать с точкой, найденной по методу наискорейшего спуска: для первого метода значение κ_k — это стационарная точка функции $\varphi_k(\kappa)$, в то время как для второго метода она есть точка минимума. Эти точки будут совпадать в том случае, когда на каждой итерации у функции $\varphi_k(\kappa)$ каждая стационарная точка будет точкой минимума. Так верно, например, для выпуклой целевой функции. #

Согласно соотношению (3.17) в методе исчерпывающего спуска (и, как следствие, в методе наискорейшего спуска) любые два последовательных направления спуска u^k и u^{k+1} ортогональны. Но вектор, ортогональный антиградиенту

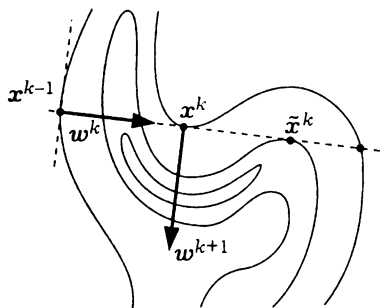


Рис. 3.1

(и градиенту), является касательным к поверхности (линии) уровня функции. Поэтому в точке x^k предыдущий вектор спуска u^k касается поверхности (линии) уровня целевой функции, проходящей через эту точку. Это показано на рис. 3.1 в двумерном случае; точка x^k соответствует методу исчерпывающего спуска, а точка \tilde{x}^k — методу наискорейшего спуска.

Выполнение условия (3.15) можно также обеспечить следующим образом. Задаем некоторое стартовое значение κ_0 (например, $\kappa_0 = 1$). На каждой k -й итерации выбираем параметр $\kappa_k = \kappa_{k-1}$ и осуществляем спуск, определяя точку x^k в соответствии с рекуррентным соотношением (3.16). Для найденной точки проверяем выполнение условия

$$f(x^k) \leq f(x^{k-1}) - \omega \kappa_k |w^k|^2, \quad (3.18)$$

где $\omega \in (0, 1)$. Если спуск оказался неудовлетворительным вследствие нарушения этого условия, то уменьшаем κ_k , умножая его на некоторое фиксированное число $\nu \in (0, 1)$ (например $\nu = 0,5$), и повторяем спуск.

Такой способ подбора значения \varkappa_k получил название **метода дробления шага**. На рис. 3.2 показаны итерации с использованием этого метода в плоском случае.

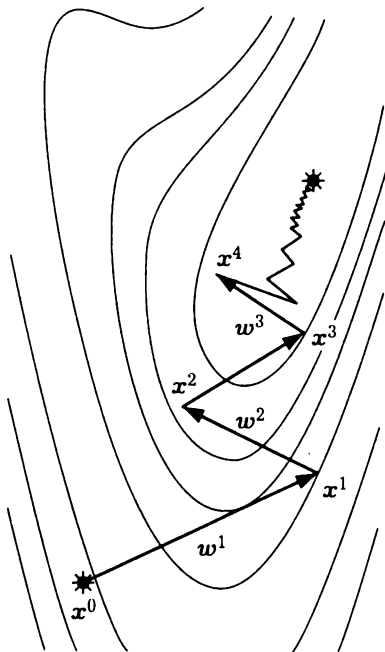


Рис. 3.2

Отметим, что условие (3.18) будет выполняться при достаточно малом значении \varkappa_k , поскольку

$$\begin{aligned} \lim_{\varkappa_k \rightarrow 0} \frac{f(x^k) - f(x^{k-1})}{\varkappa_k |w^k|^2} &= \\ &= \lim_{\varkappa_k \rightarrow 0} \frac{(\text{grad } f(x^{k-1}), \varkappa_k w^k) + o(\varkappa_k)}{\varkappa_k |w^k|^2} = -1 < -\omega. \end{aligned}$$

В то же время, если целевая функция $f(x)$ удовлетворяет условиям теоремы 3.1, последовательность $\{\varkappa_k\}$ не будет стремиться к нулю и, следовательно, коэффициенты \varkappa_k будут удовлетворять неравенствам $\varkappa_k > q$ для некоторого фиксированного числа $q > 0$. В самом деле, либо параметр \varkappa_k все время будет превышать границу $\frac{2}{L}(1 - \omega)$, либо

на какой-то итерации окажется, что $\kappa_k \leq \frac{2}{L}(1 - \omega)$. Но тогда, согласно теореме 3.1, будет выполняться неравенство

$$f(\mathbf{x}^k) \leq f(\mathbf{x}^{k-1}) - \kappa_k \left(1 - \frac{\kappa_k L}{2}\right) |\mathbf{w}^k|^2. \quad (3.19)$$

При $\kappa_k \leq \frac{2}{L}(1 - \omega)$ выполняется неравенство

$$1 - \frac{\kappa_k L}{2} \geq \omega.$$

Следовательно, неравенство (3.18) будет вытекать из неравенства (3.19). Начиная с этого момента значение κ_k уже не будет уменьшаться.

Пример 3.1. Решим задачу минимизации функции

$$f(x_1, x_2) = (x_1^2 - x_2)^2 + (x_1 - 1)^2$$

градиентным методом с дроблением шага, выбрав начальную точку $\mathbf{x}^0 = (-1, -2)$, параметр точности поиска $\varepsilon_3 = 10^{-3}$, а также параметры метода $\kappa_0 = 1$, $\omega = 0,5$. Рассмотрим несколько вариантов метода, определяемых коэффициентом ν дробления шага.

Результаты поиска точки минимума $\mathbf{x}^* = (1, 1)$ рассматриваемой функции при различных значениях коэффициента дробления шага $\nu \in (0, 1)$ (приближенное значение точки минимума, значение функции в этой точке, а также количество итераций поиска N) приведены в табл. 3.1. Траектории поиска точки минимума для каждого варианта изображены на рис. 3.3 и 3.4 (в табл. 3.1 даны ссылки на соответствующий каждой траектории рисунок).

Таблица 3.1

ν	\mathbf{x}^*	$f(\mathbf{x}^k)$	N	Рисунок
0,2	(0,999, 0,997)	$0,132 \cdot 10^{-5}$	102	3.3, а
0,4	(0,999, 0,998)	$0,119 \cdot 10^{-5}$	76	3.3, б
0,5	(0,999, 0,998)	$0,100 \cdot 10^{-5}$	72	3.3, в
0,6	(0,999, 0,997)	$0,157 \cdot 10^{-5}$	54	3.3, г
0,8	(0,999, 0,997)	$0,159 \cdot 10^{-5}$	66	3.4, а
0,5	(0,999, 0,998)	$0,066 \cdot 10^{-5}$	85	3.4, б

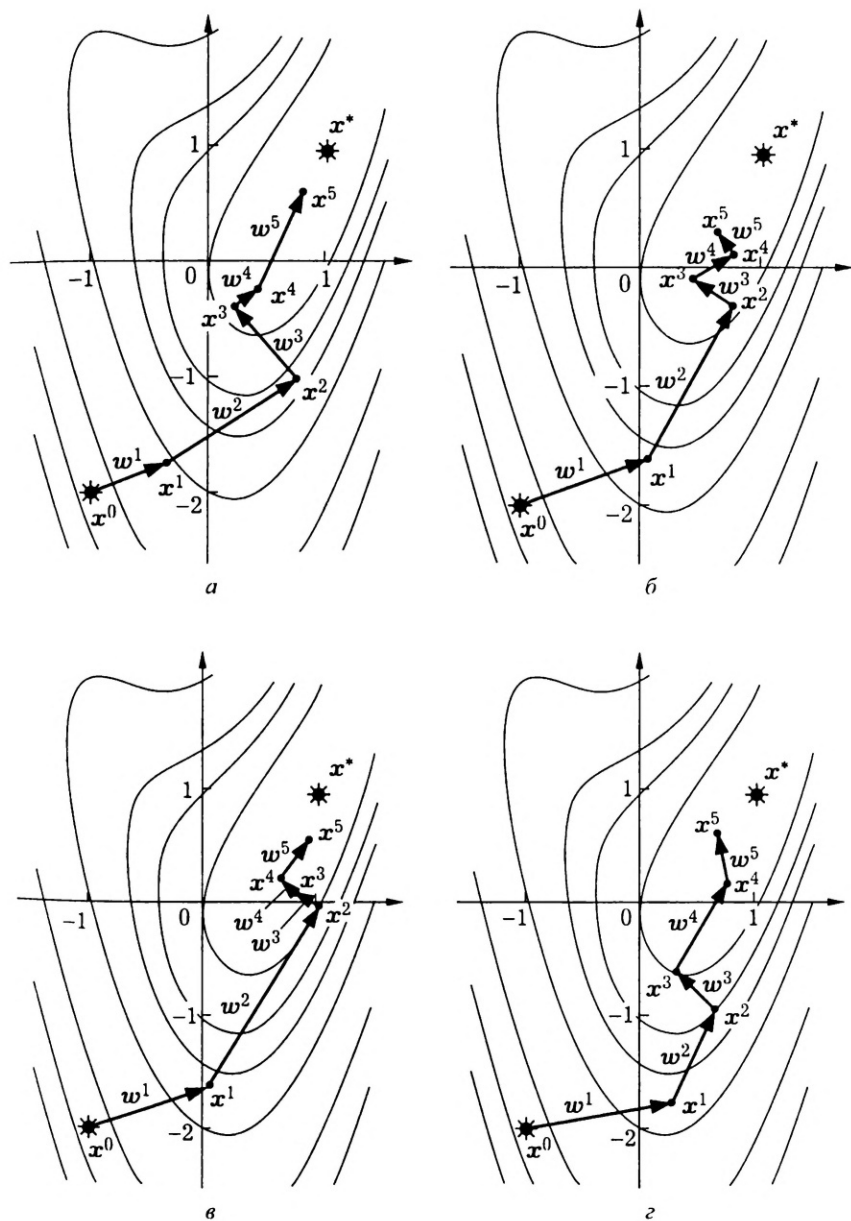


Рис. 3.3

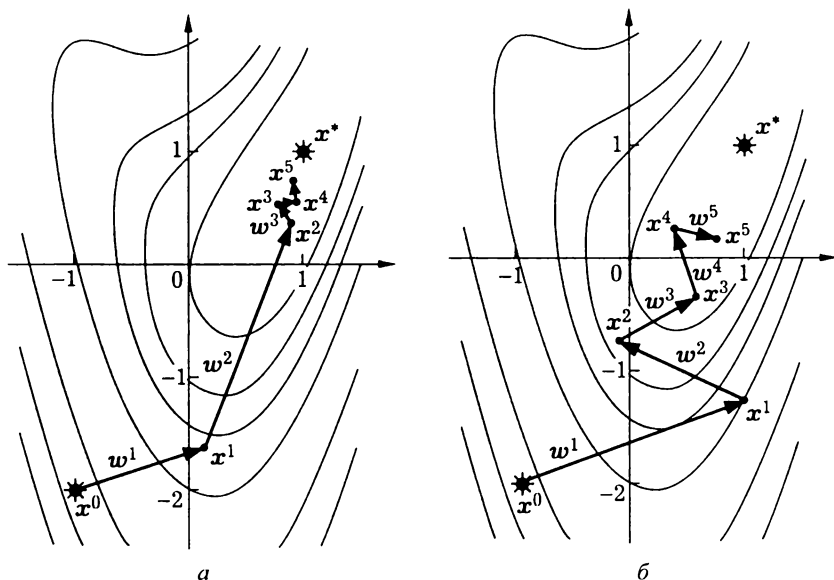


Рис. 3.4

Из результатов поиска видно, что коэффициент ν существенно влияет на количество итераций, обеспечивающее поиск точки минимума целевой функции с заданной точностью. Наилучшие результаты соответствуют значениям $\nu \in [0,5, 0,8]$.

На практике значение κ_k на k -й итерации часто выбирают, исходя не из условия (3.18), а из более слабого условия $f(x^{k-1}) - f(x^k) > 0$, которое равносильно (3.18) при $\omega = 0$. На рис. 3.4, б показан именно этот вариант алгоритма градиентного спуска при $\kappa_0 = 1$ и $\nu = 0,5$. #

В методе наискорейшего спуска шаг спуска определяется путем минимизации целевой функции в выбранном направлении спуска. Поэтому естественно предполагать, что метод наискорейшего спуска может быть более эффективным. Однако отметим, что минимизация функции — более трудоемкая задача, чем спуск с фиксированным шагом. Выигрыш в количестве итераций еще не означает снижения общего объема вычислительной работы.

Пример 3.2. Решим задачу минимизации функции из примера 3.1 методом наискорейшего спуска, приняв $x^0 = (-1, -2)$ и $\varepsilon_3 = 10^{-3}$.

Процесс поиска точки минимума показан на рис. 3.5 (обратим внимание на ортогональность двух последовательных направлений спуска). При том же значении $\varepsilon_3 = 10^{-3}$ параметра точности, что и в примере 3.1, точка минимума определена за $N = 96$ итераций.

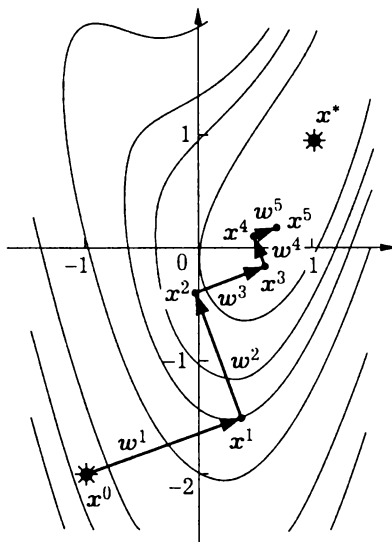


Рис. 3.5

Как видно, метод наискорейшего спуска не дает выигрыша по количеству итераций. Характерной особенностью процесса поиска в данном случае является то, что на первых итерациях происходит быстрое убывание значений целевой функции, которое затем замедляется. Это свидетельствует о том, что эффективность метода наискорейшего спуска на завершающей стадии итерационного процесса снижается.

3.3. Минимизация квадратичной функции

В разработке численных методов многомерной оптимизации большое внимание уделяется вопросам минимизации квадратичных функций. Интерес к этим вопросам вызван несколькими причинами.

Во-первых, для задачи минимизации квадратичной функции решение можно записать в аналитическом виде и использовать его в

качестве эталона при оценке погрешности численного метода, а также при сравнении различных методов между собой. Во-вторых, квадратичная функция служит своеобразным испытательным полигоном, на котором можно проверить работоспособность численного метода (если метод малопригоден для минимизации квадратичной функции, то, скорее всего, он будет неработоспособен и для более сложных целевых функций). Наконец, в-третьих, дважды непрерывно дифференцируемая функция (а значит, практически все элементарные функции) хорошо аппроксимируется в окрестности данной точки квадратичной функцией (такую аппроксимацию можно получить с помощью формулы Тейлора второго порядка). Указанная аппроксимация позволяет свести задачу минимизации сложной функции к последовательности задач минимизации аппроксимирующих квадратичных функций. Это особенно эффективно, если минимизируемая функция в окрестности точки минимума имеет положительно определенную матрицу Гессе, так как аппроксимирующие ее квадратичные функции являются строго выпуклыми функциями.

Особое место в многомерной оптимизации занимают задачи безусловной минимизации выпуклых квадратичных функций двух переменных (задачи двумерной безусловной минимизации). Изучение идей, положенных в основу вычислительных алгоритмов, на примере решения задач двумерной минимизации позволяет наглядно, с помощью геометрических построений на плоскости, представить отличительные черты каждого метода, выявить их вычислительные свойства, обсудить достоинства и недостатки.

Квадратичную функцию $f(x)$ можно представить в виде $f(x) = \frac{1}{2}A(x) + b(x) + c$, где $A(x)$ — квадратичная форма, $b(x)$ — линейная функция, c — свободный член. Квадратичную форму $A(x)$ можно записать через скалярное произведение, используя ее матрицу Q : $A(x) = (Qx, x)$. Линейную функцию также можно записать с помощью скалярного произведения в виде $b(x) = (b, x)$, где b — некоторый фиксированный вектор. Наличие свободного члена c в задачах оптимизации несущественно, т.е. можно считать, что $c = 0$. Таким образом, целевая квадратичная функция $f(x)$ может быть записана в виде

$$f(x) = \frac{1}{2}(Qx, x) + (b, x), \quad (3.20)$$

где Q — симметрическая матрица порядка n , а $b \in \mathbb{R}^n$ — заданный вектор. Нетрудно убедиться в том, что матрица Q — это матрица Гессе (матрица частных производных второго порядка) функции $f(x)$.

Непосредственной проверкой можно убедиться в том, что для квадратичной функции $\text{grad } f(x) = Qx + b$. Если матрица Q невырождена, то в силу необходимого условия экстремума квадратичная функция имеет единственную стационарную точку $x^* = -Q^{-1}b$. Эта точка будет точкой наименьшего значения функции (точкой строгого локального минимума) тогда и только тогда, когда квадратичная форма $A(x) = (Qx, x)$ положительно определена (при этом говорят о положительной определенности самой квадратичной функции). Условие положительной определенности целевой функции часто подразумевается в задачах квадратичного программирования.

При анализе тех или иных вычислительных методов всегда можно считать, что точка минимума квадратичной функции расположена в начале координат, т.е. $x^* = 0$. Это равносильно тому, что в представлении (3.20) вектор b равен нулю. Таким образом, вместо квадратичной функции общего вида (3.20) можно рассматривать частный случай

$$f(x) = \frac{1}{2}(Qx, x), \quad x \in \mathbb{R}^n. \quad (3.21)$$

При этом будем считать, что матрица Q положительно определена. Тогда квадратичная функция $\frac{1}{2}(Qx, x)$ неотрицательна в \mathbb{R}^n и достигает наименьшего значения 0 в единственной точке $x^* = 0$.

В качестве метода минимизации квадратичной функции рассмотрим метод градиентного спуска. Для функции (3.21) градиент в произвольной точке x равен Qx . Выбрав начальную точку $x^0 \neq 0$, находим в этой точке антиградиент $w^1 = -\text{grad } f(x^0) = -Qx^0$. В соответствии с рекуррентным соотношением (3.16) для следующей точки получим:

$$x^1 = x^0 + \kappa_1 w^1 = x^0 - \kappa_1 Qx^0 = (I - \kappa_1 Q)x^0, \quad (3.22)$$

где I — единичная матрица.

Из равенства $x^1 = (I - \kappa_1 Q)x^0$ можно сделать важный вывод, в значительной степени определяющий способы построения численных методов безусловной минимизации: точки минимума квадратичной функции можно достичь за одну итерацию, если в качестве начальной точки x^0 выбран один из собственных векторов матрицы Q . Действительно, если x^0 совпадает с собственным вектором матрицы Q , отвечающим собственному значению λ_j , то $Qx^0 = \lambda_j x^0$ и

$(Q - \lambda_j I)x^0 = 0$. Поэтому при $\kappa_1 = 1/\lambda_j$

$$x^1 = \left(I - \frac{1}{\lambda_j} Q\right)x^0 = -\frac{1}{\lambda_j}(Q - \lambda_j I)x^0 = 0,$$

т.е. следующая точка x^1 итерационной последовательности совпадает с точкой минимума $x^* = 0$ квадратичной функции (3.21).

В двумерном случае график квадратичной функции $f(x) = \frac{1}{2}(Qx, x)$ представляет собой эллиптический параболоид с центром в начале координат, а ее линии уровня — подобные эллипсы (рис. 3.6). Метод градиентного спуска приведет в точку $(0, 0)$ наименьшего значения за одну итерацию, если начальная точка выбрана на одной из осей эллипсов: в этом случае радиус-вектор точки является собственным вектором матрицы Q второго порядка.

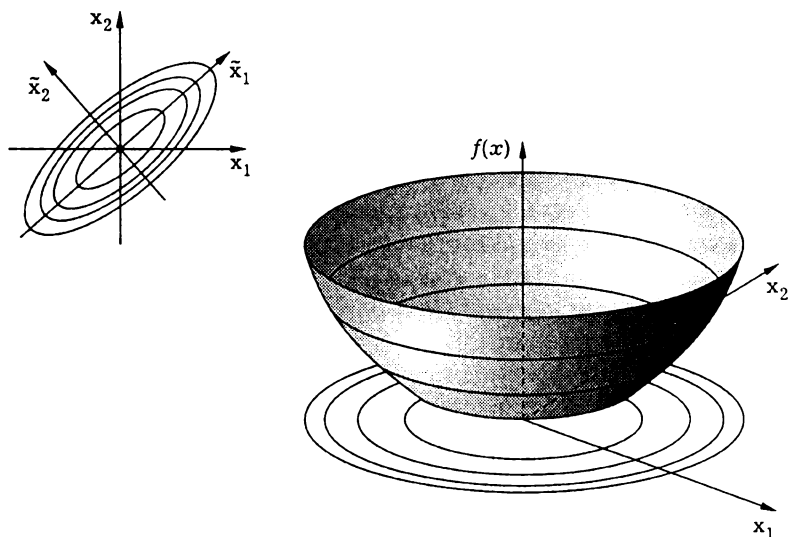


Рис. 3.6

Выделим важный частный случай $Q = \lambda I$, в котором все собственные значения матрицы Q совпадают, а каждый ненулевой вектор $x \in \mathbb{R}^n$ является собственным. В этом случае точка минимума квадратичной функции (3.21) достигается за одну итерацию при любом

выборе начальной точки. Здесь возникает простой с теоретической точки зрения путь повышения эффективности метода градиентного спуска для квадратичных функций. Квадратичную форму (3.21) с положительно определенной матрицей Q невырожденной заменой переменных можно привести к каноническому виду, причем с единичной матрицей. Например, можно ортогональным преобразованием привести квадратичную форму к виду

$$f_1(\xi) = \sum_{j=1}^n \lambda_j \xi_j^2,$$

где $\xi = (\xi_1, \dots, \xi_n)$, а $\lambda_j, j = \overline{1, n}$, — положительные собственные значения матрицы Q . Затем, изменяя масштабы переменных заменой $\zeta_j = \sqrt{\lambda_j} \xi_j, j = \overline{1, n}$, получаем

$$f_2(\zeta) = \sum_{j=1}^n \zeta_j^2,$$

где $\zeta = (\zeta_1, \dots, \zeta_n)$.

В новых переменных ζ_1, \dots, ζ_n минимум квадратичной функции методом градиентного спуска достигается за одну итерацию при любом выборе начальной точки. Однако этот теоретически простой подход может оказаться очень трудоемким с вычислительной точки зрения, так как трудоемкой является задача на собственные значения матрицы Q (особенно для матриц высокого порядка).

Описанный метод оказывается полезным, когда минимизируемая функция имеет график так называемой *овражной структуры*, т.е. скорость изменения функции по одному или нескольким направлениям существенно больше, чем по остальным. Для квадратичных функций это происходит, если собственные значения матрицы сильно различаются. В двумерном случае график функции с овражной структурой в окрестности точки минимума напоминает овраг с крутыми склонами и пологим дном, а линии уровня оказываются сильно вытянутыми (рис. 3.7). Сильное расхождение в скорости убывания функции по различным направлениям снижает эффективность практически любого метода спуска, а одним из способов борьбы с этим „недугом“ является *масштабирование*, при котором изменяются масштабы переменных. Масштабирование в случае квадратичной функции удобно выполнять после приведения ее квадратичной формы к каноническому виду.

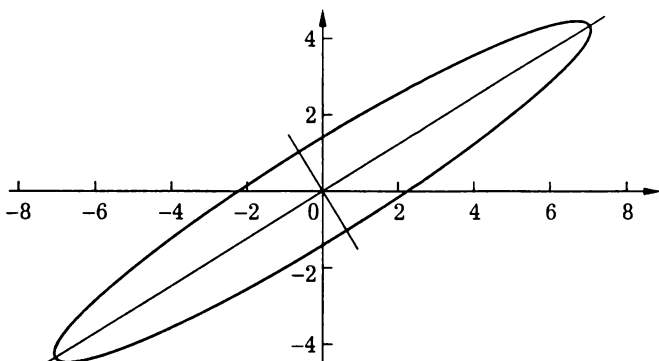


Рис. 3.7

Вернемся к задаче минимизации квадратичной функции $f(x) = \frac{1}{2}(Qx, x)$ и рассмотрим вариант метода градиентного спуска с постоянным значением $\kappa_k = \kappa$ на всех итерациях. На k -й итерации получим

$$x^k = x^{k-1} + \kappa w^k = x^{k-1} - \kappa Qx^{k-1} = (I - \kappa Q)x^{k-1}. \quad (3.23)$$

Таким образом, метод градиентного спуска свелся к методу простой итерации для линейного оператора, определяемого матрицей $I - \kappa Q$. Заключение о сходимости последовательности $\{x^k\}$ можно сделать на основании *теоремы о неподвижной точке*. Согласно этой теореме последовательность $\{x^k\}$, определяемая рекуррентным соотношением $x^k = f(x^{k-1})$, $k \in \mathbb{N}$, сходится к неподвижной точке x^* отображения $f(x)$, если это отображение является *сжимающим*, т.е. подчиняется условию Липшица

$$|f(x) - f(y)| \leq q|x - y| \quad (3.24)$$

с постоянной Липшица $q < 1$. В данном случае отображение представляет собой линейный оператор, который является сжимающим отображением, если имеет норму, меньшую единицы. Под нормой оператора (нормой матрицы) здесь понимается *спектральная норма*. Для симметрической матрицы спектральная норма совпадает с максимальным из модулей ее собственных чисел.

Итак, если итерационная последовательность $\{x^k\}$ построена с помощью рекуррентного соотношения $x^k = x^{k-1} + \kappa w^k$, $\kappa > 0$, $k \in \mathbb{N}$,

то для ее сходимости согласно равенствам (3.23) и теореме о неподвижной точке достаточно, чтобы выполнялось условие $q(\kappa) = \|I - \kappa Q\| < 1$. Для квадратичной функции с положительно определенной матрицей Q , имеющей собственные значения $0 < \lambda_1 \leq \dots \leq \lambda_n$, матрица $I - \kappa Q$ имеет собственные значения $1 - \kappa \lambda_j$, $j = \overline{1, n}$, удовлетворяющие неравенствам

$$1 - \kappa \lambda_n < 1 - \kappa \lambda_{n-1} < \dots < 1 - \kappa \lambda_1 < 1.$$

Из этих соотношений вытекает, что условие $q(\kappa) < 1$ равносильно неравенству $1 - \kappa \lambda_n > -1$ или, что то же самое, соотношению $\kappa \in (0, 2/\lambda_n)$.

Из теоремы о неподвижной точке вытекает, что для итерационной последовательности $\{x^k\}$, построенной по рекуррентному соотношению $x^k = f(x^{k-1})$, верна оценка $|x^k - x^*| \leq q^k |x^0 - x^*|$, где q — постоянная Липшица в условии (3.24). Для ускорения сходимости последовательности $\{x^k\}$ необходимо обеспечивать как можно меньшее значение q . В рассматриваемом случае $q(\kappa)$ принимает наименьшее значение, когда собственные значения $1 - \kappa \lambda_n$ и $1 - \kappa \lambda_1$ матрицы $I - \kappa Q$ совпадают по абсолютной величине и противоположны по знаку, т.е. $-(1 - \kappa \lambda_1) = 1 - \kappa \lambda_n$. Из этого равенства вытекает, что оптимальное значение параметра κ равно $\kappa_* = \frac{2}{\lambda_1 + \lambda_n} \leq \frac{2}{\lambda_n}$. При выборе $\kappa = \kappa_*$ получаем

$$q_* = q(\kappa_*) = \frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1} = \frac{c(Q) - 1}{c(Q) + 1}, \quad (3.25)$$

где $c(Q) = \lambda_n/\lambda_1$ — **число обусловленности** матрицы Q . Большое число обусловленности матрицы указывает на то, что график квадратичной функции имеет овражную структуру. В этом случае значение q_* близко к единице и последовательность $\{x^k\}$ сходится медленно. Для единичной матрицы I все собственные значения одинаковы и равны единице. Поэтому $c(I) = 1$ и $q_* = 0$, а точка минимума соответствующей квадратичной функции достигается за одну итерацию при любом выборе начальной точки.

На рис. 3.8 показан процесс сходимости метода градиентного спуска для квадратичной функции $f(x) = x_1^2 + 10x_2^2$ с двумя вариантами постоянного значения параметра κ : на рис. 3.8, а $\kappa = 0,01$, на рис. 3.8, б $\kappa = 0,09$.

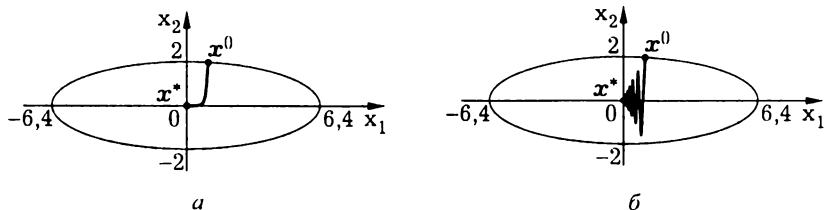


Рис. 3.8

Матрица квадратичной формы $x_1^2 + 10x_2^2$ имеет собственные значения $\lambda_1 = 2$ и $\lambda_2 = 20$. Полученное выше условие сходимости метода градиентного спуска с постоянным значением κ означает, что параметр κ нужно выбирать из интервала $(0, 2/\lambda_2) = (0, 0,1)$. Если $\kappa \geq 0,1$, то последовательность $\{x^k\}$ перестает быть релаксационной, а метод расходится или „зацикливается“. На рис. 3.9 показаны эти ситуации: *a* — „зацикливание“ при $\kappa = 0,10$; *б* — расходимость при $\kappa = 0,11$.

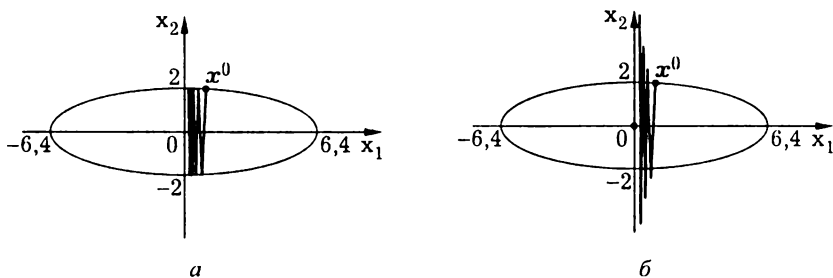


Рис. 3.9

Минимизация квадратичной функции $f(x) = \frac{1}{2}(Qx, x)$ градиентным методом с использованием исчерпывающего спуска проходит с нарушением условия $\kappa_k \in (0, 2/\lambda_n)$. Действительно, исчерпывающий спуск на k -й итерации означает поиск стационарной точки функции $\varphi_k(\kappa) = f(x^{k-1} + \kappa w^k)$. В данном случае функция

$$\begin{aligned} \varphi_k(\kappa) &= \frac{1}{2}(Q(x^{k-1} + \kappa w^k), x^{k-1} + \kappa w^k) = \\ &= f(x^{k-1}) + \kappa(Qx^{k-1}, w^k) + \frac{\kappa^2}{2}(Qw^k, w^k) \end{aligned}$$

является многочленом второй степени с положительным коэффициентом при старшей степени. Поэтому она имеет единственную стацио-

нарную точку

$$\kappa_k = -\frac{(Qx^{k-1}, w^k)}{(Qw^k, w^k)} = \frac{|w^k|^2}{(Qw^k, w^k)}. \quad (3.26)$$

С помощью формулы (3.26) нетрудно убедиться в том, что в методе исчерпывающего спуска параметр κ_k может принимать любые значения от $1/\lambda_n$ (если x^{k-1} — собственный вектор матрицы Q с собственным значением λ_n) до $1/\lambda_1$ (если x^{k-1} — собственный вектор матрицы Q с собственным значением λ_1). Ясно, что если число обусловленности $c(Q)$ превышает 2, то условие $\kappa_k \in (0, 2/\lambda_n)$ при $\kappa_k = 1/\lambda_1$ нарушается.

Различие между двумя вариантами градиентного спуска состоит в том, при постоянном параметре κ шаг спуска определяется без учета скорости убывания функции в направлении точки минимума, а потому из расчета максимально возможной скорости (т.е. пешеход спускается в овраг со скоростью, рассчитанной на самый крутой склон). В случае же наискорейшего спуска шаг спуска оказывается тем больше, чем медленнее квадратичная функция убывает в направлении точки минимума. Нетрудно убедиться в том, что для квадратичной функции с положительно определенной матрицей выполняются условия теоремы 3.1. Поэтому применение исчерпывающего спуска дает релаксационную последовательность $\{x^k\}$, для которой последовательность градиентов $\{\text{grad } f(x^k)\}$ сходится к нулю. Поскольку матрица квадратичной функции положительно определенная, релаксационная последовательность сходится к точке минимума.

Результаты применения градиентного метода с исчерпывающим спуском для минимизации функции $f(x_1, x_2) = x_1^2 + 10x_2$ показаны на рис. 3.10. Траектория спуска близка к траектории спуска с фиксированным значением $\kappa = 0,01$ (см. рис. 3.8, а), но проходится за существенное меньшее количество итераций (5 против 120).

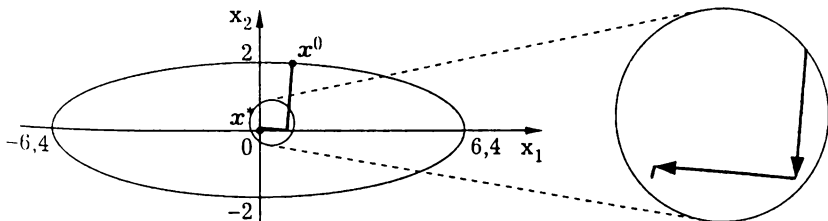


Рис. 3.10

Из приведенных рассуждений следует, что для квадратичной функции метод наискорейшего спуска эквивалентен градиентному методу с исчерпывающим спуском (т.е. оба метода дают одну и ту же итерационную последовательность). Впрочем, такой вывод можно сделать и без выкладок, имея в виду, что квадратичная функция с положительно определенной матрицей является строго выпуклой.

3.4. Метод сопряженных направлений

Продолжим рассмотрение методов спуска, заменив рекуррентное соотношение (3.4) более общим соотношением

$$x^k = x^{k-1} + \kappa_k p^k, \quad k \in \mathbb{N}, \quad (3.27)$$

где κ_k — параметр, регулирующий шаг спуска; $p^k \in \mathbb{R}^n$ — вектор, определяющий направление спуска. В этом соотношении вектор p^k может быть неединичным, а шаг спуска задается величиной $\kappa_k |p^k|$.

Рассмотрим задачу минимизации квадратичной функции, ограничившись ее частным случаем $f(x) = \frac{1}{2} (Qx, x)$, где Q — положительно определенная матрица. Как было отмечено выше (см. разд. 3.3), такую квадратичную функцию невырожденной заменой переменных можно привести к каноническому виду

$$f_2(\zeta) = \sum_{j=1}^n \zeta_j^2, \quad (3.28)$$

где $\zeta = (\zeta_1, \zeta_2, \dots, \zeta_n)$. В переменных $\zeta_1, \zeta_2, \dots, \zeta_n$ метод градиентного спуска приводит в точку минимума за одну итерацию. Однако при таком подходе градиент функции необходимо вычислять в новых переменных. Стандартная процедура приведения квадратичной формы к каноническому виду трудоемкая, а метод, базирующийся на этой процедуре, оказывается малоперспективным.

Поставим задачу иначе. Линейная невырожденная замена переменных в квадратичной форме может интерпретироваться как переход в \mathbb{R}^n от одного базиса к другому. Тогда приведение квадратичной формы к каноническому виду означает выбор соответствующего базиса в \mathbb{R}^n . Можно заметить, что если матрица Q является (симметрической)

положительно определенной, то функция $(x, y)_Q = (Qx, y)$ определяет в \mathbb{R}^n скалярное произведение, а квадратичная форма (Qx, x) есть квадрат евклидовой нормы вектора x в евклидовом пространстве с указанным скалярным произведением. Приведение этой квадратичной формы к каноническому виду (сумме квадратов) означает выбор ортонормированного базиса в евклидовом пространстве со скалярным произведением $(x, y)_Q$.

Описанная интерпретация чрезвычайно удобная: она позволяет упростить вид квадратичной функции, но при этом вместо решения задачи на собственные значения можно использовать обычную процедуру построения ортогонального базиса, хорошо известную из курса линейной алгебры и называемую *процессом ортогонализации*.

Условие ортогональности ненулевых векторов p^1 и p^2 относительно скалярного произведения $(x, y)_Q$ можно записать в виде $(Qp^1, p^2) = 0$. Векторы, удовлетворяющие этому условию, называют *сопряженными векторами* относительно положительно определенной матрицы Q , или Q -ортогональными, а направления, определяемые такими векторами, — *сопряженными направлениями*.

Рассмотрим Q -ортогональный базис p^j , $j = \overline{1, n}$. В этом базисе квадратичная функция $f(x) = \frac{1}{2}(Qx, x)$ имеет канонический вид $f_1(\xi) = \lambda_1 \xi_1^2 + \lambda_2 \xi_2^2 + \dots + \lambda_n \xi_n^2$, где $\xi = (\xi_1, \dots, \xi_n)$, а коэффициенты $\lambda_j > 0$, $j = \overline{1, n}$, определяются нормами $\|p^j\|_Q$ векторов p^j относительно скалярного произведения $(x, y)_Q$ в соответствии с формулами $\lambda_j = \frac{1}{2}\|p^j\|_Q^2$. Действительно, функция $f_1(\xi)$ представляет собой половину квадрата нормы вектора с координатами ξ_1, \dots, ξ_n . Используя координаты $0, 0, \dots, 0, 1, 0, \dots, 0$ (единица — j -я по порядку) базисного вектора p^j , заключаем, что $\lambda_j = \frac{1}{2}\|p^j\|_Q^2$.

Функция $f_1(\xi)$ является сепарабельной. Искерпывающий спуск в одном из направлений p^j означает минимизацию одного из слагаемых такой функции, а последовательность из n искерпывающих спусков в направлениях p^1, p^2, \dots, p^n минимизирует все слагаемые функции и приводит в точку ее минимума.

Выберем некоторую начальную точку x^0 . Искерпывающий спуск для квадратичной функции $f(x)$ в направлении вектора p^1 равносильен искерпывающему спуску для квадратичной функции $f_1(\xi)$ в направлении первого базисного вектора и приводит к обнулению первой координаты точки x^0 в базисе p^1, p^2, \dots, p^n . Таким образом, получаем

$x^1 = x^0 - \xi_1^0 p^1$, где первая координата ξ_1^0 точки x^0 в ортогональном (относительно скалярного произведения $(x, y)_Q$) базисе p^j может быть найдена по формуле

$$\xi_1^0 = \frac{(Qx^0, p^1)}{(Qp^1, p^1)}.$$

Сравнивая равенство $x^1 = x^0 - \xi_1^0 p^1$ с рекуррентным соотношением (3.27), заключаем, что значение ξ_1^0 отличается лишь знаком от значения κ_1 в соотношении (3.27). Повторение этих рассуждений для каждого вектора p^j приводит к следующему утверждению.

Теорема 3.2. Точка минимума квадратичной функции $f(x) = \frac{1}{2}(Qx, x)$ с положительно определенной матрицей Q достигается не более чем за n итераций спуска, если направления спуска задаются векторами $p^k \in \mathbb{R}^n$, сопряженными относительно матрицы Q , а параметры κ_k , определяющие шаг спуска в рекуррентном соотношении (3.27), вычисляются по формуле исчерпывающего спуска

$$\kappa_k = -\frac{(Qx^{k-1}, p^k)}{(Qp^k, p^k)}, \quad k = \overline{1, n}, \quad (3.29)$$

Замечание 3.3. Может оказаться, что в базисе из сопряженных векторов p^j не все векторы определяют направления спуска. В этом случае часть базисных векторов можно заменить противоположными к ним, что, конечно, не нарушит условия их сопряженности. Если и вектор p^j , и противоположный ему вектор $-p^j$ в точке x^{j-1} не определяют направление спуска, то выполняется условие $(Qx^{j-1}, p^j) = 0$. Значит, $\kappa_j = 0$ и спуск в направлении вектора p^j проводить не нужно. В этом случае количество итераций будет меньше n .

Замечание 3.4. Координаты произвольного вектора x^0 в Q -ортогональном базисе можно выразить через скалярное произведение. Это приводит к следующей формуле:

$$x^0 = \sum_{i=1}^n \frac{(Qx^0, p^i)}{(Qp^i, p^i)} p^i. \quad (3.30)$$

Если x^* — точка минимума квадратичной функции (3.20) с положительно определенной матрицей Q , то

$$x^0 - x^* = \sum_{i=1}^n \frac{(Q(x^0 - x^*), p^i)}{(Qp^i, p^i)} p^i = \sum_{i=1}^n \frac{(\text{grad } f(x^0), p^i)}{(Qp^i, p^i)} p^i.$$

Следовательно,

$$x^* = x^0 - \sum_{i=1}^n \frac{(\text{grad } f(x^0), p^i)}{(Qp^i, p^i)} p^i. \quad (3.31)$$

Используя матричную запись скалярного произведения, находим

$$(Qx^0, p^i) p^i = p^i (p^i, Qx^0) = p^i (p^i)^T Qx^0.$$

Учитывая это равенство, соотношение (3.30) можно переписать в виде

$$x^0 = \sum_{i=1}^n \frac{p^i (p^i)^T}{(Qp^i, p^i)} Qx^0,$$

причем это равенство верно для любой начальной точки x^0 . Значит, линейный оператор в \mathbb{R}^n с матрицей

$$\sum_{i=1}^n \frac{p^i (p^i)^T}{(Qp^i, p^i)} Q$$

является тождественным и

$$Q^{-1} = \sum_{i=1}^n \frac{p^i (p^i)^T}{(Qp^i, p^i)}, \quad (3.32)$$

т.е. система векторов p^i , $i = \overline{1, n}$, сопряженных относительно положительно определенной матрицы Q , позволяет достаточно просто построить матрицу Q^{-1} , обратную к Q .

Равенство (3.31) можно записать в виде

$$\begin{aligned} x^* &= x^0 - \sum_{i=1}^n \frac{(\text{grad } f(x^0), p^i)}{(Qp^i, p^i)} p^i = \\ &= x^0 - \sum_{i=1}^n \frac{p^i (p^i)^T}{(Qp^i, p^i)} \text{grad } f(x^0) = x^0 - Q^{-1} \text{grad } f(x^0). \end{aligned}$$

В силу полученного соотношения можно считать, что для квадратичной функции $f(x)$ выполнение n итераций исчерпывающего спуска в направлении сопряженных векторов p^1, \dots, p^n эквивалентно одному спуску вида $x^* = x^0 - Q^{-1} \text{grad } f(x^0)$ с одновременным обращением матрицы Q . Указанный спуск представляет собой первую итерацию метода Ньютона. #

Использование векторов, сопряженных относительно некоторой положительно определенной матрицы, составляет основу метода первого порядка, известного как *метод сопряженных направлений*. Этот метод по сравнению с методом наискорейшего спуска требует, как правило, заметно меньшего количества итераций для достижения той же точности поиска при сопоставимом объеме вычислений на каждой итерации. Различие в способах построения системы сопряженных векторов, определяющих сопряженные направления спуска, порождает несколько вариантов метода сопряженных направлений.

Сначала рассмотрим процесс минимизации квадратичной функции общего вида (3.20). Для построения системы сопряженных направлений можно использовать любой начальный базис, проводя процесс ортогонализации относительно скалярного произведения $(x, y)_Q$. Однако более эффективно исходить не из произвольного базиса, а из системы антиградиентов, вычисляемых в точках итерационной последовательности, тем самым вплетая процесс ортогонализации в процесс спуска. Рассмотрим этот подход подробнее.

Выберем начальную точку $x^0 \in \mathbb{R}^n$, найдем в этой точке антиградиент $w^1 = -\text{grad } f(x^0) = -Qx^0 - b$ и положим $p^1 = w^1$. Если $|p^1| \neq 0$, то вектор p^1 определяет направление спуска (в противном случае точка x^0 является стационарной, которая для квадратичной функции с положительно определенной матрицей является точкой минимума). Считаем, что $|p^1| \neq 0$, и, проводя исчерпывающий спуск в направлении вектора $p^1 = w^1$, по формуле (3.26) находим значение

$$\kappa_1 = \frac{|w^1|^2}{(Qw^1, w^1)} = \frac{|p^1|^2}{(Qp^1, p^1)}$$

и точку $x^1 = x^0 + \kappa_1 p^1$. Первая итерация рассматриваемого метода полностью совпадает с итерацией метода наискорейшего спуска.

На второй итерации вычисляем антиградиент $w^2 = -Qx^1 - b$ в точке x^1 . Пусть $|w^2| \neq 0$, так как в противном случае x^1 — точка минимума целевой функции. Проведем ортогонализацию системы

векторов p^1, w^2 относительно скалярного произведения $(x, y)_Q$. Для этого вычислим

$$p^2 = w^2 - \frac{(Qp^1, w^2)}{(Qp^1, p^1)} p^1. \quad (3.33)$$

Отметим, что $(w^2, p^1) = 0$, поскольку w^2 и p^1 — антиградиенты на двух последовательных итерациях исчерпывающего спуска. Следовательно, векторы w^2 и p^1 линейно независимы, а вектор p^2 отличен от нуля. Кроме того, вектор p^2 задает направление спуска из точки x^1 , так как с учетом ортогональности векторов w^2 и p^1 получаем

$$(\text{grad } f(x^1), p^2) = - (w^2, \gamma_1 p^1 + w^2) = - (w^2, w^2) = -|w^2|^2 < 0,$$

где γ_1 — коэффициент при p^1 в формуле (3.33).

Продолжая процесс исчерпывающего спуска вдоль очередного направления, полученного корректировкой антиградиента в текущей точке, в результате получим n сопряженных направлений спуска и достигнем точки минимума квадратичной функции через n итераций или даже раньше. Однако есть два важных уточнения. Во-первых, каждый очередной градиент w^k ортогонален не только предпоследнему направлению спуска p^{k-1} , но и всем остальным предшествующим направлениям $p^i, i = \overline{1, k-2}$. В самом деле, так как $w^j = -Qx^{j-1} - b$, $j = \overline{1, k}$, заключаем, что

$$w^k - w^{k-1} = -Q(x^{k-1} - x^{k-2}) = -\kappa_{k-1} Qp^{k-1}.$$

Поэтому при $k-1 > i$

$$(w^k, p^i) = (w^{k-1}, p^i) - \kappa_{k-1} (Qp^{k-1}, p^i) = (w^{k-1}, p^i).$$

Следовательно,

$$(w^k, p^i) = (w^{k-1}, p^i) = \dots = (w^{i+1}, p^i) = 0.$$

Из этих равенств, в частности, вытекает, что антиградиенты w^k и $w^i, k > i$, ортогональны, так как по построению антиградиент w^i является линейной комбинацией векторов p^1, \dots, p^i , ортогональных градиенту w^k .

Во-вторых, антиградиент w^k сопряжен с любым направлением спуска $p^i, i < k-1$. Действительно,

$$\kappa_i (Qp^i, w^k) = (w^i - w^{i+1}, w^k) = (w^i, w^k) - (w^{i+1}, w^k) = 0.$$

Это означает, что стандартная формула процесса ортогонализации

$$p^k = w^k - \sum_{i=1}^{k-1} \frac{(Qp^i, w^k)}{(Qp^i, p^i)} p^i$$

сокращается до двух слагаемых:

$$p^k = w^k - \frac{(Qp^{k-1}, w^k)}{(Qp^{k-1}, p^{k-1})} p^{k-1}.$$

Это и есть главный выигрыш при использовании в процессе ортогонализации последовательности антиградиентов.

Итак, подведем итоги. На первой итерации, выбрав начальную точку x^0 , находим $w^1 = -Qx^0 - b$, полагаем $p^1 = w^1$, $\kappa_1 = \frac{|p^1|^2}{(Qp^1, p^1)}$ и определяем $x^1 = x^0 + \kappa_1 p^1$. Затем последовательно вычисляем

$$\begin{cases} w^k = -Qx^{k-1} - b, \\ p^k = w^k - \frac{(Qp^{k-1}, w^k)}{(Qp^{k-1}, p^{k-1})} p^{k-1}, \\ x^k = x^{k-1} + \kappa_k p^k, \end{cases} \quad k = \overline{2, n}, \quad (3.34)$$

причем коэффициент κ_k находим из условия исчерпывающего спуска, например по формуле

$$\kappa_k = \frac{(w^k, p^k)}{(Qp^k, p^k)}. \quad (3.35)$$

При этом на каждой итерации выполняются соотношения

$$\begin{aligned} (w^k, w^i) &= 0, & k &\neq i; \\ (w^k, p^i) &= 0, & k &> i; \\ (Qp^k, p^i) &= 0, & k &\neq i. \end{aligned}$$

Пример 3.3. Рассмотрим описанный метод спуска на примере функции $f(x_1, x_2) = 6x_1^2 - 4x_1x_2 + 3x_2^2 + 4\sqrt{5}(x_1 + 2x_2) + 22$, выбрав в качестве начальной точку $x^0 = (-2, 1)$. На рис. 3.11, а показаны

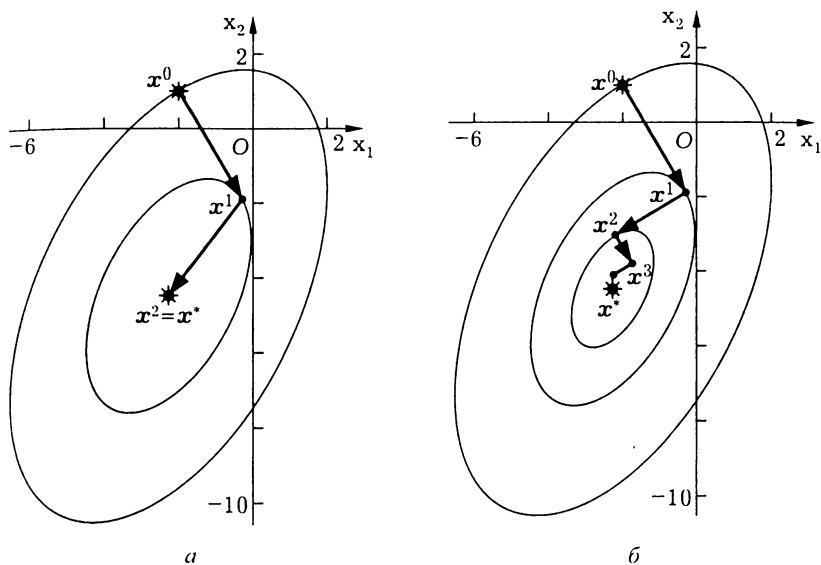


Рис. 3.11

две итерации поиска. Для сравнения на рис. 3.11, б приведены результаты поиска методом наискорейшего спуска. Обратим внимание на то, что результаты первой итерации для обоих методов одни и те же. Метод сопряженных направлений приводит в точку минимума функции уже на второй итерации, в то время как для достижения точки минимума методом наискорейшего спуска необходимо значительно больше итераций. #

Описанный метод сопряженных направлений можно использовать для неквадратичных функций. Например, учитывая, что для квадратичной функции (3.20) матрица Q является матрицей Гессе, в формулах (3.34) можно матрицу Q заменить матрицей Гессе минимизируемой функции. В результате получим метод второго порядка, состоящий в следующем. На первой итерации выбираем начальную точку x^0 и находим $w^1 = -\text{grad } f(x^0)$, $p^1 = w^1$. Затем последовательно вычисляем

$$\begin{cases} w^k = -\text{grad } f(x^{k-1}), \\ p^k = w^k + \gamma_k p^{k-1}, \\ x^k = x^{k-1} + \varkappa_k p^k, \end{cases} \quad k = 2, 3, \dots, \quad (3.36)$$

где коэффициент γ_k с помощью матрицы Гессе $H_k = H(x^{k-1})$ вычисляется по формуле

$$\gamma_k = -\frac{(H_k p^{k-1}, w^k)}{(H_k p^{k-1}, p^{k-1})}, \quad (3.37)$$

а параметр \varkappa_k определяется из условия исчерпывающего спуска (например, по формуле (3.35)). Процесс вычислений продолжается до тех пор, пока не будет выполнено условие останова.

Метод сопряженных направлений для квадратичных функций можно модифицировать так, что в нем не будет использоваться матрица Q . Например, учитывая, что для квадратичной функции выполняется равенство $w^{k-1} - w^k = \varkappa_{k-1} Q p^k$, $k > 1$, получаем

$$(Q p^{k-1}, w^k) = \frac{(w^{k-1} - w^k, w^k)}{\varkappa_{k-1}} = -\frac{|w^k|^2}{\varkappa_{k-1}}, \quad (3.38)$$

$$(Q p^{k-1}, p^{k-1}) = \frac{(w^{k-1} - w^k, p^{k-1})}{\varkappa_{k-1}} = \frac{(w^{k-1}, p^{k-1})}{\varkappa_{k-1}}. \quad (3.39)$$

Выполнив соответствующие замены в формулах (3.34), приходим к соотношениям (3.36), в которых параметр γ_k можно определить по формуле

$$\gamma_k = \frac{|w^k|^2}{(w^{k-1}, p^{k-1})}, \quad k = 2, 3, \dots \quad (3.40)$$

Такой способ построения сопряженных направлений спуска получил название *метода сопряженных градиентов*.

Формула (3.40) — не единственное соотношение, по которому можно определять коэффициенты γ_k . Учитывая, что $p^{k-1} = w^{k-1} + \gamma_{k-1} p^{k-2}$, а векторы p^{k-1} и p^{k-2} ортогональны, получаем $(p^{k-1}, w^{k-1}) = |w^{k-1}|^2 + \gamma_{k-1} (p^{k-2}, w^{k-1}) = |w^{k-1}|^2$, откуда

$$\gamma_k = \frac{|w^k|^2}{|w^{k-1}|^2}, \quad k = 2, 3, \dots \quad (3.41)$$

Использование равенства (3.41) для определения параметра γ_k дает вариант метода сопряженных градиентов, обычно называемый *методом Флетчера — Ривса*.

В силу ортогональности векторов w^k и w^{k-1} выполняется равенство $|w^k|^2 = (w^k - w^{k-1}, w^k)$, из которого получаем еще один вариант формулы для определения γ_k :

$$\gamma_k = \frac{(w^k - w^{k-1}, w^k)}{|w^{k-1}|^2}, \quad k = 2, 3, \dots \quad (3.42)$$

Использование этой формулы дает вариант метода сопряженных градиентов, который часто называют *методом Полака — Рибьера*.

Подчеркнем, что формулы (3.40)–(3.42) эквивалентны только в том случае, когда минимизируемая функция квадратичная. Для неквадратичной функции они приводят к разным итерационным процессам. При этом точку минимума не удастся найти за конечное число шагов, как это происходит в случае квадратичной функции, а сам процесс может оказаться расходящимся или „зацикливающимся“.

Сложности возникают и при определении параметра спуска γ_k . В общем случае этот параметр не выражается аналитически, и его необходимо находить тем или иным численным методом одномерной минимизации. Это вызывает неизбежные погрешности на каждой итерации, что может снижать скорость сходимости алгоритма или приводить к его расходимости. Чтобы ослабить влияние этих погрешностей, используют процедуру „обновления“ алгоритма, состоящую в том, что в итерационных соотношениях (3.36) периодически через заданное число итераций полагают $\gamma_k = 0$. Соответствующий номер итерации называют *моментом обновления алгоритма*, или *рестартом*. Обычно моменты обновления алгоритма выбирают кратными размерности n арифметического пространства. Использование в алгоритме рестартов позволяет избежать накопления вычислительных погрешностей и уменьшить вероятность построения после каждых n итераций линейно зависимых направлений спуска, но приводит к росту общего числа итераций, необходимых для достижения заданной точности поиска.

Пример 3.4. Используем метод сопряженных градиентов для минимизации функции $f(x_1, x_2) = (x_1^2 - x_2)^2 + (x_1 - 1)^2$, рассмотренной в примере 3.1. Зададим параметр точности поиска $\varepsilon_3 = 10^{-3}$ и начальную точку $x^0 = (-1, -2)$, в которой $f(x^0) = 13$.

Результаты вычислений, в которых параметр γ_k в рекуррентных соотношениях (3.36) определялся по формуле (3.41), приведены в табл. 3.2, а аналогичные результаты, в которых параметр γ_k рассчитан

Таблица 3.2

k	x^k	$f(x^k)$	κ_k	$ \text{grad } f(x^k) $
1	(0,379, -1,483)	3,031	0,086	3,475
2	(0,158, -0,103)	0,726	0,394	1,624
3	(0,792, 0,375)	0,107	0,432	0,631
4	(0,746, 0,494)	0,068	0,194	0,346
5	(0,978, 0,881)	0,006	1,028	0,828
6	(0,965, 0,924)	0,001	0,114	0,047
7	(0,974, 0,930)	0,001	0,232	0,045
8	(0,978, 0,960)	0,000	0,519	0,058
9	(0,999, 0,995)	0,000	0,286	0,012
10	(0,998, 0,996)	0,000	0,098	0,001
11	(0,999, 0,997)	0,000	0,525	0,002
12	(0,999, 0,999)	0,000	0,398	0,001
13	(1,000, 1,000)	0,000	0,166	0,000

по формуле (3.42), — в табл. 3.3. Для сравнения также даны результаты вычислений, в которых параметр γ_k определялся по формуле (3.37) (табл. 3.4). Траектории поиска точки минимума $x^* = (1, 1)$ для всех трех вариантов вычислений показаны на рис. 3.12: *a* — вариант с формулой (3.41); *b* — вариант с формулой (3.42); *c* — вариант с формулой (3.37).

Таблица 3.3

k	x^k	$f(x^k)$	κ_k	$ \text{grad } f(x^k) $
1	(0,379, -1,483)	3,031	0,086	3,475
2	(0,192, -0,064)	0,663	0,401	1,551
3	(0,893, 0,655)	0,032	0,508	0,406
4	(0,901, 0,818)	0,001	0,265	0,219
5	(0,936, 0,927)	0,000	0,133	0,334
6	(0,999, 0,997)	0,000	0,116	0,003
7	(1,000, 1,000)	0,000	0,377	0,001
8	(1,000, 1,000)	0,000	0,088	0,000

Таблица 3.4

k	x^k	$f(x^k)$	κ_k	$ \text{grad } f(x^k) $
1	(0,379, -1,483)	3,031	0,086	3,475
2	(0,531, 0,326)	0,222	0,472	1,034
3	(0,899, 0,686)	0,025	0,325	0,344
4	(0,990, 0,988)	0,000	0,443	0,050
5	(1,000, 1,000)	0,000	0,115	0,000

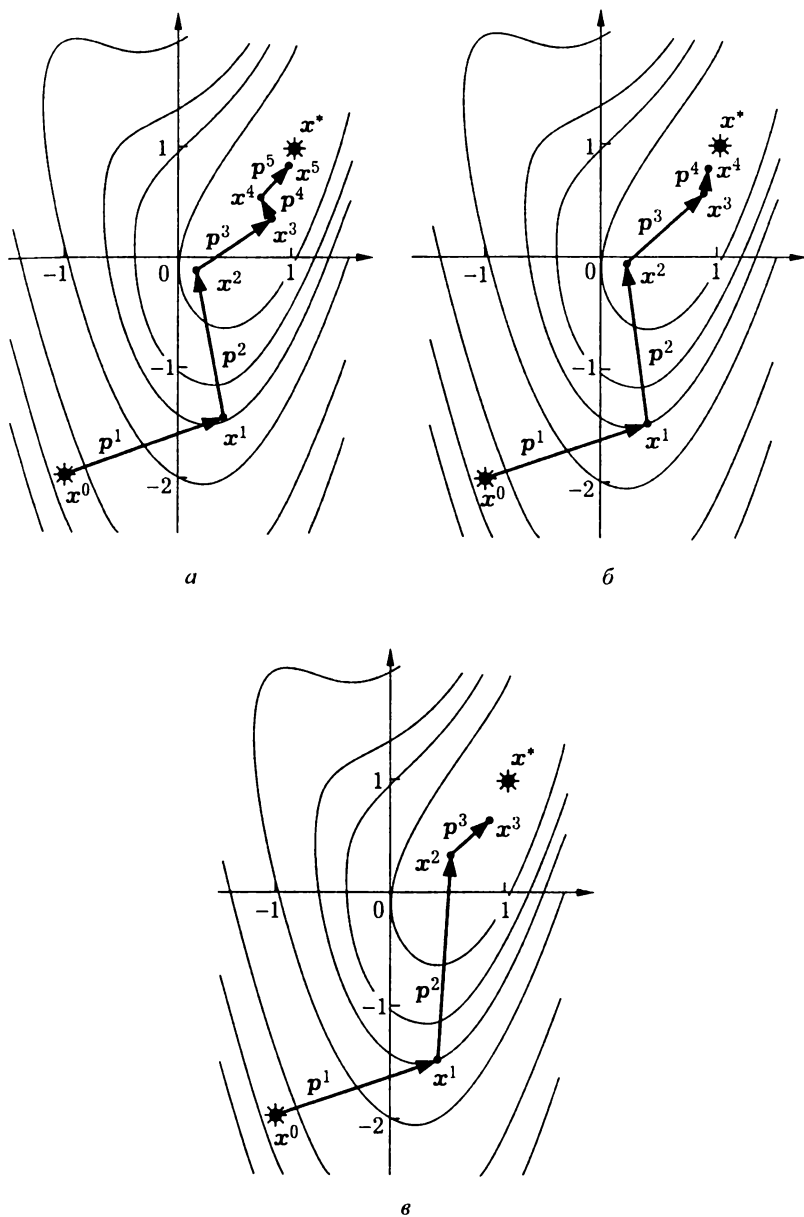


Рис. 3.12

Из приведенных результатов видно, что при поиске точки минимума наименьшее количество итераций N оказалось в варианте с формулой (3.37), а наибольшее — в варианте с формулой (3.41). Для сравнения отметим, что для достижения той же точности при использовании метода наискорейшего спуска требуется $N = 96$ итераций.

3.5. Метод Ньютона и его модификации

Метод Ньютона. Если целевая функция дважды дифференцируема в \mathbb{R}^n , то в процессе поиска точки минимума можно использовать информацию не только о градиенте этой функции, но и о ее матрице Гессе, повышая скорость сходимости итерационной последовательности. Алгоритмы такого поиска обычно относят к *методу Ньютона*. В простейшем варианте алгоритма на каждой итерации целевая функция аппроксимируется в окрестности текущей точки квадратичной функцией, а следующая точка релаксационной последовательности определяется как точка минимума аппроксимирующей квадратичной функции.

Аппроксимация целевой функции основана на использовании формулы Тейлора. Для минимизируемой функции $f(x)$, рассматриваемой на k -й итерации в окрестности текущей точки x^{k-1} , запишем формулу Тейлора с остаточным членом в форме Пеано:

$$f(x) = f(x^{k-1}) + (\text{grad } f(x^{k-1}), x - x^{k-1}) + \\ + \frac{1}{2} (H(x^{k-1})(x - x^{k-1}), x - x^{k-1}) + o(|x - x^{k-1}|^2).$$

Пренебрегая последним слагаемым в правой части, получаем квадратичную функцию

$$\varphi_k(x) = f(x^{k-1}) + (\text{grad } f(x^{k-1}), x - x^{k-1}) + \\ + \frac{1}{2} (H(x^{k-1})(x - x^{k-1}), x - x^{k-1}).$$

Если матрица Гессе $H(x^{k-1})$ целевой функции, вычисленная в точке x^{k-1} , является положительно определенной, то функция $\varphi_k(x)$ имеет единственную точку минимума, которую и выбирают в качестве следующей точки x^k итерационной последовательности. Эта точка

может быть найдена из условия, что градиент функции равен нулю. Записав равенство

$$\text{grad } \varphi_k(x) = \text{grad } f(x^{k-1}) + H(x^{k-1})(x - x^{k-1}) = 0,$$

получим

$$x^k = x^{k-1} - H^{-1}(x^{k-1}) \text{grad } f(x^{k-1}), \quad k \in \mathbb{N}. \quad (3.43)$$

Рекуррентное соотношение (3.43) можно трактовать как рекуррентное соотношение для метода Ньютона решения системы нелинейных уравнений $\text{grad } f(x) = 0$. Этим можно объяснить и название метода минимизации функции многих переменных, определяемого тем же рекуррентным соотношением.

В соотношение (3.43) входит матрица, обратная к матрице Гессе. При большом количестве n аргументов минимизируемой функции обращение матрицы Гессе* оказывается слишком трудоемким. В таком случае очередную точку x^k целесообразнее искать путем непосредственной минимизации функции $\varphi_k(x)$, например, методом сопряженных направлений или методом сопряженных градиентов. Подробнее этот вопрос рассмотрен ниже.

Точку минимума функции $\varphi_k(x)$ можно считать лишь вспомогательным приближением и, обозначив эту точку \tilde{x}^k , для построения релаксационной последовательности $\{x^k\}$ использовать рекуррентное соотношение

$$x^k = x^{k-1} + \kappa_k(\tilde{x}^k - x^{k-1}) = x^{k-1} + \kappa_k p^k, \quad k \in \mathbb{N}, \quad (3.44)$$

в котором значение $\kappa_k > 0$ можно выбрать различными способами.

Вектор $p^k = \tilde{x}^k - x^{k-1}$ задает на k -й итерации направление спуска, которое в случае положительно определенной матрицы $H(x^{k-1})$ называют *ньютоновским направлением спуска*. Действительно, с учетом соотношений (3.43) и (3.44) этот вектор можно представить в виде

$$p^k = -H^{-1}(x^{k-1}) \text{grad } f(x^{k-1}), \quad k \in \mathbb{N}, \quad (3.45)$$

так что

$$\begin{aligned} (\text{grad } f(x^{k-1}), p^k) &= \\ &= -(\text{grad } f(x^{k-1}), H^{-1}(x^{k-1}) \text{grad } f(x^{k-1})) < 0. \end{aligned}$$

* Фактически здесь речь идет не о непосредственном обращении матрицы Гессе, а о решении системы линейных уравнений $H(x^{k-1})x^k = H(x^{k-1})x^{k-1} - \text{grad } f(x^{k-1})$ относительно вектора неизвестных x^k .

Если в соотношении (3.44) выбрать $\kappa_k = 1$, то оно сведется к соотношению (3.43). Но можно использовать и другие значения параметра κ_k . Например, можно найти его как точку наименьшего значения функции $\varphi_k(\kappa) = f(x^{k-1} + \kappa p^k)$ или с помощью исчерпывающего спуска в направлении вектора p^k (напомним, что эти подходы могут привести к разным значениям). Для выбора значения κ_k можно также использовать метод дробления шага.

Если целевая функция является квадратичной с положительно определенной матрицей Q , то исчерпывающий спуск из произвольной начальной точки $x^0 \in \mathbb{R}^n$ в ньютоновском направлении приводит в точку минимума этой функции за одну итерацию. В самом деле, для квадратичной функции $f(x) = \frac{1}{2}(Qx, x) + (b, x)$ матрица Гессе совпадает с матрицей Q , а для ньютоновского направления спуска p^1 в начальной точке получаем

$$p^1 = -Q^{-1} \operatorname{grad} f(x^0) = -Q^{-1}(Qx^0 + b) = -x^0 - Q^{-1}b.$$

В результате первая итерация приводит в точку $x^1 = x^0 + p^1 = -Q^{-1}b$, оказывающуюся точкой минимума квадратичной функции,

Для неквадратичной функции ньютоновское направление в общем случае не приводит в точку минимума, хотя часто оказывается, что использование этого направления дает лучшее приближение, чем направление антиградиента. Если это так, то спуск в ньютоновском направлении за одну итерацию позволяет достичь более существенного убывания минимизируемой неквадратичной функции и получить более близкое приближение к искомой точке x^* минимума по сравнению со спуском в направлении антиградиента.

Если график целевой функции имеет овражную структуру, то вектор, определяющий ньютоновское направление спуска, может составлять с осью оврага меньший угол, чем антиградиент. Эта особенность указывает на то, что при минимизации овражных функций метод Ньютона более эффективен, чем метод градиентного спуска.

Пример 3.5. Сравним метод Ньютона в варианте (3.44) с $\kappa_k = 1$ и метод наискорейшего спуска на примере функции $f(x_1, x_2) = (x_1^2 - x_2)^2 + (x_1 - 1)^2$. В качестве начальной выберем точку $x^0 = (-1, -2)$ (в этой точке $f(x^0) = 13$) и зададим параметр точности поиска $\varepsilon_3 = 10^{-3}$.

Методом Ньютона (рис. 3.13, а) приближенное значение точки минимума $x^* \approx (0,999, 0,999)$ получено за $N = 5$ итераций, в то время как метод наискорейшего спуска (рис. 3.13, б) позволяет достигнуть приближенное значение точки минимума $x^* \approx (0,999, 0,998)$ за $N = 96$ итераций. Из результатов вычислений видно, насколько эффективнее может быть метод Ньютона по сравнению с методом наискорейшего спуска.

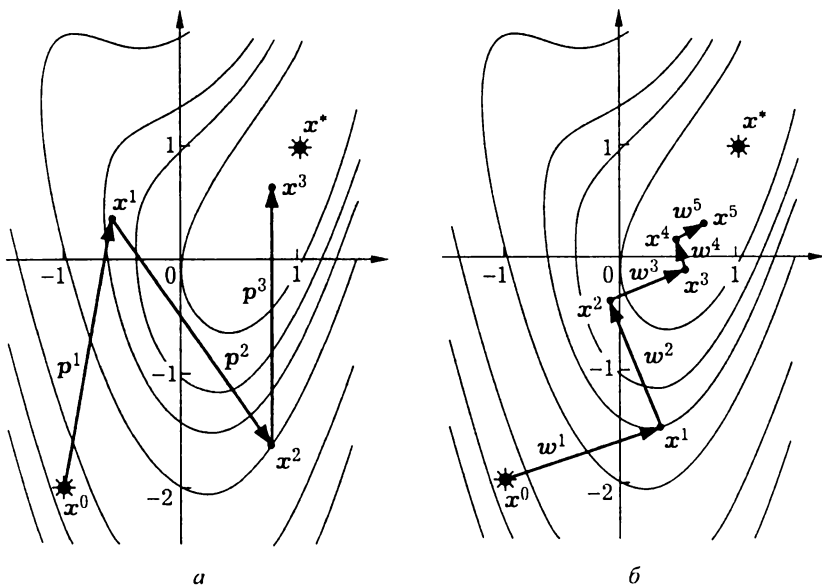


Рис. 3.13

Обратим внимание на то, что в методе Ньютона $f(x^1) < f(x^2)$ (см. рис. 3.13, а), т.е. метод Ньютона привел к последовательности, не являющейся релаксационной. Такое может происходить, если $\kappa_k = 1$ или начальная точка находится на значительном удалении от точки минимума. #

Сходимость метода Ньютона существенно зависит от выбора начального приближения. Можно показать, что если целевая функция сильно выпуклая и для любых точек $x, y \in \mathbb{R}^n$ для матрицы Гессе $H(x)$ целевой функции выполнено неравенство $\|H(x) - H(y)\| \leq L\|x - y\|$, $L > 0$, а начальное приближение выбрано удачно (точка x^0 расположена достаточно близко к точке x^* минимума), то алгоритм

метода Ньютона при значении $\varkappa_k = 1$ в соотношении (3.44) обладает *квадратичной скоростью сходимости*, т.е. справедлива оценка

$$|x^k - x^*| \leq C|x^{k-1} - x^*|^2, \quad k \in \mathbb{N}, \quad C = \text{const.}$$

Это позволяет для такой функции в качестве условия прекращения процесса поиска использовать неравенство $|x^k - x^{k-1}| < \varepsilon_1$, где ε_1 — *параметр точности поиска*. Но если целевая функция не является сильно выпуклой или начальное приближение находится далеко от искомой точки x^* , метод Ньютона может расходиться.

Квадратичная скорость сходимости, а также возможность контроля за соблюдением достаточного условия минимума целевой функции $f(x)$ на каждой k -й итерации с помощью матрицы Гессе $H(x^{k-1})$ этой функции способствуют высокой эффективности рассматриваемого метода. Однако при его практической реализации возникают две проблемы.

Первая из них — это сохранение положительной определенности матрицы Гессе $H(x^{k-1})$ целевой функции на каждой k -й итерации, так как иначе вектор $p^k = -H^{-1}(x^{k-1}) \text{grad } f(x^{k-1})$ может не определять направление спуска. Более того, матрица $H(x^{k-1})$ может быть вырожденной и не иметь обратной матрицы. Эту проблему можно решить, если направление спуска задавать вектором

$$\tilde{p}^k = -(\eta_k I + H(x^{k-1}))^{-1} \text{grad } f(x^{k-1}),$$

где I — единичная матрица порядка n , а $\eta_k \geq 0$ — параметр, выбираемый так, чтобы в точке x^{k-1} матрица

$$\tilde{H}_k = \eta_k I + H(x^{k-1}), \quad k \in \mathbb{N}, \quad (3.46)$$

была положительно определена. Такой способ выбора направления спуска часто называют *методом Марквардта*.

Вторая проблема — обращение на каждой итерации матрицы порядка n , а это при больших n — достаточно трудоемкая операция. В действительности вычислять обратную матрицу нет необходимости: вектор \tilde{p}^k проще найти как решение системы линейных алгебраических уравнений (СЛАУ)

$$\tilde{H}_k \tilde{p}^k = -\text{grad } f(x^{k-1}). \quad (3.47)$$

Систему (3.47) можно решить различными численными методами, как прямыми, так и итерационными. Можно также решать систему линейных уравнений путем минимизации квадратичной функции

$$\tilde{\varphi}_k(x) = \frac{1}{2}(\tilde{H}_k x, x) + (\text{grad } f(x^{k-1}), x)$$

методом сопряженных направлений или методом сопряженных градиентов, поскольку выполнение (3.47) является необходимым и достаточным условием минимума такой функции.

Опишем вариант метода Ньютона, в котором направление спуска определяется путем решения СЛАУ (3.47). Предварительно выбираем начальную точку $x^0 \in \mathbb{R}^n$ и параметр точности поиска $\varepsilon_3 > 0$. Полагая $k = 1$ и переходим к основной части алгоритма.

1. В точке x^{k-1} находим градиент $\text{grad } f(x^{k-1})$ и матрицу Гессе $H(x^{k-1})$ целевой функции. Проверяем условие останова $|\text{grad } f(x^{k-1})| < \varepsilon_3$. Если оно выполнено, итерации прекращаем, принимая* $x^* \approx x^{k-1}$ и $f(x^*) \approx f(x^{k-1})$. Если условие останова не выполнено, проверяем, является ли матрица $H(x^{k-1})$ положительно определенной. Если да, полагаем $\tilde{H}_k = H(x^{k-1})$ и переходим к п. 3, в противном случае переходим к п. 2.

2. Подбираем значение $\eta_k > 0$, при котором матрица \tilde{H}_k (3.46) будет положительно определенной, и переходим к п. 3.

3. Из решения СЛАУ (3.47) находим вектор p^k и затем точку $x^k = x^{k-1} + p^k$. Увеличиваем номер k на единицу и возвращаемся к п. 1.

Модификации метода Ньютона. Собственно метод Ньютона без условной минимизации можно описать рекуррентным соотношением (3.44), в котором вектор p^k задает ньютоновское направление спуска и определяется равенством (3.45), а $\varkappa_k = 1$.

Поскольку ньютоновское направление является направлением спуска, то и метод Ньютона можно рассматривать в ряду других методов спуска. В этом контексте возникает идея использовать альтернативные способы определения шага спуска, т.е. вместо значения $\varkappa_k = 1$ использовать другие значения. Стремление модифицировать

Утверждать, что x^ — точка минимума, можно лишь, если матрица $H(x^{k-1})$ положительно определенная. Иначе необходимо проводить дополнительное исследование поведения функции в окрестности точки x^* .

метод Ньютона не в последнюю очередь объясняется тем, что метод Ньютона очень чувствителен к выбору начальной точки. Отметим две модификации метода Ньютона, различающиеся выбором параметра κ_k .

Одна из указанных модификаций метода Ньютона опирается на метод дробления шага, используемый для определения параметра κ_k . В этой модификации на каждой итерации параметр κ_k выбирается так, чтобы выполнялось неравенство

$$f(x^{k-1}) - f(x^k) \geq -\omega \kappa_k (\text{grad } f(x^{k-1}), p^k), \quad k \in \mathbb{N}, \quad (3.48)$$

где $\omega \in (0, 1/2)$ — некоторый фиксированный параметр. Подбор параметра κ_k осуществляется следующим образом. Сначала полагаем $\kappa_k = 1$. Затем проверяем условие (3.48). Если оно нарушено, то корректируем значение κ_k , умножая его на постоянный коэффициент дробления шага ν . Снова проверяем условие (3.48) и снова корректируем κ_k , если условие нарушено. Процесс корректировки продолжаем до тех пор, пока условие (3.48) не будет выполнено. Ньютоновское направление спуска можно определять, например, путем решения системы (3.47).

Другая модификация метода Ньютона основана на применении исчерпывающего спуска в ньютоновском направлении. Такая модификация может привести к довольно большому объему вычислений, так как трудоемкой является минимизация функции одного переменного $\varphi_k(\kappa) = f(x^{k-1} + \kappa p^k)$.

Пример 3.6. Сравним описанные модификации метода Ньютона на примере минимизации функции $f(x_1, x_2) = (x_1^2 - x_2)^2 + (x_1 - 1)^2$ — той же, что и в примере 3.5. В качестве начальной выберем точку $x^0 = (-1, -2)$ и зададим параметр точности поиска $\varepsilon_3 = 10^{-3}$.

Процесс поиска точки минимума $x^* = (1, 1)$ целевой функции $f(x_1, x_2)$ для обеих модификаций метода Ньютона показан на рис. 3.14: *a* — спуск с дроблением шага ($\nu = 0,5, \omega = 0,25$); *б* — исчерпывающий спуск. Результаты вычислений приведены в табл. 3.5, в которой двум указанным модификациям соответствуют колонки *a* и *б*. В колонке *в* содержатся результаты расчетов „классическим“ методом Ньютона (т.е. с выбором в рекуррентном соотношении (3.44) значения $\kappa_k = 1$).

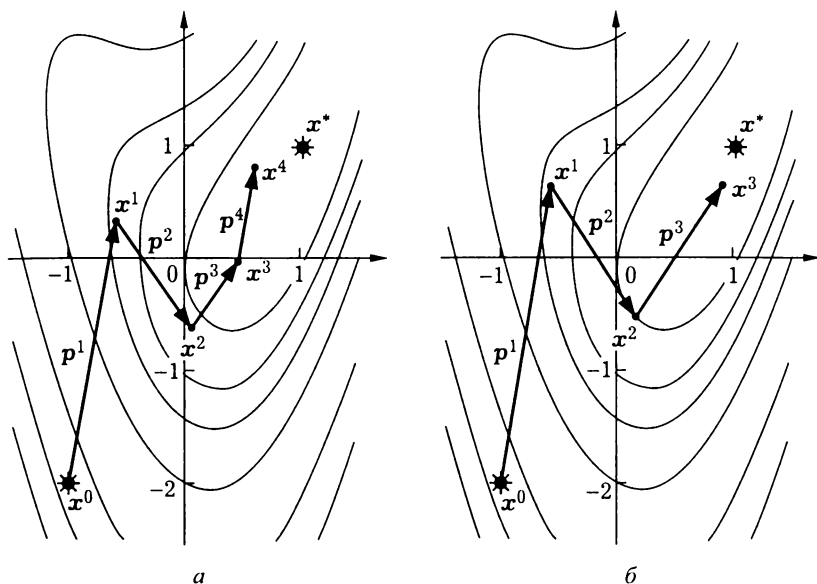


Рис. 3.14

Таблица 3.5

k	a	$б$	$в$
1	(-0,7143, 0,4286)	(-0,6888, 0,6455)	(-0,7143, 0,4286)
2	(0,0226, -0,5832)	(0,1152, -0,5155)	(0,7594, -1,5951)
3	(0,4735, 0,0209)	(0,9260, 0,6683)	(0,8044, 0,6451)
4	(0,8478, 0,5786)	(0,9821, 0,9697)	(0,9992, 0,9605)
5	(0,9667, 0,9203)	(1,0000, 1,0000)	(1,0000, 1,0000)
6	(0,9991, 0,9971)		
7	(1,0000, 1,0000)		

Из результатов расчетов видно, что метод дробления шага при спуске в ньютоновском направлении дает результаты хуже, чем метод исчерпывающего спуска и метод спуска с ньютоновским шагом. При одном и том же значении $\varepsilon_3 = 10^{-3}$ в вариантах $б$ и $в$ требуется всего пять итераций, в то время как в варианте $а$ — семь итераций. #

Две описанные модификации метода Ньютона интересны тем, что они менее чувствительны к выбору начальной точки по сравнению с „классическим“ методом Ньютона, но при этом сохраняют высокую скорость сходимости. Можно показать, что если целевая функция

дважды непрерывно дифференцируема, ее матрица Гессе $H(x)$ в некоторой окрестности $U(x^*)$ точки минимума x^* удовлетворяет условию Липшица

$$\|H(x) - H(y)\| \leq L|x - y|, \quad x, y \in U(x^*), \quad (3.49)$$

а также условию* $(H(x)h, h) \geq \mu|h|^2$, $x \in U(x^*)$, $h \in \mathbb{R}^n$, то при попадании очередной точки x^{k-1} в окрестность $U(x^*)$ верна оценка

$$|x^k - x^*| \leq \frac{L}{\mu} |x^{k-1} - x^*|^2. \quad (3.50)$$

Реализация указанных модификаций метода Ньютона наталкивается на те же трудности, которые характерны и для „классического“ метода Ньютона. Главная из них состоит в том, что на каждой итерации необходимо для определения ньютоновского направления спуска обращать матрицу Гессе или решать систему линейных уравнений, матрицей которой является матрица Гессе. И то, и другое — трудоемкая вычислительная задача. Один из способов преодоления этой трудности — использование для выбора направления спуска матрицы Гессе, найденной на одной из предыдущих итераций, что сокращает количество операций обращения матрицы. В простейшем варианте этого подхода матрицу Гессе обращают один раз в начальной точке и для определения вектора, указывающего направление спуска на k -й итерации, используют формулу

$$p^k = -H^{-1}(x^0) \operatorname{grad} f(x^{k-1}), \quad k \in \mathbb{N}. \quad (3.51)$$

Если матрица Гессе $H(x^0)$ целевой функции, вычисленная в начальной точке x^0 , не является положительно определенной, то формулу (3.51) можно модифицировать, заменив матрицу $H(x^0)$ матрицей $\tilde{H}_1 = \eta_1 I + H(x^0)$, где параметр η_1 подобран так, что \tilde{H}_1 будет положительно определенной.

Описанный подход приводит к методу с *линейной скоростью* локальной *сходимости*, которая характеризуется оценкой

$$|x^k - x^*| \leq q|x^{k-1} - x^*|, \quad k \in \mathbb{N}, \quad (3.52)$$

Это условие означает, что функция $f(x)$ сильно выпукла в $U(x^)$.

с некоторой константой $q > 0$. Выбор начальной точки x^0 влияет на параметр q : чем ближе x^0 к искомой точке x^* , тем меньше q .

Промежуточная стратегия, при которой матрица Гессе обновляется через фиксированное количество итераций (т.е. на первых m итерациях в формуле (3.51) используется матрица $H^{-1}(x^0)$, на следующих m итерациях — матрица $H^{-1}(x^m)$ и т.д.), приводит к модификации метода Ньютона, обладающей *сверхлинейной скоростью сходимости* (т.е. расстояние до точки минимума убывает быстрее геометрической прогрессии). Оценку скорости сходимости в этом случае можно получить, комбинируя оценку (3.50) квадратичной скорости сходимости, которая выполняется через каждые m итераций, с оценкой (3.52) линейной скорости сходимости, действующей на промежуточных итерациях. Это комбинирование дает оценку

$$|x^{km} - x^*| \leq C |x^{(k-1)m} - x^*|^{m+1}, \quad k \in \mathbb{N},$$

где $C > 0$ — некоторая постоянная. Отсюда видно, что при периодическом обновлении матрицы Гессе скорость сходимости итерационной последовательности возрастает с уменьшением периода обновления и дает квадратичную скорость сходимости при $m = 1$, когда матрица Гессе обращается на каждой итерации. Если период m возрастает, то скорость сходимости убывает, приближаясь к линейной.

3.6. Квазиньютоновские методы

В многомерной минимизации выделяются методы, которые объединяют в себе достоинства как метода наискорейшего спуска, так и метода Ньютона. Это так называемые *квазиньютоновские методы*. В этих методах не требуется обращение матрицы Гессе целевой функции, но в то же время сохраняется высокая скорость сходимости итерационной последовательности, присущая методу Ньютона и его модификациям.

Релаксационную последовательность $\{x^k\}$ в квазиньютоновских методах строят в соответствии с общим рекуррентным соотношением (3.27), в котором вектор p^k , задающий на k -й итерации направление спуска, определяют равенством

$$p^k = A_k w^k, \quad k \in \mathbb{N}. \quad (3.53)$$

Здесь $w^k = -\text{grad } f(x^{k-1})$ — антиградиент целевой функции в точке x^{k-1} , а A_k — положительно определенная матрица порядка n , которая на каждой итерации вычисляется по определенному алгоритму. Обратим внимание на то, что в силу положительной определенности матрицы A_k вектор p^k действительно задает направление спуска, поскольку

$$(\text{grad } f(x^{k-1}), p^k) = -(w^k, A_k w^k) = -(A_k w^k, w^k) < 0.$$

Процедуру вычисления матриц A_k строят таким образом, чтобы последовательность $\{A_k\}$ при $k \rightarrow \infty$ сходилась к матрице $H^{-1}(x^*)$, где $H(x^*)$ — матрица Гессе целевой функции, вычисленная в точке минимума x^* этой функции. На первой итерации обычно полагают $A_1 = I$, т.е. первая итерация выполняется по схеме градиентного спуска.

Последовательность матриц A_k можно строить различными способами. В частности, соотношение $A_k = H^{-1}(x^{k-1})$ приводит к методу Ньютона или одной из его модификаций. Таким образом, квазиньютоновские методы можно рассматривать как обобщение „классического“ метода Ньютона.

Параметр \varkappa_k в рекуррентном соотношении (3.27) можно выбирать по любой из типовых схем (задать постоянное значение $\varkappa_k = 1$, применять метод дробления шага), но чаще всего используют исчерпывающий спуск в направлении вектора p^k .

В силу сходимости последовательности $\{A_k\}$ к матрице $H^{-1}(x^*)$ матрицы A_k на завершающей стадии поиска близки к матрицам $H^{-1}(x^{k-1})$, используемым в методе Ньютона. Исходя из этого можно ожидать, что квазиньютоновские методы сохраняют высокую скорость сходимости, присущую методу Ньютона. Сходимость указывает и на то, как можно строить матрицы A_k : они должны быть близки к матрицам $H^{-1}(x^{k-1})$, и их естественно получать, тем или иным образом аппроксимируя матрицы $H^{-1}(x^{k-1})$.

Выбор удачной аппроксимации может существенно сократить объем вычислений по сравнению с обращением матрицы Гессе и тем самым упростить процедуру построения очередного направления спуска. Именно эти соображения лежат в основе всех квазиньютоновских методов.

Каждую очередную матрицу A_{k+1} строят, корректируя определенным способом ранее вычисленную матрицу A_k . Этот процесс можно описать рекуррентным соотношением

$$A_{k+1} = A_k + \Delta A_k, \quad k \in \mathbb{N}, \quad (3.54)$$

где ΔA_k — положительно определенная матрица порядка n , называемая *поправочной*. Способ выбора матрицы ΔA_k определяет и конкретный квазиньютоновский метод. На первой итерации, как отмечено, обычно принимают $A_1 = I$.

Пусть целевая функция является квадратичной функцией $f(x) = \frac{1}{2}(Qx, x) + (b, x)$ с положительно определенной матрицей Q . Тогда в каждой точке x матрица Гессе совпадает с матрицей Q , а $\text{grad } f(x) = Qx + b$. Для приращения антиградиента $\Delta w^k = w^k - w^{k-1}$, соответствующего приращению $\Delta x^k = x^k - x^{k-1}$ векторного аргумента функции, имеем

$$\Delta w^k = \text{grad } f(x^{k-1}) - \text{grad } f(x^k) = Q(x^{k-1} - x^k) = -Q\Delta x^k.$$

Таким образом, для квадратичной целевой функции выполняется равенство

$$Q^{-1}\Delta w^k = -\Delta x^k, \quad k \in \mathbb{N}. \quad (3.55)$$

Имея в виду это равенство, в общем случае потребуем, чтобы матрицы A_{k+1} удовлетворяли так называемому *квазиньютоновскому условию*

$$A_{k+1}\Delta w^k = -\Delta x^k, \quad k \in \mathbb{N}. \quad (3.56)$$

Для поправочных матриц это условие имеет вид

$$\Delta A_k \Delta w^k = -\Delta x^k - A_k \Delta w^k.$$

Непосредственной проверкой можно убедиться, что этому условию удовлетворяют матрицы

$$\Delta A_k = -\frac{\Delta x^k y^T}{(\Delta w^k, y)} - \frac{A_k \Delta w^k z^T}{(\Delta w^k, z)},$$

где $y, z \in \mathbb{R}^n$ можно выбрать произвольно. Полагая $y = \Delta x^k$ и $z = A_k w^k$ и учитывая (3.54), получаем

$$A_{k+1} = A_k - \frac{\Delta x^k (\Delta x^k)^T}{(\Delta w^k, \Delta x^k)} - \frac{A_k \Delta w^k (\Delta w^k)^T A_k^T}{(A_k \Delta w^k, \Delta w^k)}, \quad k \in \mathbb{N}. \quad (3.57)$$

Выбор матриц A_k по формуле (3.57) в сочетании с исчерпывающим спуском по направлениям векторов p^k — основа квазиньютоновского метода, называемого *методом Давидона — Флетчера — Пауэлла*, или *ДФП-методом*.

При построении матриц A_k в соответствии с соотношением (3.57) свойство положительной определенности сохраняется.

При минимизации квадратичной функции с положительно определенной матрицей ДФП-метод сводится к методу сопряженных направлений, поскольку первые n векторов p^k , задающие направления спуска в этом методе, являются сопряженными относительно матрицы Q . Таким образом, ДФП-метод не более чем за n итераций дает точное решение задачи безусловной минимизации квадратичной функции.

Матрицы A_k , вычисленные по ДФП-методу, связаны равенствами

$$A_k Q p^i = p^i, \quad i = \overline{1, k}, \quad k = \overline{1, n}.$$

При $k = n$ из этих равенств следует, что n векторов $p^i, i = \overline{1, n}$, являются собственными для симметрической матрицы $A_n Q$, а соответствующие собственные значения равны единице. Следовательно, $A_n Q = I$ и $A_n = Q^{-1}$, т.е. матрица A_n оказывается обратной к матрице Гессе $H(x^*) = Q$ квадратичной функции, вычисленной в точке минимума x^* . На рис. 3.15 показаны две итерации ДФП-метода для целевой функции двух переменных. Из сказанного выше ясно, почему траектория ДФП-метода совпала с траекторией метода сопряженных направлений (см. рис. 3.11, а).

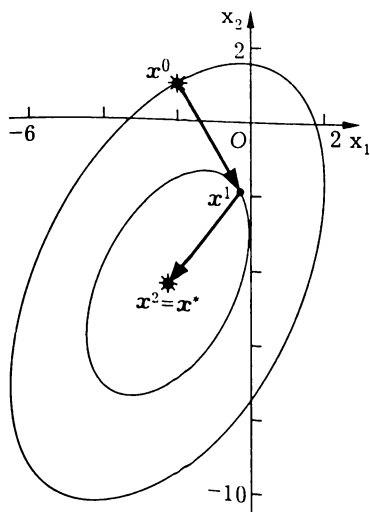


Рис. 3.15

Если целевая функция не является квадратичной, использование ДФП-метода в общем случае не позволяет найти точку минимума за конечное число итераций. Как и в методе сопряженных направлений, чтобы ослабить влияние накапливаемых погрешностей на сходимость итерационной последовательности и уменьшить вероятность появления после очередных n итераций линейно зависимых направлений спуска, в ДФП-методе применяют процедуру „обновления“ алгоритма, в соответствии с которой через каждые n итераций в качестве матрицы A_k используют единичную матрицу.

Выполнение процедуры „обновления“ алгоритма увеличивает общее число итераций, необходимых для достижения заданной точности поиска. Поэтому на практике используют варианты ДФП-метода, в которых (как и в методе сопряженных направлений) „обновление“ не используется. Поясним суть такого подхода на конкретном примере.

Пример 3.7. С помощью ДФП-метода найдем минимум функции $f(x_1, x_2) = (x_1^2 - x_2)^2 + (x_1 - 1)^2$ (см. примеры 3.1, 3.4–3.6). В качестве начальной выберем точку $x^0 = (-1, -2)$ со значением функции $f(x^0) = 13$ и зададим параметр точности поиска $\varepsilon_3 = 10^{-3}$.

Процесс поиска точки минимума $x^* = (1, 1)$ показан на рис. 3.16. Первая итерация совпадает с первой итерацией как по методу сопряженных направлений (см. рис. 3.12, а, б), так и по методу наискорейшего спуска. В табл. 3.6 приведены координаты точек x^k итерационной последовательности, построенной ДФП-методом, значения $f(x^k)$ целевой функции в этих точках, а также матрицы A_k и матрицы, обратные к матрицам Гессе $H(x^k)$ целевой функции.

Из результатов вычислений видно, что заданная точность поиска минимума достигнута за 7 итераций, матрицы A_7 и $H^{-1}(x^7)$ на конечном, седьмом шаге поиска близки друг к

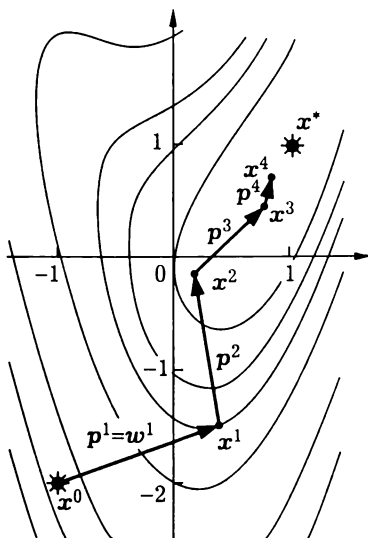


Рис. 3.16

Таблица 3.6

k	x^k	$f(x^k)$	A_k	$H^{-1}(x^k)$
1	(0,3787, -1,4830)	3,0312	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0,071 & -0,143 \\ -0,143 & 0,786 \end{pmatrix}$
2	(0,1584, -0,1027)	0,7246	$\begin{pmatrix} 0,100 & -0,127 \\ -0,127 & 0,986 \end{pmatrix}$	$\begin{pmatrix} 0,118 & 0,089 \\ 0,089 & 0,567 \end{pmatrix}$
3	(0,8593, 0,5572)	0,0526	$\begin{pmatrix} 0,073 & -0,005 \\ -0,005 & 0,456 \end{pmatrix}$	$\begin{pmatrix} 0,398 & 0,126 \\ 0,126 & 0,540 \end{pmatrix}$
4	(0,8707, 0,7660)	0,0168	$\begin{pmatrix} 0,382 & 0,382 \\ 0,382 & 0,782 \end{pmatrix}$	$\begin{pmatrix} 0,367 & 0,631 \\ 0,631 & 1,584 \end{pmatrix}$
5	(0,9942, 0,9708)	$3,4 \cdot 10^{-4}$	$\begin{pmatrix} 0,272 & 0,481 \\ 0,481 & 1,351 \end{pmatrix}$	$\begin{pmatrix} 0,508 & 0,885 \\ 0,885 & 2,040 \end{pmatrix}$
6	(0,9980, 0,9964)	$4,3 \cdot 10^{-6}$	$\begin{pmatrix} 0,494 & 0,914 \\ 0,914 & 2,160 \end{pmatrix}$	$\begin{pmatrix} 0,483 & 0,960 \\ 0,960 & 2,410 \end{pmatrix}$
7	(0,9999, 0,9999)	$3,3 \cdot 10^{-10}$	$\begin{pmatrix} 0,512 & 1,016 \\ 1,016 & 2,516 \end{pmatrix}$	$\begin{pmatrix} 0,500 & 0,999 \\ 0,999 & 2,494 \end{pmatrix}$

другу. Матрица $H^{-1}(x^*)$, вычисленная в точке минимума x^* , имеет вид

$$H^{-1}(x^*) = \begin{pmatrix} 0,5 & 1,0 \\ 1,0 & 2,5 \end{pmatrix}$$

и также мало отличается от A_7 и $H^{-1}(x^7)$. Отметим, что для достижения той же точности в методе наискорейшего спуска потребовалось 96 итераций. #

Возможны и другие способы построения последовательности матриц A_k . Так, например, в *методе Бройдена — Флетчера — Шенно (БФШ-методе)* используют соотношение

$$A_{k+1} = A_k - \frac{\Delta x^k (\Delta x^k)^T}{(\Delta w^k, \Delta x^k)} - \frac{A_k \Delta w^k (\Delta w^k)^T A_k^T}{\rho_k} + \rho_k r^k (r^k)^T, \quad k \in \mathbb{N}, \quad (3.58)$$

где

$$r^k = \frac{A_k \Delta w^k}{\rho_k} - \frac{\Delta x^k}{(\Delta x^k, \Delta w^k)}, \quad \rho_k = (A_k \Delta w^k, \Delta w^k),$$

а в *методе Пауэлла* —

$$A_{k+1} = A_k - \frac{\Delta \tilde{x}^k (\Delta \tilde{x}^k)^T}{(\Delta w^k, \Delta \tilde{x}^k)}, \quad k \in \mathbb{N}.$$

где $\Delta \tilde{x}^k = \Delta x^k + A_k \Delta w^k$, $k \in \mathbb{N}$. В этих методах, как и в ДФП-методе, в качестве A_1 обычно выбирают единичную матрицу.

Можно показать, что любой из рассмотренных способов построения матриц A_k сохраняет свойство положительной определенности, а последовательность $\{A_k\}$ при $k \rightarrow \infty$ сходится к $H^{-1}(x^*)$.

Пример 3.8. Сравним различные методы на примере минимизации функции $f(x_1, x_2) = (x_1^2 - x_2)^2 + (x_1 - 1)^2$, рассмотренной в ряде предыдущих примеров. Выберем начальную точку $x^0 = (-1, -2)$ со значением функции $f(x^0) = 13$ и зададим параметр точности поиска $\varepsilon_3 = 10^{-3}$. Как и в примере 3.7, мы не будем использовать „обновление“ алгоритма через фиксированное число шагов.

В табл. 3.7 приведены координаты точек x^k , найденных двумя квазиньютоновскими методами. Траектории поиска точки минимума для этих методов показаны на рис. 3.17 (а — БФШ-метод; б — метод Пауэлла).

Таблица 3.7

N	БФШ-метод	Метод Пауэлла
1	(0,379, -1,483)	(0,379, -1,483)
2	(0,158, -0,103)	(0,158, -0,103)
3	(0,859, 0,557)	(0,859, 0,557)
4	(0,871, 0,766)	(0,871, 0,766)
5	(0,994, 0,971)	(0,934, 0,871)
6	(0,998, 0,996)	(0,998, 0,989)
7	(0,999, 0,999)	(0,999, 0,999)
8		(0,999, 0,999)

Сравнительный анализ результатов показывает, что для рассматриваемой функции наилучший результат по количеству итераций, требуемых для достижения заданной точности, дают ДФП-метод и БФШ-метод. Метод Пауэлла по этому критерию эффективности им уступает. #

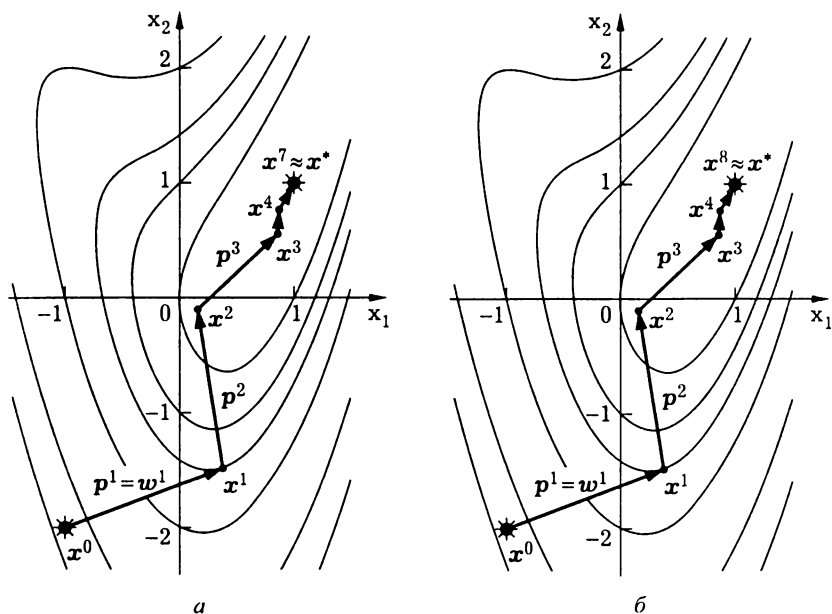


Рис. 3.17

Отметим, что сравнение эффективности алгоритмов минимизации принято проводить на специально сконструированных функциях. Графики этих функций имеют четко выраженную овражную структуру. На рис. 3.18 представлены линии уровня унимодальной функции Розенброка

$$f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2, \quad (3.59)$$

а на рис. 3.19 — функции Химмельблау

$$f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2, \quad (3.60)$$

имеющей четыре точки минимума.

Для испытания алгоритмов используют также унимодальную функцию Пауэлла

$$f(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4, \quad (3.61)$$

достигающую минимума в точке $x^* = (0, 0, 0, 0)$.

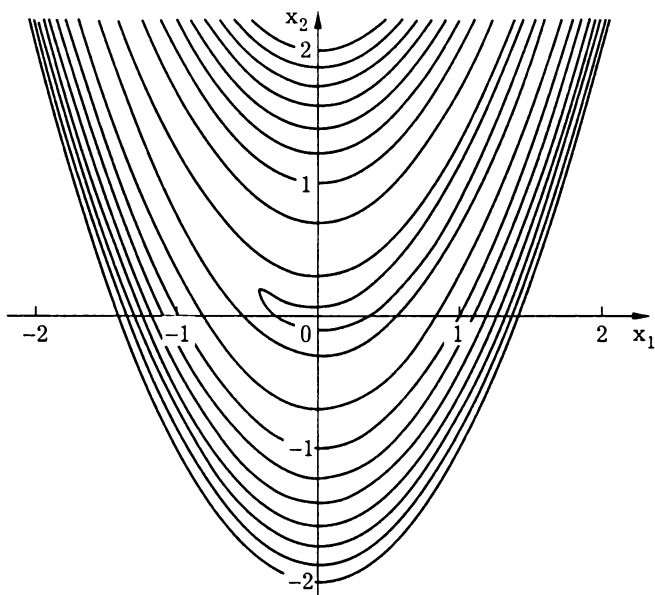


Рис. 3.18

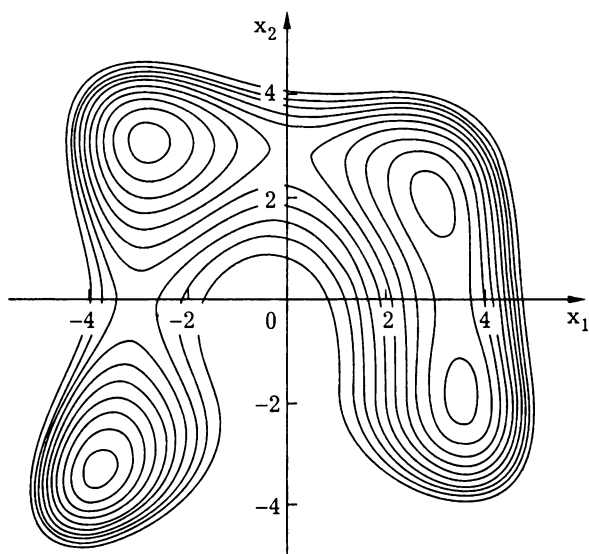


Рис. 3.19

3.7. Методы прямого поиска

В методах прямого поиска (методах нулевого порядка) используют информацию только о значениях целевой функции. Многие из этих методов не имеют теоретического обоснования и построены на эвристических соображениях. Вопросы их сходимости мало изучены, а оценки скорости сходимости обычно отсутствуют. Вместе с тем методы прямого поиска идейно связаны с методами первого и второго порядков, что в ряде случаев позволяет оценивать их эффективность для некоторых классов функций.

Распространенными способами оценки эффективности методов прямого поиска являются вычислительные эксперименты и сравнительный анализ методов по результатам таких экспериментов. Однако следует учитывать, что этот анализ не всегда приводит к однозначным выводам о преимуществах одного метода перед другим.

Это связано с тем, что, во-первых, сравнению обычно подвергаются не сами методы, а конкретные алгоритмы (программы), в которых реализованы эти методы. Хороший метод можно „загубить“ плохим программированием, неудачным выбором параметров алгоритма. Во-вторых, итерационная последовательность может иметь разное поведение на разных этапах процесса минимизации. Удовлетворительного способа учета указанных особенностей при сравнении методов нет. Единственное, что можно сделать в подобной ситуации, — привести данные о результатах вычислений в развернутой форме, позволяющей сравнивать методы по различным критериям. Кроме того, не следует забывать, что численное решение математических задач в значительной степени является искусством, которому можно научиться лишь накоплением опыта, путем проб и ошибок.

Особенности прямого поиска минимума. В методах прямого поиска вычисляются значения лишь самой функции, ее частные производные при этом не используются. Это обстоятельство существенно расширяет сферу практического применения таких методов.

Опишем один из наиболее простых методов прямого поиска. Выбрав начальную точку $x^0 \in \mathbb{R}^n$, строим релаксационную последовательность $\{x^k\}$ следующим образом. На k -й итерации отталкиваемся от найденной на предыдущей итерации точки x^{k-1} , которую в методах прямого поиска часто называют *базовой точкой*. Строим n -мерный куб с центром в базовой точке и с ребрами длиной l , параллельными векторам стандартного базиса в \mathbb{R}^n . Вершинами этого куба являются

точки вида $x^{k-1} + \sum_{i=1}^n \delta_i e_i$, где e_1, \dots, e_n — векторы стандартного базиса, а числовые коэффициенты δ_i , $i = \overline{1, n}$, принимают значение $l/2$ или $-l/2$. Множество вершин куба вместе с его центром составляют так называемый **шаблон** (или образец). Вычислив значения функции в центре и вершинах куба, выберем ту из этих точек, в которой значение функции наименьшее. Если выбранная точка оказалась одной из вершин куба, она становится базовой точкой x^k . Если же функция принимает наименьшее значение в точке x^{k-1} , то уменьшаем параметр l , например вдвое, и повторяем процедуру с кубом меньшего размера. Процесс поиска завершается, когда длина ребра куба l станет меньше заданного числа $\varepsilon > 0$.

Основной недостаток описанного метода прямого поиска (поиска по шаблону) состоит в резком росте количества вычисляемых значений целевой функции с увеличением размерности n (количество вершин n -мерного куба равно 2^n), а главное преимущество — в простоте реализации. Отталкиваясь от этого метода, можно построить более удачные методы прямого поиска, сокращая количество вершин в шаблоне. Совсем не обязательно располагать вершины столь регулярно, как в описанном методе. Главное, чтобы вершины шаблона позволяли учитывать любое направление из базовой точки.

Реализация этой идеи привела к разработке методов **симплексного поиска**, которые к началу 1970-х гг. получили широкое распространение. **Симплекс** (от лат. simplex — простой) — это простейший многогранник в \mathbb{R}^n с $n + 1$ вершинами (в \mathbb{R}^2 — треугольник, в \mathbb{R}^3 — тетраэдр и т.д.). В \mathbb{R}^n симплекс образуют любые точки y^0, y^1, \dots, y^n , для которых векторы $y^1 - y^0, y^2 - y^0, \dots, y^n - y^0$ линейно независимы. При этом точки y^0, y^1, \dots, y^n называют **вершинами симплекса**.

Использование в качестве шаблона не n -мерного куба, а симплекса уже дает заметное преимущество, так как на каждой итерации количество вычисляемых значений функции сокращается с 2^n до $n + 1$. Однако это количество можно сократить еще больше, если принять во внимание следующую идею. Вычислив в вершинах симплекса значения целевой функции, выберем наихудшую из них, т.е. ту, в которой значение функции наибольшее. Заменяв эту вершину новой по определенному правилу, получим новый симплекс, с которым повторяем такую же процедуру. В результате получим последовательность симплексов, стягивающихся к точке минимума целевой функции.

В первом варианте симплексного поиска, предложенном в 1962 г. Спендлеем, Хекстом и Химсвортом, был использован *регулярный симплекс*, у которого все вершины равноудалены друг от друга (в \mathbb{R}^2 — равносторонний треугольник, в \mathbb{R}^3 — правильный тетраэдр). Дж. Нелдер и Р. Мид в 1965 г. впервые рассмотрели произвольный (нерегулярный) симплекс и предложили способ изменения скорости его движения путем расширения или сжатия симплекса в зависимости от того, удачно был выбран шаг поиска в направлении убывания функции или нет.

Позднее в исходные варианты методов симплексного поиска были введены различные способы изменения размеров и формы симплекса. Это привело к созданию методов *управляемого прямого поиска*, в которых по определенному правилу выбирают направление смещения центра используемой конфигурации и величину этого смещения так, чтобы значения функции в вершинах или центре симплекса убывали. Метод, в котором в процессе поиска определенным образом изменяют форму симплекса, обычно называют *методом деформируемых конфигураций*. Изменение формы симплекса позволяет улучшить процесс адаптации используемой конфигурации к рельефу графика минимизируемой функции.

Использование регулярного симплекса. Сначала рассмотрим простейший вариант метода симплексного поиска с использованием регулярного симплекса. На первом шаге поиска строим регулярный симплекс с вершинами $x^{1,i}$, $i = \overline{1, n+1}$, который обозначаем S_1 . Построение симплекса можно провести двумя способами.

1. Если x^0 — заданная базовая точка, выбираем ее в качестве первой вершины $x^{1,1}$ симплекса S_1 , а координаты остальных n вершин $x^{1,i}$, $i = \overline{2, n+1}$, этого симплекса вычисляем по формулам

$$x_j^{1,i} = \begin{cases} x_j^{1,1} + \frac{\sqrt{n+1}-1}{n\sqrt{2}} l, & i = j+1; \\ x_j^{1,1} + \frac{\sqrt{n+1}+n-1}{n\sqrt{2}} l, & i \neq j+1, \end{cases} \quad (3.62)$$

где $x_j^{1,i}$, $j = \overline{1, n}$, — координаты вершины $x^{1,i}$ симплекса; l — длина ребра симплекса. Например, если в \mathbb{R}^2 выбрать базовую точку $x^0 = x^{1,1} = (0, 0)$ и длину ребра $l = 2$, то две другие вершины $x^{1,2}$ и $x^{1,3}$ двумерного симплекса будут иметь координаты $x_1^{1,2} = 0,518$, $x_2^{1,2} = 1,932$ и $x_1^{1,3} = 1,932$, $x_2^{1,3} = 0,518$ (рис. 3.20, а).

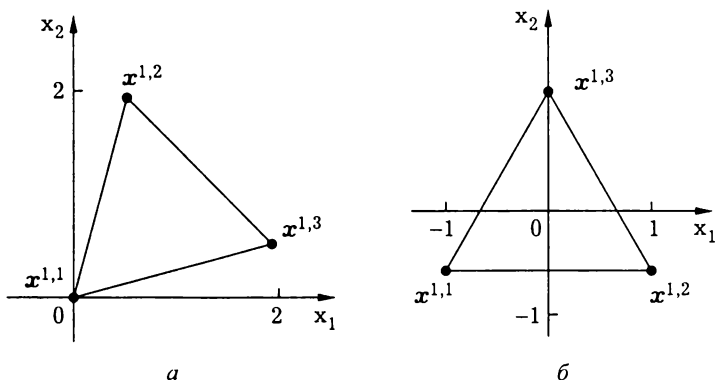


Рис. 3.20

2. Если x^0 — заданная базовая точка, то ее определяем как центр регулярного симплекса S_1 с длиной ребра l , а координаты вершин $x^{1,i}$, $i = \overline{1, n+1}$, этого симплекса находим по формулам

$$x_j^{1,i} = \begin{cases} x_j^0, & j < i-1; \\ x_j^0 + l \sqrt{\frac{j}{2(j+1)}}, & j = i-1; \\ x_j^0 - \frac{l}{\sqrt{2j(j+1)}}, & j > i-1, \end{cases} \quad (3.63)$$

где x_j^0 , $j = \overline{1, n}$, — координаты точки x^0 ; $x_j^{1,i}$, $j = \overline{1, n}$, — координаты вершины $x^{1,i}$. Например, если в \mathbb{R}^2 выбрать базовую точку $x^0 = (0, 0)$ и длину ребра симплекса $l = 2$, то вершинами симплекса будут точки $x^{1,1} = (-1,000, -0,578)$, $x^{1,2} = (1,000, -0,578)$, $x^{1,3} = (0,000, 1,156)$ (рис. 3.20, б).

Определив вершины симплекса S_1 , вычисляем в них значения минимизируемой функции. По результатам вычислений выбираем вершину, в которой значение функции является наибольшим (если таких вершин несколько, можно взять любую из них). Пусть для определенности это будет вершина $x^{1,n+1}$. По формуле

$$x^{2,n+1} = 2x_c^1 - x^{1,n+1} \quad (3.64)$$

находим точку $x^{2,n+1}$, противолежащую вершине $x^{1,n+1}$ относительно противоположной грани симплекса S_1 . Точка $x_c^1 = \frac{1}{n} \sum_{i=1}^n x^{k,i}$

представляет собой центр грани симплекса, противоположной точке $x^{1,n+1}$.

Преобразование симплекса, состоящее в замене одной его вершины (в данном случае $x^{1,n+1}$) симметричной ей относительно противоположной грани (в данном случае вершиной $x^{2,n+1}$), называют **отражением вершины**. В результате получают новый регулярный симплекс S_2 , образованный новой вершиной $x^{2,n+1}$ и n вершинами $x^{1,i}$, $i = \overline{1, n}$, принадлежавшими симплексу S_1 . На рис. 3.21 показано построение нового регулярного симплекса в двумерном случае.

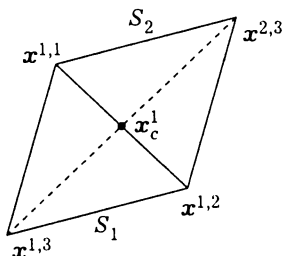


Рис. 3.21

На втором шаге поиска вычисляют значение целевой функции в новой вершине симплекса S_2 . В остальных вершинах значения функции уже известны. Можно повторить описанную процедуру преобразования симплекса, состоящую в отражении худшей (с наибольшим значением функции) вершины.

При записи алгоритма описанного метода удобно, чтобы номера вершин симплекса на k -й итерации поиска были присвоены по возрастанию значений функции, т.е. так, чтобы выполнялись неравенства

$$f(x^{k,1}) \leq \dots \leq f(x^{k,i}) \leq \dots \leq f(x^{k,n+1}).$$

Такую нумерацию вершин симплекса будем называть **правильной**. При правильной нумерации вершин отражению подлежит последняя вершина, т.е. $x^{k,n+1}$. После того как построен новый симплекс, изменяют порядок вершин, устанавливая правильную их нумерацию.

Если на k -м шаге в результате отражения вершины $x^{k,n+1}$ симплекса S_k в точку $x^{k+1,n+1}$ окажется, что $f(x^{k+1,n+1}) \geq f(x^{k,n+1})$, отражение признают неудачным и возвращаются к исходному симплексу S_k . В этом случае пробуют провести отражение вершины $x^{k,n}$, сравнивая значение в новой вершине $x^{k+1,n}$ с наибольшим значением

$f(x^{k,n+1})$ в симплексе S_k . Если и это отражение неудачно, пробуют отражение следующей вершины $x^{k,n-1} \in S_k$ и т.д. Если отражения всех вершин симплекса S_k оказались неудачными, можно прекратить поиск минимума, выбрав в качестве приближения точку $x^{k,1}$ с наименьшим значением функции среди всех вершин последнего симплекса S_k .

Особенность симплексного поиска состоит в том, что его результатом является последовательность симплексов, а не точек, как в методах спуска. Однако и в симплексном поиске можно выделить релаксационную последовательность. Действительно, в силу описанного метода выбора симплексов $\{S_k\}$ последовательность $\{x^{k,n+1}\}$ „наихудших“ вершин является релаксационной. Отметим, что в большинстве случаев последовательность симплексов оказывается конечной, так как в окрестности точки минимума возникает ситуация, когда любое отражение вершины очередного симплекса является неудачным и этот симплекс будет последним.

Точность приближения, полученного с помощью простейшего симплексного поиска, описанного выше, определяется размерами ребер симплекса l . Повышение точности поиска требует уменьшения значения этого параметра, а это приводит к увеличению количества итераций, поскольку шаг поиска напрямую связан со значением l . В этой связи возникает идея изменять размеры симплекса в процессе поиска. Реализация этой идеи наделяет процесс поиска высокими адаптивными свойствами.

Уменьшение размера симплекса при сохранении его формы называют *редукцией симплекса*. Редукция симплекса может использоваться следующим образом. Сначала делают попытку выполнить отражение вершины симплекса S_k , полученного на предыдущей итерации, действуя так, как это описано выше. Затем, если отражение оказалось неудачным, выполняют редукцию симплекса, уменьшая длину его ребер с помощью некоторого фиксированного *коэффициента редукции* $\delta \in (0, 1)$. В этом случае процесс редукции можно описать равенствами

$$x^{k+1,i} = x^{k,1} + \delta(x^{k,i} - x^{k,1}), \quad i = \overline{2, n+1}.$$

Из этих равенств видно, что симплекс сжимается к вершине $x^{k,1}$, в которой значение целевой функции меньше, чем в других вершинах симплекса.

Использование в симплексном поиске редукции симплекса приводит к изменению и условия прекращения поиска. Здесь возможны

различные варианты, выбор определяется конкретной постановкой задачи. Один из вариантов — прекращение поиска в момент, когда в результате очередной редукции длина ребер симплекса окажется меньше заданной точности поиска.

На рис. 3.22 показан процесс симплексного поиска с помощью регулярного симплекса.

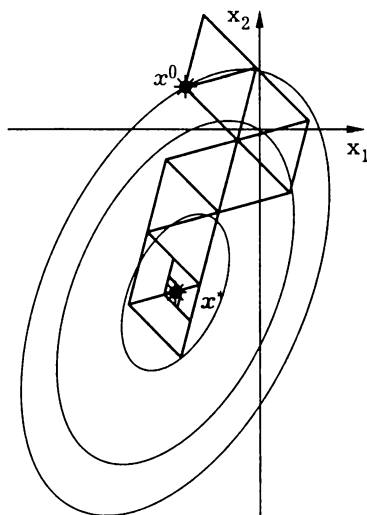


Рис. 3.22

Использование нерегулярного симплекса. Симплексный поиск с использованием регулярного симплекса эффективен лишь в случаях, когда график целевой функции имеет достаточно простую топографию. Если график целевой функции имеет овражную структуру, использование регулярного симплекса менее эффективно: регулярный симплекс, скатываясь по поверхности уровня, плохо в нее вписывается.

Примером функции, график которой обладает овражной структурой, является функция

$$f(x_1, x_2) = 10(x_1^2 - x_2)^2 + (x_1 - 1)^2,$$

имеющая минимум в точке $x^* = (1, 1)$. Топография линий уровня $f(x_1, x_2) = C$ этой функции приведена на рис. 3.23. Препятствием

для успешного поиска минимума $f(x)$ является в этом случае форма симплекса: регулярный симплекс нельзя изменить так, чтобы он „вытянулся“ вдоль „оврага“, а это не позволяет успешно продолжить поиск минимума. Естественной в этих условиях выглядит идея деформирования симплекса в процессе поиска, т.е. изменения его формы и размера.

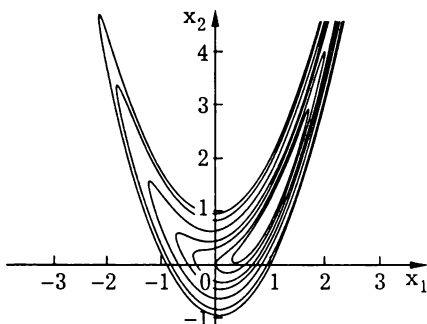


Рис. 3.23

Существует много вычислительных конструкций прямого поиска с помощью нерегулярного симплекса, различающихся способами изменения размеров и формы симплекса. Рассмотрим одну из простейших конструкций подобного рода — *метод Нелдера — Мида*.

Обозначим через S_k симплекс, полученный на предыдущей, $(k-1)$ -й итерации. Этот симплекс может иметь произвольную форму, однако будем считать, что он имеет правильную нумерацию вершин. Построение нового симплекса S_{k+1} включает несколько процедур: отражение „худшей“ вершины $x^{k,n+1}$; изменение размера и формы симплекса, полученного в результате отражения этой вершины, и редукция исходного симплекса S_k , выполняемая при неудачном отражении вершины. Рассмотрим эти процедуры подробнее.

1. Отражение вершины $x^{k,n+1} \in S_k$ осуществляют в соответствии с формулой, аналогичной формуле (3.64):

$$x^{k+1,n+1} = x_c^k + \alpha(x_c^k - x^{k,n+1}),$$

где $\alpha > 0$ — заданный *коэффициент отражения*; $x_c^k = \frac{1}{n} \sum_{i=1}^n x^{k,i}$ — центр грани симплекса, противоположной „худшей“ вершине $x^{k,n+1}$. В результате получают новый симплекс \tilde{S}_k , образованный новой вер-

шиной $x^{k+1,n+1}$ и вершинами $x^{k,i}$, $i = \overline{1, n}$, исходного симплекса. На рис. 3.24 показана процедура построения симплекса \tilde{S}_k в двумерном случае при $\alpha = 1$.

В новой вершине вычисляют значение целевой функции $f(x)$. Если $f(x^{k+1,n+1}) < f(x^{k,1})$, то шаг поиска считают удачным: по-

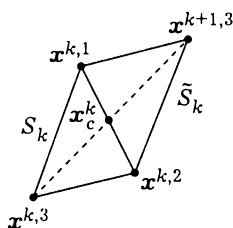


Рис. 3.24

лучен новый симплекс \tilde{S}_k , у которого значение целевой функции в новой вершине меньше, чем наименьшее значение функции в вершинах исходного симплекса. При удачном отражении функция убывает в направлении новой вершины, поэтому целесообразно эту вершину отодвинуть еще дальше, увеличивая тем самым шаг поиска. Это

реализуется процедурой растяжения симплекса. Если выполнены неравенства $f(x^{k,n}) \geq f(x^{k+1,n+1}) \geq f(x^{k,1})$, то замена „худшей“ вершины новой уменьшает среднее по вершинам значение функции и отражение следует признать приемлемым. Однако целесообразно выполнить процедуру сжатия симплекса \tilde{S}_k , чтобы получить новую вершину с меньшим значением функции. Если же $f(x^{k+1,n+1}) > f(x^{k,n})$, то отражение неудачно и можно попробовать процедуру сжатия исходного симплекса S_k : может оказаться, что сжатие приведет к более удачному варианту симплекса.

2. Растяжение симплекса \tilde{S}_k осуществляют в соответствии с формулой

$$x_*^{k+1,n+1} = x_c^k + \beta(x^{k+1,n+1} - x_c^k),$$

где $\beta > 1$ — заданный коэффициент растяжения; x_c^k — центр грани, противоположной вершине $x^{k+1,n+1}$ в симплексе \tilde{S}_k . Геометрически растяжение можно интерпретировать как увеличение длины вектора $x^{k+1,n+1} - x_c^k$ в β раз.

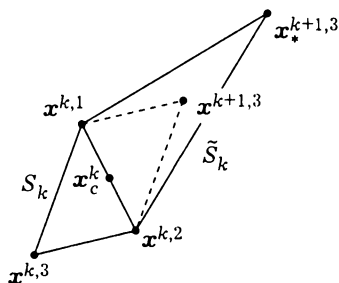


Рис. 3.25

После растяжения сравнивают значение $f(x_*^{k+1,n+1})$ со значениями целевой функции в остальных вершинах симплекса. Если $f(x_*^{k+1,n+1}) < f(x^{k,1})$, выбирают симплекс S_{k+1} с вершинами $x_*^{k+1,n+1}$ и $x^{k,i}$, $i = \overline{1, n}$ (на рис. 3.25 построен симплекс S_{k+1} в двумерном случае с применением операции растяжения при $\beta = 2$). При

$f(x_*^{k+1,n+1}) \geq f(x^{k,1})$ операцию растяжения признают неудачной и выбирают симплекс $S_{k+1} = \tilde{S}_k$.

3. Сжатие симплекса \tilde{S}_k проводят в соответствии с формулой

$$x_{**}^{k+1,n+1} = x_c^k + \gamma(x^{k+1,n+1} - x_c^k),$$

где $\gamma \in (0, 1)$ — заданный *коэффициент сжатия*. Сжатие симплекса \tilde{S}_k показано на рис. 3.26, а в двумерном случае при $\gamma = 1/2$.

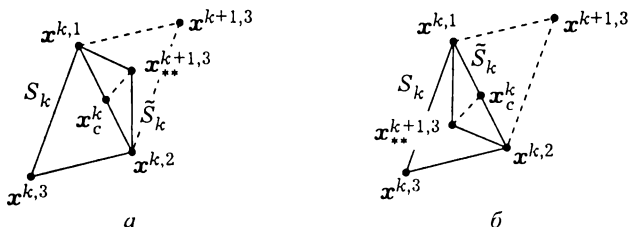


Рис. 3.26

Сжатие симплекса S_k выполняют по аналогичной формуле:

$$x_{**}^{k+1,n+1} = x_c^k + \gamma(x^{k,n+1} - x_c^k).$$

На рис. 3.26, б показано такое сжатие при $\gamma = 1/2$.

После сжатия симплекса сравнивают значение $f(x_{**}^{k+1,n+1})$ целевой функции в найденной вершине со значением $f(x^{k+1,n+1})$ или $f(x^{k,n+1})$ в отброшенной вершине. Если значение в новой вершине меньше, то в качестве S_{k+1} выбирают результат сжатия симплекса. Если сжатие симплекса не уменьшило значение функции, то оставляют симплекс, подвергшийся сжатию, т.е. либо $S_{k+1} = \tilde{S}_k$, либо $S_{k+1} = S_k$. В последнем случае все попытки не привели к изменению симплекса. Поэтому выполняют операцию редукции симплекса, которая осуществляется по тем же правилам, что и для регулярного симплекса. Симплекс S_{k+1} , полученный редукцией симплекса S_k , изображен на рис. 3.27.

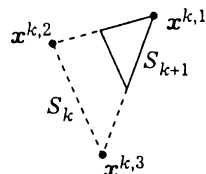


Рис. 3.27

Одна из проблем, связанных с применением нерегулярного симплекса, состоит в том, что через несколько итераций симплекс может

оказаться сильно вытянутым и потеряет способность адекватно отражать изменение целевой функции в разных направлениях. Учитывая это, в методах симплексного поиска, базирующихся на использовании нерегулярного симплекса, периодически (через фиксированное число итераций) проводят так называемую *операцию восстановления симплекса*, в которой, выбрав две „лучшие“ точки текущего симплекса S_k , строят новый симплекс, используя расстояние между выбранными точками как длину ребер нового симплекса, а одну из этих точек — как базовую.

Симплексный поиск в методе Нелдера — Мида прекращается, когда размеры текущего симплекса оказываются меньше некоторой критической величины. В отличие от методов с регулярным симплексом, в которых все ребра одинаковы и по их длине можно контролировать размеры симплекса, в методах с нерегулярным симплексом необходима некоторая общая характеристика размеров симплекса. Например, в качестве условия останова можно использовать неравенство

$$\frac{1}{n+1} \sum_{i=1}^{n+1} (f(x^{k,i}) - f(x_c^k))^2 < \varepsilon^2,$$

где $x_c^k = \frac{1}{n} \sum_{i=1}^n x^{k,i}$ — центр грани симплекса, противоположной его „худшей“ вершине; $\varepsilon > 0$ — параметр точности поиска.

Построение начального симплекса в методах поиска с нерегулярным симплексом проводят различными способами. Можно в качестве начального выбрать регулярный симплекс, вершины которого находят по формуле (3.62) или (3.63). Другой способ состоит в выборе базовой точки x^0 в качестве первой вершины $x^{1,1}$ симплекса и вычислении остальных n вершин симплекса по формуле

$$x^{1,i+1} = x^0 + l e_i, \quad i = \overline{1, n}, \quad (3.65)$$

где e_1, \dots, e_n — стандартный базис в \mathbb{R}^n ; l — заданное число, регулирующее размеры симплекса. При этом способе построения начальный симплекс будет нерегулярным (например, в двумерном случае получим равнобедренный треугольник).

С теоретической точки зрения методы поиска, использующие нерегулярные симплексы, изучены недостаточно полно. Однако практика свидетельствует, что эти методы вполне работоспособны. В задачах

безусловной минимизации рекомендуют выбирать $\alpha = 1$, $\beta = 2$, $\gamma = 1/2$ или $\alpha = 2$, $\beta = 5/2$, $\gamma = 1/4$. Эта общая рекомендация может не учитывать особенностей конкретной целевой функции. Уточнить значения параметров симплексного поиска можно, проведя дополнительный анализ целевой функции или вычислительный эксперимент.

Отметим, что симплексный поиск на основе нерегулярных симплексов чувствителен к выбору значений параметров. При правильном выборе параметров такой поиск может оказаться более эффективным, чем поиск с помощью регулярного симплекса. В то же время неудачный выбор параметров может привести не только к чрезмерному увеличению количества итераций, но и к неверным результатам вследствие того, что процесс поиска прекратится до достижения точки минимума.

На рис. 3.28 показан симплексный поиск с использованием нерегулярного симплекса.

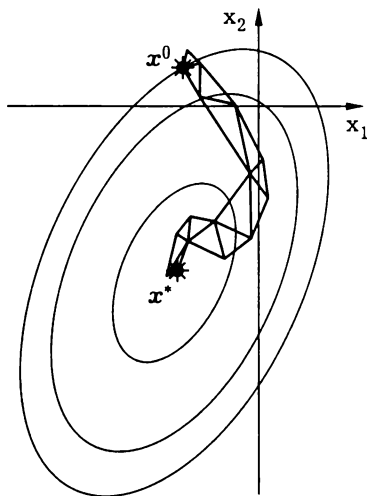


Рис. 3.28

Циклический по координатам спуск. В этом методе направление спуска последовательно выбирают вдоль одной из координатных осей. Вдоль выбранного направления проводят исчерпывающий спуск (или наискорейший спуск), определяя тем самым шаг спуска.

Метод циклического покоординатного спуска — один из самых простых в ряду методов оптимизации. Алгоритм вычислений можно описать следующим образом. Обозначим через e_i , $i = \overline{1, n}$, векторы стандартного базиса в \mathbb{R}^n . Выберем начальную точку x^0 , параметр точности поиска $\varepsilon_1 > 0$ и положим $k = j = 1$, $\tilde{x}_1^k = x^0$.

1. Минимизируя функцию $\varphi_j^{(k)}(\beta) = f(\tilde{x}_j^k + \beta e_j)$, находим значение $\beta_j^{(k)}$, вычисляем $\tilde{x}_{j+1}^k = \tilde{x}_j^k + \beta_j^{(k)} e_j$ и переходим к п. 2.

2. Если $j < n$, то увеличиваем номер j на единицу и возвращаемся к п. 1. В противном случае полагаем $x^k = \tilde{x}_{n+1}^k$ и переходим к п. 3.

3. Если $|x^k - x^{k-1}| < \varepsilon_1$, то поиск прекращаем, выбирая в качестве приближения к точке минимума x^* точку x^k , т.е. полагая $x^* \approx x^k$, $f(x^*) \approx f(x^k)$. В противном случае полагаем $j = 1$, $\tilde{x}_1^{k+1} = x^k$, увеличиваем номер k на единицу и переходим к п. 1.

В описанном алгоритме одной итерацией следует считать последовательное использование всех координатных направлений: номером итерации является параметр k , а параметр j определяет порядковый номер координатного направления и на каждой итерации пробегает все значения от 1 до n . Обратим также внимание на то, что значение $\beta_j^{(k)}$ может быть как положительным, так и отрицательным, а шагом спуска следует считать величину $|x^k - x^{k-1}|$. Но эти положения в достаточной мере условны. С неменьшим основанием можно считать, что одной итерацией прямого поиска является спуск по одному из координатных направлений, а шаг спуска определяется абсолютной величиной параметра $\beta_j^{(k)}$. Правда, в этом случае условие останова следует проверять перед каждой сменой координатного направления.

Условие останова, указанное в алгоритме, не является единственно возможным. Часто используют два условия останова:

$$|x^k - x^{k-1}| < \varepsilon_1, \quad |f(x^k) - f(x^{k-1})| < \varepsilon_2.$$

На рис. 3.29 показаны первые два шага процесса покоординатного циклического спуска в случае функции двух переменных.

Хотя метод циклического покоординатного спуска прост в реализации, он не всегда эффективен. Метод дает хорошие результаты, если минимизируемая функция является сепарабельной, т.е. представляет

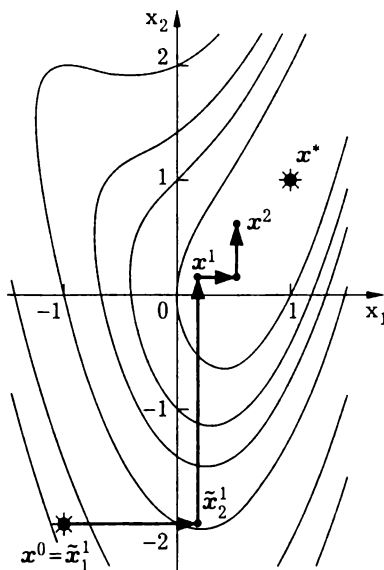


Рис. 3.29

собой сумму функций одного переменного:

$$f(x) = \sum_{j=1}^n h_j(x_j), \quad x \in \mathbb{R}^n.$$

В этом случае решение задачи минимизации можно получить за один шаг поиска (т.е. за один полный цикл перебора всех координатных направлений).

В самом деле, поскольку

$$f(x^*) = \min_{x \in \mathbb{R}^n} f(x) = \sum_{j=1}^n \min_{x_j \in \mathbb{R}} h_j(x_j) = \sum_{j=1}^n h_j(x_j^*),$$

достаточно на первом шаге поиска последовательно решить n задач одномерной минимизации функций $h_j(x_j)$, $j = \overline{1, n}$, что позволит найти все n координат x_j^* искомой точки x^* .

Метод Хука — Дживса. При прямом поиске точки минимума целевой функции важную роль играет выбор направления спуска. Для этой цели на каждой итерации можно организовать предварительный

этап *исследующего поиска*. На этом этапе, анализируя значения целевой функции $f(x)$ в точках, близких к текущей точке x^{k-1} , находят точку \tilde{x}^k , удовлетворяющую условию $f(\tilde{x}^k) < f(x^{k-1})$. В качестве направления спуска выбирают то, которое определяется вектором $\tilde{x}^k - x^{k-1}$. Такая стратегия прямого поиска, предложенная в 1961 г., получила название метода Хука — Дживса (эту стратегию иногда называют методом конфигураций).

Опишем один из алгоритмов исследующего поиска, в котором используется фиксированный вектор $b = (b_1 \dots b_n)^T$, удовлетворяющий условию $\max_{j=1, n} |b_j| \geq \varepsilon$, где $\varepsilon > 0$ — параметр точности исследующего поиска. Координаты этого вектора, называемого *вектором перемещений*, определяют приращения координат точки на этапе исследующего поиска. На k -й итерации полагаем $j = 1$, $\tilde{x}_j^k = x^{k-1}$, $f_j^k = f(x^{k-1})$ и переходим к основной части алгоритма исследующего поиска.

1. Вычисляем $f_{+j}^k = f(\tilde{x}_j^k + b_j e_j)$ и $f_{-j}^k = f(\tilde{x}_j^k - b_j e_j)$, где e_1, \dots, e_n — стандартный базис в \mathbb{R}^n , находим точку

$$\tilde{x}_{j+1}^k = \begin{cases} \tilde{x}_j^k + b_j e_j, & f_{+j}^k < f_j^k \text{ и } f_{+j}^k \leq f_{-j}^k; \\ \tilde{x}_j^k - b_j e_j, & f_{-j}^k < f_j^k \text{ и } f_{-j}^k < f_{+j}^k; \\ \tilde{x}_j^k, & \text{во всех остальных случаях,} \end{cases}$$

и переходим к п. 2.

2. Если $j < n$, то номер j увеличиваем на единицу, полагаем $f_j^k = f(\tilde{x}_j^k)$ и переходим к п. 1. В противном случае переходим к п. 3.

3. Если $\tilde{x}_{n+1}^k \neq \tilde{x}^k$, переходим к п. 4. В противном случае уменьшаем длину вектора b , умножая его на число $1/\gamma$, где $\gamma > 1$ — коэффициент дробления шага исследующего поиска. Затем, полагая $j = 1$, $\tilde{x}_j^k = x^{k-1}$, $f_j^k = f(x^{k-1})$, возвращаемся к п. 1.

4. Полагаем $\tilde{x}^k = \tilde{x}_{n+1}^k$ и переходим к этапу спуска, в котором направление спуска определяется вектором $\tilde{x}^k - x^{k-1}$.

На этапе спуска, подбирая так называемый *ускоряющий множитель* $a_k > 0$, по формуле

$$x^k = x^{k-1} + a_k(\tilde{x}^k - x^{k-1}) \quad (3.66)$$

находим такую точку x^k , чтобы $f(x^k) \leq f(\tilde{x}^k)$. Условием останова в методе Хука — Дживса может служить неравенство $|\tilde{x}^k - x^{k-1}| < \varepsilon$ (ε — параметр точности исследующего поиска), которое можно проверять сразу после этапа исследующего поиска. При выполнении этого условия полагают $x^* \approx x^{k-1}$ и $f(x^*) \approx f(x^{k-1})$.

Ускоряющий множитель a_k позволяет регулировать величину шага спуска $a_k |\tilde{x}^k - x^{k-1}|$, обеспечивая тем самым более высокую эффективность метода. Его можно подбирать из разных соображений, например, выполняя исчерпывающий спуск в направлении вектора $\tilde{x}^k - x^{k-1}$. В простейшем варианте метода Хука — Дживса значение a_k задают постоянным, обычно равным 2. На рис. 3.30 показаны этапы исследующего поиска и спуска для первых трех итераций метода Хука — Дживса. Рассмотрены два варианта метода: с постоянным значением ускоряющего множителя a_k (рис. 3.30, а) и с ускоряющим множителем, выбираемым в соответствии с исчерпывающим спуском в выбранном направлении (рис. 3.30, б).

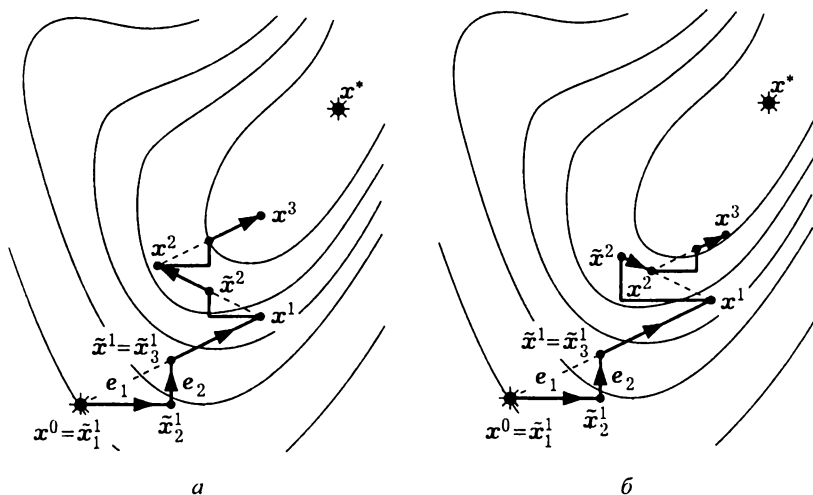


Рис. 3.30

Известно много модификаций метода Хука — Дживса, отличающихся как способом исследующего поиска, так и выбором ускоряющего множителя. Отметим вариант метода, при котором исследующий поиск проводится с помощью метода циклического покоординатного спуска (рис. 3.31).

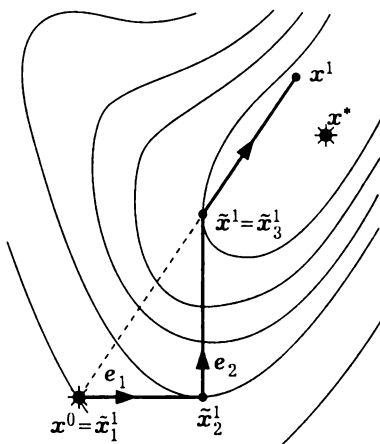


Рис. 3.31

В некоторых модификациях метода Хука — Дживса вводят повторный этап исследующего поиска. В этом случае на каждом k -м шаге поиска после того, как найдена точка \tilde{x}^k , выполняют так называемый **поиск по образцу**, который включает две процедуры. Сначала выполняют спуск в направлении вектора $\tilde{x}^k - x^{k-1}$, выбирая тем или иным способом ускоряющий множитель. Затем, получив в результате спуска некоторую точку $\tilde{x}_{\text{сп}}^k$, в окрестности этой точки организуют повторный исследующий поиск, который дает еще одну точку $\tilde{x}_{\text{пв}}^k$. Поиск по образцу считают успешным, если точки $\tilde{x}_{\text{сп}}^k$ и $\tilde{x}_{\text{пв}}^k$ различны, или, другими словами, $f(\tilde{x}_{\text{пв}}^k) < f(\tilde{x}_{\text{сп}}^k)$. В этом случае направление спуска задают вектором $\tilde{x}_{\text{пв}}^k - \tilde{x}^k$, регулируя шаг спуска ускоряющим множителем. Если две дополнительные точки совпали, поиск считают неуспешным и в качестве результата выполнения итерации выбирают точку \tilde{x}^k , т.е. полагают $x^k = \tilde{x}^k$.

Поиск по образцу может заметно ускорить сходимость итерационной последовательности. Например, если при минимизации квадратичной функции двух переменных на этапах исследующего поиска используется циклический покоординатный спуск, а на этапах спуска — исчерпывающий спуск, то точка минимума достигается за одну итерацию (рис. 3.32, а). Сравним эту стратегию с модификацией метода Хука — Дживса, в которой исследующий поиск основан на методе циклического покоординатного спуска, а ускоряющий множи-

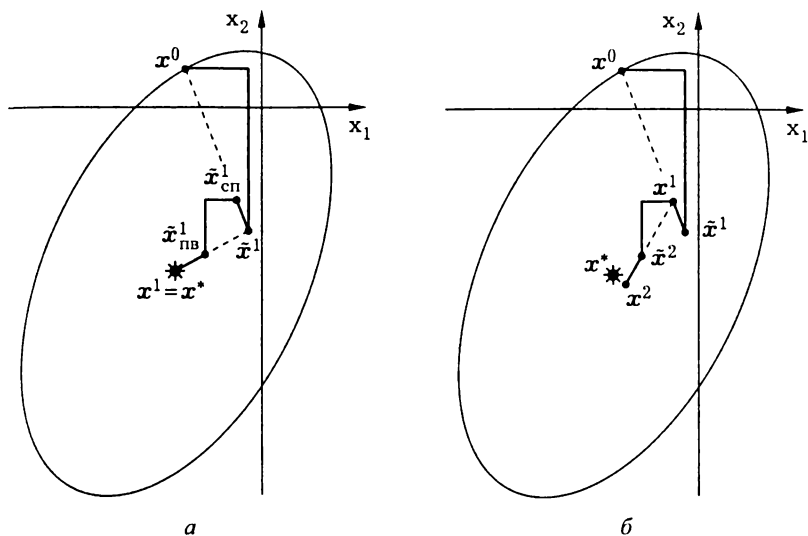


Рис. 3.32

тель на этапе спуска выбирается исходя из исчерпывающего спуска (рис. 3.32, б). Оба варианта метода на первом шаге дают одну и ту же точку x^1 , но различаются правилами выбора текущего направления спуска.

Метод Розенброка. Этот метод, как и метод Хука — Дживса, основан на проведении исследующего поиска на каждой итерации, но отличается от последнего способом выбора направлений исследующего поиска. Если в методе Хука — Дживса эти направления фиксированы и определяются векторами стандартного базиса в \mathbb{R}^n , то в методе Розенброка направления исследующего поиска задаются ортонормированным базисом, который на каждой итерации строят заново в процессе минимизации целевой функции.

Такая стратегия поиска впервые была реализована в 1960 г. и получила название метода Розенброка. В первоначальном варианте метода при исследующем поиске использовались дискретные шаги по направлениям, заданным векторами текущего ортонормированного базиса. Дискретные шаги поиска определяются, как и в методе Хука — Дживса, некоторым вектором b . Изменяя длину этого вектора, уменьшают или увеличивают шаги исследующего поиска.

Опишем одну из модификаций метода Розенброка, в которой исследующий поиск на каждом шаге проводится циклическим исчерпывающим спуском по направлениям текущего ортонормированного базиса. Пусть выбраны начальная точка $x^0 \in \mathbb{R}^n$ и параметр $\varepsilon_1 > 0$ точности поиска. В качестве начального ортонормированного базиса p_j^1 , $j = \overline{1, n}$, задающего направления исследующего поиска на первом шаге, выберем стандартный базис в \mathbb{R}^n . Полагаем $k = j = 1$, $\tilde{x}_1^k = x^0$ и переходим к основной части алгоритма.

1. Минимизируя функцию $\varphi_j^{(k)}(\beta) = f(\tilde{x}_j^k + \beta p_j^k)$ на всей числовой оси, находим значение $\beta_j^{(k)} \in \mathbb{R}$, вычисляем $\tilde{x}_{j+1}^k = \tilde{x}_j^k + \beta_j^{(k)} p_j^k$ и переходим к п. 2.

2. Если $j < n$, то номер j увеличиваем на единицу и возвращаемся к п. 1. В противном случае полагаем $x^k = \tilde{x}_{n+1}^k$ и переходим к п. 3.

3. Если $|x^k - x^{k-1}| < \varepsilon_1$, то дальнейший поиск точки минимума прекращаем, принимая $x^* \approx x^k$ и $f(x^*) \approx f(x^k)$. В противном случае переходим к п. 4.

4. На k -м шаге поиска строим новый ортонормированный базис p_j^{k+1} , $j = \overline{1, n}$, векторы которого задают направления исследующего поиска на $(k+1)$ -м шаге. Для этого строим систему векторов a_j^{k+1} , $j = \overline{1, n}$, в соответствии с формулами

$$a_j^{k+1} = \begin{cases} p_j^k, & \beta_j^{(k)} = 0; \\ \sum_{i=j}^n \beta_i^{(k)} p_i^k, & \beta_j^{(k)} \neq 0. \end{cases} \quad j = \overline{1, n},$$

и проводим ортогонализацию этой системы:

$$b_1^{k+1} = a_1^{k+1},$$

$$b_j^{k+1} = a_j^{k+1} - \sum_{i=1}^{j-1} \frac{(a_j^{k+1}, b_i^{k+1})}{|b_i^{k+1}|^2} b_i^{k+1}, \quad j = \overline{2, n}.$$

Затем выполняем нормировку полученной системы векторов:

$$p_j^{k+1} = \frac{b_j^{k+1}}{|b_j^{k+1}|}, \quad j = \overline{1, n}.$$

Построив систему векторов p_j^{k+1} , $j = \overline{1, n}$, переходим к п. 1, полагая $j = 1$, $\tilde{x}_j^{k+1} = x^k$ и затем увеличивая номер k на единицу.

На рис. 3.33 показана первая итерация поиска методом Розенброка в двумерном случае.

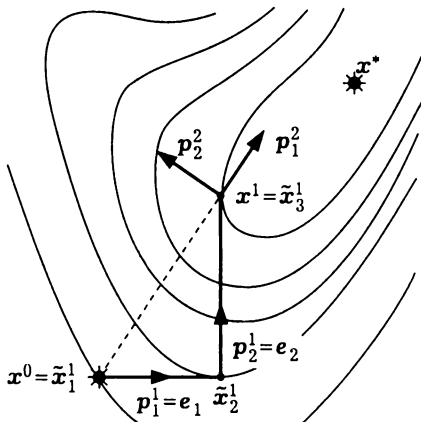


Рис. 3.33

Метод Пауэлла. К методу Розенброка близок метод прямого поиска, предложенный в 1964 г. Пауэллом. Различие двух методов состоит в выборе системы векторов, определяющих направления спуска на каждом последующем шаге поиска. В методе Пауэлла построение системы векторов p_j^k , $j = \overline{1, n}$, на одной итерации выполняется n раз. Опишем алгоритм метода. Полагаем $k = j = i = 1$, $\tilde{x}_1^k = x^0$, в качестве системы векторов p_j^1 , $j = \overline{1, n}$, выбираем стандартный базис в \mathbb{R}^n и переходим к основной части алгоритма.

1. Минимизируя функцию $\varphi_j^{(k)}(\beta) = f(\tilde{x}_j^k + \beta p_j^k)$, находим значение $\beta_j^{(k)} \in \mathbb{R}$, вычисляем $\tilde{x}_{j+1}^k = \tilde{x}_j^k + \beta_j^{(k)} p_j^k$ и переходим к п. 2.
2. Если $j < n$, то увеличиваем номер j на единицу и возвращаемся к п. 1. В противном случае переходим к п. 3.
3. Полагаем $p = \tilde{x}_{n+1}^k - \tilde{x}_1^k$ и находим значение $\hat{\beta}$, минимизируя функцию $\varphi(\beta) = f(\tilde{x}_{n+1}^k + \beta p)$, а затем вычисляем $z_i = \tilde{x}_{n+1}^k + \hat{\beta} p$. Если $i < n$, то для всех $j = \overline{1, n-1}$ заменяем p_j^k на p_{j+1}^k , полагаем

$p_n^k = p$, $\tilde{x}_1^k = z_i$, $j = 1$, номер i увеличиваем на единицу и переходим к п. 1 рассматриваемого алгоритма. В противном случае, т.е. при $i = n$, переходим к п. 4.

4. Принимаем $x^k = z_n$. Если $|x^k - x^{k-1}| < \varepsilon_1$, вычисления прекращаем и полагаем $x^* \approx x^k$ и $f(x^*) \approx f(x^k)$. В противном случае полагаем $\tilde{x}_1^{k+1} = x^k$, $i = j = 1$, увеличиваем номер k на единицу и возвращаемся к п. 1 алгоритма.

На рис. 3.34 показан первый шаг метода Пауэлла в случае функции двух переменных.

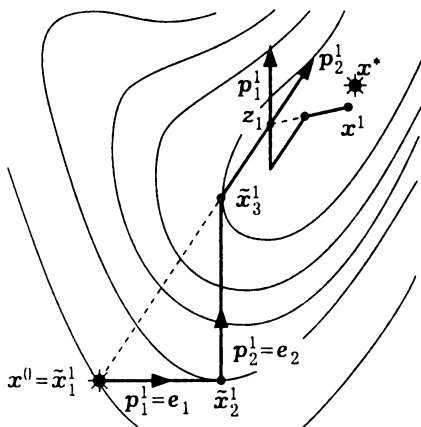


Рис. 3.34

Можно доказать, что при минимизации квадратичной функции с положительно определенной матрицей Q за один шаг поиска методом Пауэлла строится система векторов, сопряженных относительно Q , а точка минимума достигается за одну итерацию. На рис. 3.35 это показано в двумерном случае. Здесь сопряженными являются векторы $p^1 = z_1 - x^0$ и $p^2 = x^1 - z_1$.

Известно много модификаций метода Пауэлла. Например, на каждом шаге поиска можно включить дополнительный спуск в направлении вектора p^n (рис. 3.36). Такая итерация для неквадратичной функции обеспечивает линейную независимость векторов, определяющих очередное направление спуска.

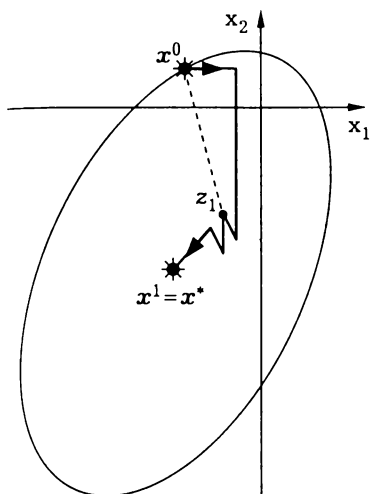


Рис. 3.35

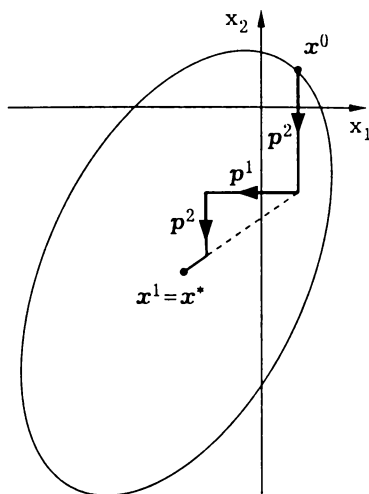


Рис. 3.36

3.8. Методы случайного поиска

Под *случайным поиском* понимают группу методов спуска, в которых направление спуска выбирается случайным образом. Построение релаксационной последовательности осуществляется по формуле (3.4), общей для всех методов спуска:

$$x^k = x^{k-1} + \beta_k \xi^k, \quad k \in \mathbb{N}, \quad (3.67)$$

но в этом случае единичный вектор ξ , определяющий направление спуска, есть случайный вектор, имеющий заданное распределение на единичной сфере $\|\xi\| = 1$.

Поскольку на каждом шаге итерационного процесса направление спуска есть случайная величина, то и очередная точка x^k релаксационной последовательности есть функция случайных векторов $\xi^1, \xi^2, \dots, \xi^k$, т.е. сама является многомерной случайной величиной. В этой ситуации говорить о сходимости метода можно только в вероятностном смысле.

Методы случайного поиска не имеют очевидных преимуществ перед другими методами спуска, да и сравнивать их в теоретическом

плане сложно, поскольку утверждения о сходимости в детерминированных и стохастических методах качественно разные. Однако методы случайного поиска просты в реализации и в силу этого находят применение на практике.

Напомним два важных понятия из теории вероятностей, связанных со сходимостью.

Последовательность $\{\xi_k\}$ случайных величин, заданных на одном вероятностном пространстве, сходится по вероятности к случайной величине ξ , если $P\{|\xi_n - \xi| \leq \varepsilon\} \rightarrow 1$ при $n \rightarrow \infty$ для любого числа $\varepsilon > 0$. *Сходимость по вероятности* записывают в виде $\xi_n \xrightarrow[n \rightarrow \infty]{P} \xi$. Случайные величины могут быть n -мерными векторами. В этом случае знак модуля означает какую-либо норму в \mathbb{R}^n .

Последовательность $\{\xi_k\}$ случайных величин, заданных на одном вероятностном пространстве, сходится к случайной величине ξ почти наверное, если событие $\{\omega: \xi_n(\omega) \rightarrow \xi(\omega)\}$ имеет вероятность 1. *Сходимость почти наверное* записывают в виде $\xi_n \xrightarrow[n \rightarrow \infty]{\text{п.н.}} \xi$.

Лемма 3.1. Пусть функция $f(x)$ удовлетворяет условиям теоремы 3.1, а релаксационная последовательность $\{x^k\}$, построенная по рекуррентному соотношению (3.4), подчиняется условию (3.6), в котором $0 < \lambda \leq \lambda_k \leq 1$. Тогда для некоторого числа $C > 0$ выполняются неравенства

$$f(x^k) - f(x^{k-1}) \geq C\alpha_k^2 |w^k|^2,$$

где $\alpha_k = \frac{(u^k, w^k)}{|w^k|}$ — косинус угла между антиградиентом w^k и направлением спуска u^k .

◀ Из неравенства (3.6) заключаем, что

$$f(x^k) - f(x^{k-1}) \geq \lambda(f(x^{k-1}) - f(\tilde{x}^k)),$$

где \tilde{x}^k — точка, в которой достигается минимум $f(x^{k-1} + \beta u^k)$, $\beta \in \mathbb{R}_*$. Следовательно, достаточно разность $f(x^{k-1}) - f(\tilde{x}^k)$ оценить снизу.

Пусть точке \tilde{x}^k соответствует значение $\tilde{\beta}$. Отметим, что

$$(\text{grad } f(\tilde{x}^k), u^k) = 0,$$

поскольку функция $f(x)$ в точке \tilde{x}^k имеет минимум по направлению u^k . Поэтому

$$\begin{aligned}\alpha_k |w^k| &= (w^k, u^k) = -(\operatorname{grad} f(x^{k-1}), u^k) = \\ &= (\operatorname{grad} f(\tilde{x}^k) - \operatorname{grad} f(x^{k-1}), u^k).\end{aligned}$$

Но в соответствии с неравенством Коши — Буняковского

$$\begin{aligned}|(\operatorname{grad} f(\tilde{x}^k) - \operatorname{grad} f(x^{k-1}), u^k)| &\leq \\ &\leq |\operatorname{grad} f(\tilde{x}^k) - \operatorname{grad} f(x^{k-1})| |u^k| \leq L |\tilde{x}^k - x^{k-1}| = L\tilde{\beta}.\end{aligned}$$

Следовательно,

$$\alpha_k |w^k| \leq L\tilde{\beta}.$$

Выберем значение $\beta = \frac{1}{L} \alpha_k |w^k|$ и используем неравенство*

$$f(x^{k-1}) - f(x^{k-1} + \beta u^k) \geq \beta (w^k, u^k) - \frac{L}{2} \beta^2 = \beta \alpha_k |w^k| - \frac{L}{2} \beta^2.$$

Для выбранного значения β получаем

$$f(x^{k-1}) - f(x^{k-1} + \beta u^k) \geq \frac{1}{L} \alpha_k |w^k|^2 - \frac{1}{2L} \alpha_k |w^k|^2 = \frac{1}{2L} \alpha_k |w^k|^2.$$

Значит,

$$f(x^{k-1}) - f(\tilde{x}^k) \geq f(x^{k-1}) - f(\tilde{x}^k) \geq \frac{1}{2L} \alpha_k |w^k|^2$$

и

$$f(x^{k-1}) - f(x^k) \geq \lambda (f(x^{k-1}) - f(\tilde{x}^k)) \geq C \alpha_k |w^k|^2,$$

где $C = \frac{\lambda}{2L}$. ►

Далее мы рассматриваем направления спуска u^k как случайные векторы, распределенные на единичной сфере в \mathbb{R}^n . В этом случае элементы x^k также будут случайными векторами, а точнее, функциями случайных векторов u^k , поскольку по направлениям спуска в

*Это уточнение формулы Тейлора для функции $f(x^{k-1} + \beta u^k)$ (см.: Аттетков А.В., Галкин С.В., Зарубин В.С., лемма 4.4).

рамках конкретного алгоритма спуска релаксационная последовательность однозначно восстанавливается. Также случайными величинами являются параметры α_k .

Будем использовать следующую стратегию спуска с помощью векторов u^k . Если на k -й итерации $\alpha_k > 0$, т.е. вектор u^k является допустимым направлением спуска, то выбираем точку $x^k = x^{k-1} + \beta_k u^k$, выбирая шаг спуска β_k тем или иным образом, но так, чтобы выполнялось условие (3.6), в котором $0 < \lambda \leq \lambda_k \leq 1$. Если $\alpha_k < 0$, то допустимым направлением спуска является $-u^k$. Следующую точку релаксационной последовательности вычисляем по формуле $x^k = x^{k-1} - \beta_k u^k$. Наконец, при $\alpha_k = 0$ итерацию пропускаем, т.е. полагаем $x^k = x^{k-1}$ и переходим к следующей итерации.

Очевидно, что при такой стратегии распределение случайного вектора u^k должно допускать значения, близкие с направлением антиградиента или градиента. Будем предполагать, что выполняется условие

$$\mathbf{P}\{\alpha_k^2 \geq \alpha^2 | u^1, \dots, u^{k-1}\} \geq p > 0, \quad k \in \mathbb{N}, \quad (3.68)$$

где $\alpha \in (0, 1)$ — некоторое число.

Теорема 3.3. Пусть функция $f(x)$ удовлетворяет условиям теоремы 3.1, а распределения случайных векторов u^k таковы, что выполняются условия (3.68). Тогда для релаксационной последовательности $\{x^k\}$ имеем

$$|\text{grad } f(x^k)| \xrightarrow{\mathbf{P}} 0 \quad \text{при } k \rightarrow \infty.$$

◀ Согласно лемме 3.1 имеем неравенство

$$f(x^{k-1}) - f(x^k) \geq C \alpha_k^2 |w^k|^2, \quad k = 1, 2, \dots$$

Это неравенство следует рассматривать как неравенство случайных величин. Полагая уже заданными векторы u^1, \dots, u^{k-1} , рассмотрим условные распределения обеих частей неравенства. Из неравенства вытекает неравенство условных математических ожиданий случайных величин:

$$\mathbf{M}(f(x^{k-1}) - f(x^k) | u^1, \dots, u^{k-1}) \geq C |w^k|^2 \mathbf{M}(\alpha_k^2 | u^1, \dots, u^{k-1}).$$

Согласно первому неравенству Чебышева

$$\mathbf{M}(\alpha_k^2 | u^1, \dots, u^{k-1}) \geq \alpha^2 \mathbf{P}\{\alpha_k^2 \geq \alpha^2 | u^1, \dots, u^{k-1}\} \geq p \alpha^2.$$

Поэтому

$$\mathbf{M}(f(x^{k-1}) - f(x^k) | u^1, \dots, u^{k-1}) \geq C |w^k|^2 p \alpha^2.$$

Условные математические ожидания в обеих частях неравенства представляют собой случайную величину (в данном случае функцию случайных векторов u^1, \dots, u^{k-1}). Поскольку эти случайные величины связаны неравенством, то и их математические ожидания связаны таким же неравенством:

$$\mathbf{M}(f(x^{k-1}) - f(x^k)) \geq C p \alpha^2 \mathbf{M} |w^k|^2, \quad k = 1, 2, \dots$$

Сложим полученные неравенства:

$$\mathbf{M}(f(x^0) - f(x^k)) = \sum_{i=1}^k \mathbf{M}(f(x^{i-1}) - f(x^i)) \geq C p \alpha^2 \sum_{i=1}^k \mathbf{M} |w^i|^2.$$

Из последнего неравенства нетрудно сделать вывод, что знакоположительный числовой ряд $\sum_{i=1}^{\infty} \mathbf{M} |w^i|^2$ сходится. Следовательно, его общий член стремится к нулю. К утверждению теоремы приходим, применяя первое неравенство Чебышева:

$$\mathbf{P}\{|w^k|^2 \geq \varepsilon^2\} \leq \frac{\mathbf{M} |w^k|^2}{\varepsilon^2}. \quad \blacktriangleright$$

Замечание 3.5. Доказанная теорема не дает гарантий сходимости релаксационной последовательности к точке минимума или стационарной точке. Однако можно утверждать, что если релаксационная последовательность сходится, то с вероятностью, сколь угодно близкой к 1, последовательность сходится к стационарной точке.

Чтобы обеспечить сходимость последовательности, нужны дополнительные условия. Например, достаточно, чтобы функция имела единственную стационарную точку, а множество точек $x \in \mathbb{R}^n$, в которых $f(x) \leq f(x^0)$, было ограниченным. Тогда это множество компактно, и любая подпоследовательность релаксационной последовательности будет содержать сходящуюся подпоследовательность, по которой последовательность градиентов (по вероятности) сходится к нулю. Следовательно, эта подпоследовательность сходится к стационарной точке. Но тогда и вся последовательность по вероятности сходится к стационарной точке.

Замечание 3.6. Условие (3.68) не так легко проверить. Однако его можно заменить другим условием. Например, пусть $\|x\|$ — какая-либо норма в \mathbb{R}^n . Обозначим $\left\| \frac{w^k}{\|w^k\|} - u^k \right\| = \varepsilon$, где вектор u^k , указывающий направление спуска, имеет единичную норму: $\|u^k\| = 1$. Обозначим $\tilde{w}^k = \frac{w^k}{\|w^k\|}$ и $z = \tilde{w}^k - u^k$.

Как известно из линейной алгебры, любая норма $\|x\|$ эквивалентна евклидовой норме $|x|$, т.е. существуют такие константы C_1 и C_2 , что $C_1|x| \leq \|x\| \leq C_2|x|$ для любого $x \in \mathbb{R}^n$. С учетом этого получаем

$$\begin{aligned} \frac{(\tilde{w}^k, u^k)}{|\tilde{w}^k||u^k|} &= \frac{(z + u^k, u^k)}{|u^k||u^k + z|} = \\ &= \frac{|u^k|^2 + (z, u^k)}{|u^k||u^k + z|} \geq \frac{|u| - |z|}{|u| + |z|} \geq \frac{1 - C\varepsilon}{1 + C\varepsilon} \geq 1 - 2C\varepsilon, \end{aligned}$$

где $C = C_2/C_1$. Отсюда следует, что если выполняется условие

$$P\left\{ \left\| \frac{w^k}{\|w^k\|} - u^k \right\| \leq \varepsilon \mid u^1, \dots, u^{k-1} \right\} \geq p > 0, \quad k \in \mathbb{N}, \quad (3.69)$$

где u^k — случайный вектор единичной нормы, а $\varepsilon < \frac{1}{2C}$, то выполняется и условие (3.68) с $\alpha = 1 - 2C\varepsilon$.

Нетрудно показать, что и условие (3.68) вытекает из условия (3.69), если α достаточно близко к 1. #

Остановимся на способах построения последовательности $\{u^k\}$ случайных векторов, определяющих направления спуска. Очевидный способ состоит в построении случайных векторов, равномерно распределенных на $(n-1)$ -мерной единичной сфере. Непосредственный подсчет площадей $(n-1)$ -мерной сферы показывает, что условие (3.68) выполняется.

Отметим, что моделирование случайного вектора, равномерно распределенного на многомерной сфере, — непростая задача. Согласно замечанию 3.6 можно выбрать произвольную норму $\|x\|$ в \mathbb{R}^n и использовать случайный вектор, равномерно распределенный на единичной сфере $\{x \in \mathbb{R}^n: \|x\| = 1\}$. Например, если $\|x\| = \|x\|_\infty = \max_{i=1, n} |x_i|$,

то единичная сфера представляет собой n -мерный куб, имеющий $2n$ граней. Нетрудно смоделировать $(n-1)$ -мерный случайный вектор

$z = (z_1, \dots, z_{n-1})$, равномерно распределенный на прямоугольном параллелепипеде

$$-1 \leq z_i \leq 1, \quad i = \overline{1, n-2}, \quad -1 \leq z_{n-1} \leq 4n-1,$$

который представляет собой $2n$ состыкованных $(n-1)$ -мерных кубов. Этот параллелепипед затем можно отобразить на границу n -мерного куба с сохранением площади и тем самым получить равномерное распределение на границе куба.

Можно отказаться от равномерного распределения единичного вектора u^k , обеспечивая при этом выполнение условия (3.68). Например, моделируем случайный вектор ξ , равномерно распределенный в единичном кубе $\|\xi\|_\infty \leq 1$ (достаточно взять n независимых случайных величин, равномерно распределенных на отрезке $[-1, 1]$). Если полученный вектор попал в куб $\|\xi\|_\infty \leq 1/2$, отбрасываем его и снова моделируем случайный вектор. В результате такой процедуры получим случайный вектор, равномерно распределенный на множестве $1/2 \leq \|\xi\|_\infty \leq 1$. В качестве направления спуска u^k выбираем вектор $u^k = \frac{\xi}{\|\xi\|_\infty}$. Можно проверить, что для таким образом построенного вектора выполняется условие (3.69). Значит, согласно замечанию 3.6 выполняется и условие (3.68).

Рассмотренные способы случайного поиска — это варианты *метода случайного поиска с возвратом при неудачном шаге*.

В *методе наилучшей пробы* выбираются s реализаций $u^{k,1}, u^{k,2}, \dots, u^{k,s}$ вектора u^k и сравниваются значения целевой функции в точках $x^{k-1} + \beta u^{k,i}$, $i = \overline{1, s}$. В качестве окончательного направления спуска на k -й итерации выбирается то из s анализируемых направлений, которое обеспечивает наименьшее значение целевой функции, т.е.

$$f(x^{k-1} + \beta u^k) = \min_{i=\overline{1,s}} f(x^{k-1} + \beta u^{k,i}).$$

Возможны модификации метода с переменным шагом спуска β_k .

В *методе статистического градиента* несколько построенных случайных направлений используются для получения статистической оценки градиента функции в текущей точке. Предположим, что вектор ξ равномерно распределен на сфере $|\xi| = 1$. Можно показать, что тогда случайный вектор

$$\eta = \frac{\partial f}{\partial \xi} \xi = (\text{grad } f(x^{k-1}), \xi) \xi$$

имеет математическое ожидание $\mathbf{M}\boldsymbol{\eta} = \rho_n \operatorname{grad} f(\mathbf{x}^{k-1})$. Следовательно, среднее значение нескольких реализаций этого вектора является статистической оценкой вектора, коллинеарного градиенту, и может использоваться для определения направления спуска.

В методе статистического градиента используют величину

$$\frac{f(\mathbf{x}^{k-1} + \gamma \boldsymbol{\xi}) - f(\mathbf{x}^{k-1})}{\gamma}$$

в качестве приближенного значения производной функции $f(\mathbf{x})$ по направлению $\boldsymbol{\xi}$. Суть метода в следующем. Строятся s реализаций $\boldsymbol{\xi}^1, \dots, \boldsymbol{\xi}^s$ вектора, равномерно распределенного на единичной сфере, и вычисляется вектор

$$\hat{\boldsymbol{\xi}} = - \sum_{i=1}^s \frac{f(\mathbf{x}^{k-1} + \gamma \boldsymbol{\xi}^i) - f(\mathbf{x}^{k-1})}{\gamma} \boldsymbol{\xi}^i.$$

В качестве направления спуска \mathbf{u}^k выбирается вектор, получающийся нормированием $\hat{\boldsymbol{\xi}}$, т.е.

$$\mathbf{u}^k = \frac{\hat{\boldsymbol{\xi}}}{|\hat{\boldsymbol{\xi}}|}.$$

Величины γ , s и β (шаг спуска) являются параметрами метода.

Возможен вариант метода статистического градиента, в котором случайный вектор $\boldsymbol{\xi}$ имеет более сложное распределение (например, равномерное на множестве $\|\boldsymbol{\xi}\| = 1$, определяемом некоторой нормой в \mathbb{R}^n , или равномерное на множестве $1/2 \leq \|\boldsymbol{\xi}\|_\infty \leq 1$, как в методе случайного поиска с возвратом при неудачном шаге). В этом случае математическое ожидание вектора $\boldsymbol{\eta}$ может не совпадать с направлением градиента функции, но при выполнении условия (3.69) будет возможным направлением спуска. Следовательно, его статистическая оценка может использоваться для задания направления спуска.

Вопросы для самопроверки

1. Перечислите известные вам численные методы безусловной оптимизации. Какого порядка эти методы?
2. Какую последовательность называют релаксационной?

3. Что называют: а) градиентным спуском; б) исчерпывающим спуском; в) наискорейшим спуском? Чем исчерпывающий спуск отличается от наискорейшего спуска? Когда они совпадают?

4. Дайте определение векторов, сопряженных относительно симметрической матрицы. Какова связь этого понятия с понятием ортонормированного базиса?

5. Опишите варианты метода сопряженных направлений, предназначенные для минимизации неквадратичных функций.

6. Что такое ньютоновское направление спуска? Чем это направление отличается от направления спуска в квазиньютоновских методах?

7. Опишите основную суть методов симплексного поиска.

8. Дайте определение сходимости случайных величин: а) по вероятности; б) почти наверное; в) в среднем квадратичном.

9. Опишите способы построения векторов, определяющих направления спуска в методах случайного поиска.

10. Решите задачу безусловной оптимизации

$$11x^2 + 3y^2 + 6xy - 2\sqrt{10}(x - 3y) - 22 \rightarrow \min,$$

выбрав в качестве начальной точку $(\sqrt{10}, 0)$ и параметр точности $\varepsilon = 0,01$. Для решения задачи используйте несколько методов первого и второго порядков, несколько методов прямого поиска, какой-либо метод случайного поиска. Сравните полученные результаты по количеству итераций, необходимых для достижения заданной точности.

АНАЛИТИЧЕСКИЕ МЕТОДЫ НЕЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

Общая задача нелинейного программирования может включать ограничения типа равенства или ограничения типа неравенства, а также оба вида ограничений одновременно. В этой главе сначала кратко обсудим задачу нелинейного программирования, в которую входят только ограничения типа равенства, а затем перейдем к общему случаю.

4.1. Минимизация целевой функции на заданном множестве

Рассмотрим общую задачу математического программирования

$$f_0(x) \rightarrow \min, \quad x \in \Omega, \quad (4.1)$$

где $\Omega \subset \mathbb{R}^n$ — допустимое множество, входящее в область определения целевой функции $f_0(x)$. Пока не будем уточнять, каким способом задано множество Ω , но будем считать, что оно не пусто.

Если множество Ω компактное, то существует хотя бы одна точка $x^* \in \Omega$, в которой непрерывная на этом множестве функция $f_0(x)$ достигает своего наименьшего значения. Следовательно, задача (4.1) в случае непрерывной целевой функции и компактного допустимого множества имеет решение.

Теорема 4.1. Пусть функция $f_0(x)$ непрерывна на замкнутом множестве $\Omega \subset \mathbb{R}^n$. Если для некоторого $\alpha \in \mathbb{R}$ множество $X = \{x \in \Omega: f_0(x) \leq \alpha\}$ не пустое и ограниченное в \mathbb{R}^n , то задача (4.1) имеет решение.

◀ Так как функция $f_0(x)$ непрерывна в Ω , множество X , определенное нестрогим неравенством $f_0(x) \leq \alpha$, является замкнутым. Действительно, если $x^0 \in \Omega$ — предельная точка множества X , то в любой

окрестности этой точки можно указать точки множества X . Возьмем произвольное число $\varepsilon > 0$ и согласно условию непрерывности функции $f_0(x)$ в точке x^0 выберем окрестность $U(x^0, \delta)$, для которой $|f_0(x) - f_0(x^0)| < \varepsilon$ при $x \in U(x^0, \delta) \cap \Omega$. В выбранной окрестности существует точка x^1 , принадлежащая множеству X , т.е. для этой точки выполняется неравенство $f_0(x^1) \leq \alpha$. Значит, $f_0(x^0) < f_0(x^1) + \varepsilon \leq \alpha + \varepsilon$. Поскольку записанное неравенство верно для любого числа $\varepsilon > 0$, то $f_0(x^0) \leq \alpha$ и $x^0 \in X$. Итак, множество X содержит все свои предельные точки, а потому замкнуто.

Функция $f_0(x)$, непрерывная на замкнутом ограниченном множестве X , достигает на X наименьшего значения в некоторой точке x^* . Из условия $x^* \in X$ вытекает, что $f_0(x^*) \leq \alpha$. Если $x \in \Omega \setminus X$, то в соответствии с определением множества X выполняется неравенство $f_0(x) > \alpha$. Поэтому

$$f_0(x) > \alpha \geq f_0(x^*), \quad x \in \Omega \setminus X,$$

и значение целевой функции в точке x^* является наименьшим на всем множестве Ω . ►

Рассмотрим произвольную точку $x \in \Omega$. Будем говорить, что единичный вектор $e \in \mathbb{R}^n$ задает *допустимое направление* в точке x , если для некоторого числа $\delta > 0$ каждая точка $x + te$, $t \in (0, \delta)$, принадлежит множеству Ω . Множество $C(x)$ всех единичных векторов, каждый из которых задает в точке $x \in \Omega$ допустимое направление, называют *конусом допустимых направлений* в точке x .

Если множество Ω выпуклое, то допустимое направление в точке $x \in \Omega$ можно задать любым вектором $\tilde{x} - x$, определяемым произвольной точкой $\tilde{x} \in \Omega$, не совпадающей с x . Действительно, в этом случае для $e = \frac{\tilde{x} - x}{|\tilde{x} - x|}$ в качестве $\delta > 0$ можно выбрать $\delta = |\tilde{x} - x|$.

Теорема 4.2. Пусть функция $f_0(x)$ дифференцируема* в точке $x^* \in \mathbb{R}^n$. Если x^* является точкой наименьшего значения функции $f_0(x)$ на множестве Ω , то

$$(\text{grad } f_0(x^*), e) \geq 0, \quad e \in C(x^*). \quad (4.2)$$

*Дифференцируемость функции в точке предполагает, что функция определена в некоторой окрестности этой точки, т.е. все точки дифференцируемости функции являются внутренними точками ее области определения.

◄ Выберем вектор $e \in C(x^*)$ и соответствующее этому вектору число $\delta > 0$, так что $x^* + te \in \Omega$ при $t \in (0, \delta)$. Функция $\varphi_e(t) = f_0(x^* + te)$ определена на полуинтервале $[0, \delta)$ и достигает наименьшего значения при $t = 0$, поскольку $\varphi_e(0) = f_0(x^*) \leq f_0(x) = \varphi(t)$, где $x = x^* + te \in \Omega$, $t \in (0, \delta)$. Отсюда вытекает, что

$$\frac{\varphi_e(t) - \varphi_e(0)}{t} \geq 0, \quad t \in (0, \delta).$$

Так как функция $f_0(x)$ дифференцируема в точке x^* , то сложная функция $\varphi_e(t)$ дифференцируема в точке $t = 0$. Поэтому существует предел

$$\varphi'_e(0) = \lim_{t \rightarrow +0} \frac{\varphi_e(t) - \varphi_e(0)}{t} \geq 0.$$

Производная функции $\varphi_e(t)$ в точке $t = 0$ представляет собой производную функции $f_0(x)$ в точке x^* по направлению вектора e , которую с помощью градиента функции можно записать в виде $(\text{grad } f_0(x^*), e)$. Таким образом, неравенство $\varphi'_e(0) \geq 0$ равносильно неравенству (4.2). ►

Утверждение теоремы 4.2 можно рассматривать как обобщение необходимого условия локального экстремума. Чтобы придать такой трактовке более точный смысл, введем следующее понятие. Будем говорить, что функция $f_0(x)$, определенная на множестве Ω , имеет в точке $x^* \in \Omega$ *локальный минимум на множестве Ω* , если существует такая окрестность U точки x^* , что $f_0(x) \geq f_0(x^*)$ при $x \in U \cap \Omega$. Если последнее неравенство при $x \neq x^*$ является строгим, то будем говорить о *строгом локальном минимуме на множестве Ω* . Введенные понятия расширяют понятия локального минимума и строгого локального минимума: согласно определению точка локального (строгого локального) минимума должна быть внутренней точкой множества Ω .

Повторяя почти дословно доказательство теоремы 4.2, можно показать, что если $x^* \in \Omega$ — точка локального минимума функции $f_0(x)$ на множестве Ω , а функция $f_0(x)$ дифференцируема в точке x^* , то имеет место условие (4.2).

Если точка x^* является внутренней точкой множества Ω , то конус допустимых направлений $C(x^*)$ содержит все n -мерные единичные векторы, а условие (4.2) легко преобразовать в обычное условие ло-

кального экстремума функции, записанное в виде $(\text{grad } f_0(x^*), e) = 0$, $e \in \mathbb{R}^n$. Действительно, в этом случае условие (4.2) выполняется для любой пары единичных векторов e и $-e$, а потому в этом условии знак нестрогого неравенства можно заменить знаком равенства. Кроме того, условие $|e| = 1$ не является существенным, так как множитель, равный длине вектора e , можно вынести за знак скалярного произведения.

Отметим, что условие $(\text{grad } f_0(x^*), e) = 0$, $e \in \mathbb{R}^n$, эквивалентно условию $\text{grad } f_0(x^*) = 0$. В самом деле, первое условие непосредственно вытекает из второго. Наоборот, если верно первое условие, то равенство $(\text{grad } f_0(x^*), e) = 0$ верно и при $e = \text{grad } f_0(x^*)$, т.е. $|\text{grad } f_0(x^*)|^2 = 0$, что равносильно равенству $\text{grad } f_0(x^*) = 0$ — обычной формулировке необходимого условия локального минимума.

Теорема 4.3. Пусть функция $f_0(x)$ определена на выпуклом множестве Ω и дифференцируема в точке $x^* \in \Omega$. Если для некоторого числа $\alpha > 0$ выполнено условие

$$(\text{grad } f_0(x^*), e) \geq \alpha, \quad e \in C(x^*), \quad (4.3)$$

то x^* — точка строгого локального минимума этой функции на множестве Ω .

◀ Условие дифференцируемости функции $f_0(x)$ в точке x^* означает, что функция $f_0(x)$ определена в некоторой окрестности $U(x^*, \delta)$ точки x^* и в этой окрестности имеет место представление

$$f_0(x) - f_0(x^*) = (\text{grad } f_0(x^*), \Delta x) + o(\Delta x),$$

где $\Delta x = x - x^*$, а $\frac{o(\Delta x)}{|\Delta x|}$ — бесконечно малая при $\Delta x \rightarrow 0$. Уменьшим, если необходимо, радиус окрестности δ настолько, чтобы в этой окрестности выполнялось неравенство $|o(\Delta x)| < \frac{\alpha}{2} |\Delta x|$.

Пусть $x \in \Omega \cap U(x^*, \delta)$, $x \neq x^*$. Тогда вектор $e = \frac{x - x^*}{|x - x^*|}$ задает допустимое направление. Следовательно,

$$\begin{aligned} f_0(x) - f_0(x^*) &= (\text{grad } f_0(x^*), \Delta x) + o(\Delta x) \geq \\ &\geq |\Delta x| (\text{grad } f_0(x^*), e) - \frac{\alpha}{2} |\Delta x| \geq |\Delta x| \alpha - \frac{\alpha}{2} |\Delta x| = \frac{\alpha}{2} |\Delta x| > 0. \end{aligned}$$

Но это и означает, что в точке x^* функция $f_0(x)$ имеет локальный минимум на множестве Ω . ►

Отметим, что условие (4.3) может выполняться лишь в граничной точке множества Ω . Действительно, если бы это условие выполнялось во внутренней точке x^* множества Ω , то точка x^* была бы точкой локального минимума функции $f_0(x)$ и, значит, в этой точке выполнялось бы равенство $(\text{grad } f_0(x^*), e) = 0$, $e \in \mathbb{R}^n$. Но это противоречит условию (4.3).

Доказанные необходимое и достаточное условия локального минимума функции на множестве можно использовать при решении задач минимизации функции на заданном множестве Ω . Если целевая функция достигает наименьшего значения на Ω , то для определения точки минимума можно придерживаться следующего порядка действий.

1. Находим все стационарные точки функции, являющиеся внутренними точками множества Ω .

2. С помощью необходимого условия (см. теорему 4.2) находим все точки на границе множества Ω , в которых функция может иметь локальный минимум на Ω .

3. Сравнивая значения минимизируемой функции в найденных точках, находим ту точку, в которой функция имеет наименьшее значение.

При решении задачи выпуклого программирования, в которой выпуклая функция минимизируется на выпуклом множестве, достаточно найти любую точку ее локального минимума, так как выпуклая функция в любой точке локального минимума достигает наименьшего значения. В этом случае уместно использовать достаточное условие локального минимума на множестве (см. теорему 4.3), которое не требует вычисления частных производных второго порядка и легко проверяется.

Узким местом предложенной схемы является применение необходимого условия локального минимума на множестве. Если необходимое условие экстремума функции записывается в виде системы уравнений, то необходимое условие локального минимума на множестве представлено в виде системы неравенств, в которых варьируется не только неизвестная точка, но и вектор e допустимого направления. Выделить с помощью теоремы 4.2 точки, „подозрительные“ на локальный минимум, можно лишь в случае, когда допустимое множество Ω имеет границу простой структуры.

4.2. Минимизация при ограничениях типа равенства

Задачу нелинейного программирования с ограничениями типа равенства можно записать следующим образом:

$$\begin{cases} f_0(x) \rightarrow \min; \\ f_l(x) = 0, \quad l = \overline{1, k}. \end{cases} \quad (4.4)$$

В данном случае допустимое множество Ω имеет вид

$$\Omega = \{x \in \mathbb{R}^n: f_l(x) = 0, \quad l = \overline{1, k}\}.$$

Сформулированную задачу для дифференцируемых функций $f_l(x)$ можно связать с задачей исследования целевой функции на условный экстремум. Если точка x^* есть решение задачи (4.4), т.е. если целевая функция $f_0(x)$ достигает в этой точке наименьшего значения на множестве Ω , то точка x^* является точкой условного локального минимума функции $f_0(x)$ при условиях $f_l(x) = 0, \quad l = \overline{1, k}$ (эти уравнения называют *уравнениями связи*). Поэтому решать задачу (4.4) можно следующим образом. Найти все точки, в которых функция может иметь условный локальный минимум, а затем путем сравнения значений функции выбрать из этих точек ту, в которой функция имеет наименьшее значение. Если задача имеет решение (т.е. целевая функция при заданных ограничениях достигает наименьшего значения), то этим решением будет найденная точка.

Поиск точек, „подозрительных“ на условный экстремум, выполняют с помощью необходимого условия условного локального экстремума, суть которого в следующем. Если точка x^* является точкой условного экстремума функции $f_0(x)$ при условиях $f_l(x) = 0, \quad l = \overline{1, k}$, причем функции $f_l(x), \quad l = \overline{0, k}$, непрерывно дифференцируемы в окрестности точки x^* и матрица Якоби $J(x)$ векторной функции $f(x) = (f_1(x) \dots f_k(x))^T$ в точке x^* имеет ранг k (максимальный ранг, равный количеству строк), то существуют такие множители Лагранжа $\lambda_1, \dots, \lambda_k$, что для функции Лагранжа

$$L(x) = f_0(x) + \sum_{l=1}^k \lambda_l f_l(x) \quad (4.5)$$

в точке x^* выполняется необходимое условие экстремума

$$\text{grad } L(x^*) = 0,$$

или

$$\frac{\partial L(x^*)}{\partial x_1} = \frac{\partial L(x^*)}{\partial x_2} = \dots = \frac{\partial L(x^*)}{\partial x_n} = 0.$$

Поиск стационарных точек функции Лагранжа сводится к решению системы $n + k$ уравнений

$$\begin{cases} \frac{\partial L(x)}{\partial x_j} = 0, & j = \overline{1, n}; \\ f_l(x) = 0, & l = \overline{1, k}, \end{cases}$$

содержащей $n + k$ неизвестных $x_1, \dots, x_n, \lambda_1, \dots, \lambda_k$.

Таким образом, исследование функции на условный экстремум состоит в выделении некоторого набора точек, „подозрительных“ на экстремум, и в проверке каждой точки, является ли она в действительности точкой условного экстремума. Набор точек, „подозрительных“ на экстремум, можно разделить на три группы:

- 1) точки, в которых хотя бы одна из функций $f_l(x)$, $l = \overline{0, k}$, не дифференцируема или одна из ее частных производных разрывна;
- 2) точки, в которых ранг матрицы Якоби $J(x)$ системы уравнений связи меньше количества k уравнений связи;
- 3) стационарные точки функции Лагранжа.

Отличие задачи минимизации при ограничениях типа равенства от задачи исследования на условный экстремум состоит в том, что в задаче минимизации нет необходимости проверять, является ли точка, „подозрительная“ на экстремум, точкой условного локального минимума. Можно просто сравнить значения минимизируемой функции в этих точках и выбрать ту, в которой значение функции наименьшее. Эта точка и будет решением задачи минимизации. Правда, последнее утверждение верно лишь в случае, когда задача минимизации имеет решение. В действительности возможна ситуация, когда целевая функция имеет точки условного локального минимума, но не достигает наименьшего значения.

Необходимое условие условного локального экстремума можно модифицировать так, что определение стационарных точек функции Лагранжа будет включать в себя и поиск точек, в которых нарушается

условие на ранг матрицы Якоби системы уравнений связи (необходимое условие условного локального экстремума формулируется в предположении, что ранг системы уравнений связи максимален и совпадает с числом уравнений связи). Для этого изменим понятие функции Лагранжа, используя вместо (4.5) формулу

$$\tilde{L}(x) = \lambda_0 f_0(x) + \sum_{l=1}^k \lambda_l f_l(x), \quad (4.6)$$

в которую добавлен еще один множитель Лагранжа λ_0 . Следующее утверждение называют *обобщенным правилом множителей Лагранжа*.

Теорема 4.4. Если $x^* \in \mathbb{R}^n$ — точка условного экстремума функции $f_0(x)$ при условиях $f_l(x) = 0, l = \overline{1, k}$, причем функции $f_l(x), l = \overline{0, k}$, непрерывно дифференцируемы в некоторой окрестности точки x^* , то существуют такие множители Лагранжа $\lambda_l, l = \overline{0, k}$, не все равные нулю, что для функции Лагранжа $\tilde{L}(x)$ точка x^* является стационарной, т.е.

$$\text{grad } \tilde{L}(x^*) = 0. \quad (4.7)$$

◀ Если ранг матрицы Якоби $J(x^*)$ системы функций $f_l(x), l = \overline{1, k}$, в точке x^* равен их количеству k , то достаточно положить $\lambda_0 = 1$. Тогда утверждение теоремы будет вытекать из необходимого условия условного экстремума. Предположим, что $\text{Rg } J(x^*) < k$. Тогда строки матрицы $J(x^*)$ линейно зависимы, например, в силу теоремы о базисном миноре. Это означает, что для некоторых значений $\lambda_1, \dots, \lambda_k$, одновременно не равных нулю, выполняется равенство

$$\lambda_1 \text{grad } f_1(x^*) + \lambda_2 \text{grad } f_2(x^*) + \dots + \lambda_k \text{grad } f_k(x^*) = 0.$$

Положив $\lambda_0 = 0$, запишем

$$\lambda_0 \text{grad } f_0(x^*) + \sum_{l=1}^k \lambda_l \text{grad } f_l(x^*) = 0,$$

что в силу равенства (4.6) совпадает с равенством (4.7). ►

Нетрудно увидеть, что множители Лагранжа, существование которых вытекает из теоремы 4.4, определяются неоднозначно: их можно,

не нарушая утверждения теоремы, умножить на любой постоянный множитель. Поэтому для определения $k + 1$ множителя Лагранжа $\lambda_0, \dots, \lambda_k$ и n координат точки x^* достаточно $k + n$ уравнений. Такое количество дают условие (4.7) (это n уравнений) и совокупность уравнений связи (их количество равно k).

4.3. Общая задача нелинейного программирования

Общую задачу нелинейного программирования можно записать следующим образом:

$$\begin{cases} f_0(x) \rightarrow \min; \\ f_l(x) = 0, \quad l = \overline{1, k}; \quad g_i(x) \leq 0, \quad i = \overline{1, m}. \end{cases} \quad (4.8)$$

При этом предполагается, что одна из фигурирующих в задаче функций не является линейной. Считаем, что все функции $f_l(x)$, $l = \overline{0, k}$, и $g_i(x)$, $i = \overline{1, m}$, определены на некотором открытом множестве $X \subset \mathbb{R}^n$. Упрощая изложение, далее ограничимся случаем $X = \mathbb{R}^n$. В конкретной ситуации можно добиться выполнения этого ограничения, доопределив все функции вне множества X , например, большим постоянным значением, которое не повлияет на процесс минимизации целевой функции.

Сформулированная задача представляет собой частный случай общей задачи математического программирования, заключающейся в определении наименьшего значения целевой функции $f_0(x)$ на допустимом множестве

$$\Omega = \{x \in X: f_l(x) = 0, l = \overline{1, k}, g_i(x) \leq 0, i = \overline{1, m}\}. \quad (4.9)$$

Отметим, что если все функции $f_l(x)$, $l = \overline{1, k}$, и $g_i(x)$, $i = \overline{1, m}$, непрерывны в \mathbb{R}^n , то множество Ω замкнуто. В самом деле, это множество можно представить как пересечение конечного числа множеств вида $\{x \in \mathbb{R}^n: \varphi(x) \leq \alpha\}$, образованных непрерывной в \mathbb{R}^n функцией $\varphi(x)$. В данном случае в качестве $\varphi(x)$ используются функции $f_l(x)$, $-f_l(x)$, $l = \overline{1, k}$, и $g_i(x)$, $i = \overline{1, m}$, а $\alpha = 0$. Такие множества являются замкнутыми (см. доказательство теоремы 4.1). Следовательно, и множество Ω , как пересечение конечного числа замкнутых множеств, является замкнутым.

Большинство известных методов решения задачи нелинейного программирования позволяет найти лишь точки локального минимума целевой функции на заданном множестве. В этой связи важную роль играет априорная информация о существовании решения задачи, т.е. информация о том, достигает ли целевая функция наименьшего значения на допустимом множестве. Если допустимое множество Ω в задаче нелинейного программирования замкнуто, то доказательство существования решения в этой задаче можно строить с помощью теоремы 4.1, проверяя непрерывность целевой функции $f_0(x)$ и ограниченность при некотором α множества $\Omega_\alpha = \{x \in \Omega: f_0(x) \leq \alpha\}$. В частности, если множество Ω ограничено, а целевая функция непрерывна на Ω , то задача нелинейного программирования имеет решение.

Однако проверить ограниченность множества Ω (или Ω_α) не просто. Поэтому интересны другие условия, налагаемые на целевую функцию и ограничения, которые позволяют сделать заключение о существовании решения без использования ограниченности допустимого множества. В задаче выпуклого программирования, в которой целевая функция выпукла и допустимое множество Ω выпукло, любая точка локального минимума целевой функции является решением этой задачи.

Теорема 4.5. Если функции $g_i(x)$, $i = \overline{1, m}$, выпуклы в \mathbb{R}^n , то множество

$$\tilde{\Omega} = \{x \in \mathbb{R}^n: g_i(x) \leq 0, i = \overline{1, m}\} \quad (4.10)$$

является выпуклым.

◀ Утверждение теоремы очевидно, если множество $\tilde{\Omega}$ пусто или содержит всего лишь одну точку. Поэтому ограничимся случаем, когда это множество содержит по крайней мере две точки. Выберем в $\tilde{\Omega}$ произвольные точки x^1 и x^2 и докажем, что для любого числа $\alpha \in (0, 1)$ выполняется включение $\alpha x_1 + (1 - \alpha)x_2 \in \tilde{\Omega}$.

Из определения множества $\tilde{\Omega}$ вытекает, что $g_i(x^1) \leq 0$ и $g_i(x^2) \leq 0$, $i = \overline{1, m}$. Поэтому в силу условия выпуклости функций $g_i(x)$ для любого числа $\alpha \in (0, 1)$

$$g_i(\alpha x_1 + (1 - \alpha)x_2) \leq \alpha g_i(x_1) + (1 - \alpha)g_i(x_2) \leq 0, \quad i = \overline{1, m}.$$

Значит, точка $\alpha x_1 + (1 - \alpha)x_2$ принадлежит множеству $\tilde{\Omega}$. ►

Замечание 4.1. Из теоремы 4.5 следует, что если допустимое множество Ω определяется только ограничениями типа неравенства $g_i(x) \leq 0, i = \overline{1, m}$, причем все функции $g_i(x)$ выпуклые в \mathbb{R}^n , то множество Ω выпукло. Наличие ограничений типа равенства усложняет проверку допустимого множества на выпуклость. Однако если функции $f_i(x)$ во всех ограничениях типа равенства являются линейными, а все функции $g_i(x)$ в ограничениях типа неравенства — выпуклыми, то допустимое множество выпукло. Действительно, в этом случае допустимое множество Ω является пересечением множества $\tilde{\Omega}$ вида (4.10) с гиперплоскостями $f_i(x) = 0, i = \overline{1, m}$. Множество $\tilde{\Omega}$ выпукло в силу теоремы 4.5, гиперплоскости также являются выпуклыми множествами. Следовательно, множество Ω , как пересечение выпуклых множеств, выпукло. #

Ограничения типа неравенства в задаче нелинейного программирования введением дополнительных переменных можно преобразовать в ограничения типа равенства. Например, ограничения $g_i(x) \leq 0, i = \overline{1, m}$, с помощью дополнительных переменных z_i можно записать в виде

$$g_i(x) + z_i^2 = 0, \quad i = \overline{1, m}. \quad (4.11)$$

Таким образом, общую задачу нелинейного программирования можно свести к частному случаю, когда в задаче нет ограничений типа неравенства. Это позволяет в решении задачи использовать технику исследования целевой функции на условный экстремум (см. разд. 4.2). Впрочем, подобная техника может использоваться и непосредственно в задаче, имеющей ограничения типа неравенства.

Пример 4.1. Рассмотрим задачу нелинейного программирования

$$\begin{cases} f_0(x_1, x_2) \rightarrow \min; \\ g_1(x_1, x_2) \leq 0, \quad g_2(x_1, x_2) \leq 0, \quad g_3(x_1, x_2) \leq 0, \end{cases}$$

в которой функции $f_0(x_1, x_2), g_1(x_1, x_2), g_2(x_1, x_2), g_3(x_1, x_2)$ непрерывно дифференцируемы в \mathbb{R}^2 . Предположим, что множество Ω , которое определяется тремя неравенствами, ограничено и имеет вид, изображенный на рис. 4.1.

В этой задаче наименьшее значение функции $f_0(x_1, x_2)$ может достигаться либо во внутренней точке множества Ω , т.е. в точке

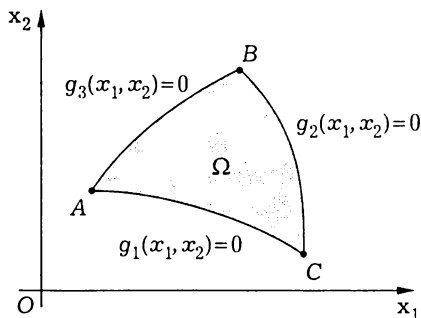


Рис. 4.1

множества

$$\Omega^c = \{(x_1, x_2): g_i(x_1, x_2) < 0, i = 1, 2, 3\},$$

либо на границе множества Ω на одной из дуг AB , AC , BC , либо в угловых точках границы A , B , C . Поэтому решение задачи можно строить по следующей схеме.

1. Определяем все стационарные точки целевой функции в области Ω^c .

2. Решаем три задачи на условный экстремум:

$$\begin{cases} f_0(x_1, x_2) \rightarrow \text{extr}, \\ g_1(x_1, x_2) = 0; \end{cases} \quad \begin{cases} f_0(x_1, x_2) \rightarrow \text{extr}, \\ g_2(x_1, x_2) = 0; \end{cases} \quad \begin{cases} f_0(x_1, x_2) \rightarrow \text{extr}, \\ g_3(x_1, x_2) = 0. \end{cases}$$

Среди найденных точек отбираем те, которые попадают на соответствующую дугу — AB , AC , BC . Например, из решений первой задачи нужно оставить такие точки (x_1, x_2) , которые удовлетворяют условиям $g_2(x_1, x_2) < 0$, $g_3(x_1, x_2) < 0$.

3. К точкам, отобранным в первых двух пунктах, добавляем угловые точки A , B , C , которые находятся из систем

$$\begin{cases} g_1(x_1, x_2) = 0, \\ g_2(x_1, x_2) < 0, \\ g_3(x_1, x_2) = 0; \end{cases} \quad \begin{cases} g_1(x_1, x_2) < 0, \\ g_2(x_1, x_2) = 0, \\ g_3(x_1, x_2) = 0; \end{cases} \quad \begin{cases} g_1(x_1, x_2) = 0, \\ g_2(x_1, x_2) = 0, \\ g_3(x_1, x_2) < 0. \end{cases}$$

4. Во всех отобранных точках вычисляем значение целевой функции и отбираем ту (или те), в которой значение функции наименьшее. #

Описанный в примере 4.1 процесс анализа внутренних точек допустимого множества и различных частей границы можно объединить в одну формальную процедуру. Рассмотрим общую задачу нелинейного программирования (4.8).

Теорема 4.6. Если точка $x^* \in \Omega$ является точкой локального минимума функции $f_0(x)$ на множестве Ω вида (4.9), причем функции $f_l(x)$, $l = \overline{0, k}$, и $g_i(x)$, $i = \overline{1, m}$, непрерывно дифференцируемы в окрестности точки x^* , то существуют такие числа $\lambda_0 \geq 0$, λ_l , $l = \overline{1, k}$, и $\mu_i \geq 0$, $i = \overline{1, m}$, одновременно не равные нулю, что для функции

$$\tilde{L}(x) = \lambda_0 f_0(x) + \sum_{l=1}^k \lambda_l f_l(x) + \sum_{i=1}^m \mu_i g_i(x)$$

выполняется необходимое условие экстремума

$$\text{grad } \tilde{L}(x^*) = 0$$

и, кроме того, выполняются условия дополняющей нежесткости

$$\mu_i g_i(x^*) = 0, \quad i = \overline{1, n}. \quad \#$$

Теорему 4.6 принято называть *теоремой Куна — Таккера*, а функцию $\tilde{L}(x)$ — *функцией Лагранжа*. Хотя функция Лагранжа определена для любых значений параметров λ_l , $l = \overline{0, k}$, и μ_i , $i = \overline{1, m}$, мы, учитывая утверждение теоремы Куна — Таккера, будем рассматривать ее лишь при $\mu_i \geq 0$, $i = \overline{1, n}$.

Отметим особую роль в этой теореме условий дополняющей нежесткости. В рассматриваемой точке x^* локального минимума функции $f_0(x)$ на множестве Ω каждое из ограничений $g_i(x) \leq 0$, $i = \overline{1, m}$, выполняется либо в виде равенства $g_i(x^*) = 0$, либо в виде строгого неравенства $g_i(x^*) < 0$. Ограничения первого вида называют *активными ограничениями*, в то время как остальные — *неактивными (пассивными) ограничениями*. Для активных ограничений соответствующее условие дополняющей нежесткости выполнено. Неактивные ограничения в силу непрерывности фигурирующих в задаче функций выполняются в виде строгого неравенства не только в точке x^* , но и в некоторой окрестности этой точки. Поэтому после удаления этих ограничений из задачи точка x^* останется точкой локального минимума. Полагая для таких ограничений $\mu_i = 0$, мы, с одной сторо-

ны, обеспечиваем выполнение условия дополняющей нежесткости, а с другой — удаляем соответствующее слагаемое из функции Лагранжа. В конечном счете утверждение теоремы Куна — Таккера сводится к обобщенному принципу Лагранжа.

Вернемся к примеру 4.1 и поясним, как работает сформулированная теорема. В данном случае ограничений типа равенства нет, а есть три ограничения типа неравенства. Если точка x^* находится внутри множества Ω , то утверждение теоремы будет верно при $\mu_1 = \mu_2 = \mu_3 = 0$, $\lambda_0 = 1$. Если точка x^* находится на одной из дуг, например на дуге AB , то в этой точке $g_3 = 0$. Утверждение теоремы вытекает из обобщенного принципа Лагранжа для задачи с ограничениями типа равенства при $\mu_1 = \mu_2 = 0$. Наконец, в угловой точке, например в точке A , выполняются два равенства $g_1 = 0$ и $g_3 = 0$. Чтобы обеспечить условия дополняющей нежесткости, полагаем $\mu_2 = 0$. При этом необходимое условие экстремума для функции Лагранжа состоит из двух уравнений, в которых неизвестны три параметра λ_0 , μ_1 и μ_3 , причем относительно этих параметров система уравнений является линейной и однородной. Ясно, что такая система имеет ненулевые решения.

Пример 4.2. Рассмотрим задачу нелинейного программирования

$$\begin{cases} f_0(x_1, x_2) = (x_1 - 3)^2 + (x_2 - 3)^2 \rightarrow \min; \\ -x_1 \leq 0, \quad -x_2 \leq 0, \quad x_1 + x_2 - 4 \leq 0. \end{cases}$$

В этой задаче допустимое множество представляет собой замкнутую треугольную область на плоскости (рис. 4.2), в которой непрерывная целевая функция достигает наименьшего значения. Кроме

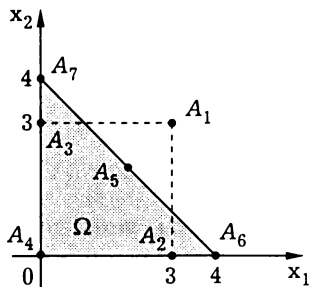


Рис. 4.2

того, нетрудно увидеть, что ограничения задачи линейные, а целевая функция, как квадратичная функция с положительно определенной матрицей, строго выпукла. Таким образом, рассматриваемая задача относится к задачам выпуклого программирования.

Проследим на примере этой простейшей задачи, как работает теорема Куна — Таккера. В этой задаче нет ограничений типа равенства, функция Лагранжа имеет вид

$$\tilde{L}(x_1, x_2) = \lambda_0((x_1 - 3)^2 + (x_2 - 3)^2) - \mu_1 x_1 - \mu_2 x_2 + \mu_3(x_1 + x_2 - 4).$$

Записываем систему уравнений, включающую необходимое условие экстремума функции Лагранжа и условия дополняющей нежесткости:

$$\begin{cases} 2\lambda_0(x_1 - 3) - \mu_1 + \mu_3 = 0, \\ 2\lambda_0(x_2 - 3) - \mu_2 + \mu_3 = 0, \\ \mu_1 x_1 = 0, \\ \mu_2 x_2 = 0, \\ \mu_3(x_1 + x_2 - 4) = 0. \end{cases}$$

В случае $\lambda_0 = 0$ из двух первых уравнений находим $\mu_1 = \mu_2 = \mu_3$, причем общее значение этих трех коэффициентов ненулевое, так как по теореме Куна — Таккера хотя бы один из множителей Лагранжа должен быть ненулевым. Из последних трех уравнений получаем: $x_1 = x_2 = 0$, $x_1 + x_2 = 4$, а это невозможно. Поэтому решений системы с $\lambda_0 = 0$ нет.

Пусть $\lambda_0 \neq 0$. В этом случае можно взять для λ_0 любое ненулевое значение. Положим $\lambda_0 = 1/2$. Тогда из первых двух уравнений можно выразить x_1 и x_2 и подставить в остальные. Получим систему трех уравнений

$$\begin{cases} \mu_1(3 + \mu_1 - \mu_3) = 0, \\ \mu_2(3 + \mu_2 - \mu_3) = 0, \\ \mu_3(2 + \mu_1 + \mu_2 - 2\mu_3) = 0. \end{cases}$$

Рассматривая различные варианты, находим семь решений системы (табл. 4.1).

Первое решение соответствует точке локального минимума целевой функции, причем эта точка не попадает в допустимое множество Ω .

Второе, третье и пятое решения соответствуют точкам условного локального минимума на сторонах треугольника, ограничивающего допустимое множество. Наконец, четвертое, шестое и седьмое решения соответствуют трем угловым точкам множества (вершинам треугольника). Сравнение значений функции показывает, что единственным решением рассматриваемой задачи является точка A_5 с координатами $x_1 = x_2 = 2$. К этим выводам легко прийти, если учесть, что значением целевой функции в точке (x_1, x_2) является квадрат расстояния от этой точки до точки $(3, 3)$.

4.4. Седловая точка функции Лагранжа

Рассмотрим задачу нелинейного программирования

$$\begin{cases} f_0(x) \rightarrow \min; \\ g_i(x) \leq 0, \quad i = \overline{1, m}, \end{cases} \quad (4.12)$$

в которой нет ограничений типа равенства. Предполагаем, что функции $f_0(x)$ и $g_i(x)$, $i = \overline{1, m}$, определенные в \mathbb{R}^n , могут быть недифференцируемыми. Как и выше, сформируем функцию Лагранжа

$$L(x, \mu) = f_0(x) + \sum_{i=1}^m \mu_i g_i(x), \quad (4.13)$$

где $\mu = (\mu_1, \dots, \mu_m) \in \mathbb{R}_*^m$, \mathbb{R}_*^m — неотрицательный ортант в \mathbb{R}^m . Будем рассматривать функцию Лагранжа как функцию всей совокупности переменных x и μ . Отметим, что в данном случае в функцию Лагранжа не включен множитель λ_0 .

Определение 4.1. Точку (x^*, μ^*) , где $x^* \in \Omega$, $\mu \in \mathbb{R}_*^m$, называют *седловой точкой* функции Лагранжа $L(x, \mu)$, если

$$L(x^*, \mu) \leq L(x^*, \mu^*) \leq L(x, \mu^*), \quad x \in \Omega, \quad \mu \in \mathbb{R}_*^m. \quad (4.14)$$

Таблица 4.1

Номер решения	1	2	3	4	5	6	7
μ_1	0	0	-3	-3	0	0	-4
μ_2	0	-3	0	-3	0	-4	0
μ_3	0	0	0	0	1	-1	-1
x_1	3	3	0	0	2	4	0
x_2	3	0	3	0	2	0	4

Теорема 4.7. Если (x^*, μ^*) — седловая точка функции Лагранжа $L(x, \mu)$, то x^* — точка наименьшего значения функции $f_0(x)$ на множестве

$$\Omega = \{x \in \mathbb{R}^n: g_i(x) \leq 0, i = \overline{1, m}\}.$$

◀ Из определения функции Лагранжа вытекает, что для любых $x \in \Omega$ и $\mu \in \mathbb{R}_*^m$ выполняется неравенство $L(x, \mu) \leq f_0(x)$, так как $\mu_i g_i(x) \leq 0$, $i = \overline{1, m}$, $x \in \Omega$. При этом указанное неравенство превращается в равенство, если $\mu = 0$, т.е. $L(x, 0) = f_0(x)$. Таким образом, если (x^*, μ^*) — седловая точка функции Лагранжа, то для любой точки $x \in \Omega$

$$f_0(x^*) = L(x^*, 0) \leq L(x^*, \mu^*) \leq L(x, \mu^*) \leq f_0(x). \quad \blacktriangleright$$

Теорема 4.7 указывает метод решения задачи нелинейного программирования (4.12). Достаточно определить седловую точку функции Лагранжа, чтобы прийти к решению задачи нелинейного программирования. Следующая теорема устанавливает критерий для седловых точек функции Лагранжа.

Теорема 4.8. Точка (x^*, μ^*) , $x^* \in \Omega$, $\mu^* = (\mu_1, \dots, \mu_m) \in \mathbb{R}_*^m$, является седловой для функции Лагранжа $L(x, \mu)$ в том и только в том случае, если выполнены условия

$$L(x^*, \mu^*) = \min_{x \in \Omega} L(x, \mu^*); \quad (4.15)$$

$$\mu_i^* g_i(x^*) = 0, \quad i = \overline{1, m}. \quad (4.16)$$

◀ Если (x^*, μ^*) — седловая точка функции Лагранжа, причем $x^* \in \Omega$, $\mu^* \in \mathbb{R}_*^m$, то соотношение (4.15) верно в силу определения седловой точки. Кроме того, согласно этому определению $L(x^*, \mu^*) \geq L(x^*, 0)$, что равносильно неравенству

$$f_0(x^*) + \sum_{i=1}^m \mu_i^* g_i(x^*) \geq f_0(x^*).$$

Таким образом,

$$\sum_{i=1}^m \mu_i^* g_i(x^*) \geq 0. \quad (4.17)$$

Но из условия $x^* \in \Omega$ вытекает, что $g_i(x^*) \leq 0$, а условие $\mu^* \in \mathbb{R}_*^m$ означает, что $\mu_i^* \geq 0$, $i = \overline{1, m}$. Следовательно, каждое слагаемое суммы в неравенстве (4.17) неположительно. Поэтому и сумма, и каждое ее слагаемое равны нулю, т.е. верны соотношения (4.16).

Теперь предположим, что выполнены условия (4.15) и (4.16). Тогда $L(x^*, \mu^*) = f_0(x^*)$ и в то же время $L(x^*, \mu) \leq f_0(x^*)$ для любого $\mu \in \mathbb{R}_*^m$, откуда $L(x^*, \mu) \leq L(x^*, \mu^*)$. Это неравенство вместе с соотношением (4.15) приводит к неравенствам (4.14), т.е. к утверждению, что (x^*, μ^*) — седловая точка функции Лагранжа. ►

Теорему 4.7 можно использовать в широком круге задач нелинейного программирования. При этом на функции $f_0(x)$ и $g_i(x)$, $i = \overline{1, m}$, не накладываются никакие ограничения. Однако следует иметь в виду, что достаточное условие, которое устанавливается теоремой 4.7, лишь заменяет одну задачу минимизации другой, которая в некоторых случаях может быть проще исходной. Действительно, поиск седловой точки функции Лагранжа в соответствии с равенством (4.15) означает минимизацию функции Лагранжа по переменным x .

4.5. Двойственная функция

Рассмотрим задачу нелинейного программирования (4.12), не имеющую ограничений типа равенства, предполагая, что целевая функция $f_0(x)$, а также функции $g_i(x)$, $i = \overline{1, m}$, определены в \mathbb{R}^n . Как и выше (см. разд. 4.4), сформируем функцию Лагранжа $L(x, \mu)$ в виде (4.13), определенную при $x \in \mathbb{R}^n$ и $\mu \in \mathbb{R}_*^m$, где \mathbb{R}_*^m — неотрицательный ортант в \mathbb{R}^m .

На множестве \mathbb{R}_*^m определим функцию

$$w(\mu) = \inf_{x \in \Omega} L(x, \mu). \quad (4.18)$$

Функции $f_0(x)$ и $w(\mu)$ связаны соотношением

$$w(\mu) \leq f_0(x), \quad \mu \in \mathbb{R}_*^m, \quad x \in \Omega. \quad (4.19)$$

Действительно, функция Лагранжа $L(x, \mu)$ удовлетворяет неравенству $L(x, \mu) \leq f_0(x)$, $x \in \Omega$, $\mu \in \mathbb{R}_*^m$. Из этого неравенства вытекает, что

$$w(\mu) = \inf_{x \in \Omega} L(x, \mu) \leq \inf_{x \in \Omega} f_0(x) \leq f_0(x).$$

Теорема 4.9. Если для некоторых $x^* \in \Omega$ и $\mu^* \in \mathbb{R}_*^m$ верно равенство $f_0(x^*) = w(\mu^*)$, то функция $f_0(x)$ в точке x^* достигает наименьшего значения на множестве Ω , а функция $w(\mu)$ в точке μ^* — наибольшего значения на множестве \mathbb{R}_*^m .

◀ Из неравенства (4.19) для $x = x^*$ и равенства $f_0(x^*) = w(\mu^*)$ следует, что

$$w(\mu) \leq w(\mu^*), \quad \mu \in \mathbb{R}_*^m,$$

т.е. значение функции $w(\mu)$ в точке μ^* является наибольшим на множестве \mathbb{R}_*^m .

Аналогично, полагая в неравенстве (4.19) $\mu = \mu^*$ и учитывая, что $w(\mu^*) = f_0(x^*)$, находим

$$f_0(x) \geq f_0(x^*), \quad x \in \Omega,$$

т.е. в точке x^* функция $f(x)$ достигает наименьшего значения. ►

Любая точка локального максимума функции $w(\mu)$ на множестве \mathbb{R}_*^m является точкой ее наибольшего значения на этом множестве. Чтобы доказать это свойство функции $w(\mu)$, достаточно показать, что функция $-w(\mu)$ выпукла на выпуклом множестве \mathbb{R}_*^m . Выберем произвольные несовпадающие точки μ^1, μ^2 в \mathbb{R}_*^m и произвольное число $\alpha \in [0, 1]$. Пусть $\mu = \alpha\mu^1 + (1 - \alpha)\mu^2$. В соответствии с определением функции $w(\mu)$ для произвольного числа $\varepsilon > 0$ существует такая точка $\tilde{x} \in \Omega$, что $w(\mu) \geq L(\tilde{x}, \mu) - \varepsilon$. Из определения функции Лагранжа (она линейна по множителям Лагранжа) следует равенство

$$L(x, \mu) = L(x, \alpha\mu^1 + (1 - \alpha)\mu^2) = \alpha L(x, \mu^1) + (1 - \alpha)L(x, \mu^2).$$

С учетом этого равенства имеем

$$\begin{aligned} w(\mu) &\geq \alpha L(\tilde{x}, \mu^1) + (1 - \alpha)L(\tilde{x}, \mu^2) - \varepsilon \geq \\ &\geq \alpha w(\mu^1) + (1 - \alpha)w(\mu^2) - \varepsilon. \end{aligned}$$

Ввиду того, что число $\varepsilon > 0$ выбрано произвольно, заключаем, что

$$w(\mu) \geq \alpha w(\mu^1) + (1 - \alpha)w(\mu^2),$$

а это равносильно выпуклости функции $-w(\mu)$.

Теорема 4.10. Функция Лагранжа (4.13) имеет седловую точку (x^*, μ^*) тогда и только тогда, когда

$$w(\mu^*) = f_0(x^*). \quad (4.20)$$

◀ Пусть (x^*, μ^*) — седловая точка функции Лагранжа. Тогда согласно определению седловой точки $L(x^*, \mu^*) \leq L(x, \mu^*)$, $x \in \Omega$, т.е. при заданном $\mu = \mu^*$ функция Лагранжа достигает на множестве Ω наименьшего значения в точке x^* . Следовательно, $w(\mu^*) = L(x^*, \mu^*)$. Далее, снова используя определение седловой точки, заключаем, что $L(x^*, \mu^*) \geq L(x^*, 0) = f_0(x^*)$, откуда $f_0(x^*) \leq w(\mu^*)$. Сопоставляя последнее неравенство с неравенством (4.19), получаем равенство $f_0(x^*) = w(\mu^*)$.

Пусть для $x^* \in \Omega$ и $\mu^* \in \mathbb{R}_*^m$ верно равенство (4.20). Тогда согласно теореме 4.9 точка x^* является точкой наименьшего значения функции $f_0(x)$ на Ω , а точка μ^* — точкой наибольшего значения функции $w(\mu)$ на \mathbb{R}_*^m . При этом

$$w(\mu^*) = \inf_{x \in \Omega} L(x, \mu^*) \leq L(x^*, \mu^*) \leq f_0(x^*) = w(\mu^*).$$

Следовательно,

$$L(x^*, \mu^*) = \inf_{x \in \Omega} L(x, \mu^*).$$

Кроме того, $L(x^*, \mu^*) = f_0(x^*)$, откуда, используя равенство (4.13), определяющее функцию Лагранжа, получаем

$$\sum_{i=1}^m \mu_i^* g_i(x^*) = 0,$$

где $\mu^* = (\mu_1^*, \dots, \mu_m^*)$. Так как все слагаемые в левой части последнего равенства неотрицательны, все они равны нулю, т.е. выполняется условие (4.16). Согласно теореме 4.8 точка (x, μ) является седловой точкой функции Лагранжа. ►

Функцию $w(\mu)$, определенную на \mathbb{R}_*^m соотношением (4.18), называют *двойственной функцией* по отношению к целевой функции $f_0(x)$. Задачу поиска точки $\mu^* \in \mathbb{R}_*^m$ наибольшего значения этой функции называют *двойственной задачей* по отношению к задаче минимизации функции $f_0(x)$ на множестве Ω , а теорему 4.10 — *теоремой двойственности*. Установленные свойства функции $w(\mu)$ дают возможность вместо задачи поиска точки наименьшего значения целевой функции решать двойственную задачу, которая может быть проще исходной. Найденное решение двойственной задачи далее можно использовать в качестве набора множителей Лагранжа и затем решать задачу минимизации функции Лагранжа.

Пример 4.3. В задаче

$$\begin{cases} x_1^2 + x_2^2 \rightarrow \min; \\ 2x_1 + x_2 + 4 \leq 0 \end{cases}$$

найдем функцию $w(\mu)$, двойственную по отношению к целевой функции $f_0(x) = x_1^2 + x_2^2$.

В данном случае допустимое множество Ω представляет собой полуплоскость $2x_1 + x_2 + 4 \leq 0$. Чтобы найти двойственную функцию, нужно определить наименьшее значение функции Лагранжа

$$L(x_1, x_2, \mu) = x_1^2 + x_2^2 + \mu(2x_1 + x_2 + 4)$$

при ограничении $2x_1 + x_2 + 4 \leq 0$. Такую задачу можно решить с помощью теоремы Куна — Таккера:

- для функции $L(x_1, x_2, \mu)$, считая μ фиксированным, построить функцию Лагранжа (т.е. функцию Лагранжа для функции Лагранжа), для чего необходимо ввести параметры λ_0 и $\tilde{\mu}$;
- на основании теоремы Куна — Таккера записать систему четырех уравнений относительно неизвестных x_1 , x_2 , λ_0 и $\tilde{\mu}$ и решить ее;
- среди найденных точек путем сравнения выбрать ту, в которой функция $L(x_1, x_2, \mu)$ принимает наименьшее значение.

Так как функция $L(x_1, x_2, \mu)$ при фиксированном μ является квадратичной с положительно определенной матрицей, а значит, сильно выпуклой функцией, можно утверждать, что она достигает на полуплоскости $2x_1 + x_2 + 4 \leq 0$ (это выпуклое множество) наименьшего значения. Следовательно, выбранная точка будет точкой наименьшего значения функции $L(x_1, x_2, \mu)$. Значение функции $L(x_1, x_2, \mu)$ в выбранной точке, зависящее от μ , и есть значение функции $w(\mu)$ при фиксированном μ .

Отступим, однако, от этой стандартной последовательности действий и учтем специфику рассматриваемой задачи. В выражении для функции Лагранжа $L(x_1, x_2, \mu)$ выделим полные квадраты по переменным x_1 и x_2 и запишем эту функцию следующим образом:

$$L(x_1, x_2, \mu) = (x_1 + \mu)^2 + \left(x_2 + \frac{\mu}{2}\right)^2 + 4\mu - \frac{5}{4}\mu^2.$$

Из этого представления вытекает, что значением функции Лагранжа является квадрат расстояния от точки $(-\mu, -\mu/2)$ до точки

(x_1, x_2) плюс величина $4\mu - \frac{5}{4}\mu^2$, не зависящая от точки (x_1, x_2) . Следовательно, наименьшее значение функции Лагранжа (т.е. значение двойственной функции $w(\mu)$) равно квадрату расстояния от точки $(-\mu, -\mu/2)$ до полуплоскости Ω плюс $4\mu - \frac{5}{4}\mu^2$. Если точка $(-\mu, -\mu/2)$ попадает в Ω , то расстояние от нее до Ω равно нулю. Если же $(-\mu, -\mu/2) \notin \Omega$, то это расстояние есть расстояние от точки $(-\mu, -\mu/2)$ до границы множества Ω — прямой $2x_1 + x_2 + 4 = 0$. Условие $(-\mu, -\mu/2) \in \Omega$ означает, что

$$2(-\mu) + \left(-\frac{\mu}{2}\right) + 4 \leq 0,$$

или $\mu \geq 8/5$. Расстояние $\rho(\mu)$ от точки $(-\mu, -\mu/2)$ до прямой $2x_1 + x_2 + 4 = 0$ равно

$$\rho(\mu) = \frac{|-2\mu - \mu/2 + 4|}{\sqrt{5}}.$$

Следовательно,

$$\rho^2(\mu) = \frac{(4 - 5\mu/2)^2}{5} = \frac{16}{5} - 4\mu + \frac{5}{4}\mu^2.$$

В результате получаем

$$w(\mu) = \begin{cases} \frac{16}{5}, & 0 \leq \mu < \frac{8}{5}; \\ 4\mu - \frac{5}{4}\mu^2, & \mu \geq \frac{8}{5}. \end{cases}$$

Вопросы для самопроверки

1. Что называют допустимым направлением в задачах нелинейного программирования?
2. Что такое конус допустимых направлений?
3. При каких условиях можно утверждать, что целевая функция на заданном множестве достигает наименьшего значения?
4. Какую роль в задачах минимизации играет функция Лагранжа? Сформулируйте обобщенное правило множителей Лагранжа.

5. Какие ограничения в задаче нелинейного программирования называют активными, а какие — неактивными?

6. Какую функцию в задачах минимизации называют двойственной и каковы ее свойства? Что такое двойственная задача?

7. Минимизируйте функцию $f(x_1, x_2) = x_1 - x_2$ при ограничении $x_1^2 + x_2^2 = 1$. Найдите стационарные точки и точки минимума. Проанализируйте поведение функции Лагранжа в окрестностях найденных точек. Классифицируйте найденные точки (точки минимума, максимума, седловые точки для функции Лагранжа по переменным x, λ).

ЧИСЛЕННЫЕ МЕТОДЫ НЕЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

Известно достаточно много численных методов решения общей задачи нелинейного программирования. Во многих из них использованы идеи, реализованные в численных методах безусловной минимизации. Главное отличие задач нелинейного программирования состоит в том, что при решении этих задач необходимо учитывать ограничения, задающие допустимое множество, в пределах которого определяется минимум целевой функции. В этой главе рассмотрены наиболее часто применяемые на практике численные методы решения задач нелинейного программирования.

5.1. Метод условного градиента

Для численного решения общей задачи нелинейного программирования

$$f_0(x) \rightarrow \min, \quad x \in \Omega \subset \mathbb{R}^n, \quad (5.1)$$

где $f_0(x)$ — целевая функция, определенная на допустимом множестве Ω , можно применить модификацию одного из методов спуска. Однако непосредственное использование методов спуска может привести к тому, что точка x^k , найденная на некоторой k -й итерации, например, с помощью рекуррентных соотношений вида (3.4) или (3.10), окажется вне пределов допустимого множества. Избежать этого в случае замкнутого ограниченного выпуклого множества Ω и непрерывно дифференцируемой на нем выпуклой целевой функции можно следующим образом.

На k -й итерации в окрестности исходной точки $x^{k-1} \in \Omega$ запишем приращение целевой функции в соответствии с условием ее дифференцируемости:

$$\Delta f_k = f_0(x) - f_0(x^{k-1}) = (\text{grad } f_0(x^{k-1}), x - x^{k-1}) + o(|x - x^{k-1}|).$$

Отбрасывая в этом представлении остаток, получим главную линейную часть приращения целевой функции, которая дает ее линейное приближение:

$$f_k(x) = (\text{grad } f_0(x^{k-1}), x - x^{k-1}). \quad (5.2)$$

Минимизируя это приближение на допустимом множестве Ω , получим очередное приближение к точке минимума целевой функции:

$$f_k(\tilde{x}^k) = \min_{x \in \Omega} (\text{grad } f_0(x^{k-1}), x - x^{k-1}). \quad (5.3)$$

В силу замкнутости и ограниченности множества Ω и непрерывности линейной функции $f_k(x)$ точка \tilde{x}^k ее минимума на Ω всегда существует (возможно, не единственная, тогда можно выбрать одну из них). Поскольку линейная функция не имеет локальных экстремумов, точка \tilde{x}^k лежит на границе $\partial\Omega$ допустимого множества.

Для некоторых множеств, имеющих достаточно простую структуру, решение задачи (5.3) удастся представить в явном виде. Например, в n -мерном замкнутом шаре радиуса R с центром в точке $x^0 \in \mathbb{R}^n$, т.е. на множестве $\Omega = \{x \in \mathbb{R}^n: |x - x^0| \leq R\}$, точкой

минимума линейной функции $\psi(x) = (b, x)$ является точка $x^* = x^0 - R \frac{b}{|b|}$ (на рис. 5.1 вектор b указывает направление возрастания функции). В этом нетрудно убедиться, используя неравенство Коши — Буняковского. В самом деле,

$$\begin{aligned} (b, x) - (b, x^*) &= (b, x - x^*) = \\ &= (b, x - x^0) + \left(b, R \frac{b}{|b|}\right) = (b, x - x^0) + R|b|. \end{aligned}$$

Из неравенства Коши — Буняковского вытекает, что для любой точки $x \in \Omega$

$$(b, x - x^0) \geq -|b| |x - x^0| \geq -R|b|,$$

поэтому

$$(b, x) - (b, x^*) \geq -R|b| + R|b| = 0.$$

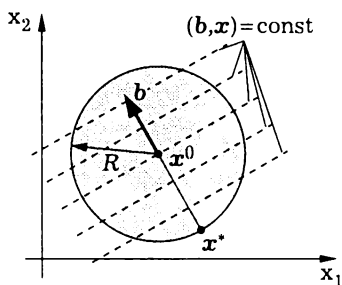


Рис. 5.1

Для задачи (5.3), в которой допустимым множеством является n -мерный шар $\Omega = \{x \in \mathbb{R}^n: |x - x_0| \leq R\}$, получаем

$$\tilde{x}^k = x^0 - R \frac{\text{grad } f_0(x^{k-1})}{|\text{grad } f_0(x^{k-1})|}.$$

Если допустимое множество является n -мерным параллелепипедом $\Omega = \{(x_1, \dots, x_n) \in \mathbb{R}^n: x_j \in [a_j, b_j], j = \overline{1, n}\}$, то решением задачи (5.3) является точка \tilde{x}^k с координатами

$$\tilde{x}_j^{(k)} = \begin{cases} a_j, & \frac{\partial f_0(x^{k-1})}{\partial x_j} \geq 0; \\ b_j, & \frac{\partial f_0(x^{k-1})}{\partial x_j} < 0, \end{cases} \quad j = \overline{1, n}. \quad (5.4)$$

Если $\frac{\partial f_0(x^{k-1})}{\partial x_j} = 0$ для некоторого номера j , то в качестве координаты

$\tilde{x}_j^{(k)}$ точки минимума можно выбрать любое число из отрезка $[a_j, b_j]$.

Если допустимое множество определено системой линейных ограничений типа равенства и неравенства, т.е.

$$\Omega = \{x \in \mathbb{R}^n: (a_i, x) \leq b_i, i = \overline{1, m}, (c_j, x) = d_j, j = \overline{1, k}\},$$

то задача (5.3) будет представлять собой задачу линейного программирования. Для ее решения можно использовать симплекс-метод.

Предположим, что задача (5.3) решена и точка \tilde{x}^k найдена. Если $\tilde{x}^k = x^{k-1}$, то $f_k(\tilde{x}^k) = 0$. Следовательно, $f_k(x) \geq f_k(x^*) = 0, x \in \Omega$. Учитывая вид функции $f_k(x)$, заключаем, что

$$(\text{grad } f_0(x^{k-1}), x - x^{k-1}) \geq 0, \quad x \in \Omega.$$

Записанное условие означает, что точка x^{k-1} является точкой минимума целевой функции на множестве Ω . В этом случае процесс поиска точки минимума целевой функции можно прекратить.

Пусть $\tilde{x}^k \neq x^{k-1}$. Тогда в точке \tilde{x}^k функция $f_k(x)$ имеет значение, меньшее, чем в точке x^{k-1} , т.е.

$$f_k(\tilde{x}^k) = (\text{grad } f_0(x^{k-1}), \tilde{x}^k - x^{k-1}) < 0.$$

Следовательно, вектор $p^k = \tilde{x}^k - x^{k-1}$ определяет направление спуска. Положив

$$x^k = x^{k-1} + \kappa_k(\tilde{x}^k - x^{k-1}) = (1 - \kappa_k)x^{k-1} + \kappa_k\tilde{x}^k, \quad (5.5)$$

для любого значения $\kappa_k \in [0, 1]$ получим точку x^k , принадлежащую допустимому множеству Ω , поскольку множество Ω выпукло.

Таким образом, вспомогательное приближение \tilde{x}^k на k -й итерации позволило определить вектор p^k , задающий, с одной стороны, допустимое направление, а с другой — направление спуска. Однако это направление в общем случае не совпадает с направлением антиградиента функции $f_0(x)$ в точке x^{k-1} (рис. 5.2). Поэтому нахождение точки x^k в соответствии с соотношением (5.5) путем выбора значения κ_k отличается от итерации метода градиентного спуска. Это отличие послужило основанием для того, чтобы описанный метод минимизации функции на заданном множестве назвать *методом условного градиента*.

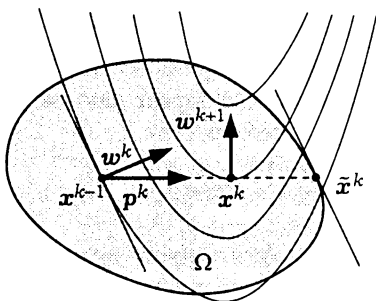


Рис. 5.2

Допустимое направление, определяемое вектором p^k , назовем *возможным направлением спуска* для функции $f_0(x)$ из точки $x^{k-1} \in \Omega$ на множестве $\Omega \subset \mathbb{R}^n$, если это направление является направлением спуска, т.е. существует такое число $\kappa_k > 0$, что $x^{k-1} + \kappa p^k \in \Omega$ при $\kappa \in (0, \kappa_k]$ и выполнено неравенство $f_0(x^{k-1} + \kappa p^k) < f_0(x^{k-1})$. Спуск в возможном направлении из точки x^{k-1} позволяет на k -й итерации продолжить построение релаксационной последовательности $\{x^k\}$.

Рассмотрим вопрос о выборе значения $\kappa_k \in (0, 1]$ в соотношении (5.5). Один из возможных вариантов — это минимизация на отрезке $[0, 1]$ функции

$$\varphi_k(\kappa) = f_0(x^{k-1} + \kappa p^k). \quad (5.6)$$

Для квадратичной функции вида $f_0(x) = \frac{1}{2}(Qx, x) + (b, x)$ с положительно определенной матрицей Q точка $x_k^* \in (0, 1]$ минимума функции $\varphi_k(\kappa)$ единственна и может быть представлена в аналитическом виде. Действительно, подставляя $x = x^{k-1} + \kappa p^k$ в квадратичную функцию и учитывая, что матрица Q симметрическая, получаем

$$\varphi_k(\kappa) = \frac{\kappa^2}{2}(Qp^k, p^k) + \kappa(Qx^{k-1} + b, p^k) + f_0(x^{k-1}),$$

или, учитывая равенство $Qx^{k-1} + b = \text{grad } f_0(x^{k-1}) = -w^k$,

$$\varphi_k(\kappa) = \frac{\kappa^2}{2}(Qp^k, p^k) - \kappa(w^k, p^k) + f_0(x^{k-1}).$$

Случай $(Qp^k, p^k) = 0$ для положительно определенной матрицы соответствует равенству $p^k = \tilde{x}^k - x^{k-1} = 0$ и уже рассмотрен выше. При $(Qp^k, p^k) > 0$ имеем относительно κ квадратный трехчлен, достигающий минимума при значении

$$\kappa_k^* = \frac{(w^k, p^k)}{(Qp^k, p^k)}. \quad (5.7)$$

Поскольку вектор p^k определяет направление спуска и в силу этого

$$(w^k, p^k) = -(\text{grad } f_0(x^{k-1}), \tilde{x}^k - x^{k-1}) \geq 0,$$

имеем $\kappa_k \geq 0$. При $\kappa_k = 0$, как отмечено выше, точка x^{k-1} оказывается точкой минимума целевой функции на допустимом множестве Ω . В иной ситуации $\kappa_k > 0$. Может быть так, что $\kappa_k^* > 1$. Тогда функция $\varphi_k(\kappa)$ достигает своего минимума на отрезке $[0, 1]$ при $\kappa = 1$. Значит, следует принять $\kappa_k = 1$ и $x^k = \tilde{x}^k$. В противном случае точка x^k соответствует значению $\kappa_k = \kappa_k^*$, вычисляемому по формуле (5.7).

Для выпуклой целевой функции $f_0(x)$ выбор значения κ_k из условия минимума функции $\varphi_k(\kappa)$ потребует применения методов одномерной минимизации. Отметим, что для выпуклой целевой функции $f_0(x)$ и функция $\varphi_k(\kappa)$ выпукла, а значит, достигает минимума в любой своей стационарной точке κ^* . Если $f_0(x)$ — строго выпуклая функция, то и функция $\varphi_k(\kappa)$ строго выпукла на отрезке $[0, 1]$, а потому имеет на этом отрезке единственную точку минимума.

Минимизация функции $\varphi_k(x)$ — трудоемкая задача. Но можно обойти ее решение и выбирать значение x_k для определения точки x^k по формуле (5.5) из других соображений (например, применяя процедуру дробления шага). Единственное требование при этом заключается в выполнении условия $f_0(x^k) \leq f_0(x^{k-1})$, обеспечивающего релаксационность последовательности $\{x^k\}$.

Условием прекращения итераций в методе условного градиента может быть выполнение одного или всех неравенств (3.2) и (3.3).

При некоторых ограничениях метод условного градиента работоспособен и в случае, когда целевая функция не является выпуклой.

Пример 5.1. Решим задачу квадратичного программирования

$$f_0(x_1, x_2) = x_1^2 - 4x_1 + x_2^2 - 2x_2 \rightarrow \min, \\ 0 \leq x_1 \leq 1, \quad 0 \leq x_2 \leq 2.$$

В данном случае $f_0(x) = \frac{1}{2}(Qx, x) + (b, x)$, где

$$b = \begin{pmatrix} -4 \\ -2 \end{pmatrix}, \quad Q = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix},$$

а допустимое множество Ω можно записать в виде

$$\Omega = \left\{ (x_1, x_2) \in \mathbb{R}^2: x_1 \in [0, 1], x_2 \in [0, 2] \right\}.$$

В качестве начальной выберем точку $x^0 = (0, 0)$, а условие прекращения итераций примем в виде $|x^k - x^{k-1}| < \varepsilon$, положив $\varepsilon = 0,1$. Рассмотрим подробнее первую итерацию метода условного градиента.

1. Вычислим в начальной точке $x^0 = (0, 0)$ градиент целевой

функции $\text{grad } f_0(x^0) = (-4 \ -2)^T$ и найдем в соответствии с формулой (5.2) линейное приближение целевой функции в указанной точке:

$$f_1(x) = (\text{grad } f_0(x^0), x - x^0) = -4x_1 - 2x_2.$$

2. Вспомогательное приближение \tilde{x}^1 найдем, минимизируя функцию $f_1(x)$ на заданном множестве Ω (рис. 5.3). Для этого используем соотношения (5.4), в которых $a_1 = a_2 = 0$, $b_1 = 1$ и $b_2 = 2$.

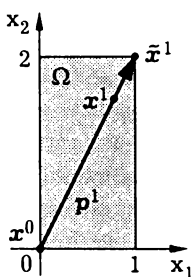


Рис. 5.3

Так как частные производные $\frac{\partial f_0(x^0)}{\partial x_1} = -4$ и $\frac{\partial f_0(x^0)}{\partial x_2} = -2$ целевой функции в точке x^0 отрицательны, то в соответствии с формулой (5.4) имеем $\tilde{x}^1 = (1, 2)$.

3. На первой итерации направление спуска определяется вектором $p^1 = \tilde{x}^1 - x^0 = (1 \ 2)^T$. Учитывая выражение для антиградиента $w^1 = -\text{grad } f_0(x^0) = (4 \ 2)^T$ целевой функции в точке x^0 и используя равенство (5.7), находим точку x_1^* минимума функции $\varphi_1(x)$:

$$x_1^* = \frac{(w^1, p^1)}{(Qp^1, p^1)} = \frac{(4 \ 2)(1 \ 2)^T}{(1 \ 2) \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix}} = \frac{8}{10} = 0,8.$$

Поскольку $x_1^* \leq 1$, принимаем $x_1 = x_1^* = 0,8$ и по формуле (5.5) находим точку $x^1 = x^0 + x_1 p^1 = (0,8 \ 1,6)^T$. Ясно, что $x^1 \in \Omega$ (см. рис. 5.3).

4. Проверяя условие прекращения итераций, убеждаемся, что $|x^1 - x^0| = \sqrt{(0,8)^2 + (1,6)^2} > 0,1 = \varepsilon$, т.е. требуемая точность еще не достигнута и поиск точки минимума следует продолжить, используя в качестве исходной найденную на первой итерации точку $x^1 = (0,8, 1,6)$.

Результаты расчетов на первой и последующих итерациях приведены в табл. 5.1, а графическая иллюстрация решения задачи методом условного градиента представлена на рис. 5.4. Описанный алгоритм за восемь итераций с заданной точностью приводит к точке $x^* = (1, 1)$, являющейся точным решением задачи. В точке $x^8 = (0,957, 0,953)$ значение $f_0(x^8) \approx -3,910$ отличается от минимального значения $f_0(x^*) = -4$ менее чем на 0,1.

Обратим внимание на характерное зигзагообразное движение текущей точки в направлении искомой точки минимума x^* . Такое движение вызвано тем обстоятельством, что в данном случае в результате решения задачи минимизации линейной функции $f_k(x)$ на выпуклом множестве Ω вспомогательными приближениями поочередно являются крайние точки $(1, 2)$ и $(1, 0)$ этого множества. #

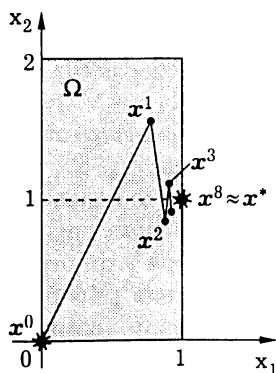


Рис. 5.4

Таблица 5.1

k	x^k	$ x^k - x^{k-1} $	k	x^k	$ x^k - x^{k-1} $
1	(0,800, 1,600)	1,790	5	(0,939, 1,069)	0,152
2	(0,892, 0,861)	0,745	6	(0,947, 0,940)	0,129
3	(0,915, 1,103)	0,243	7	(0,952, 1,053)	0,113
4	(0,929, 0,917)	0,187	8	(0,957, 0,953)	0,100

Поиск вспомогательного приближения $\tilde{x}^k \in \Omega$ на k -й итерации метода условного градиента можно вести не только на основе линеаризации целевой функции $f_0(x)$ в окрестности точки $x^{k-1} \in \Omega$. Если $f_0(x)$ — выпуклая функция, дважды дифференцируемая на выпуклом ограниченном замкнутом множестве Ω , то ее приращение в окрестности точки x^{k-1} приближенно можно представить квадратичной функцией

$$\tilde{f}_k(x) = (\text{grad } f_0(x^{k-1}), x - x^{k-1}) + (H(x^{k-1})(x - x^{k-1}), x - x^{k-1}),$$

где $H(x^{k-1})$ — матрица Гессе функции $f_0(x)$, вычисленная в точке x^{k-1} . Такое представление приводит к необходимости вместо решения задачи (5.3) минимизации линейной функции на допустимом множестве Ω решать на том же множестве задачу минимизации квадратичной функции. Это существенно усложняет поиск точки \tilde{x}^k , но благодаря более точной аппроксимации целевой функции в окрестности точки x^{k-1} может уменьшить общее число итераций.

5.2. Использование приведенного градиента

Если в общей задаче нелинейного программирования допустимое множество $\Omega \subset \mathbb{R}^n$ задано с помощью линейных ограничений, то для решения этой задачи можно использовать подход, аналогичный по координатному спуску, применяемому в задачах безусловной минимизации. При этом ограничения удобно привести к форме, содержащей ограничения типа равенства, которые можно записать в виде $Ax = b$ с известными матрицей A и вектором b , и условие неотрицательности параметров оптимизации. Последнее условие можно записать в виде $x \in \mathbb{R}_*^n$, где \mathbb{R}_*^n — *неотрицательный ортант*, содержащий все арифметические векторы с неотрицательными координатами.

Записав допустимое множество в виде

$$\Omega = \{x \in \mathbb{R}_+^n: Ax = b\}, \quad (5.8)$$

можно, не ограничивая общности, считать, что ранг матрицы A равен числу m ее строк и меньше числа n ее столбцов, т.е. $\text{Rg } A = m < n$. В самом деле, если ранг матрицы равен количеству столбцов, то система $Ax = b$ несовместна или имеет единственное решение. В этом случае задача нелинейного программирования не представляет интереса. Если система $Ax = b$ совместна, а $\text{Rg } A = r < m$, то в матрице можно выделить r линейно независимых строк. Уравнения системы, соответствующие остальным $m - r$ строкам матрицы A , будут следствиями уравнений, соответствующих выделенным r строкам матрицы A . Такие уравнения можно из системы удалить, не меняя множество решений системы и не меняя, по существу, задачу линейного программирования.

Итак, далее считаем, что $\text{Rg } A = m < n$. Выберем в матрице A базисный минор B . Для удобства будем считать, что этот минор расположен в первых m столбцах матрицы A (этого можно добиться, изменив при необходимости порядок переменных в задаче нелинейного программирования). Матрицу A можно записать как блочную матрицу $(B | N)$, где B — квадратная матрица, составленная из первых m столбцов матрицы A , а матрица N объединяет оставшиеся $n - m$ столбцов. Матрица B , соответствующая базисному минору, является невырожденной. Выбор базисного минора в системе линейных алгебраических уравнений (СЛАУ) $Ax = b$ приводит к разделению неизвестных x_i , $i = \overline{1, n}$, на **базисные неизвестные** x_1, \dots, x_m и **свободные неизвестные** x_{m+1}, \dots, x_n . Первую группу неизвестных сгруппируем в вектор-столбец x^B , а вторую — в вектор-столбец x^N . Введем также множества индексов $\mathcal{J}^B = \{1, 2, \dots, m\}$, $\mathcal{J}^N = \{m+1, \dots, n\}$ и $\mathcal{J} = \mathcal{J}^B \cup \mathcal{J}^N$.

С учетом введенных обозначений СЛАУ $Ax = b$ можно записать в виде матричного уравнения $Bx^B + Nx^N = b$. Поскольку матрица B невырождена, то это матричное уравнение можно преобразовать к виду

$$x^B = \hat{b} - \hat{N}x^N, \quad (5.9)$$

где $\hat{b} = B^{-1}b$, $\hat{N} = B^{-1}N$. Параметры оптимизации, составляющие векторы x^B и x^N , называют обычно **базисными** и **свободными переменными** в задаче нелинейного программирования.

Матрицу Якоби целевой функции в некоторой точке $x^0 \in \Omega$ представим в виде блочной матрицы-строки $(g^B \ g^N)$, где g^B и g^N — матрицы-строки длины m и $n - m$. Элементами этих матриц-строк являются частные производные $\frac{\partial f_0}{\partial x_j}$, $j = \overline{1, n}$, по базисным и свободным переменным соответственно. Тогда, используя равенство (5.9), для полного дифференциала целевой функции получаем

$$\begin{aligned} df_0(x^0) &= (g^B \ g^N) \begin{pmatrix} x^B & x^N \end{pmatrix}^T = g^B dx^B + g^N dx^N = \\ &= -g^B \hat{N} dx^N + g^N dx^N = (g^N - g^B \hat{N}) dx^N = u^T dx^N, \end{aligned} \quad (5.10)$$

где $u = (g^N - g^B \hat{N})^T \in \mathbb{R}^{n-m}$ — вектор, называемый *приведенным градиентом* целевой функции $f_0(x)$ относительно свободных переменных в точке $x^0 \in \Omega$.

В задаче оптимизации с линейными ограничениями возможны разные стратегии поиска точки минимума с использованием приведенного градиента. Простейшая из них — применение покоординатного спуска в пространстве свободных переменных. Последовательный перебор свободных переменных проводят по тем же правилам, что и при решении задачи линейного программирования симплекс-методом, а метод, основанный на этой стратегии, называют *выпуклым симплекс-методом*.

Можно также на каждой итерации процесса минимизации изменять все свободные переменные. Метод, основанный на подобной стратегии, называют *методом приведенного градиента*.

Из равенства (5.10) следует, что при изменении свободных переменных и соответствующем изменении базисных переменных значение целевой функции быстрее всего уменьшается в направлении, противоположном приведенному градиенту. Но при произвольном изменении свободных переменных может быть нарушено условие их неотрицательности. Чтобы этого избежать, приведенный градиент можно спроектировать на неотрицательный ортант \mathbb{R}_*^{n-m} — допустимое множество для этих переменных. Поскольку \mathbb{R}_*^{n-m} — достаточно простое по структуре множество, проектирование на него выполнить несложно. Это позволяет сформировать возможное направление спуска, т.е. найти координаты p_j , $j \in \mathcal{J}$, вектора $p \in \mathbb{R}^n$, определяющего это направление. При $u_j \leq 0$, $j \in \mathcal{J}^N$, увеличение свободного переменного x_j в окрестности рассматриваемой точки $x^0 \in \Omega$ приведет

к уменьшению значения целевой функции. Поэтому для этого индекса принимаем $p_j = -u_j$. Если же $u_j > 0$, $j \in \mathcal{J}^N$, то увеличение свободного переменного x_j в окрестности рассматриваемой точки может привести к ее возрастанию. В этом случае положим $p_j = -u_j x_j^{(0)}$. Таким образом, для координат вектора $\mathbf{p} \in \mathbb{R}^n$, соответствующих свободным переменным и составляющих вектор $\mathbf{p}_N \in \mathbb{R}^{n-m}$, имеем

$$p_j = \begin{cases} -u_j, & u_j \leq 0; \\ -u_j x_j^{(0)}, & u_j > 0, \end{cases} \quad j \in \mathcal{J}^N. \quad (5.11)$$

Чтобы при движении точки

$$\mathbf{x} = \mathbf{x}^0 + \kappa \mathbf{p}, \quad \kappa > 0, \quad (5.12)$$

в направлении, которое задано вектором $\mathbf{p} \in \mathbb{R}^n$, не были нарушены ограничения $A\mathbf{x} = \mathbf{b}$, определяющие допустимое множество Ω вида (5.8), необходимо, чтобы выполнялось условие $A\mathbf{x} = A\mathbf{x}^0 + \kappa(B|N)\mathbf{p} = \mathbf{b}$, или $B\mathbf{p}_B = -N\mathbf{p}_N$, поскольку $A\mathbf{x}^0 = \mathbf{b}$. Отсюда находим вектор

$$\mathbf{p}_B = -B^{-1}N\mathbf{p}_N = -\hat{N}\mathbf{p}_N \quad (5.13)$$

с координатами p_j , $j \in \mathcal{J}^B$, соответствующими базисным переменным. Значение κ в равенстве (5.12) выбираем, минимизируя функцию $\varphi(\kappa) = f_0(\mathbf{x}^0 + \kappa \mathbf{p})$, $\kappa > 0$, но при выполнении условия $\mathbf{x} \in \mathbb{R}_*^n$, приводящего с учетом (5.12) для всех $j \in \mathcal{J}$ к соотношениям $0 \leq x_j = x_j^{(0)} + \kappa p_j$, где x_j и $x_j^{(0)}$ — координаты точек \mathbf{x} и \mathbf{x}^0 соответственно. Из этих соотношений в предположении, что множество $\mathcal{I}^* = \{j \in \mathcal{J}: p_j < 0\}$ не пусто, получаем ограничение на максимально возможное значение κ в виде

$$\kappa \leq \bar{\kappa} = \min_{j \in \mathcal{I}^*} \left(-\frac{x_j^{(0)}}{p_j} \right). \quad (5.14)$$

Если $\mathcal{I}^* = \emptyset$, то κ может иметь любое неотрицательное значение, т.е. $\kappa \in (0, +\infty)$.

Покажем, что равенство $\mathbf{p}_N = \mathbf{0}$ равносильно выполнению в рассматриваемой точке \mathbf{x}^0 необходимых условий локального экстремума на множестве Ω , установленных теоремой 4.6.

Для рассматриваемой целевой функции $f_0(x)$ функция Лагранжа имеет вид

$$L(x, \lambda, \mu) = f_0(x) + (\lambda, Ax - b) - (\mu, x),$$

где $\lambda \in \mathbb{R}^m$ и $\mu \in \mathbb{R}^n$ — векторы множителей Лагранжа, соответствующие ограничениям $Ax - b = 0$ типа равенства и $-x \leq 0$ типа неравенства (последнее неравенство означает, что $-x_j \leq 0$, $j = \overline{1, n}$). Необходимые условия локального экстремума на множестве Ω , установленные теоремой 4.6, можно записать следующим образом:

$$\begin{cases} \text{grad } L(x^*, \lambda, \mu) = \text{grad } f_0(x^*) + A^T \lambda - \mu = 0; \\ \mu_j x_j^* = 0, \quad j = \overline{1, n}. \end{cases} \quad (5.15)$$

При $p_N = 0$ эти условия будут выполнены, если положить

$$\lambda^T = -g^B B^{-1}, \quad \mu^B = 0, \quad \mu^N = u,$$

где μ^B и μ^N — вектор-столбцы, полученные группированием базисных и свободных координат вектора μ . В самом деле, в соответствии с соотношениями (5.11) из равенства $p_N = 0$ следует, что $u_j \geq 0$ для всех $j \in \mathcal{J}^N$ и, значит, $\mu_j \geq 0$, $j = \overline{1, n}$. Так как $\mu^B = 0$, то $\mu_j x_j^{(0)} = 0$, $j \in \mathcal{J}^B$. Из соотношений (5.11) и равенства $p_N = 0$ вытекает, что $u_j x_j^{(0)} = 0$, $j \in \mathcal{J}^N$, откуда в силу равенства $\mu^N = u$ заключаем, что $\mu_j x_j^{(0)} = 0$, $j \in \mathcal{J}^N$. Таким образом, вторая система равенств в системе (5.15) выполнена. Из выражений для λ^T и μ^T находим

$$-\lambda^T A + \mu^T = g^B B^{-1} (B^T N) + (0^T \ u^T) = (g^B \ g^B B^{-1} N + u^T).$$

Так как $u^T = g^N - g^B \hat{N} = g^N - g^B B^{-1} N$, то $-\lambda^T A + \mu^T = (g^B \ g^N)$. Но $\text{grad } f(x^0) = (g^B \ g^N)^T$. Следовательно,

$$\text{grad } f(x^0) + A^T \lambda - \mu = (-\lambda^T A + \mu^T)^T + A^T \lambda - \mu = 0,$$

т.е. выполнено и первое равенство (5.15).

Проведенное рассуждение позволяет рассматривать равенство $p_N = 0$ в процессе поиска как условие достижения точки минимума на множестве Ω . Учитывая это, можно построить алгоритм поиска точки минимума целевой функции на множестве Ω с использованием приведенного градиента.

На предварительном этапе устанавливаем начальный номер итерации $k = 1$ и выбираем начальную точку $x^0 \in \Omega$. Анализируя линейные ограничения типа равенства, выделяем базисные и свободные переменные в задаче, формируем множества \mathcal{J}_k^B и \mathcal{J}_k^N индексов базисных и свободных переменных и разделяем матрицу A на две матрицы B_k и N_k , группируя столбцы, соответствующие базисным и свободным неизвестным. Вычисляем матрицы B_k^{-1} и $\hat{N}_k = B_k^{-1} N_k$ и переходим к основной части этого алгоритма.

1. На k -й итерации в точке $x^{k-1} \in \Omega$ вычисляем блоки g_k^B и g_k^N матрицы Якоби целевой функции, соответствующие базисным и свободным переменным. Затем находим приведенный градиент $u^k = (g_k^N - g_k^B \hat{N}_k)^T$. По формулам (5.11), используя приведенный градиент, находим вектор $p_N^k \in \mathbb{R}^{n-m}$. Если $|p_N^k| < \varepsilon_3$, то итерации прекращаем, полагая $x^* \approx x^{k-1}$ и $f_0(x^*) \approx f_0(x^{k-1})$. В противном случае переходим к п. 2.

2. Вычисляем вектор $p_B^k = -\hat{N}_k p_N^k \in \mathbb{R}^m$ и формируем вектор p^k , объединяя координаты векторов p_B^k и p_N^k с учетом их порядка, определяемого множествами индексов \mathcal{J}_k^B и \mathcal{J}_k^N . Этот вектор задает возможное направление спуска из точки x^{k-1} . Формируем множество \mathcal{I}_k^* индексов j , для которых $p_j^{(k)} < 0$. Если это множество непусто, вычисляем значение

$$\bar{\kappa}_k = \min_{j \in \mathcal{I}_k^*} \left(-\frac{x_j^{(0)}}{p_j} \right) > 0. \quad (5.16)$$

Если же $\mathcal{I}_k^* = \emptyset$, то в качестве $\bar{\kappa}_k$ можно принять любое достаточно большое положительное значение. Минимизируя функцию $\varphi_k(\kappa) = f_0(x^{k-1} + \kappa p^k)$ в полуинтервале $(0, \bar{\kappa}_k]$, находим значение $\kappa_k^* \in (0, \bar{\kappa}_k]$ и новую точку $x^k = x^{k-1} + \kappa_k^* p^k \in \Omega$. Если $\kappa_k^* = \bar{\kappa}_k$, то переходим к п. 3. В противном случае номер итерации k увеличиваем на единицу и возвращаемся к п. 1.

3. Устанавливаем индекс i , для которого $-x_i^{(k-1)}/p_i^{(k)} = \bar{\kappa}_k$. Таких индексов может быть несколько. Если все они принадлежат множеству

\mathcal{J}^N , то увеличиваем номер итерации k на единицу и возвращаемся к п. 1. В противном случае равно нулю хотя бы одно базисное переменное $x_i^{(k)}$, $i \in \mathcal{J}^B$, и мы переходим к п. 4.

4. Установленный в п. 3 индекс i (если их несколько, то один из них) переводим из множества \mathcal{J}_k^B в множество \mathcal{J}_k^N , а из \mathcal{J}_k^N в \mathcal{J}_k^B переводим индекс l наибольшего по значению свободного переменного x_l . Для новых множеств индексов \mathcal{J}_k^B и \mathcal{J}_k^N разделяем матрицу A на матрицы B_k и N_k и вычисляем матрицы B_k^{-1} и $\hat{N}_k = B_k^{-1} N_k$. Увеличив номер итерации k на единицу, переходим к п. 1.

Отсутствие изменений в составе базисных переменных в течение нескольких итераций указывает на то, что достигнута достаточно близкая окрестность искомой точки $x^* \in \Omega$. В этом случае для ускорения сходимости можно применить спуск по сопряженным направлениям в пространстве \mathbb{R}^{n-m} свободных переменных.

Пример 5.2. Рассмотрим задачу нелинейного программирования

$$\begin{aligned} f_0(x_1, x_2, x_3) &= x_1^2 - 2x_1x_2 + x_2^2 + x_3^2 \rightarrow \min; \\ x_1 + 2x_2 + x_3 &= 3; \\ x_j &\geq 0, \quad j = \overline{1, 3}. \end{aligned}$$

Сначала используем модифицированный алгоритм метода приведенного градиента, выполняя спуск по сопряженным направлениям в пространстве свободных переменных. Выберем начальную точку $x^0 = (1, 1/2, 1)$ и зададим параметр точности поиска $\varepsilon_3 = 0,1$.

Целевая функция представляет собой неотрицательно определенную квадратичную форму. Запишем ее в виде $f_0(x) = \frac{1}{2} (Qx, x)$, где

$$Q = \begin{pmatrix} 2 & -2 & 0 \\ -2 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix}.$$

В задачу входит одно линейное ограничение типа равенства. Поэтому лишь одно переменное может быть базисным. Выберем в качестве базисного переменное x_1 . Тогда $\mathcal{J}_1^B = \{1\}$, $\mathcal{J}_1^N = \{2, 3\}$, $B_1 = 1$ и $N_1 = (2 \ 1)$. Вычисляем $B_1^{-1} = 1$ и $\hat{N}_1 = B_1^{-1} N_1 = (2 \ 1)$.

Первая итерация. 1. В точке x^0 вычисляем приведенный градиент:

$$\begin{aligned} (u^1)^T &= g_1^N - g_1^B \hat{N}_1 = \left(\frac{\partial f_0}{\partial x_2} \frac{\partial f_0}{\partial x_3} \right) - \frac{\partial f_0}{\partial x_1} \hat{N}_1 = \\ &= (-2x_1 + 2x_2 \ 2x_3) - (2x_1 - 2x_2) \hat{N}_1 = (-1 \ 2) - (2 \ 1) = (-3 \ 1). \end{aligned}$$

Так как $u_2^{(1)} = -3$ и $u_3^{(1)} = 1$, то в соответствии с соотношениями (5.11) имеем $p_2^{(1)} = -u_2^{(1)} = 3$ и $p_3^{(1)} = -u_3^{(1)} x_3^{(0)} = -1$. Следовательно, $p_N^1 = (3 \ -1)^T$. Поскольку $|p_N^1| = \sqrt{10} > \varepsilon_3 = 0,1$, переходим к п. 2 алгоритма.

2. По формуле (5.13) находим

$$p_B^1 = -\hat{N}_1 p_N^1 = -(2 \ 1)(3 \ -1)^T = -5$$

и формируем вектор $p^1 = (-5 \ 3 \ -1)^T$, который определяет возможное направление спуска из точки x^0 . Формируем множество $\mathcal{I}_1^* = \{j \in \mathcal{J}: p_j^{(1)} < 0\} = \{1, 3\}$ и по формуле (5.16) вычисляем

$$\bar{\kappa}_1 = \min \left\{ -\frac{x_1^{(0)}}{p_1^{(1)}}, -\frac{x_3^{(0)}}{p_3^{(1)}} \right\} = \min \left\{ \frac{1}{5}, 1 \right\} = \frac{1}{5}.$$

Находим минимум функции $\varphi_1(\kappa) = f_0(x^0 + \kappa p^1)$. Для этого, учитывая, что целевая функция квадратичная, можно использовать формулу (5.7):

$$\kappa_1^* = -\frac{(Qx^0, p^1)}{(Qp^1, p^1)} = \frac{1}{13}.$$

Поскольку $\kappa_1^* < \bar{\kappa}_1 = 1/5$, функция $\varphi_1(\kappa)$ достигает минимума на отрезке $[0, \bar{\kappa}_1] = [0, 1/5]$ при $\kappa = \kappa_1^*$. Вычисляем точку $x^1 = x^0 + \kappa_1^* p^1 = (8/13, 19/26, 12/13)$ и переходим ко второй итерации, возвращаясь к п. 1 алгоритма.

Вторая итерация. 1. Множества индексов \mathcal{J}^B и \mathcal{J}^N не изменились, поэтому $\hat{N}_2 = \hat{N}_1 = (2 \ 1)$. Вычисляем приведенный градиент в новой точке x^1 :

$$(u^2)^T = g_2^N - g_2^B \hat{N}_2 = \left(\frac{3}{13} \ \frac{24}{13} \right) + \frac{3}{13} (2 \ 1) = \frac{9}{13} (1 \ 3).$$

Поскольку базисное переменное осталось прежним, вектор p_N^2 найдем как сопряженный с вектором p_N^1 :

$$\begin{aligned} p_N^2 &= -u^2 + \frac{|u^2|^2}{|u^1|^2} p_N^1 = \\ &= -\frac{9}{13} \begin{pmatrix} 1 \\ 3 \end{pmatrix} + \frac{(9/13)^2(1^2 + 3^2)}{3^2 + 1^2} \begin{pmatrix} 3 \\ -1 \end{pmatrix} = \frac{18}{169} \begin{pmatrix} 7 \\ -24 \end{pmatrix}. \end{aligned}$$

Так как $|p_N^2| = 450/169 > \varepsilon_3 = 0,1$, переходим к п. 2.

2. По формуле (5.13) находим вектор

$$p_B^2 = -\hat{N}_1 p_N^2 = -\frac{18}{169} (2 \ 1) (7 \ -24)^T = \frac{180}{169}$$

и формируем вектор $p^2 = \frac{18}{169} (10 \ 7 \ -24)^T$, задающий направление спуска из точки x^1 . Длина этого вектора для дальнейших вычислений на этой итерации не имеет значения. Поэтому примем $p^2 = (10 \ 7 \ -24)^T$. Сформировав множество $\mathcal{I}_2^* = \{j \in \mathcal{J}: p_j^{(2)} < 0\} = \{3\}$, по формуле (5.16) вычисляем

$$\bar{x}_2 = -\frac{x_3^{(1)}}{p_3^{(2)}} = \frac{1}{26}.$$

Минимизируя функцию $\varphi_2(x) = f_0(x^1 + x p^2)$ на отрезке $[0, \bar{x}_2]$, находим

$$x_2^* = -\frac{(Qx^1, p^2)}{(Qp^2, p^2)} = \frac{1}{26} = \bar{x}_2$$

и вычисляем точку $x^2 = x^1 + x_2^* p^1 = (1, 1, 0)$. Поскольку $x_2^* = \bar{x}_2$, переходим к п. 3.

3. Так как индекс $i = 3$ принадлежит множеству $\mathcal{J}_2^N = \{2, 3\}$, т.е. обратилось в нуль свободное переменное x_3 , переходим к третьей итерации, возвращаясь к п. 1 алгоритма.

Третья итерация. 1. Снова множества индексов \mathcal{J}^B и \mathcal{J}^N остались прежними, поэтому $\hat{N}_3 = \hat{N}_1 = (2 \ 1)$. Находим приведенный градиент в новой точке x^2 :

$$(u^3)^T = g_3^N - g_3^B \hat{N}_3 = (0 \ 0) - 0 \cdot (2 \ 1) = (0 \ 0).$$

Поскольку вектор u^3 оказался нулевым, то и вектор p_N^3 в соответствии с соотношениями (5.11) будет нулевым. Так как $|p_N^3| = 0 < \varepsilon_3 = 0,1$, дальнейшие вычисления прекращаем и полагаем $x^* = x^3 = (1, 1, 0)$, $f_0(x^*) = f_0(x^3) = 0$.

Из равенства $p_N^3 = 0$ вытекает, что в точке x^3 выполняются необходимые условия локального экстремума (5.15). В общем случае из этого не следует, что x^3 — точка минимума. Но в данном случае, учитывая, что целевая функция выпуклая, а ограничения задачи линейные, такой вывод сделать можно.

Применение метода приведенного градиента в сочетании с использованием сопряженных направлений позволило найти точку минимума целевой функции всего за две итерации. Это связано с тем, что целевая функция является квадратичной, а свободных переменных всего два.

Выполним в рассматриваемой задаче вторую итерацию, используя основной алгоритм, в котором в пространстве свободных переменных выполняется градиентный спуск, и убедимся в том, что для достижения точки минимума целевой функции необходимо большее число итераций. В этой итерации исходной является точка $x^1 = (8/13, 19/26, 12/13)$, а приведенный градиент в этой точке равен $u^2 = (9/13 \ 27/13)$. По формуле (5.11) находим

$$\tilde{p}_2^{(2)} = -u_2^{(2)} x_2^{(1)} = -\frac{171}{328}, \quad \tilde{p}_3^{(2)} = -u_3^{(2)} x_3^{(1)} = -\frac{324}{169}$$

и определяем вектор $\tilde{p}_N^2 = \frac{9}{328}(-19 \ -72)^T$. Поскольку $|\tilde{p}_N^2| \approx 2,04 > \varepsilon_3 = 0,1$, то переходим к п. 2 алгоритма.

Вычислив вектор $\tilde{p}_B^2 = -\hat{N}_2 \tilde{p}_N^2 = \frac{495}{169}$ (в данном случае этот вектор — число), сформируем вектор $\tilde{p}^2 = (110 \ -19 \ -72)^T$, опустив множитель $9/328$. На множестве $\tilde{I}_2^* = \{2, 3\}$ в соответствии с равенством (5.16) находим

$$\tilde{\kappa}_2 = \min \left\{ \frac{19}{26 \cdot 19}, \frac{12}{13 \cdot 72} \right\} = \frac{1}{78} \approx 0,01282.$$

Функция $\tilde{\varphi}_2(\kappa) = f_0(x^1 + \kappa \tilde{p}^2)$ достигает минимума при

$$\tilde{\kappa}_2^* = -\frac{(Qx^1, \tilde{p}^2)}{(Q\tilde{p}^2, \tilde{p}^2)} = \frac{4230}{26 \cdot 43450} \approx 0,00374 < \tilde{\kappa}_2,$$

и мы получаем точку $\tilde{x}^2 = x^1 + \tilde{x}_2 \tilde{p}^2 = (1,0333, 0,6596, 0,6535)$. Эта точка отличается от точного решения $x^2 = (1, 1, 0)$, полученного с использованием сопряженных направлений, т.е. за две итерации основного алгоритма точка минимума не была достигнута, получено лишь некоторое приближение к ней.

5.3. Проектирование точки на множество

Выше (см. разд. 5.1) отмечено, что непосредственное применение к численному решению общей задачи нелинейного программирования одного из методов спуска может привести к тому, что очередная точка итерационной последовательности выйдет за пределы допустимого множества. Чтобы этого избежать, необходимо корректировать точку, полученную в результате выполнения очередной итерации. Одним из способов такой корректировки является операция проектирования точки на допустимое множество.

Проекцией точки $x \in \mathbb{R}^n$ на множество $\Omega \subset \mathbb{R}^n$ называют ближайшую к x точку y этого множества и обозначают ее $P_\Omega(x)$. Для точки $y = P_\Omega(x)$ выполняются равенства

$$|y - x| = \inf_{z \in \Omega} |z - x| = \rho(x, \Omega). \quad (5.17)$$

Величину $\rho(x, \Omega)$ называют *расстоянием от точки x до множества Ω* .

На рис. 5.5 дана геометрическая иллюстрация операции проектирования точки на множество. Проекция точки $x \notin \Omega$ на множество Ω может и не существовать. Например, для открытого множества Ω ни одна точка вне Ω не имеет проекции на это множество: нетрудно показать, что проекция точки $x \notin \Omega$ на Ω не может быть внутренней точкой Ω . В то же время может существовать несколько (и даже бесконечное множество) проекций точки на множество (рис. 5.6).

Перечислим основные свойства операции проектирования точки на множество.

1. Если $x \in \Omega$, то $P_\Omega(x) = x$.
2. Если Ω — замкнутое множество, то любая точка $x \in \mathbb{R}^n$ имеет проекцию на это множество. Если при этом Ω — выпуклое множество, то проекция любой точки $x \in \mathbb{R}^n$ на Ω единственна.

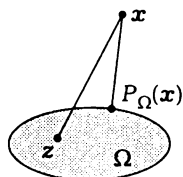


Рис. 5.5

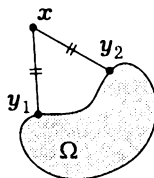


Рис. 5.6

3. Точка y является проекцией точки $x \in \mathbb{R}^n$ на замкнутое выпуклое множество Ω тогда и только тогда, когда для любой точки $z \in \Omega$ выполнено неравенство (рис. 5.7)

$$(z - y, x - y) \leq 0. \quad (5.18)$$

4. Если Ω — выпуклое замкнутое множество, $y_1 = P_\Omega(x_1)$ и $y_2 = P_\Omega(x_2)$ — проекции точек x_1 и x_2 на Ω , то

$$|y_1 - y_2| \leq |x_1 - x_2|,$$

т.е. длина проекции отрезка на выпуклое множество не превосходит длины самого отрезка (рис. 5.8).

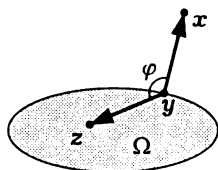


Рис. 5.7

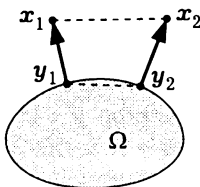


Рис. 5.8

Непосредственно из определения следует, что проекцию заданной точки $x \in \mathbb{R}^n$ на замкнутое множество $\Omega \subset \mathbb{R}^n$ можно найти путем решения задачи минимизации

$$\Phi(z) = |z - x|^2 \rightarrow \min, \quad z \in \Omega. \quad (5.19)$$

Согласно свойству 2 эта задача имеет решение, а при дополнительном условии выпуклости Ω решение задачи единственно. Трудность решения задачи связана главным образом с видом множества Ω и способом его описания. Рассмотрим несколько примеров, когда определение проекции точки на множество не составляет труда.

Пример 5.3. Если $\Omega = \{z \in \mathbb{R}^n: |z - x_0| \leq R\}$ — замкнутый шар с центром в точке $x_0 \in \mathbb{R}^n$ и радиусом R , то из геометрических соображений нетрудно понять, что проекция точки $x \notin \Omega$ на Ω расположена на отрезке, соединяющем точку x с центром шара x_0 (рис. 5.9). Отсюда несложно получить

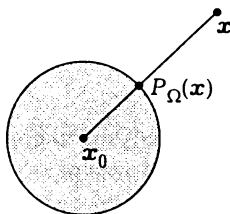


Рис. 5.9

$$P_{\Omega}(x) = x_0 + \frac{x - x_0}{|x - x_0|} R. \quad (5.20)$$

Множество $\tilde{\Omega} = \{z \in \mathbb{R}^n: |z - x_0| = R\}$ точек, равноудаленных от точки x_0 (называемое $(n+1)$ -мерной сферой), является замкнутым, но не выпуклым. Для точки x_0 любая точка множества является проекцией. Для точек $x \notin \tilde{\Omega}$ проекция на сферу единственна и может быть найдена по формуле (5.20).

Пример 5.4. Гиперплоскость

$$\Omega = \{z \in \mathbb{R}^n: (n, z) = b\}, \quad (5.21)$$

где $n \in \mathbb{R}^n$ — нормальный вектор этой гиперплоскости, а b — постоянное число, является замкнутым выпуклым множеством. Проекцию точки $x \notin \Omega$ на множество Ω , как и в предыдущем примере, можно найти из геометрических соображений. Из этих соображений нетрудно понять, что вектор $y - x$, где $y = P_{\Omega}(x)$, коллинеарен вектору n , а поэтому $y = x + \lambda n$, где λ — некоторое пока не известное число. Подставляя это представление в уравнение гиперплоскости, получаем $(n, y) = (n, x) + \lambda n^2 = b$. Отсюда легко определить значение λ , и в результате мы получаем

$$y = P_{\Omega}(x) = x + \frac{b - (n, x)}{n^2} n. \quad (5.22)$$

Найденное решение подтверждается с помощью теоремы Пифагора. Для любой точки $z \in \Omega$ имеем $(n, z) = (n, y)$, откуда $(n, z - y) = 0$, т.е. векторы n и $z - y$, а значит, и векторы $y - x$ и $z - y$, ортогональны. По теореме Пифагора

$$|z - x|^2 = |(z - y) + (y - x)|^2 = |z - y|^2 + |y - x|^2 \geq |y - x|^2.$$

Следовательно, точка $y = P_{\Omega}(x)$ оказывается ближайшей к точке x .

Несложно проверить, что проекция точки x на полупространство

$$\Omega^* = \{z \in \mathbb{R}^n: (n, z) \leq b\} \quad (5.23)$$

(при условии $x \in \Omega^*$) определяется той же формулой (5.22).

Пример 5.5. Расширим предыдущий пример и рассмотрим множество

$$\Omega = \{z \in \mathbb{R}^n: (n_i, z) = b_i, i = \overline{1, m}\},$$

определяемое системой из $m < n$ линейно независимых векторов n_i , $i = \overline{1, m}$. Это множество, как пересечение замкнутых выпуклых множеств, является замкнутым выпуклым. Проекцию произвольной точки $x \notin \Omega$ на множество Ω можно найти методами линейной алгебры.

Рассмотрим линейную оболочку $\mathcal{H} \subset \mathbb{R}^n$ векторов n_1, \dots, n_m и ортогональное дополнение \mathcal{H}^\perp к подпространству \mathcal{H} . Выберем произвольную точку $z \in \Omega$ и представим вектор $z - x$ в виде $z - x = a_H + a^\perp$, где $a_H \in \mathcal{H}$, $a^\perp \in \mathcal{H}^\perp$. Поскольку $a^\perp \in \mathcal{H}^\perp$, то $(n_i, a^\perp) = 0$, $i = \overline{1, m}$. Отсюда нетрудно заключить, что вектор $y = z - a^\perp = x + a_H$ принадлежит множеству Ω . Кроме того, из ортогональности векторов a_H и a^\perp вытекает, что

$$|z - x|^2 = |a_H|^2 + |a^\perp|^2 \geq |a^\perp|^2 = |y - x|^2.$$

Следовательно, точка $y \in \Omega$ является ближайшей к точке x .

Чтобы найти вектор y , вернемся к разложению $z - x = a_H + a^\perp$ и запишем условия ортогональности $(z - x - a_H, n_j) = 0$, $j = \overline{1, m}$, вектора a^\perp векторам n_j . Записав вектор a_H в виде линейной комбинации векторов n_j , т.е. $a_H = \sum_{i=1}^m \lambda_i n_i$, и используя равенства $(z, n_j) = b_j$, $j = \overline{1, m}$, получим

$$b_j - (x, n_j) - \sum_{i=1}^m \lambda_i (n_i, n_j) = 0, \quad j = \overline{1, m}.$$

Таким образом, мы пришли к квадратной системе линейных уравнений с неизвестными коэффициентами λ_i :

$$\sum_{i=1}^m \lambda_i (n_i, n_j) = b_j - (x, n_j), \quad j = \overline{1, m}. \quad (5.24)$$

Матрицей этой системы является матрица Грама системы векторов n_1, \dots, n_m , т.е. матрица скалярных произведений (n_i, n_j) . Из курса линейной алгебры известно, что матрица Грама невырождена тогда и только тогда, когда система векторов линейно независима. Это значит, что в рассматриваемой ситуации система уравнений (5.24) имеет решение, и притом единственное. Это решение позволяет вычислить вектор $a_H = \sum_{i=1}^m \lambda_i n_i$, а вместе с ним и вектор $y = x + a_H$.

Пример 5.6. Рассмотрим n -мерный параллелепипед

$$\Omega = \{x \in \mathbb{R}^n: a_j \leq x_j \leq b_j, j = \overline{1, n}\},$$

где a_j и b_j — заданные числа. Координаты y_j проекции $y = P_\Omega(x)$ точки $x = (x_1, \dots, x_n)$ на множество Ω можно найти по формулам $y_j = \min \{ \max \{ x_j, a_j \}, b_j \}$, $j = \overline{1, n}$, или в развернутой форме

$$y_j = \begin{cases} a_j, & x_j \leq a_j; \\ x_j, & a_j < x_j < b_j; \\ b_j, & x_j \geq b_j, \end{cases} \quad j = \overline{1, n}.$$

5.4. Метод проекции точки на множество

Сочетание метода градиентного спуска, используемого при безусловной минимизации, и операции проектирования точки на множество составляет существо *метода проекции точки на множество*, применяемого для решения общей задачи нелинейного программирования. Такое сочетание позволяет на каждой итерации обеспечивать соблюдение ограничений, определяющих множество Ω .

В методе проекции точки на множество релаксационную последовательность $\{x^k\}$ строят с помощью рекуррентного соотношения

$$x^k = P_\Omega(x^{k-1} + \kappa_k w^k), \quad (5.25)$$

сперва определяя точку $\tilde{x}^k = x^{k-1} + \kappa_k w^k$ каким-либо методом градиентного спуска, а затем проектируя ее на допустимое множество. Здесь $w^k = -\text{grad } f_0(x^{k-1})$ — антиградиент целевой функции, вычисленный в точке $x^{k-1} \in \Omega$. Процесс построения релаксационной последовательности начинается с выбора начальной точки $x^0 \in \Omega$.

Если Ω — замкнутое выпуклое множество, процедуры градиентного спуска и последующего проектирования на допустимое множество можно объединить в одной задаче минимизации. Действительно, рассмотрим минимизацию на множестве Ω квадратичной функции

$$\Phi_k(z) = \frac{|z - x^{k-1}|^2}{2} - \kappa_k(z - x^{k-1}, w^k), \quad z \in \Omega.$$

Эта функция выпуклая как сумма двух выпуклых функций. Она достигает минимума в точке $x^k \in \Omega$, в которой неравенство

$$(\text{grad } \Phi_k(x^k), z - x^k) \geq 0$$

выполнено для любой точки $z \in \Omega$. В данном случае $\text{grad } \Phi_k(z) = z - x^{k-1} - \kappa_k w^k$, поэтому

$$\begin{aligned} 0 \leq (\text{grad } \Phi_k(x^k), z - x^k) &= (x^k - x^{k-1} - \kappa_k w^k, z - x^k) = \\ &= -(\tilde{x}^k - x^k, z - x^k). \end{aligned} \quad (5.26)$$

Сравнивая неравенство $(\tilde{x}^k - x^k, z - x^k) \leq 0$ с условием (5.18), убеждаемся, что точка x^k минимума функции $\Phi_k(z)$ на множестве Ω является проекцией точки $\tilde{x}^k = x^{k-1} + \kappa_k w^k$ на это множество.

Варианты метода проекции градиента различаются способами выбора значения параметра κ_k , регулирующего на k -й итерации шаг спуска в направлении антиградиента. Могут быть использованы и фиксированное значение параметра, и его выбор исходя из исчерпывающего спуска либо в соответствии с методом дробления шага.

Критерием прекращения поиска в методе проекции точки на множество служат те же условия (3.2) и (3.3), что и в случае безусловной минимизации.

Опишем подробнее последовательность действий при выполнении k -й итерации варианта этого метода, в котором используется исчерпывающий спуск в направлении антиградиента. Предположим, что допустимое множество Ω задано ограничением $g(x) = 0$ типа равенства, где $g(x)$ — функция, дифференцируемая в \mathbb{R}^n . Графически выполнение k -й итерации в двумерном случае показано на рис. 5.10. Здесь Ω есть множество точек плоской гладкой кривой Γ , заданной уравнением $g(x) = 0$.

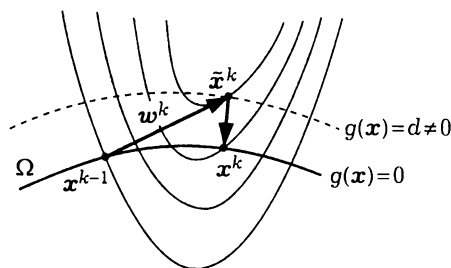


Рис. 5.10

После вычисления антиградиента w^k в исходной точке x^{k-1} (найденной на предыдущей итерации или начальной в случае первой итерации), ортогонального линии уровня $f_0(x) = f_0(x^{k-1})$ целевой функции, проведем исчерпывающий спуск в направлении антиградиента до точки \tilde{x}^k , в которой направление спуска касается линии уровня целевой функции (см. рис. 5.10). Поскольку $g(\tilde{x}^k) = d \neq 0$, ограничение типа равенства, задающее допустимое множество Ω , нарушено. Для завершения k -й итерации и нахождения точки x^k необходимо спроектировать точку \tilde{x}^k на множество Ω . В данном случае это означает, что нужно найти точку плоской кривой Γ , наименее удаленную от точки \tilde{x}^k . Так как функция $g(x)$ дифференцируема, вектор $x^k - \tilde{x}^k$ будет ортогонален касательной к кривой Γ , построенной в точке x^k (или, проще говоря, этот вектор ортогонален кривой Γ).

Аналогична последовательность действий и в случае, когда допустимое множество Ω задано ограничением типа неравенства $g(x) \leq 0$. На рис. 5.10 это множество — часть плоскости, ограниченная кривой Γ , которая описывается уравнением $g(x) = 0$. При $g(\tilde{x}^k) = d > 0$ имеем $\tilde{x}^k \notin \Omega$, а проекция точки \tilde{x}^k на множество Ω снова является основанием кратчайшего перпендикуляра, опущенного из этой точки на кривую Γ .

Рассмотрим условия сходимости последовательности $\{x^k\}$.

Теорема 5.1. Если Ω — замкнутое выпуклое множество, а целевая функция $f_0(x)$ на этом множестве ограничена снизу, непрерывно дифференцируема и для любых $x, y \in \Omega$ удовлетворяет условию

$$|\text{grad } f_0(x) - \text{grad } f_0(y)| \leq L|x - y|, \quad L > 0, \quad (5.27)$$

то при любой начальной точке $x^0 \in \Omega$ для релаксационной последовательности $\{x^k\}$, построенной по формуле (5.25), где $0 < \varepsilon_0 \leq \kappa_k \leq \frac{2}{L+2\varepsilon}$, $\varepsilon > 0$, верно равенство $\lim_{k \rightarrow \infty} |x^{k-1} - x^k| = 0$. Если при этом множество $X_0 = \{x \in \Omega: f_0(x) \leq f_0(x^0)\}$ ограничено, то такая последовательность имеет хотя бы одну предельную точку \tilde{x}^* , в которой для любых $x \in \Omega$ выполнено неравенство

$$(\text{grad } f_0(\tilde{x}^*), x - \tilde{x}^*) \geq 0. \quad \# \quad (5.28)$$

Можно показать, что если выполнены все условия теоремы 5.1 и, кроме того, функция $f_0(x)$ выпукла на допустимом множестве Ω , то справедлива оценка $0 \leq f_0(x^k) - f_* \leq C/k$, $k \in \mathbb{N}$, $C = \text{const} > 0$. Если при этом целевая функция сильно выпукла, а релаксационная последовательность $\{x^k\}$ построена по рекуррентному соотношению (5.25) так, что выполняется условие $0 < \varepsilon_0 \leq \kappa_k = \kappa < 2\gamma/L^2$, где $\gamma \leq L$ — параметр сильной выпуклости функции $f_0(x)$, то последовательность $\{x^k\}$ сходится к точке x^* минимума целевой функции на множестве Ω и справедлива оценка $|x^k - x^*| \leq q^k |x^0 - x^*|$, $k \in \mathbb{N}$, где $q = \sqrt{1 - 2\gamma\kappa + \kappa^2 L^2} \in (0, 1)$. Отметим, что при выборе $\kappa = \gamma/L^2$ знаменатель геометрической прогрессии принимает минимально возможное значение $q = \sqrt{1 - \gamma^2/L^2}$.

Проиллюстрируем возможности метода проекции точки на множество на конкретных примерах.

Пример 5.7. Рассмотрим задачу нелинейного программирования

$$\begin{aligned} f_0(x_1, x_2) &= 10x_1^2 - 4x_1x_2 + 7x_2^2 - 4\sqrt{5}(5x_1 - x_2) - 16 \rightarrow \min, \\ x_1 - x_2 - \sqrt{5} &= 0. \end{aligned}$$

В качестве начальной точки выберем $x^0 = (0, -\sqrt{5})$. Проектирование точки \tilde{x}^k на множество Ω выполняем по формуле (5.22), где $b = \sqrt{5}$ и $n = (1, -1)$ — нормальный вектор прямой $x_1 - x_2 - \sqrt{5} = 0$. Условием останова выберем выполнение неравенства $|x^k - x^{k-1}| < \varepsilon_1$, где $\varepsilon_1 = 0,01$, а в качестве приближенного решения задачи — последнюю точку релаксационной последовательности: $x^* \approx x^k$ и $f(x^*) \approx f(x^k)$.

Точное решение рассматриваемой задачи: $x^* = (\sqrt{5}, 0)$ и $f(x^*) = -66$, где $\sqrt{5} \approx 2,2361$. При численном решении задачи используем три варианта выбора значения κ_k в рекуррентном соотношении

Таблица 5.2

κ_k	N	x^N
$\kappa = 0,10$	6	(2,234, -0,002)
$\kappa = 0,05$	7	(2,235, -0,001)
$\tilde{\kappa}_k$	4	(2,236, 0,000)

(5.25): первый и второй варианты — постоянные значения $\kappa = 0,1$ и $\kappa = 0,05$; третий — выбор κ_k с помощью исчерпывающего спуска. Результаты численного решения представлены табл. 5.2. Траектории спуска в трех вариантах метода показаны на рис. 5.11: *а* — $\kappa = 0,1$; *б* — $\kappa = 0,05$; *в* — $\kappa = \tilde{\kappa}_k$ (параметр определяется исчерпывающим спуском). Расчеты показывают, что по числу N итераций и достигнутому приближению

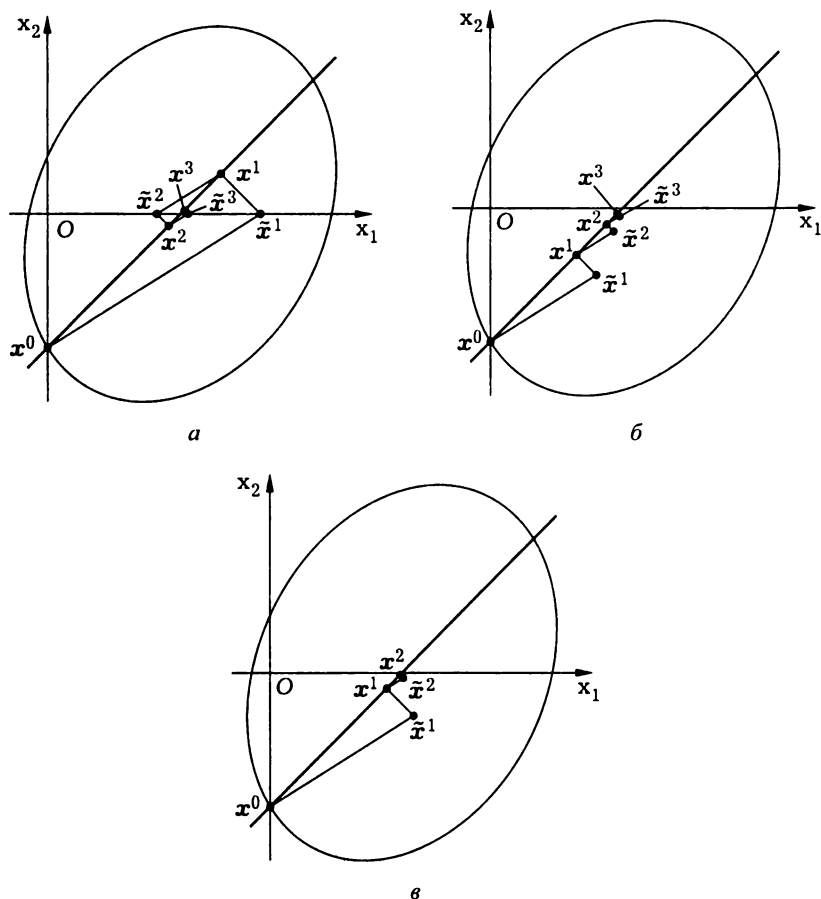


Рис. 5.11

к точке минимума в данном примере предпочтение следует отдать варианту метода с использованием исчерпывающего спуска.

Обратим внимание на различие в траекториях поиска точки x^* при постоянных значениях \varkappa (см. рис. 5.11, а и б). Видно, что при большем из двух значений (при $\varkappa = 0,1$) точка минимума целевой функции достигнута за меньшее число итераций.

Пример 5.8. Методом проекции точки на множество решим задачу нелинейного программирования

$$f_0(x_1, x_2) = 10(x_1^2 - x_2)^2 + (x_1 - 1)^2 \rightarrow \min;$$

$$(x_1 - 1)^2 + (x_2 - 3)^2 \leq 1.$$

В качестве начальной выберем точку $x^0 = (0, 3) \in \Omega$, а параметр \varkappa_k на каждой итерации будем находить путем исчерпывающего спуска. Проектирование точки \tilde{x}^k на замкнутый круг Ω осуществляем в соответствии с формулой (5.20). Таким образом,

$$x^k = P_{\Omega}(\tilde{x}^k) = x_0 + \frac{\tilde{x}^k - x_0}{|\tilde{x}^k - x_0|},$$

где $x_0 = (1, 3)$ — центр круга Ω . Прекращение поиска определяется условием $|x^k - x^{k-1}| < \varepsilon_1$, где $\varepsilon_1 = 10^{-3}$ — параметр точности поиска. Процесс поиска в рассматриваемой задаче показан на рис. 5.12.

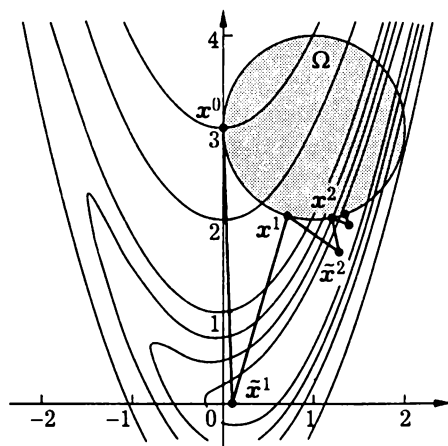


Рис. 5.12

Пример 5.9. Методом проекции точки на множество найдем минимум функции $f_0(x_1, x_2)$ из примера 5.8 в прямоугольнике $\Omega \subset \mathbb{R}^2$, заданном соотношениями $x_1 \in [-2, 5, 0]$, $x_2 \in [-1, 2]$. В качестве начальной возьмем точку $x^0 = (-2, 2)$, а способ выбора параметра α_k оставим тем же, что и в примере 5.8. Для прямоугольника построение проекции рассмотрено в примере 5.6. На рис. 5.13 показаны первые итерации процесса поиска точки минимума.

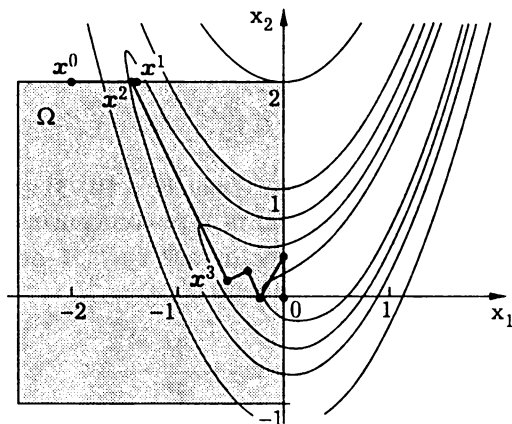


Рис. 5.13

5.5. Метод проекции антиградиента

Результатом проектирования точки $x \in \mathbb{R}^n$ на замкнутое множество $\Omega \subset \mathbb{R}^n$ является точка $y \in \Omega$, либо совпадающая с x , если $x \in \Omega$, либо лежащая на его границе $\partial\Omega$ (см. разд. 5.3). Это свойство операции проектирования позволяет при численном решении задачи оптимизации с ограничениями на каждой k -й итерации использовать обычный метод градиентного спуска в направлении антиградиента из точки x^{k-1} в точку \tilde{x}^k , а затем находить проекцию $x^k \in \Omega$ точки \tilde{x}^k на множество Ω (см. разд. 5.4). Если при этом $\tilde{x}^k \notin \Omega$, то операция проектирования как бы возвращает ее обратно в допустимое множество Ω .

Естественен вопрос: нельзя ли предварительно найти такое *направление спуска*, движение точки по которому при определенных ограничениях вообще не выводило бы ее за пределы допустимого

множества Ω ? Сначала рассмотрим случай, когда множество Ω описывается системой линейных ограничений типа равенства:

$$\Omega = \{x \in \mathbb{R}^n: (a_i, x) = b_i, i = \overline{1, m}\}. \quad (5.29)$$

Отметим, что в силу линейности ограничений множество Ω является выпуклым.

Составим вектор $b = (b_1 \dots b_m)^T$ и матрицу A размера $m \times n$, строками которой являются матрицы-строки $a_i^T, i = \overline{1, m}$. Без ограничения общности можно принять, что $\text{Rg } A = m < n$ (см. разд. 5.2). В этом случае векторы $a_i, i = \overline{1, m}$, линейно независимы.

Отметим, что для любой матрицы A матрица AA^T является симметрической, поскольку $(AA^T)^T = (A^T)^T A^T = AA^T$. Если векторы $a_i \in \mathbb{R}^n, i = \overline{1, m}$, линейно независимы, то матрица AA^T невырождена и имеет обратную матрицу $(AA^T)^{-1}$, причем симметрическую, так как матрица AA^T симметрическая.

Квадратная матрица

$$P = A^T(AA^T)^{-1}A \quad (5.30)$$

порядка n является симметрической, поскольку

$$\begin{aligned} P^T &= (A^T(AA^T)^{-1}A)^T = A^T(A^T(AA^T)^{-1})^T = \\ &= A^T((AA^T)^{-1})^T(A^T)^T = A^T(AA^T)^{-1}A = P. \end{aligned}$$

Кроме того,

$$\begin{aligned} PP &= (A^T(AA^T)^{-1}A)A^T(AA^T)^{-1}A = \\ &= A^T(AA^T)^{-1}(AA^T)(AA^T)^{-1}A = A^T(AA^T)^{-1}A = P, \end{aligned}$$

т.е. $P^2 = P$.

Квадратную матрицу M , удовлетворяющую соотношению $M^2 = M$, называют **идемпотентной матрицей**, а симметрическую идемпотентную матрицу — **проекционной**. Отметим, что проекционная матрица является неотрицательно определенной, так как

$$(Mw, w) = (M^2w, w) = (Mw, Mw) = |Mw|^2 \geq 0.$$

Собственными значениями идемпотентной матрицы могут быть только числа 0 и 1. В самом деле, если $x \neq 0$ — собственный вектор идемпотентной матрицы M с собственным значением λ , то

$$\lambda x = Mx = M^2x = M(\lambda x) = \lambda^2 x.$$

Следовательно, $(\lambda^2 - \lambda)x = 0$ и $\lambda^2 - \lambda = 0$, так как $x \neq 0$. Уравнению $\lambda^2 - \lambda = 0$ удовлетворяют только числа 0 и 1. Очевидно, что собственными значениями проекционной матрицы также могут быть только числа 0 и 1.

Нетрудно показать, что если P — проекционная матрица, то и $P^* = I - P$ — также проекционная матрица (здесь I — единичная матрица). Действительно,

$$(P^*)^2 = (I - P)^2 = I - 2P + P^2 = I - P = P^* \quad (5.31)$$

и

$$(P^*)^T = (I - P)^T = I - P = P^*. \quad (5.32)$$

С помощью матрицы P любой вектор $w \in \mathbb{R}^n$ можно представить в виде суммы $w = Pw + (I - P)w$, в которой слагаемые Pw и $(I - P)w$ ортогональны. Ортогональность слагаемых вытекает из равенств $P(I - P) = P - P^2 = \Theta$, в соответствии с которыми

$$(Pw, (I - P)w) = (w, P(I - P)w) = 0. \quad (5.33)$$

Вектор Pw является проекцией вектора w на линейное подпространство $\mathcal{L}_P = \{x \in \mathbb{R}^n: x = Pw\}$. Если матрица P имеет вид (5.30), то базисом в пространстве \mathcal{L}_P являются векторы $a_i, i = \overline{1, m}$, так как в этом случае для любого вектора $y \in \mathcal{L}_P$ имеем

$$y = Pw = A^T(AA^T)^{-1}Aw = A^Td = \sum_{i=1}^m d_i a_i, \quad (5.34)$$

где $d = (AA^T)^{-1}Aw$, а $d_i, i = \overline{1, m}$, — координаты вектора d .

Обозначим $P^* = I - P$. Линейное подпространство \mathcal{L}_{P^*} всех векторов вида $P^*w, w \in \mathbb{R}^n$, является ортогональным дополнением линейного подпространства \mathcal{L}_P , поскольку для любых $w_1, w_2 \in \mathbb{R}^n$

$$(Pw_1, P^*w_2) = (w_1, P(I - P)w_2) = 0.$$

Поэтому вектор $w^\perp = P^*w$ ортогонален каждому вектору $a_i, i = \overline{1, m}$. Таким образом, вектор w^\perp ортогонален нормальному вектору $a_i \in \mathbb{R}^n$ каждой гиперплоскости $(a_i, x) = b_i, i = \overline{1, m}$, ограничивающей множество Ω (5.29). Следовательно, направление, задаваемое вектором w^\perp , параллельно этим гиперплоскостям и движение в этом направлении из любой точки $x^0 \in \Omega$ не выводит за пределы множества Ω .

Линейные ограничения типа равенства. Пусть в задаче оптимизации допустимое множество Ω определено ограничениями типа равенства, т.е. имеет вид (5.29), а целевая функция $f_0(x)$ дифференцируема в \mathbb{R}^n . Предположим, что задана точка $x^0 \in \Omega$. Рассмотрим функцию

$$\varphi(y) = f_0(x^0 + P^*y), \quad y \in \mathbb{R}^n, \quad (5.35)$$

где матрица P определена соотношением (5.30). Для этой функции в силу правила дифференцирования сложной функции и симметричности матрицы P справедливо равенство*

$$\text{grad } \varphi(y) = P^* \text{grad } f_0(x), \quad (5.36)$$

где x и y связаны соотношением

$$x = x^0 + P^*y, \quad x, y \in \mathbb{R}^n. \quad (5.37)$$

Теорема 5.2. Если y^* — стационарная точка функции $\varphi(y)$, то соответствующая ей точка

$$x^* = x^0 + P^*y^* \quad (5.38)$$

удовлетворяет необходимому условию минимума целевой функции $f_0(x)$ на допустимом множестве Ω вида (5.29).

◀ Прежде всего убедимся, что $x^* \in \Omega$. Используя представление (5.38) и условие $x^0 \in \Omega$, заключаем, что

$$(a_i, x^*) = (a_i, x^0) + (a_i, P^*y^*) = (a_i, x^0) = b_i, \quad i = \overline{1, m},$$

поскольку вектор P^*y^* ортогонален каждому вектору линейного подпространства \mathcal{L}_P , в том числе и векторам a_i . Следовательно, $x^* \in \Omega$.

*Здесь и далее градиент функции трактуется как вектор-столбец.

Так как \mathbf{y}^* является стационарной точкой функции $\varphi(\mathbf{y})$, то $\text{grad } \varphi(\mathbf{y}^*) = \mathbf{0}$, и поэтому, учитывая равенства (5.30) и (5.36) при $\mathbf{y} = \mathbf{y}^*$ и $\mathbf{x} = \mathbf{x}^*$, находим

$$\begin{aligned} P^* \text{grad } f_0(\mathbf{x}^*) &= \text{grad } f_0(\mathbf{x}^*) - A^T (AA^T)^{-1} A \text{grad } f_0(\mathbf{x}^*) = \\ &= \text{grad } f_0(\mathbf{x}^*) + A^T \boldsymbol{\lambda} = \text{grad } f_0(\mathbf{x}^*) + \sum_{i=1}^m \lambda_i \mathbf{a}_i = \mathbf{0}, \end{aligned} \quad (5.39)$$

где $\boldsymbol{\lambda} = -(AA^T)^{-1} A \text{grad } f_0(\mathbf{x}^*)$, а λ_i — координаты вектора $\boldsymbol{\lambda} \in \mathbb{R}^m$. Представив ограничения (5.29) в виде $f_i(\mathbf{x}) = (\mathbf{a}_i, \mathbf{x}) - b_i = 0$, $i = \overline{1, m}$, и учитывая равенства $\text{grad } f_i(\mathbf{x}^*) = \mathbf{a}_i$, заключаем, что точка \mathbf{x}^* является стационарной точкой функции Лагранжа

$$L(\mathbf{x}, \boldsymbol{\lambda}) = f_0(\mathbf{x}) + \sum_{i=1}^m \lambda_i ((\mathbf{a}_i, \mathbf{x}) - b_i), \quad (5.40)$$

т.е. удовлетворяет необходимому условию минимума целевой функции $f_0(\mathbf{x})$ на множестве Ω (5.29). ►

В соответствии с теоремой 5.2 стационарные точки целевой функции $f_0(\mathbf{x})$ на допустимом множестве можно определять, решая задачу безусловной минимизации функции $\varphi(\mathbf{y})$. В общем случае функция $\varphi(\mathbf{y})$ может иметь несколько стационарных точек, причем каждой ее стационарной точке \mathbf{y}^* будет соответствовать стационарная точка \mathbf{x}^* (5.38) функции Лагранжа (5.40). При этом теорема 5.2 не гарантирует, что через минимизацию функции $\varphi(\mathbf{y})$ можно найти все стационарные точки целевой функции. Однако если целевая функция является выпуклой функцией на выпуклом множестве Ω , то для решения задачи минимизации такой функции достаточно найти хотя бы одну стационарную точку целевой функции в Ω , поскольку в любой стационарной точке выпуклая функция достигает наименьшего значения. Отметим, что строго выпуклая функция может иметь лишь одну стационарную точку, так как она может достигать наименьшего значения на выпуклом множестве только в одной точке.

Пример 5.10. Применим теорему 5.2 для поиска минимума квадратичной функции $f_0(\mathbf{x}) = \frac{1}{2} (Q\mathbf{x}, \mathbf{x})$ с положительно определенной матрицей Q порядка n на множестве Ω вида (5.29).

Поскольку в данном случае $\text{grad } f_0(\mathbf{x}) = Q\mathbf{x}$, то в соответствии с соотношениями (5.36) и (5.37) можно записать

$$\text{grad } \varphi(\mathbf{y}) = P^*Q(\mathbf{x}^0 + P^*\mathbf{y}).$$

Из условия $\text{grad } \varphi(\mathbf{y}^*) = \mathbf{0}$ получим СЛАУ

$$P^*QP^*\mathbf{y} = -P^*Q\mathbf{x}^0, \quad (5.41)$$

которой должна удовлетворять стационарная точка \mathbf{y}^* функции $\varphi(\mathbf{y})$ (5.35).

Остановимся на частном случае функции двух переменных. Пусть $f_0(\mathbf{x}) = f_0(x_1, x_2) = x_1^2 + x_2^2$, а допустимое множество задано одним ограничением $x_2 = 2$. Тогда

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

а в представлении (5.29) $m = 1$, $\mathbf{a}_1 = (0 \ 1)^T$ и $b_1 = 2$. Ограничению типа равенства удовлетворяет, например, точка $\mathbf{x}^0 = (2, 2)$, лежащая на прямой, задаваемой уравнением $(\mathbf{a}_1, \mathbf{x}) = 2$, или $x_2 = 2$. В данном случае имеем $A = \mathbf{a}_1^T = (0 \ 1)$, $AA^T = \mathbf{a}_1^T \mathbf{a}_1 = |\mathbf{a}_1|^2 = 1$ и

$$P = A^T(AA^T)^{-1}A = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \cdot 1 \cdot (0 \ 1) = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix};$$

$$P^* = I - P = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix};$$

$$P^*QP^* = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix};$$

$$P^*Q\mathbf{x}^0 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} 2 \\ 2 \end{pmatrix} = \begin{pmatrix} 4 \\ 0 \end{pmatrix}.$$

Система линейных уравнений (5.41) относительно координат точки $\mathbf{y}^* = (y_1^*, y_2^*)$ принимает вид

$$\begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} y_1^* \\ y_2^* \end{pmatrix} = \begin{pmatrix} -4 \\ 0 \end{pmatrix},$$

или $2y_1^* + 0 \cdot y_2^* = -4$ и $0 \cdot y_1^* + 0 \cdot y_2^* = 0$. Из первого уравнения находим $y_1^* = -2$, а второму уравнению удовлетворяет произвольное значение y_2^* . Используя равенство (5.38), получаем

$$\begin{pmatrix} x_1^* \\ x_2^* \end{pmatrix} = \begin{pmatrix} x_1^{(0)} \\ x_2^{(0)} \end{pmatrix} + P^* \begin{pmatrix} y_1^* \\ y_2^* \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \end{pmatrix} + \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} -2 \\ y_2^* \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \end{pmatrix}.$$

Таким образом, $x^* = (0, 2)$ и $f_0(x^*) = 4$. Это точка минимума функции $f_0(x)$ на прямой $x_2 = 2$, так как функция строго выпуклая. Полученный результат очевиден с геометрической точки зрения: $x^* = (0, 2)$ является точкой касания прямой $x_2 = 2$ и линии уровня $f_0(x_1, x_2) = x_1^2 + x_2^2 = 4$ целевой функции (рис. 5.14). #

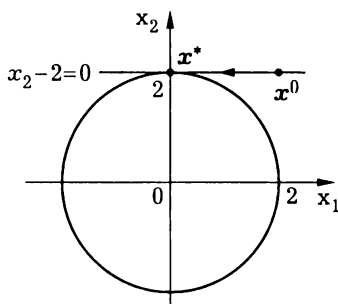


Рис. 5.14

Для поиска стационарной точки $y^* \in \Omega$ функции $\varphi(y)$ можно использовать численные методы безусловной оптимизации. В частности, можно применить один из вариантов метода градиентного спуска. Однако при решении прикладных задач иногда удобнее минимизировать исходную целевую функцию $f_0(x)$, поскольку координаты точки $x \in \mathbb{R}^n$ являются *параметрами оптимизации* и имеют вполне определенную содержательную интерпретацию.

Рассмотрим процедуру поиска минимума дифференцируемой целевой функции $f_0(x)$ на множестве Ω (5.29). Выберем начальную точку $x^0 \in \Omega$, вычислим в этой точке антиградиент $w^1 = -\text{grad } f_0(x^0)$ и антиградиент $p^1 = P^* w^1$ функции $\varphi(y)$ в точке $y = w^1$. Если $p^1 = 0$, то градиент функции $\varphi(y)$ в точке w^1 равен нулевому вектору, т.е. $w^1 = y^*$ — стационарная точка этой функции, а в силу теоремы 5.2

x^0 — точка, удовлетворяющая необходимому условию минимума целевой функции на множестве Ω (для выпуклой целевой функции x^0 — точка ее наименьшего значения на Ω).

При численном решении целесообразно считать $y = w^1$ стационарной точкой функции $\varphi(y)$ при выполнении условия, аналогичного неравенству (3.3): $|p^1| < \varepsilon_3$, где $\varepsilon_3 > 0$ — параметр точности поиска. Если $|p^1| \geq \varepsilon_3$, то вектор p^1 определяет направление спуска для функции $\varphi(y)$ при ее безусловной минимизации. Но этот же вектор задает *возможное направление спуска* для целевой функции, если в соответствии с (5.37) для первой итерации записать

$$x^1 = x^0 + \kappa_1 P^* w^1 = x^0 + \kappa_1 p^1, \quad \kappa_1 > 0. \quad (5.42)$$

Действительно, точка x^1 принадлежит множеству Ω при любом значении κ_1 , так как, учитывая равенства $(a_i, P^* w) = 0$, $i = \overline{1, m}$, для любого вектора w^1 имеем

$$(a_i, x^1) = (a_i, x^0) + (a_i, P^* w^1) = (a_i, x^0) = b_i, \quad i = \overline{1, m}.$$

Поскольку с учетом равенств (5.31) и (5.32)

$$\begin{aligned} (p^1, \text{grad } f_0(x^0)) &= -(P^* w^1, w^1) = -((P^*)^2 w^1, w^1) = \\ &= -(P^* w^1, P^* w^1) = -|p^1|^2 < 0, \end{aligned}$$

вектор p^1 задает направление спуска и для целевой функции, в котором ее значение уменьшается.

Значение κ_1 в соотношении (5.42) можно найти с помощью исчерпывающего спуска в направлении p^1 или методом дробления шага, задаваясь некоторым первоначальным значением κ_0 и при необходимости уменьшая его так, чтобы добиться выполнения неравенства $f_0(x^1) < f_0(x^0)$. Это позволит начать формировать релаксационную последовательность $\{x^k\}$.

Найдя точку x^1 по соотношению (5.42), переходим ко второй итерации и т.д. На k -й итерации, вычисляя в точке $x^{k-1} \in \Omega$ антиградиент $w^k = -\text{grad } f_0(x^{k-1})$, находим возможное направление спуска $p^k = P^* w^k$; в этом направлении (если $|p^k| \geq \varepsilon_3$) ищем точку

$$x^k = x^{k-1} + \kappa_k P^* w^k = x^{k-1} + \kappa_k p^k, \quad \kappa_k > 0, \quad (5.43)$$

подбирая соответствующим образом значение κ_k . Можно показать, что $x^k \in \Omega$ при любом значении κ_k . Следовательно, можно переходить

к следующей, $(k+1)$ -й итерации. Если $|p^k| < \varepsilon_3$, то итерации прекращаем, полагая $w^* \approx w^k$ и $x^* \approx x^{k-1}$.

Поскольку на каждой k -й итерации вектор p^k является проекцией антиградиента w^k целевой функции на ортогональное дополнение подпространства \mathcal{L}_P , то описанную процедуру поиска минимума этой функции на множестве Ω назовем *методом проекции антиградиента*.

Общий случай линейных ограничений. Пусть задача оптимизации содержит линейные ограничения типа неравенства и равенства. В этом случае допустимое множество можно записать в виде

$$\Omega = \{x \in \mathbb{R}^n: (a_i, x) \leq b_i, i \in I^-, (a_i, x) = b_i, i \in I^0\}, \quad (5.44)$$

где I^- и I^0 — конечные множества индексов. Множество Ω (5.44) согласно замечанию 4.1 является выпуклым. Предположим, что на k -й итерации точка x^{k-1} удовлетворяет m_k равенствам $(a_i, x^{k-1}) - b_i = 0$, $i \in \mathcal{J}_k$, где $\mathcal{J}_k \subseteq I = I^- \cup I^0$ — некоторое подмножество множества I индексов всех векторов a_i в представлении (5.44), причем $I^0 \subseteq \mathcal{J}_k$.

Составим матрицу A_k размера $m_k \times n$, строками которой служат матрицы-строки a_i^T , $i \in \mathcal{J}_k$. Если $\text{Rg } A_k = m_k \leq n$, то матрица $A_k A_k^T$ невырожденная и можно определить квадратные проекционные матрицы $P_k = A_k^T (A_k A_k^T)^{-1} A_k$ и $P_k^* = I - P_k$ порядка n .

После вычисления антиградиента $w^k = -\text{grad } f_0(x^{k-1})$ целевой функции $f_0(x)$ в точке $x^{k-1} \in \Omega$ находим

$$p^k = P_k^* w^k. \quad (5.45)$$

Отметим, что в записи (5.44) могут отсутствовать ограничения типа равенства, т.е. множество I^0 индексов может быть пусто. Возможна ситуация, в которой на k -й итерации все ограничения типа неравенства не являются активными. При этом $m_k = 0$, т.е. пусто и множество \mathcal{J}_k , а x^{k-1} — внутренняя точка множества Ω , для которой $(a_i, x^{k-1}) < b_i$ при всех $i \in I^-$. В этом случае полагаем, что P_k^* совпадает с единичной матрицей I порядка n , а в равенстве (5.45) $p^k = w^k$. Для внутренней точки антиградиент w^k задает возможное направление спуска, по которому на этой итерации можно двигаться вплоть до границы $\partial\Omega$ выпуклого множества Ω (на рис. 5.15 такая ситуация показана на плоскости). Чтобы найти точку $\tilde{x}^k \in \partial\Omega$, в выражения

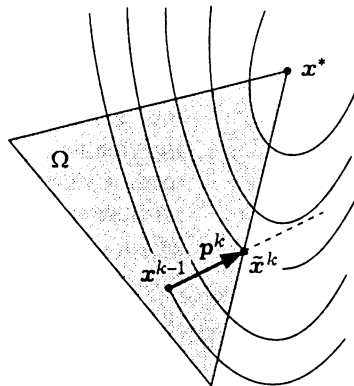


Рис. 5.15

для ограничений типа неравенства в записи (5.44) подставим $x = x^k$ из соотношений (5.43) при $P_k^* = I$:

$$b_i \geq (a_i, x^{k-1} + \kappa_k w^k) = (a_i, x^{k-1}) + \kappa_k (a_i, w^k), \quad i \in \mathcal{I}^-.$$

Как видно, при $(a_i, w^k) > 0$ выбор $\kappa_k > 0$ ограничен значением $\frac{b_i - (a_i, x^{k-1})}{(a_i, w^k)}$, иначе ограничение с этим индексом будет нарушено. С учетом всех ограничений типа неравенства, для которых $(a_i, w^k) > 0$, получим

$$\tilde{\kappa}_k = \min_{i \in \mathcal{I}^+} \frac{b_i - (a_i, x^{k-1})}{(a_i, w^k)}, \quad (5.46)$$

где $\mathcal{I}^+ = \{i \in \mathcal{I}^- : (a_i, w^k) > 0\}$.

После вычисления $\tilde{\kappa}_k$ следует выбрать такое значение $\kappa_k \in (0, \tilde{\kappa}_k]$, которому соответствует точка $x^k = x^{k-1} + \kappa_k w^k$, обеспечивающая возможно меньшее значение $f_0(x^k) < f_0(x^{k-1})$ целевой функции. Отметим, что можно выбрать $\kappa_k = \tilde{\kappa}_k$, что соответствует пересечению направления антиградиента w^k и гиперплоскостей, для которых отношение

$$\frac{b_i - (a_i, x^{k-1})}{(a_i, w^k)}$$

в представлении (5.46) оказалось наименьшим. В результате найдем точку $\tilde{x}^k = x^{k-1} + \tilde{\kappa}_k w^k \in \Omega$ (см. рис. 5.15). Но при этом возможно, что $f_0(\tilde{x}^k) > f_0(x^{k-1})$.

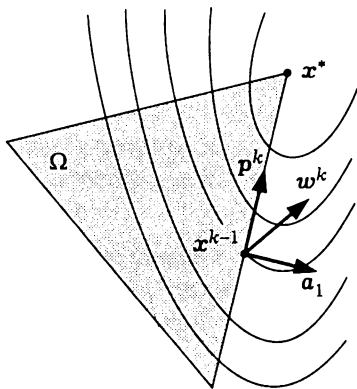


Рис. 5.16

Пусть $\mathcal{J}_k \neq \emptyset$. Если $|p^k| \geq \varepsilon_3 > 0$, то вектор p^k определяет возможное направление спуска. При движении в этом направлении не будут нарушены ограничения типа равенства и активные ограничения типа неравенства, но опять возникает опасность нарушения неактивных в точке x^{k-1} ограничений (на рис. 5.16 это показано в двумерном случае при одном активном и двух неактивных ограничениях в точке x^{k-1}). В связи с этим аналогично формуле (5.46) вычисляем

предельно допустимое для \varkappa_k значение

$$\bar{\varkappa}_k = \min_{i \in \mathcal{J}_k^+} \frac{b_i - (a_i, x^{k-1})}{(a_i, p^k)}, \quad (5.47)$$

где $\mathcal{J}_k^+ = \{i \in \mathcal{I}^- : (a_i, p^k) > 0\}$, и затем в рекуррентном соотношении (5.43) подбираем $\varkappa_k \in (0, \bar{\varkappa}_k]$ так, чтобы получить возможно меньшее значение $f_0(x^k)$ целевой функции. После выбора значения \varkappa_k и нахождения точки x^k можно переходить к следующей, $(k+1)$ -й итерации.

Если $|p^k| < \varepsilon_3$, то при $\mathcal{J}_k = \emptyset$ итерации прекращаем и полагаем $x^* \approx x^{k-1}$, а при $\mathcal{J}_k \neq \emptyset$ вычисляем вектор

$$\lambda^k = (A_k A_k^T)^{-1} A_k w^k \in \mathbb{R}^{m_k}, \quad (5.48)$$

координаты $\lambda_i^{(k)}$, $i \in \mathcal{J}$, которого являются множителями Лагранжа в функции Лагранжа

$$L_k(x, \lambda) = f_0(x) + \sum_{i \in \mathcal{J}_k} \lambda_i^{(k)} ((a_i, x) - b_i). \quad (5.49)$$

Если $\lambda_i^{(k)} \geq 0$ для всех $i \in \mathcal{J}_k \setminus \mathcal{I}^0$, т.е. для всех активных ограничений типа неравенства, то согласно теореме 4.6 это означает, что точку x^{k-1} можно приближенно считать стационарной точкой функции Лагранжа (5.49) и можно положить $x^* \approx x^{k-1}$. Но если найдутся индексы

$i \in \mathcal{J}_k \setminus \mathcal{I}^0$, для которых $\lambda_i^{(k)} < 0$, то один из них (обозначим его j) следует удалить из множества \mathcal{J}_k и для нового множества $\hat{\mathcal{J}}_k = \mathcal{J}_k \setminus \{j\}$ индексов найти проекционную матрицу \hat{P}_k^* размера $(m_k - 1) \times n$. Если при этом $\hat{\mathcal{J}}_k = \emptyset$, т.е. $m_k - 1 = 0$, то $\hat{P}_k^* = I$.

Покажем, что новый вектор $\hat{p}^k = \hat{P}_k^* w^k$ ненулевой и определяет возможное направление спуска на k -й итерации. Этот вектор можно представить в виде $\hat{p}^k = w^k - \hat{A}_k^T \hat{\lambda}^k$, где $\hat{\lambda}^k = (\hat{A}_k \hat{A}_k^T)^{-1} \hat{A}_k w^k$ и \hat{A}_k — матрица размера $(m_k - 1) \times n$, состоящая из матриц-строк a_i^T , $i \in \hat{\mathcal{J}}_k$. Аналогично для вектора $p^k = 0$ с учетом равенства (5.48) имеем

$$p^k = P_k^* w^k = w^k - A_k^T \lambda^k = w^k - \lambda_j^{(k)} a_j - \sum_{i \in \hat{\mathcal{J}}_k} \lambda_i^{(k)} a_i = 0. \quad (5.50)$$

Предположим, что $\hat{p}^k = 0$, откуда $w^k = \hat{A}_k^T \hat{\lambda}^k = \sum_{i \in \hat{\mathcal{J}}_k} \hat{\lambda}_i^{(k)} a_i$. Подставляя w^k в равенство (5.50), находим

$$\sum_{i \in \hat{\mathcal{J}}_k} \hat{\lambda}_i^{(k)} a_i - \lambda_j^{(k)} a_j - \sum_{i \in \hat{\mathcal{J}}_k} \lambda_i^{(k)} a_i = -\lambda_j^{(k)} a_j + \sum_{i \in \hat{\mathcal{J}}_k} (\hat{\lambda}_i^{(k)} - \lambda_i^{(k)}) a_i = 0,$$

т.е. нетривиальная (поскольку $\lambda_j^{(k)} \neq 0$) линейная комбинация линейно независимых векторов a_i , $i \in \mathcal{J}^k$, равна нулевому вектору. Это противоречие доказывает, что $\hat{p}^k \neq 0$.

Спуск в направлении вектора \hat{p}^k не нарушит ограничение $(a_j, x) \leq b_j$, которое при построении матрицы \hat{P}_k^* уже не рассматривается как активное, если $(a_j, \hat{p}^k) \leq 0$. Убедимся в этом, для чего умножим равенство (5.50) скалярно на вектор $\hat{P}_k^* a_j$ и запишем

$$(\hat{P}_k^* a_j, w^k) - \lambda_j^{(k)} (\hat{P}_k^* a_j, a_j) - \sum_{i \in \hat{\mathcal{J}}_k} \lambda_i^{(k)} (\hat{P}_k^* a_j, a_i) = 0. \quad (5.51)$$

Так как $(\hat{P}_k^* a_j, a_i) = 0$, $i \in \hat{\mathcal{J}}_k$, то сумма в левой части (5.51) равна нулю. Учитывая, что матрица \hat{P}_k^* симметрическая и неотрицательно определенная, из равенства (5.51) получаем

$$(\hat{P}_k^* a_j, w^k) = (a_j, \hat{P}_k^* w^k) = (a_j, \hat{p}^k) = \lambda_j^{(k)} (\hat{P}_k^* a_j, a_j) \leq 0,$$

поскольку $\lambda_j^{(k)} < 0$. Следовательно, вектор \hat{p}^k задает возможное направление спуска.

Если $|\hat{p}^k| < \varepsilon_3$, то предстоит новая проверка условия прекращения последующих итераций, приводящая либо к завершению решения задачи, либо к удалению из множества $\hat{\mathcal{J}}_k$ еще одного индекса \tilde{j} , для которого множитель Лагранжа $\hat{\lambda}_{\tilde{j}}^{(k)} < 0$, и построению очередной проекционной матрицы и т.д. Если же $|\hat{p}^k| \geq \varepsilon_3$, то для определения направления спуска, которое задано вектором \hat{p}^k , с помощью формул (5.43) и (5.47) проводим описанную выше процедуру выбора значения κ_k и нахождения точки x^k , что позволяет затем перейти к $(k+1)$ -й итерации.

Пример 5.11. Решим методом проекции антиградиента задачу квадратичного программирования

$$\begin{cases} f_0(x_1, x_2) = 2x_1^2 + 2x_2^2 - 2x_1x_2 - 4x_1 - 6x_2 \rightarrow \min; \\ x_1 + x_2 - 2 \leq 0, \quad x_1 + 5x_2 - 5 \leq 0, \quad -x_1 \leq 0, \quad -x_2 \leq 0. \end{cases}$$

Целевую функцию представим в виде $f_0(x) = \frac{1}{2}(Qx, x) + (b, x)$, где

$$Q = \begin{pmatrix} 4 & -2 \\ -2 & 4 \end{pmatrix}, \quad b = \begin{pmatrix} -4 \\ -6 \end{pmatrix}.$$

Вычислим антиградиент этой функции:

$$\begin{aligned} w &= -\text{grad } f_0(x) = -Qx - b = \\ &= -\begin{pmatrix} 4 & -2 \\ -2 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} -4 \\ -6 \end{pmatrix} = \begin{pmatrix} 4 - 4x_1 + 2x_2 \\ 6 + 2x_1 - 4x_2 \end{pmatrix} \quad (5.52) \end{aligned}$$

и в соответствии с записью (5.44) обозначим $a_1 = (1 \ 1)^T$, $b_1 = 2$, $a_2 = (1 \ 5)^T$, $b_2 = 5$, $a_3 = (-1 \ 0)^T$, $a_4 = (0 \ -1)^T$ и $b_3 = b_4 = 0$. В качестве начальной точки возьмем $x^0 = (0, 0)$, а параметр точности поиска $\varepsilon_3 = 0,1$.

Первая итерация. В точке x^0 в соответствии с равенствами (5.52) имеем $w^1 = (4 \ 6)^T$. В этой точке активны лишь ограничения неотрицательности параметров оптимизации x_1 и x_2 , т.е. $\mathcal{J}_1 = \{3, 4\}$.

Поэтому

$$A_1 = (a_3 \ a_4)^T = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}, \quad A_1 A_1^T = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

$$P_1 = A_1^T (A_1 A_1^T)^{-1} A_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad P_1^* = I - P_1 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

Так как $p^1 = P_1^* w^1 = (0 \ 0)^T$, то по формуле (5.48) вычисляем

$$\lambda^1 = (A_1 A_1^T)^{-1} A_1 w^1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 4 \\ 6 \end{pmatrix} = \begin{pmatrix} -4 \\ -6 \end{pmatrix}.$$

В данной задаче ограничения типа равенства отсутствуют, так что отрицательными оказались оба множителя Лагранжа при активных ограничениях типа неравенства. Удалим из множества \mathcal{J}_1 индекс $i = 4$ и для нового множества $\hat{\mathcal{J}}_1 = \{3\}$ найдем $\hat{A}_1 = (-1 \ 0)$, $\hat{A}_1 \hat{A}_1^T = (-1 \ 0)(-1 \ 0)^T = 1$,

$$\hat{P}_1 = \hat{A}_1^T (\hat{A}_1 \hat{A}_1^T)^{-1} \hat{A}_1 = \begin{pmatrix} -1 \\ 0 \end{pmatrix} \cdot 1 \cdot (-1 \ 0) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix},$$

$$\hat{P}_1^* = I - \hat{P}_1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}.$$

Теперь получим

$$\hat{p}^1 = \hat{P}_1^* w^1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 4 \\ 6 \end{pmatrix} = \begin{pmatrix} 0 \\ 6 \end{pmatrix},$$

причем $|\hat{p}^1| = 6 > \varepsilon_3 = 0,1$, так что вектор \hat{p}^1 определяет возможное направление спуска.

Для ограничений, неактивных в точке $x^0 = (0, 0)$, вычислим $(a_1, \hat{p}^1) = 6 > 0$, $(a_2, \hat{p}^1) = 30 > 0$. При этом

$$\frac{b_1 - (a_1, x^0)}{(a_1, p^1)} = \frac{1}{3}, \quad \frac{b_2 - (a_2, x^0)}{(a_2, p^1)} = \frac{1}{6}.$$

В соответствии с формулой (5.47) $\bar{\pi}_1 = 1/6$, т.е. при спуске в направлении вектора \hat{p}^1 первым будет нарушено ограничение с индексом $i = 2$ (рис. 5.17).

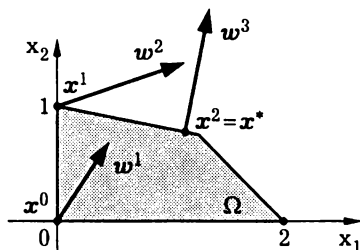


Рис. 5.17

Подставим $x = x^0 + \kappa \hat{p}^1$ в матричное представление целевой функции:

$$f_0(x) = \frac{\kappa^2}{2} (Q \hat{p}^1, \hat{p}^1) + \kappa (b, \hat{p}^1) = \varphi_1(\kappa).$$

Функция $\varphi_1(\kappa)$ принимает минимальное значение при

$$\kappa_1^* = -\frac{(b, \hat{p}^1)}{(Q \hat{p}^1, \hat{p}^1)} = \frac{1}{4}.$$

Так как $\kappa_1^* > \bar{\kappa}_1$, то полагаем $\kappa_1 = \bar{\kappa}_1 = 1/6$ и находим $x^1 = x^0 + \kappa_1 \hat{p}^1 = (0, 1)$ (см. рис. 5.17).

Вторая итерация. В точке $x^1 = (0, 1)$ согласно равенствам (5.52) находим $w^2 = (6 \ 2)^T$. В этой точке активными являются ограничения с индексами $i = 2$ и $i = 3$, т.е. $\mathcal{J}_2 = \{2, 3\}$,

$$A_2 = (a_2, a_3)^T = \begin{pmatrix} 1 & 5 \\ -1 & 0 \end{pmatrix}, \quad P_2^* = I - A_2^T (A_2 A_2^T)^{-1} A_2 = \Theta_2$$

и $p^2 = 0$. По формуле (5.48) вычислим $\lambda^2 = (A_2 A_2^T)^{-1} A_2 w^2 = (2/5 \ -28/5)^T$. Из множества \mathcal{J}_2 удаляем индекс $i = 3$, соответствующий отрицательному множителю Лагранжа, и получаем $\hat{A}_2 = (1 \ 5)$, $\hat{A}_2 \hat{A}_2^T = (1 \ 5)(1 \ 5)^T = 26$,

$$\hat{P}_2^* = I - \hat{A}_2^T (\hat{A}_2 \hat{A}_2^T)^{-1} \hat{A}_2 = \frac{1}{26} \begin{pmatrix} 25 & -5 \\ -5 & 1 \end{pmatrix}$$

и $\hat{p}^2 = \hat{P}_2^* w^2 = \frac{14}{13} (5 \ -1)^T$, причем $|\hat{p}^2| = \sqrt{26} > \varepsilon_3 = 0,1$. Следовательно, вектор \hat{p}^2 определяет возможное направление спуска. Для

дальнейших вычислений на этой итерации длина вектора \hat{p}^2 не имеет значения, поэтому для упрощения примем $\hat{p}^2 = (5 \ -1)^T$.

Для ограничений, неактивных в точке $x^1 = (0, 1)$, вычисляем $(a_1, \hat{p}^2) = 4 > 0$, $(a_4, \hat{p}^2) = 1 > 0$. При этом

$$\frac{b_1 - (a_1, x^2)}{(a_1, p^2)} = \frac{1}{4}, \quad \frac{b_4 - (a_4, x^1)}{(a_4, p^2)} = 1.$$

Таким образом, в соответствии с формулой (5.47) $\bar{\kappa}_2 = 1/4$, т.е. при спуске в направлении вектора p^2 первым будет нарушено ограничение с индексом $i = 1$ (см. рис. 5.17).

Подставляя $x = x^1 + \kappa \hat{p}^2$ в матричное выражение для целевой функции и учитывая симметричность матрицы Q , находим

$$f_0(x) = \frac{\kappa^2}{2} (Q \hat{p}^2, \hat{p}^2) + \kappa ((Q x^1, \hat{p}^2) + (b, \hat{p}^2) + (b, x^1)) = \varphi_2(\kappa).$$

Функция $\varphi_2(\kappa)$ достигает минимального значения при

$$\kappa_2^* = - \frac{(Q x_1, \hat{p}^2) + (b, \hat{p}^2)}{(Q \hat{p}^2, \hat{p}^2)} = \frac{7}{31} < \frac{1}{4} = \bar{\kappa}_2.$$

Поэтому принимаем $\kappa_2 = \frac{7}{31}$ и получаем $x^2 = x^1 + \kappa_2 \hat{p}^2 = \left(\frac{35}{31}, \frac{24}{31}\right)$.

Третья итерация. В точке x^2 по формуле (5.52) вычисляем $w^3 = \frac{32}{31} (1 \ 5)^T$ и устанавливаем, что активным является лишь ограничение с индексом $i = 2$, т.е. $\mathcal{J}_3 = \{2\}$. Поэтому $A_3 = a_2^T = (1 \ 5) = \hat{A}_2$, $(A_3 A_3^T)^{-1} = 26$, $P_3^* = \hat{P}_2^*$ и

$$p^3 = P_3^* w^3 = \frac{1}{26} \begin{pmatrix} 25 & -5 \\ -5 & 1 \end{pmatrix} \cdot \frac{32}{31} \begin{pmatrix} 1 \\ 5 \end{pmatrix} = 0.$$

Так как $|p^3| = 0 < \varepsilon_3 = 0,1$, то по формуле (5.48) находим $\lambda^3 = (A_3 A_3^T)^{-1} A_3 w^3 = \frac{1}{26} (1 \ 5) \cdot \frac{32}{31} (1 \ 5)^T = \frac{32}{31} > 0$. Таким образом, единственный множитель Лагранжа в функции Лагранжа (5.49), имеющей в данном случае вид

$$L_3(x, \lambda) = f_0(x) + \frac{32}{31} ((a_2, x) - b_2),$$

является положительным. Поскольку на этой итерации $p^3 = 0$, точка $x^2 = (35/31, 24/31)$ является стационарной для функции Лагранжа. Квадратичная целевая функция $f_0(x)$ с положительно определенной матрицей Q сильно выпуклая. Поэтому $x^2 = x^*$ — точка наименьшего значения целевой функции при заданных ограничениях (см. рис. 5.17). В этой точке целевая функция принимает значение $f_0(x^*) = \frac{6882}{961} \approx 7,1613$.

Нелинейные ограничения. Если среди ограничений в задаче оптимизации есть нелинейные, то поиск минимума целевой функции $f_0(x)$ на допустимом множестве Ω усложняется. Предположим для определенности, что Ω задано ограничениями $g_i(x) \leq 0$, $i = 1, m$, где $g_i(x)$ — дифференцируемые функции. На k -й итерации известна точка $x^{k-1} \in \Omega$. Составим множество \mathcal{I}_k индексов i , для которых i -е ограничение в точке x^{k-1} активно, т.е. $g_i(x^{k-1}) = 0$. Предполагая, что матрицы-строки $(\text{grad } g_i(x^{k-1}))^T$, $i \in \mathcal{I}_k$, линейно независимы, составим из них матрицу \tilde{A}_k . Тогда проекционная матрица $\tilde{P}_k^* = I - \tilde{A}_k^T (\tilde{A}_k \tilde{A}_k^T)^{-1} \tilde{A}_k$ спроектирует антиградиент $w^k = -\text{grad } f_0(x^{k-1})$ целевой функции в точке x^{k-1} на направление, ортогональное всем градиентам $\text{grad } g_i(x^{k-1})$, $i \in \mathcal{I}_k$. Однако в общем случае может оказаться, что это направление не будет допустимым направлением спуска, поскольку оно будет касаться в точке x^{k-1} границы $\partial\Omega$ множества Ω (на рис. 5.18 эта ситуация представлена на плоскости в случае одного активного ограничения в точке x^{k-1}). Ясно, что любое продвижение точки в найденном направлении спуска

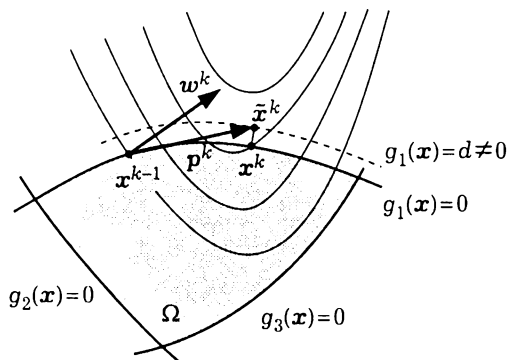


Рис. 5.18

может вывести ее из допустимого множества. В этой ситуации шаг спуска приходится сочетать с проектированием точки \tilde{x}^k , полученной в результате этого спуска, на множество Ω .

Рассмотрим на конкретном примере некоторые особенности поиска точки минимума целевой функции в случае, когда допустимое множество Ω задано одним нелинейным ограничением типа неравенства, т.е. $\Omega = \{x \in \mathbb{R}^n: g(x) \leq 0\}$.

Пример 5.12. Решим задачу выпуклого программирования

$$\begin{cases} f_0(x_1, x_2) = 2x_1^2 - 4x_1 + x_2^2 - 2x_2 \rightarrow \min; \\ x_1 + x_2^2 - 2x_2 \leq 0. \end{cases}$$

В данном случае целевая функция является квадратичной, и ее можно записать в виде $f_0(x) = \frac{1}{2}(Qx, x) + (b, x)$, где

$$Q = \begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix}, \quad b = \begin{pmatrix} -4 \\ -2 \end{pmatrix}.$$

Единственное ограничение задачи имеет вид $g(x) \leq 0$, где $g(x_1, x_2) = x_1 + x_2^2 - 2x_2$. В качестве начальной возьмем точку $x^0 = (0, 0)$, а в условии прекращения поиска $|x^k - x^{k-1}| < \varepsilon_1$ положим $\varepsilon = 0,01$.

Вычислим антиградиент целевой функции:

$$\begin{aligned} w &= -\text{grad } f_0(x) = -Qx - b = \\ &= -\begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} -4 \\ -2 \end{pmatrix} = \begin{pmatrix} 4 - 4x_1 \\ 2 - 2x_2 \end{pmatrix} \end{aligned}$$

и градиент левой части ограничения

$$\text{grad } g(x) = (1 \quad 2x_2 - 2)^T. \quad (5.53)$$

Остановимся подробнее на первой итерации при решении этой задачи методом проекции антиградиента.

1. Поскольку $w^1 = -\text{grad } f_0(x^{k-1}) = (4 \quad 2)^T$ и $\text{grad } g(x^0) = (1 \quad -2)^T$, то $(w^1, \text{grad } g(x^0)) = 0$, т.е. векторы w^1 и $\text{grad } g(x^0)$ ортогональны. Значит, нет необходимости вычислять проекционную матрицу P_1^* , которая в данном случае является единичной, а $p^1 = P_1^* w^1 = w^1 = (4 \quad 2)^T$.

2. С помощью исчерпывающего спуска из точки x^0 в направлении вектора p^1 находим значение

$$\kappa_1 = \frac{(w^1, p^1)}{(Qp^1, p^1)} = \frac{5}{18}$$

и точку $\tilde{x}^1 = x^0 + \kappa_1 p^1 = \left(\frac{10}{9}, \frac{5}{9}\right)$. Продвижение из точки x^0 в направлении вектора p^1 выводит за пределы допустимого множества, так как идет по касательной к кривой Γ , заданной уравнением $g(x) = 0$, в точке x^0 . В частности, $g(\tilde{x}^1) = \frac{25}{81} > 0$ и $\tilde{x}^1 \notin \Omega$.

3. Спроектируем точку $\tilde{x}^1 \notin \Omega$ на допустимое множество Ω , т.е. найдем точку x^1 кривой Γ , наименее удаленную от точки \tilde{x}^1 . Эту точку можно найти, решая задачу минимизации функции $\Phi(z) = \min |z - \tilde{x}^k|^2$. Один из способов решения такой задачи — метод последовательных приближений, основанный на линеаризации ограничения $g(x) = 0$ в окрестности текущей точки z_0 согласно формуле $(\text{grad } f(z_0), x - z_0) = 0$ и построении приближения с помощью проектирования на гиперплоскость по формуле (5.22). В результате приходим к рекуррентному соотношению

$$\hat{x}_s^1 = \hat{x}_{s-1}^1 - g(\hat{x}_{s-1}^1) \frac{\text{grad } g(\hat{x}_{s-1}^1)}{|\text{grad } g(\hat{x}_{s-1}^1)|^2}, \quad s \in \mathbb{N}.$$

В качестве начальной точки \hat{x}_0^1 итерационного процесса можно принять точку \tilde{x}^1 , а условием прекращения процесса последовательных приближений можно считать выполнение неравенства $|\hat{x}_s^1 - \hat{x}_{s-1}^1| < \delta$, где $\delta > 0$ — заданный параметр точности. При выполнении этого неравенства полагаем $x^1 = \hat{x}_s^1$ и $f(x^1) = f(\hat{x}_s^1)$. Выбирая $\delta = 0,01$, получаем $x^1 = (0,9211, 0,7191)$, при этом $g(x^1) \approx 0,0003$.

4. Проверяем условие прекращения итераций:

$$|x^1 - x^0| = \sqrt{(0,9211)^2 + (0,7191)^2} \approx 1,168 > \varepsilon = 0,01.$$

Так как требуемая точность поиска еще не достигнута, поиск точки $x^* \in \Omega$ минимума целевой функции $f(x)$ на множестве Ω следует продолжить. Для этого переходим к п. 1, заменив точку x^0 на найденную на первой итерации точку x^1 .

Результаты расчетов на первой и последующих трех итерациях приведены в табл. 5.3, траектория поиска точки x^* представлена на рис. 5.19.

Таблица 5.3

k	x^k	$f(x^k)$	$ x^k - x^{k-1} $
1	(0,9211, 0,7191)	-2,9086	1,168
2	(0,9976, 0,9512)	-2,9976	0,244
3	(1,0000, 0,9995)	-3,0000	0,048
4	(1,0000, 1,0000)	-3,0000	0,0005

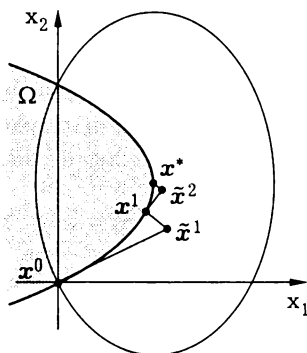


Рис. 5.19

Таким образом, результат четвертой итерации с заданной точностью совпал с точным решением задачи $x^* = (1, 1)$. #

При реализации изложенного алгоритма может возникнуть ситуация, когда на очередной k -й итерации будет выполняться неравенство $f_0(x^k) \geq f_0(x^{k-1})$, т.е. будет нарушено условие релаксационности последовательности $\{x^k\}$. В этом случае можно использовать модифицированный алгоритм, в котором выбор значения α_k проводится с помощью метода дробления шага, а не с помощью исчерпывающего спуска.

Проектирование сопряженных направлений. Эффективность поиска точки $x^* \in \Omega$ минимума дифференцируемой целевой функции $f_0(x)$, $x \in \mathbb{R}^n$, на допустимом множестве Ω можно повысить, если на каждой k -й итерации возможное направление спуска определять не по антиградиенту $w^k = -\text{grad } f_0(x^{k-1})$ в точке $x^{k-1} \in \Omega$, а по одному

из сопряженных направлений, которые находят последовательно в процессе итераций.

Рассмотрим допустимое множество вида

$$\Omega = \{x \in \mathbb{R}^n: (a_i, x) = b_i, i = \overline{1, m}\},$$

где векторы $a_i \in \mathbb{R}^n$ линейно независимы. Согласно теореме 5.2, определять точку минимума функции $f_0(x)$ на множестве Ω можно, решая задачу безусловной минимизации функции $\varphi(y) = f_0(x^0 + P^*y)$, где $x^0 \in \Omega$ — начальная точка; $P^* = I - A^T(AA^T)^{-1}A$ — проекционная матрица порядка n ; I — единичная матрица того же порядка; A — матрица размера $m \times n$, составленная из матриц-строк $a_i^T, i = \overline{1, m}$. Опишем метод сопряженных направлений применительно к поиску минимума функции $\varphi(y)$.

Выбираем начальную точку $y^0 = 0$, соответствующую заданной начальной точке x^0 . На первой итерации направление спуска находим по формуле $p^1 = -\text{grad } \varphi(y^0)$, на последующих — по формуле

$$p^k = -\text{grad } \varphi(y^{k-1}) + \frac{|\text{grad } \varphi(y^{k-1})|^2}{|\text{grad } \varphi(y^{k-2})|^2} p^{k-1}. \quad (5.54)$$

Спуск по выбранному направлению на каждой итерации осуществляем по стандартному рекуррентному соотношению

$$y^k = y^{k-1} + \kappa_k p^k, \quad k \in \mathbb{N}, \quad (5.55)$$

где значение κ_k можно выбирать либо с помощью исчерпывающего спуска, либо методом дробления шага. Условием прекращения итераций может быть выполнение неравенства $|p^k| < \varepsilon_3$, где $\varepsilon_3 > 0$ — параметр точности поиска.

Описанную последовательность действий можно уточнить, используя конкретный вид функции $\varphi(y) = f_0(x^0 + P^*y)$. В данном случае

$$\text{grad } \varphi(y) = P^* \text{grad } f_0(x^0 + P^*y).$$

Поэтому $p^1 = -P^* \text{grad } f_0(x^0)$. Вычитая из равенства $x^k = x^0 + P^*y^k$ равенство $x^{k-1} = x^0 + P^*y^{k-1}$, получаем $x^k - x^{k-1} = P^*(y^k - y^{k-1}) = \kappa_k P^* p^k$. Таким образом,

$$x^k = x^{k-1} + \kappa_k P^* p^k, \quad k \in \mathbb{N}. \quad (5.56)$$

Для направления спуска p^k при $k > 1$, заменяя в формуле (5.54) градиенты функции $\varphi(y)$ соответствующими выражениями, находим

$$p^k = -P^* \text{grad } f_0(x^{k-1}) + \frac{|P^* \text{grad } f_0(x^{k-1})|^2}{|P^* \text{grad } f_0(x^{k-2})|^2} p^{k-1}. \quad (5.57)$$

Подход, в котором используется проектирование сопряженных направлений, можно обобщить на случай, когда допустимое множество Ω задано линейными ограничениями типа неравенства и линейными ограничениями типа равенства. Можно показать, что в этом случае для поиска точки минимума квадратичной функции требуется не более n итераций.

5.6. Метод возможных направлений

Метод проекции антиградиента, а также проектирование сопряженных направлений применимы только в случае, когда в задаче оптимизации присутствуют лишь линейные ограничения. Для этих методов характерно, что на очередной итерации движение из текущей точки в выбранном направлении спуска приводит к нарушению нелинейных ограничений. Чтобы указанные методы можно было применять в случае нелинейных ограничений, необходимо их модифицировать, используя такой алгоритм выбора возможного направления спуска, что перемещение точки по этому направлению на некоторое расстояние не будет приводить к нарушению ограничений задачи.

Сначала рассмотрим задачу оптимизации, в которой целевая функция $f_0(x)$ непрерывно дифференцируема и ограничена снизу на допустимом множестве Ω вида

$$\Omega = \{x \in \mathbb{R}^n: g_i(x) \leq 0, i = \overline{1, m}\}, \quad (5.58)$$

где $g_i(x)$ — непрерывно дифференцируемые функции. На k -й итерации известна точка $x^{k-1} \in \Omega$. Обозначим через \mathcal{I}_k множество индексов активных ограничений, т.е. тех индексов i , для которых $g_i(x^{k-1}) = 0$. Отметим, что если $\mathcal{I}_k = \emptyset$, то x^{k-1} — внутренняя точка множества Ω .

Рассмотрим вспомогательную задачу оптимизации, в которой необходимо найти минимальное значение функции

$$\eta(p) = \max \left\{ (\text{grad } f_0(x^{k-1}), p), \max_{i \in \mathcal{I}_k} (\text{grad } g_i(x^{k-1}), p) \right\} \quad (5.59)$$

на множестве \mathcal{W}_k векторов $\mathbf{p} = (p_1 \dots p_n)^T$, удовлетворяющих ограничениям $|p_j| \leq 1, j = \overline{1, n}$.

Допустимое множество \mathcal{W}_k задачи оптимизации не пусто. Оно содержит нулевой вектор, на котором значение целевой функции равно нулю. Поэтому точная нижняя грань целевой функции $\eta(\mathbf{p})$ на множестве \mathcal{W}_k не превышает нуля. Поскольку множество \mathcal{W}_k замкнуто и ограничено, а целевая функция непрерывна, задача оптимизации имеет решение и функция η достигает наименьшего значения $\eta_k \leq 0$ в некоторой точке \mathbf{p}^k . Отметим, что рассматриваемую задачу оптимизации введением дополнительного переменного η можно представить как задачу линейного программирования

$$\begin{cases} \eta \rightarrow \min; \\ (\text{grad } f_0(\mathbf{x}^{k-1}), \mathbf{p}) \leq \eta; \\ (\text{grad } g_i(\mathbf{x}^{k-1}), \mathbf{p}) \leq \eta, \quad i \in \mathcal{I}_k; \\ p_j \leq 1, \quad j = \overline{1, n}; \quad -p_j \leq 1, \quad j = \overline{1, n}, \end{cases} \quad (5.60)$$

которую можно решить, например, симплекс-методом.

Предположим, что эта задача решена и получена точка минимума $\mathbf{p}^k \in \mathcal{W}_k$, в которой $\eta_k < 0$. Тогда вектор \mathbf{p}^k задает возможное направление спуска из точки $\mathbf{x}^{k-1} \in \Omega$. Действительно, из вида целевой функции (5.59) заключаем, что выполняются неравенства

$$(\text{grad } f_0(\mathbf{x}^{k-1}), \mathbf{p}^k) \leq \eta_k < 0; \quad (5.61)$$

$$(\text{grad } g_i(\mathbf{x}^{k-1}), \mathbf{p}^k) \leq \eta_k < 0, \quad i \in \mathcal{I}_k. \quad (5.62)$$

Отсюда, в частности, следует, что $\mathbf{p}^k \neq 0$.

Для каждой функции $g_i(\mathbf{x})$, $i \in \mathcal{I}_k$, учитывая ее дифференцируемость и равенство $g_i(\mathbf{x}^{k-1}) = 0$, можем записать

$$\begin{aligned} g_i(\mathbf{x}^{k-1} + \alpha \mathbf{p}^k) &= g_i(\mathbf{x}^{k-1} + \alpha \mathbf{p}^k) - g_i(\mathbf{x}^{k-1}) = \\ &= (\text{grad } g_i(\mathbf{x}^{k-1}), \alpha \mathbf{p}^k) + o(\alpha). \end{aligned}$$

Отсюда, а также из соотношений (5.62) заключаем, что для любого индекса $i \in \mathcal{I}_k$ найдется такое $\alpha_i > 0$, что при $\alpha \in (0, \alpha_i)$ выполняется неравенство $g_i(\mathbf{x}^{k-1} + \alpha \mathbf{p}^k) < 0$. Но это же верно и для каждой функции $g_l(\mathbf{x})$ при $l \notin \mathcal{I}_k$, поскольку эта функция непрерывна и $g_l(\mathbf{x}^{k-1}) < 0$.

Таким образом, если $\alpha_0 = \min_{i=1, \overline{m}} \alpha_i > 0$, то $g_i(x^{k-1} + \alpha p^k) < 0, i = \overline{1, m}$, $\alpha \in (0, \alpha_0)$. Это означает, что вектор p^k определяет допустимое направление в точке $x^{k-1} \in \Omega$ относительно множества Ω .

Повторяя рассуждения для целевой функции $f_0(x)$, приходим к выводу, что число $\alpha_0 > 0$, если необходимо, можно уменьшить настолько, что при $\alpha \in (0, \alpha_0)$ будет выполняться и неравенство

$$f_0(x^{k-1} + \alpha p^k) - f_0(x^{k-1}) < 0.$$

Поэтому вектор p^k задает возможное направление спуска из точки $x^{k-1} \in \Omega$ на множестве Ω .

Перемещение в этом направлении приводит на k -й итерации в точку

$$x^k = x^{k-1} + \kappa_k p^k, \quad \kappa_k \in (0, \overline{\kappa}_k), \quad (5.63)$$

где $\overline{\kappa}_k > 0$ — наибольшее значение κ , при котором верно соотношение $x^{k-1} + \tau p^k \in \Omega, \tau \in (0, \kappa]$. Если Ω — выпуклое множество, параметр $\overline{\kappa}_k > 0$ можно определить формулой

$$\overline{\kappa}_k = \max \{ \kappa > 0: x^{k-1} + \kappa p^k \in \Omega \}. \quad (5.64)$$

Отметим, что в случае неограниченного множества Ω (в том числе и выпуклого) для некоторых направлений спуска значение $\overline{\kappa}_k$ может быть бесконечно.

Различные способы выбора значения κ_k приводят к различным вариантам описанного выше метода, который обычно называют *методом возможных направлений*. Один из способов выбора κ_k состоит в минимизации функции $\varphi_k(\kappa) = f_0(x^{k-1} + \kappa p^k)$ в полуинтервале $(0, \overline{\kappa}_k]$, для чего можно использовать методы одномерной оптимизации. Для целевой функции, удовлетворяющей условию

$$|\text{grad } f_0(x) - \text{grad } f_0(y)| \leq L|x - y|, \quad x, y \in \Omega,$$

где $L > 0$, полагают $\kappa_k = \min \left\{ \overline{\kappa}_k, \frac{|\eta_k|}{L} \right\}$. Наконец, если оценка значения $\overline{\kappa}_k$ затруднена, можно применить *метод дробления шага*: выбрать некоторое исходное значение κ_0 , найти при $\kappa_k = \kappa_0$ по формуле (5.63) точку x^k и, если $x^k \notin \Omega$, то дроблением значения κ_k добиться выполнения условия $x^k \in \Omega$. После этого нужно проверить условие $f_0(x^k) < f_0(x^{k-1})$ и, если оно нарушено, продолжить дро-

бление значения \varkappa_k до тех пор, пока это условие не будет выполнено. Если множество Ω не является выпуклым, необходимо дополнительно проверять, не нарушается ли при последующем дроблении условие $x^k \in \Omega$.

Вернемся к задаче минимизации функции (5.59) и рассмотрим случай, когда ее решением является точка p^k со значением $\eta_k = \eta(p^k) = 0$. Равенство $\eta_k = 0$ можно рассматривать как необходимое условие достижения в точке $x^{k-1} \in \Omega$ минимума целевой функции $f_0(x)$ на множестве Ω . Действительно, если x^{k-1} — точка минимума $f_0(x)$ на Ω , то в силу теоремы 4.6 существуют неотрицательные и не все равные нулю множители Лагранжа λ_0 и μ_i , $i \in \mathcal{I}_k$, для которых выполнено равенство

$$\lambda_0 \operatorname{grad} f_0(x^*) + \sum_{i \in \mathcal{I}_k} \mu_i \operatorname{grad} g_i(x^*) = 0. \quad (5.65)$$

Из этого равенства, учитывая соотношения (5.61), (5.62) и неотрицательность множителей Лагранжа, заключаем, что

$$\begin{aligned} 0 &= \left(\lambda_0 \operatorname{grad} f_0(x^*) + \sum_{i \in \mathcal{I}_k} \mu_i \operatorname{grad} g_i(x^*), p^k \right) = \\ &= \lambda_0 (\operatorname{grad} f_0(x^*), p^k) + \sum_{i \in \mathcal{I}_k} \mu_i (\operatorname{grad} g_i(x^*), p^k) \leq \eta_k \left(\lambda_0 + \sum_{i \in \mathcal{I}_k} \mu_i \right). \end{aligned}$$

Так как коэффициент при η_k положителен, то $\eta_k \geq 0$. Но выше установлено, что $\eta_k \leq 0$. Значит, $\eta_k = 0$.

Итак, условие $\eta_k = 0$ является необходимым для того, чтобы точка x^{k-1} была точкой минимума $f_0(x)$ на множестве Ω . Можно показать, что условие $\eta_k = 0$ будет достаточным при дополнительном предположении, что функции $f_0(x)$ и $g_i(x)$, $i = 1, m$, являются выпуклыми, а множество Ω имеет внутренние точки.

Проведенные рассуждения показывают, что в качестве условия прекращения итераций в методе возможных направлений можно принять выполнение неравенства $|\eta_k| < \varepsilon_\eta$, где $\varepsilon_\eta > 0$ — заданный параметр точности поиска.

Пример 5.13. Решим методом возможных направлений уже рассмотренную в примере 5.11 задачу квадратичного программирования

$$\begin{cases} f_0(x_1, x_2) = 2x_1^2 + 2x_2^2 - 2x_1x_2 - 4x_1 - 6x_2 \rightarrow \min; \\ x_1 + x_2 - 2 \leq 0, \quad x_1 + 5x_2 - 5 \leq 0, \quad -x_1 \leq 0, \quad -x_2 \leq 0. \end{cases}$$

Как и в примере 5.11, целевую функцию представим в виде $f_0(\mathbf{x}) = \frac{1}{2}(\mathbf{Q}\mathbf{x}, \mathbf{x}) + (\mathbf{b}, \mathbf{x})$, $\mathbf{x} \in \mathbb{R}^2$, где

$$\mathbf{Q} = \begin{pmatrix} 4 & -2 \\ -2 & 4 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} -4 \\ -6 \end{pmatrix}.$$

В соответствии с равенствами (5.52) ее градиент имеет вид

$$\text{grad } f_0(\mathbf{x}) = \begin{pmatrix} 4x_1 - 2x_2 - 4 \\ -2x_1 + 4x_2 - 6 \end{pmatrix}. \quad (5.66)$$

Положим $\mathbf{x} = (x_1, x_2)$ и $g_1(\mathbf{x}) = x_1 + x_2 - 2$, $g_2(\mathbf{x}) = x_1 + 5x_2 - 5$, $g_3(\mathbf{x}) = -x_1$, $g_4(\mathbf{x}) = -x_2$ (это функции, задающие левые части ограничений задачи типа неравенства). Запишем градиенты этих функций: $\text{grad } g_1(\mathbf{x}) = (1 \ 1)^T$, $\text{grad } g_2(\mathbf{x}) = (1 \ 5)^T$, $\text{grad } g_3(\mathbf{x}) = (-1 \ 0)^T$, $\text{grad } g_4(\mathbf{x}) = (0 \ -1)^T$. Допустимое множество Ω , заданное линейными ограничениями, является выпуклым. Его изображение в плоскости x_1Ox_2 (рис. 5.20) показывает, что оно ограничено. Выберем в качестве начальной точку $\mathbf{x}^0 = (0, 0)$ и параметр точности поиска $\varepsilon_\eta = 0,01$.

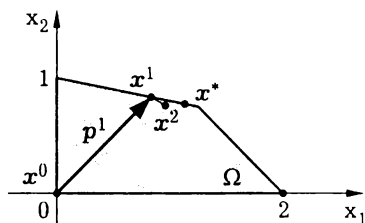


Рис. 5.20

Первая итерация. В точке \mathbf{x}^0 в соответствии с формулой (5.66) имеем $\text{grad } f_0(\mathbf{x}^0) = (-4 \ -6)^T$, а активными являются лишь ограничения неотрицательности переменных x_1 и x_2 , т.е. множество индексов $\mathcal{I}_1 = \{3, 4\}$. Поэтому для поиска направления спуска из точки \mathbf{x}^0 , определяемого вектором $\mathbf{p}^1 = (p_1^{(1)} \ p_2^{(1)})^T$, следует решить задачу (5.60), т.е. минимизировать переменное η на множестве, заданном неравенствами

$$-4p_1 - 6p_2 \leq \eta, \quad -p_1 \leq \eta, \quad -p_2 \leq \eta, \quad |p_1| \leq 1, \quad |p_2| \leq 1. \quad (5.67)$$

Наименьшее возможное значение η ограничено в (5.67) вторым и третьим неравенствами $-p_1 \leq \eta$ и $-p_2 \leq \eta$ при максимально допустимых значениях $p_1 = p_2 = 1$, т.е. $\eta = -1$. При этом первое неравенство в (5.67) также будет выполнено. Таким образом, $p^1 = (1 \ 1)^T$, причем этот вектор определен однозначно.

Ясно, что при движении точки $x = x^0 + \kappa p^1$ в направлении вектора p^1 ограничения $g_3(x) \leq 0$ и $g_4(x) \leq 0$ неотрицательности переменных не будут нарушены. Найдем, при каких значениях κ выполняются ограничения $g_1(x) \leq 0$ и $g_2(x) \leq 0$. Учитывая вид функций $g_1(x)$ и $g_2(x)$, запишем

$$g_1(x) = x_1^{(0)} + \kappa p_1^{(1)} + x_2^{(0)} + \kappa p_2^{(1)} - 2 = 2\kappa - 2 \leq 0,$$

$$g_2(x) = x_1^{(0)} + \kappa p_1^{(1)} + 5x_2^{(0)} + 5\kappa p_2^{(1)} - 5 = 6\kappa - 5 \leq 0.$$

Отсюда следует, что $\kappa \leq \bar{\kappa}_1 = \min \{1, 5/6\} = 5/6$.

Значение κ_1 выберем, минимизируя на отрезке $[0, \bar{\kappa}_1]$ функцию $\varphi_1(\kappa) = f_0(x^0 + \kappa p^1)$. Учитывая вид функции $f_0(x)$ и найденный вектор p^1 , находим

$$\varphi_1(\kappa) = \frac{1}{2} (Q(x^0 + \kappa p^1), x^0 + \kappa p^1) + (b, x^0 + \kappa p^1) = 2\kappa^2 - 10\kappa.$$

Минимума на отрезке $[0, 5/6]$ функция $\varphi_1(\kappa)$ достигает при $\kappa_1 = \bar{\kappa}_1 = 5/6$. Значит, $x^1 = x^0 + \kappa_1 p^1 = (5/6, 5/6)$ и $f_0(x^1) = -125/18 \approx -6,9444$ (см. рис. 5.20).

Вторая итерация. В точке x^1 по формуле (5.66) находим $\text{grad } f_0(x^1) = (-7/3 \ -13/3)^T$. Активным является лишь ограничение $g_2(x) \leq 0$, так что множество индексов \mathcal{I}_2 содержит лишь один номер 2. Следовательно, чтобы найти вектор $p^2 = (p_1^{(2)} \ p_2^{(2)})^T$, задающий направление спуска из точки x^1 , следует минимизировать переменное η на множестве \mathcal{W}_2 , заданном неравенствами

$$-\frac{7}{3}p_1 - \frac{13}{3}p_2 \leq \eta, \quad p_1 + 5p_2 \leq \eta, \quad |p_1| \leq 1, \quad |p_2| \leq 1. \quad (5.68)$$

Умножив второе неравенство на $7/3$ и сложив с первым неравенством, получим $\eta \geq \frac{11}{5}p_2$. Так как $\eta \leq 0$, то и $p_2 \leq 0$. Теперь умножим второе неравенство на $13/15$ и после сложения с первым найдем $p_1 \geq -\frac{14}{11}\eta$,

т.е. $p_1 \geq 0$. Отсюда следует, что наибольшему возможному значению $p_1 = 1$ соответствуют наименьшее возможное значение $\eta = -\frac{11}{14}$ и $p_2 = -\frac{5}{14}$. При этом все неравенства в (5.68) выполнены. Таким образом, получаем $p^2 = (1 - 5/14)^T$.

При движении точки $x = x^1 + \kappa p^2$ в направлении вектора p^2 ограничение $g_2(x) \leq 0$ не будет нарушено, поскольку оно является активным в точке x^1 . Ограничение $g_3(x) \leq 0$ также не нарушается, так как оно означает, что $x_1 \geq 0$, а в точке $x^1 + \kappa p^2$ для первой координаты имеем $x_1^{(1)} + \kappa p_1^{(2)} = \frac{5}{6} + \kappa \geq 0$. Выясним, при каких значениях κ выполняются ограничения $g_1(x) \leq 0$ и $g_4(x) \leq 0$, т.е. когда

$$g_1(x) = x_1^{(1)} + \kappa p_1^{(2)} + x_2^{(1)} + \kappa p_2^{(2)} - 2 = \frac{9}{14}\kappa - \frac{1}{3} \leq 0,$$

$$g_4(x) = -x_2^{(1)} - \kappa p_2^{(2)} = \frac{5}{14}\kappa - \frac{5}{6} \leq 0.$$

Из этих неравенств следует, что $\bar{\kappa}_2 = 14/27$.

Функция $\varphi_2(\kappa) = f_0(x^1 + \kappa p^2)$ имеет вид

$$\begin{aligned} \varphi_2(\kappa) &= \frac{1}{2}(Q(x^0 + \kappa p^2), x^0 + \kappa p^2) + (b, x^0 + \kappa p^2) = \\ &= \frac{291}{98}\kappa^2 - \frac{11}{14}\kappa - \frac{125}{18}. \end{aligned}$$

На отрезке $[0, \bar{\kappa}_2]$ точкой ее минимума будет $\kappa_2 = \frac{77}{582}$. Поэтому $x^2 = x^1 + \kappa_2 p^2 = \left(\frac{281}{291}, \frac{935}{1164}\right)$ и $f_0(x^2) \approx -7,0850$ (см. рис. 5.20).

Третья итерация. Используя формулу (5.66), вычисляем

$$\text{grad } f_0(x^2) = \left(-\frac{1015}{582} \quad -\frac{1373}{291}\right)^T.$$

Так как в точке x^2 нет активных ограничений, то $\mathcal{I}_3 = \emptyset$, и переменное η следует минимизировать на множестве, заданном неравенствами

$$-\frac{1015}{582}p_1 - \frac{1373}{291}p_2 \leq \eta, \quad |p_1| \leq 1, \quad |p_2| \leq 1.$$

Иначе говоря, нужно найти минимальное значение левой части первого неравенства при условии, что выполнены два других неравенства.

Ясно, что минимальное значение достигается при $p_1 = p_2 = 1$, т.е. $p^3 = (1 \ 1)^T$, и минимальным значением η будет $\eta = -\frac{3761}{582} \approx -6.4622$.

При движении точки $x = x^2 + \kappa p^3$ в направлении вектора p^3 ограничения $g_3(x) \leq 0$ и $g_4(x) \leq 0$, означающие неотрицательность переменных x_1 и x_2 , выполняются при любом κ . Выполнение ограничений $g_1(x) \leq 0$ и $g_2(x) \leq 0$ означает, что

$$\begin{aligned} g_1(x) &= x_1^{(2)} + \kappa p_1^{(3)} + x_2^{(2)} + \kappa p_2^{(3)} - 2 = 2\kappa - \frac{269}{1164} \leq 0, \\ g_2(x) &= x_1^{(2)} + \kappa p_1^{(3)} + 5x_2^{(2)} + 5\kappa p_2^{(3)} - 5 = 6\kappa - \frac{21}{1164} \leq 0. \end{aligned}$$

Отсюда устанавливаем, что $\bar{\kappa}_3 = \frac{7}{2328} \approx 0,0030$.

Функция $\varphi_3(\kappa) = f_0(x^2 + \kappa p^3)$ на отрезке $[0, \bar{\kappa}_3]$ достигает наименьшего значения при $\kappa_3 = \bar{\kappa}_3$. Поэтому

$$x^3 = x^2 + \kappa_3 p^3 = \left(\frac{2255}{2328}, \frac{1877}{2328} \right) \approx (0,9686, 0,8063)$$

и $f_0(x^3) \approx -7,0975$.

Рассматриваемая задача с квадратичной целевой функцией не является сложной, и ее решение нетрудно найти аналитическими методами. Наименьшего значения на допустимом множестве целевая функция $f_0(x)$ достигает в точке $x^* = \left(\frac{35}{31}, \frac{24}{31} \right) \approx (1,1290, 0,7742)$, при этом $f_0(x^*) = -\frac{6882}{961} \approx -7,1613$ (см. пример 5.11). Для приближения к точке x^* с заданной точностью ε_η потребуется еще довольно много итераций.

Случай линейных ограничений. Остановимся на частном случае, когда в задаче оптимизации допустимое множество Ω задано линейными ограничениями $g_i(x) = (a_i, x) \leq b_i$, $i = \overline{1, m}$. В этом случае задачу поиска возможного направления спуска можно упростить, видоизменив ее следующим образом. На каждой k -й итерации вектор $p^k \in \mathbb{R}^n$, определяющий это направление, можно искать из условия минимума функции $\zeta_k(p) = (\text{grad } f_0(x^{k-1}), p)$ на множестве

$$\widetilde{W}_k = \{p \in \mathbb{R}^n: (a_i, p) \leq 0, i \in \mathcal{I}_k; |p_j| \leq 1, j = \overline{1, n}\}, \quad (5.69)$$

где \mathcal{I}_k — по-прежнему множество индексов, соответствующих активным ограничениям в точке $x^{k-1} \in \Omega$. Можно показать, что если

минимальное значение функции $\zeta_k(p)$ на множестве $\widetilde{\mathcal{W}}_k$ равно нулю, то точка x^{k-1} удовлетворяет необходимому условию минимума целевой функции $f_0(x)$ на множестве Ω .

Пример 5.14. Вернемся к задаче из примера 5.13 и решим ее, выбирая возможное направление спуска путем минимизации функции $\zeta_k(p)$ на множестве (5.69). Отметим, что в данном случае $a_i = \text{grad } g_i(x)$, $i = \overline{1, 4}$; $b_1 = 2$; $b_2 = 5$; $b_3 = b_4 = 0$.

Результат выполнения первой итерации рассматриваемым методом останется таким же, как и в примере 5.13. В самом деле, если выбрать начальную точку $x^0 = (0, 0)$, то получим $\zeta_1(p) = (\text{grad } f_0(x^0), p) = -4p_1 - 6p_2$, а множество $\widetilde{\mathcal{W}}_1$ будет задано неравенствами $-p_1 \leq 0$, $-p_2 \leq 0$, $|p_1| \leq 1$ и $|p_2| \leq 1$. Отсюда находим минимальное значение $\zeta(p^1) = -10$ при $p^1 = (1 \ 1)$ и затем точку $x^1 = x^0 + \kappa p^1 = (5/6, 5/6)$ (см. рис. 5.20).

На второй итерации в точке $x^1 = (5/6, 5/6)$ имеем $\zeta_2(p) = (\text{grad } f_0(x^1), p) = -7p_1/3 - 13p_2/3$. Поскольку в точке x^1 активно лишь ограничение с индексом $i = 2$, функцию $\zeta_2(p)$ необходимо минимизировать на множестве $\widetilde{\mathcal{W}}_2$, заданном неравенствами

$$p_1 + 5p_2 \leq 0, \quad |p_1| \leq 1, \quad |p_2| \leq 1. \quad (5.70)$$

Множество $\widetilde{\mathcal{W}}_2$ ограничено трапецией $ABCD$ (рис. 5.21), а линиями уровня функции $\zeta_2(p)$ являются прямые, параллельные прямой $7p_1 + 13p_2 = 0$ (прямая $\zeta_2 = 0$ на рис. 5.21), на которой эта функция имеет нулевое значение. Из геометрических соображений ясно, что функция $\zeta_2(p)$ принимает на множестве $\widetilde{\mathcal{W}}_2$ наименьшее значение в точке $p^2 = (1 \ -1/5)$ и это значение равно $-22/15$.

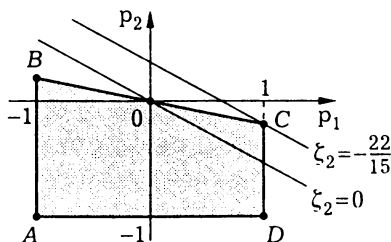


Рис. 5.21

При движении точки $x = x^1 + \kappa p^2$ в направлении вектора p^2 ограничение $g_2(x) \leq 0$, оставаясь активным, не нарушается. По-

этому необходимо убедиться в выполнении ограничений $g_1(x) \leq 0$ и $g_4(x) \leq 0$. Это означает выполнение условий

$$g_1(x) = x_1^{(1)} + \kappa p_1^{(2)} + x_2^{(1)} + \kappa p_2^{(2)} - 2 = \frac{4}{5}\kappa - \frac{1}{3} \leq 0,$$

$$g_4(x) = -x_2^{(1)} - \kappa p_2^{(2)} = \frac{1}{5}\kappa - \frac{5}{6} \leq 0.$$

Отсюда заключаем, что $\bar{\kappa}_2 = 5/12$.

Поскольку

$$\varphi_2(\kappa) = f_0(x^1 + \kappa p^2) = \frac{62}{25}\kappa^2 - \frac{22}{15}\kappa - \frac{125}{8},$$

функция $\varphi_2(\kappa)$ достигает наименьшего значения на отрезке $[0, \bar{\kappa}_2] = [0, 5/12]$ при $\kappa = \frac{55}{186}$. Следовательно, $\kappa_2 = \frac{55}{186}$ и $x^2 = x^1 + \kappa_2 p^2 = \left(\frac{35}{31}, \frac{24}{31}\right)$.

В точке x^2 по-прежнему активно лишь ограничение с индексом $i = 2$. В этой точке в соответствии с формулой (5.66) $\text{grad } f_0(x^2) = \left(-\frac{32}{31} \quad -\frac{160}{31}\right)^T$. Поэтому на третьей итерации необходимо минимизировать функцию

$$\zeta_3(p) = (\text{grad } f_0(x^2), p) = -\frac{32}{31}p_1 - \frac{160}{31}p_2$$

на множестве $\widetilde{W}_3 = \widetilde{W}_2$, заданном теми же неравенствами (5.70). Линии уровня функции $\zeta_3(p)$ — прямые, параллельные прямой $p_1 + 5p_2 = 0$, на которой функция принимает нулевое значение. Но прямая $p_1 + 5p_2 = 0$ проходит по стороне DC трапеции, ограничивающей множество \widetilde{W}_3 (см. рис. 5.21). Значения функции в точках трапеции, не лежащих на стороне BC , положительны. Таким образом, минимальное значение функции $\zeta_3(p)$ равно нулю. Поэтому точка $x^* = x^2 = \left(\frac{35}{31}, \frac{24}{31}\right)$ удовлетворяет необходимому условию минимума функции $f_0(x)$ на допустимом множестве. А так как функция $f_0(x)$ является выпуклой, допустимое множество Ω также выпуклое, то в точке x^* функция $f_0(x)$ достигает наименьшего значения на множестве Ω .

Случай нелинейных ограничений. На практике при наличии нелинейных ограничений обычно применяют несколько модифицированные варианты метода возможных направлений. Дело в том, что при малых значениях $|\eta_k|$ направление спуска, задаваемое вектором p^k , который находят из решения задачи линейного программирования (5.60), хотя и является допустимым с теоретической точки зрения, может быть неприемлемым с практической точки зрения, так как оно может оказаться почти ортогональным направлению антиградиента $w^k = -\text{grad } f_0(x^{k-1})$ целевой функции. При близких к нулю значениях (w^k, p^k) направление спуска p^k почти „касается“ границы $\partial\Omega$ множества Ω . Поэтому значения $\bar{\kappa}_k$, а следовательно, и значения κ_k , будут весьма малыми, а сходимость процесса поиска минимума — очень медленной. Более того, возможно „зацикливание“ алгоритма или сходимость последовательности $\{x^k\}$ к такой точке $\tilde{x} \in \Omega$, в которой не выполнены необходимые условия минимума.

Один из способов ускорения сходимости метода возможных направлений заключается в том, чтобы при выборе возможного направления спуска учитывать все ограничения, а не только активные в точке x^{k-1} . Для этого направление спуска p^k на k -й итерации находят как решение задачи минимизации

$$\begin{cases} \eta \rightarrow \min; \\ (\text{grad } f_0(x^{k-1}), p) \leq \eta; \\ g_i(x^{k-1}) + (\text{grad } g_i(x^{k-1}), p) \leq \beta_i \eta, \quad i = \overline{1, m}; \\ |p_j| \leq 1, \quad j = \overline{1, n}, \end{cases}$$

где $\beta_i > 0$ — некоторые весовые коэффициенты. Так как в задании допустимого множества \widehat{W}_k этой задачи оптимизации участвуют все ограничения типа неравенства исходной задачи, то направление спуска не испытывает резких изменений, когда от итерации к итерации изменяется множество индексов активных ограничений. Это позволяет избежать зигзагообразной траектории поиска точки x^* и уменьшает опасность „зацикливания“ алгоритма. Можно показать, что и для этой модификации метода возможных направлений равенство $\eta_k = 0$ является необходимым условием минимума целевой функции на множестве Ω .

Недостаток описанной модификации метода возможных направлений состоит в том, что задача линейного программирования, на основе которой определяется направление спуска, усложняется, поскольку в

нее включаются все m ограничений. Существует промежуточная модификация метода, в которой учитываются лишь так называемые ε -активные ограничения в окрестности точки x^{k-1} , т.е. такие ограничения, для которых в точке x^{k-1} выполняется неравенство $-\varepsilon_k \leq g_i(x^{k-1}) \leq 0$. В этой модификации вместо множества индексов \mathcal{I}_k используют множество

$$\mathcal{I}_k^* = \{i \in \{1, \dots, m\} : -\varepsilon_k \leq g_i(x^{k-1}) \leq 0\}. \quad (5.71)$$

Значение $\varepsilon_k > 0$ следует выбирать так, чтобы оно было существенно больше значения ε_η . Оно может меняться от итерации к итерации, но можно также считать, что $\varepsilon_k = \varepsilon_0 = \text{const}$. Множество \mathcal{I}_k^* включает в себя множество \mathcal{I}_k , но при этом с помощью выбора подходящего значения ε_k число ε -активных ограничений можно заметно уменьшить по сравнению с общим числом ограничений.

Поясним смысл введения ε -активных ограничений на примере минимизации целевой функции $f_0(x)$ двух переменных при трех ограничениях $g_i(x) \leq 0$, $i = 1, 2, 3$ (рис. 5.22, а). Пусть в точке $x^{k-1} \in \Omega$ активно лишь первое ограничение, т.е. $\mathcal{I}_k = \{1\}$. Тогда из решения задачи линейного программирования (5.60) получим направление спуска, задаваемое вектором p^k и являющееся промежуточным между направлениями в точке x^{k-1} антиградиентов $w_0^k = -\text{grad } f_0(x^{k-1})$ и $w_1^k = -\text{grad } g_1(x^{k-1})$ целевой функции и функции $g_1(x)$ соответственно. Но выбор значения κ_k в (5.63) будет существенно ограничен, так как это направление спуска пересекает границу множества Ω , определяемую ограничением с индексом $i = 2$, на малом расстоянии

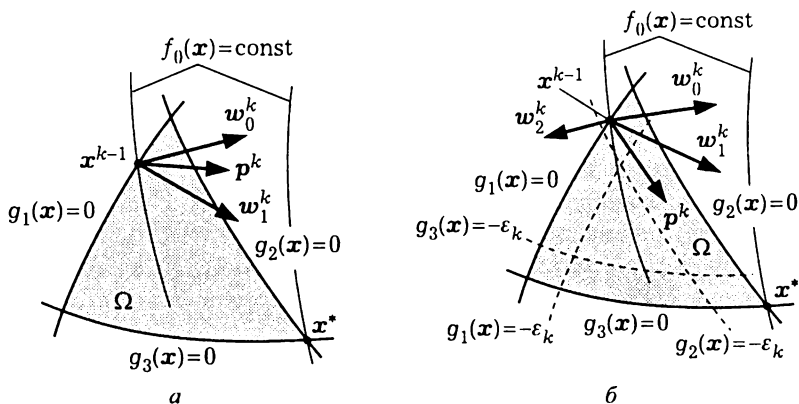


Рис. 5.22

от точки x^{k-1} . Если учесть ε -активные в этой точке ограничения (рис. 5.22, б), то будем иметь $\mathcal{I}_k^* = \{1, 2\}$. Тогда решением задачи (5.60), в которой \mathcal{I}_k заменено на \mathcal{I}_k^* и учтено влияние антиградиента $w_2^k = -\text{grad } g_2(x^{k-1})$, можно получить вектор p^k , задающий более удачное направление спуска. Это направление позволяет на k -й итерации сделать более длинный шаг на пути к точке x^* минимума целевой функции на множестве Ω .

В общем случае решение $z^k = (p^k, \eta_k)$ задачи линейного программирования (5.60), в которой множество индексов \mathcal{I}_k заменено множеством \mathcal{I}_k^* , будет удовлетворять одному из двух условий: $\eta_k \leq -\varepsilon_k$ или $-\varepsilon_k < \eta_k \leq 0$. Если выполнено первое условие, считают, что соответствующий значению η_k вектор p^k задает достаточно удачное направление спуска, и этот вектор используют для нахождения очередной точки x^k . Если же выполнено второе условие, то полагают, что найденное направление спуска неудачно, и ищут другое направление спуска, уменьшая значение ε_k , например, вдвое. Уменьшение ε_k приводит, вообще говоря, к сужению множества индексов \mathcal{I}_k^* , в результате чего во вспомогательной задаче линейного программирования часть ограничений снимается, а допустимое множество расширяется. Естественно предполагать, что на более широком множестве \mathcal{W}_k можно найти более удачное направление спуска, которое будет удовлетворять первому условию. Процедура дробления ε_k может и не дать положительного результата вплоть до выполнения неравенства $|\varepsilon_k| < \varepsilon_\eta$, что равносильно условию $|\eta_k| < \varepsilon_\eta$ прекращения последующих итераций.

Если множество Ω задано с помощью не только ограничений типа неравенства, но и q линейных ограничений типа равенства, т.е.

$$\Omega = \{x \in \mathbb{R}^n: g_i(x) \leq 0, i = \overline{1, m}; Ax = b\},$$

где A — матрица размера $q \times n$, $b \in \mathbb{R}^q$, то это следует учесть во вспомогательной задаче линейного программирования, добавив в нее ограничение $Ap = 0$. Тогда при спуске в направлении, определяемом вектором p^k , удовлетворяющим последнему ограничению, точка x^k не выйдет за пределы допустимого множества Ω . Действительно, согласно равенству (5.63) для любого значения κ_k верно равенство

$$Ax^k = Ax^{k-1} + \kappa_k Ap^k = b,$$

поскольку $Ap^k = 0$, а точка x^{k-1} по условию принадлежит множеству Ω и, значит, удовлетворяет равенству $Ax^{k-1} = b$.

При наличии r нелинейных ограничений вида $f_l(x) = 0$, $l = \overline{1, r}$, метод возможных направлений можно применить, если на каждой k -й итерации линейаризовать эти ограничения в окрестности точки x^{k-1} . Но тогда спуск в направлении, найденном этим методом, при любом значении $\kappa_k > 0$ может выводить точку x^k за пределы Ω . Это нарушение можно скорректировать, заменяя найденную точку ее проекцией на множество Ω . Другой подход состоит в том, чтобы ослабить нелинейные ограничения типа равенства, для чего полагают $|f_l(x)| \leq \delta_k$, $l = \overline{1, r}$. По мере приближения к искомой точке $x^* \in \Omega$ минимума целевой функции на множестве Ω значение $\delta_k > 0$ можно уменьшать.

5.7. Методы последовательной безусловной минимизации

Один из подходов к решению общей задачи нелинейного программирования состоит в преобразовании ее в последовательность задач безусловной минимизации для сконструированных специальным образом вспомогательных функций. Методы, в которых используется такой подход, принято называть *методами последовательной безусловной минимизации*.

Рассмотрим общую схему, лежащую в основе методов последовательной безусловной минимизации. Формально задача минимизации функции $f(x)$, определенной в \mathbb{R}^n , на допустимом множестве Ω эквивалентна задаче безусловной минимизации

$$\min_{x \in \mathbb{R}^n} (f(x) + \delta_\Omega(x)), \quad (5.72)$$

где $\delta_\Omega(x)$ — так называемая *индикаторная функция* множества Ω :

$$\delta_\Omega(x) = \begin{cases} 0, & x \in \Omega; \\ +\infty, & x \notin \Omega. \end{cases}$$

Символ $+\infty$ на практике можно считать просто очень большим числом, превышающим, например, любое значение целевой функции на допустимом множестве.

Точки минимума функций $f(x)$ и $f(x) + \delta_\Omega(x)$ на множестве Ω совпадают, но индикаторная функция, даже если $+\infty$ трактовать

как большое число, имеет точки разрыва на границе допустимого множества. Это не позволяет применять для решения задачи (5.72) известные методы безусловной минимизации. Выход из этой ситуации состоит в построении такой последовательности непрерывных (или даже дифференцируемых определенное число раз) функций $\delta_k(x)$, что в каждой фиксированной точке $x \in \mathbb{R}^n$

$$\lim_{k \rightarrow \infty} \delta_k(x) = \delta_\Omega(x). \quad (5.73)$$

Можно рассчитывать на то, что последовательность $\{x^k\}$, составленная из точек минимума x^k функций $f(x) + \delta_k(x)$, сходится к точке минимума функции $f(x) + \delta_\Omega(x)$. Таким образом, исходную задачу нелинейного программирования можно свести к последовательности задач безусловной минимизации.

При таком подходе функции $\delta_k(x)$ называют *функциями штрафа*, целевые функции $f(x) + \delta_k(x)$ в последовательности задач безусловной минимизации — *штрафными функциями*, а методы, основанные на этом подходе, — *методами штрафных функций*.

Построение последовательности $\{\delta_k(x)\}$ сильно зависит от способа представления допустимого множества и в общем случае может быть весьма сложным. Если допустимое множество Ω задано ограничениями типа равенства и неравенства, штрафные функции можно строить, используя фигурирующие в ограничениях функции. В данном случае выделяются два способа построения штрафных функций и соответственно две разновидности метода штрафных функций.

Для первой разновидности метода характерным является совпадение функций штрафа с предельной функцией δ_Ω вне допустимого множества Ω , т.е. $\delta_k(x) = +\infty$ при $x \notin \Omega$. Это означает, что штрафная функция неограниченно возрастает при приближении точки $x \in \Omega$ к границе допустимого множества, а ее точка минимума (если существует) принадлежит допустимому множеству. Такую разновидность метода называют *методом внутренних штрафных функций* (также *методом барьерных функций* или *методом внутренней точки*), а функцию штрафа — *функцией внутреннего штрафа* или *барьерной функцией*.

Для второй разновидности метода штрафных функций характерно совпадение функций штрафа с предельной функцией $\delta_\Omega(x)$ внутри допустимого множества, т.е. $\delta_k(x) = 0$ при $x \in \Omega$. Это означает, что штрафные функции в Ω совпадают с функцией $f(x)$, а их точки

минимума могут оказаться вне множества Ω . В данном случае говорят о *методе внешних штрафных функций* (иногда — о *методе внешней точки*). При этом функции штрафа называют *функциями внешнего штрафа*.

Рассмотрим вычислительные конструкции методов штрафных функций, наиболее широко применяемых в приложениях, считая, что допустимое множество Ω задано с помощью системы ограничений типа неравенства в виде

$$\Omega = \{x \in \mathbb{R}^n: g_i(x) \leq 0, i = \overline{1, m}\}, \quad (5.74)$$

где $g_i(x)$, $x \in \mathbb{R}^n$, — непрерывные функции.

Метод внутренних штрафных функций. Пусть допустимое множество Ω содержит внутренние точки, т.е. точки, в которых все ограничения выполнены как строгие неравенства. Функцию внутреннего штрафа строят таким образом, чтобы она была неотрицательной и принимала значение $+\infty$ за пределами допустимого множества, т.е.

$$\delta_k(x) = \begin{cases} > 0, & x \in \Omega \setminus \partial\Omega; \\ +\infty, & x \notin \Omega \setminus \partial\Omega. \end{cases}$$

Роль функции штрафа состоит в создании „барьера“ вдоль границы допустимого множества. Чаще всего в качестве такой функции принимают

$$\delta_k(x) = -r_k \sum_{i=1}^m \frac{1}{g_i(x)}, \quad x \in \Omega \setminus \partial\Omega, \quad (5.75)$$

или

$$\delta_k(x) = -r_k \sum_{i=1}^m \ln(-g_i(x)), \quad x \in \Omega \setminus \partial\Omega, \quad (5.76)$$

где последовательность коэффициентов $r_k > 0$, называемых *параметрами штрафа*, стремится к нулю при $k \rightarrow \infty$.

Можно показать, что если допустимое множество Ω замкнуто и ограничено, то при выборе функций штрафа в виде (5.75) или (5.76) последовательность $\{x_k^*\}$ точек минимума штрафных функций $f_k(x) = f(x) + \delta_k(x)$, $k \in \mathbb{N}$, сходится к точке минимума целевой функции $f(x)$.

Применение метода внутренних штрафных функций состоит в последовательной минимизации штрафных функций $f_k(x) = f(x) + \delta_k(x)$ и построении последовательности точек минимума x_k^* этих функций. Итерационный процесс прекращается, если выполняется одно из условий останова

$$|x_k^* - x_{k-1}^*| < \varepsilon_1, \quad |f(x_k^*) - f(x_{k-1}^*)| < \varepsilon_2 \quad (5.77)$$

или оба вместе.

Эффективность метода внутренних штрафных функций в значительной степени зависит от выбора значений параметров штрафа r_k . Общих рекомендаций здесь не существует. Часто параметры штрафа задают в соответствии с формулой $r_k = \frac{r_{k-1}}{\gamma}$, где $\gamma > 1$ — некоторый фиксированный коэффициент. Тогда процесс оптимизации регулируется начальным значением параметра штрафа r_0 и коэффициентом γ .

Эффективность метода внутренних штрафных функций зависит не только от выбора параметров штрафа, но и от способа задания самих функций штрафа. Универсальных рекомендаций в этом вопросе также не существует. В одних случаях способ задания этой функции не оказывает существенного влияния, а в других — его влияние оказывается решающим.

Пример 5.15. Методом внутренних штрафных функций решим задачу оптимизации

$$\begin{cases} f(x_1, x_2) = 10x_1^2 - 4x_1x_2 + 7x_2^2 - 4\sqrt{5}(5x_1 - x_2) - 16 \rightarrow \min; \\ (x_1 - 1)^2 + (x_2 - 3)^2 - 4 \leq 0. \end{cases}$$

Допустимое множество Ω в этой задаче представляет собой замкнутый круг с радиусом 2 и центром в точке (1, 3). Вводя функцию штрафа в соответствии с формулой (5.75) или (5.76), получим два варианта штрафных функций:

$$f_k(x) = f(x_1, x_2) - \frac{r_k}{g(x_1, x_2)}$$

или

$$\tilde{f}_k(x) = f(x_1, x_2) - r_k \ln(-g(x_1, x_2)).$$

Параметры штрафа r_k определим соотношениями $r_0 = 32$, $r_k = r_{k-1}/2$, $k \in \mathbb{N}$. На рис. 5.23, *а* изображены линии уровня штрафной функции $f_k(x)$ при значении параметра штрафа $r_k = 32$, а на рис. 5.23, *б* — линии уровня функции $\tilde{f}_k(x)$ при $r_k = 1$.

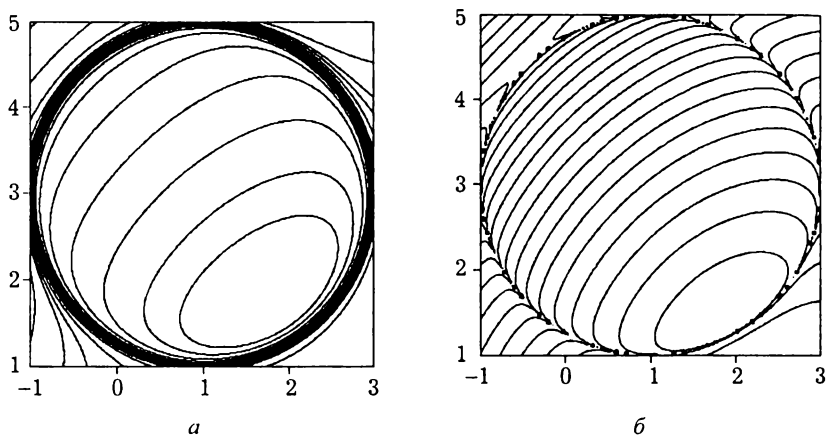


Рис. 5.23

Решение вспомогательных задач безусловной минимизации было проведено численно с применением метода Ньютона при выборе в качестве начальной точки $x^0 = (1, 3)$. Поиск точки минимума $x^* \in \Omega$ прекращался при выполнении второго условия (5.77) с параметром $\varepsilon_2 = 10^{-3}$.

Результаты расчетов приведены в табл. 5.4, в которой N — число решенных задач минимизации штрафных функций в каждом варианте метода. Числа в двух последних колонках таблицы позволяют сравнить значения N -й штрафной функции в точке $x_N^* \approx x^*$ и целевой функции $f(x)$ в этой же точке. Из табл. 5.4 видно, что в данном случае способ задания функции штрафа практически не влияет на эффективность реализуемого алгоритма.

Таблица 5.4

Штрафная функция	N	x_N^*	$f_N(x_N^*)$	$f(x_N^*)$
$f_k(x)$	17	(1,970, 1,254)	−52,932	−52,957
$\tilde{f}_k(x)$	16	(1,970, 1,251)	−52,997	−53,004

Пример 5.16. Найдем решение задачи квадратичного программирования

$$\begin{cases} f(x_1, x_2) = 10x_1^2 - 4x_1x_2 + 7x_2^2 - 4\sqrt{5}(5x_1 - x_2) - 16 \rightarrow \min; \\ x_1 - 2 \leq 0, \quad x_1 + x_2 - 1 \leq 0. \end{cases}$$

Граница допустимого множества Ω для этой задачи изображена на рис. 5.24. Используя функции штрафа (5.75) и (5.76), получаем штрафные функции

$$f_k(x) = f(x_1, x_2) - \frac{r_k}{x_1 - 2} - \frac{r_k}{x_1 + x_2 - 1}$$

и

$$\tilde{f}_k(x) = f(x_1, x_2) - r_k \left(\ln(-(x_1 - 2)) + \ln(-(x_1 + x_2 - 1)) \right).$$

Параметры штрафа r_k определим, как и в примере 5.15, соотношениями $r_0 = 32$, $r_k = r_{k-1}/2$, $k \in \mathbb{N}$.

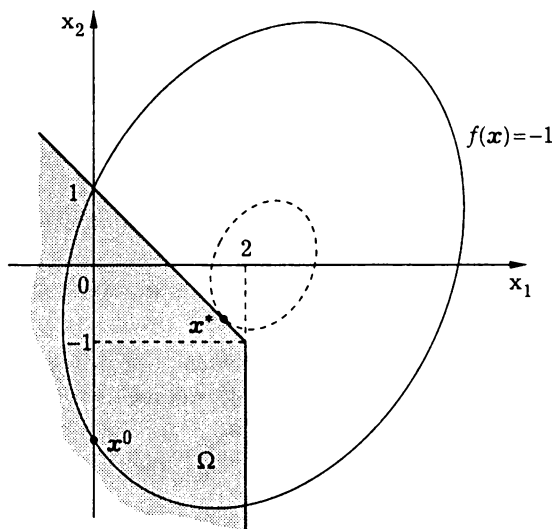


Рис. 5.24

На рис. 5.25, а показаны линии уровня вспомогательной функции $f_k(x)$ при $r_k = 32$, а на рис. 5.25, б — линии уровня вспомогательной функции $\tilde{f}_k(x)$ при $r_k = 1$. Все расчеты проведены так же, как и в

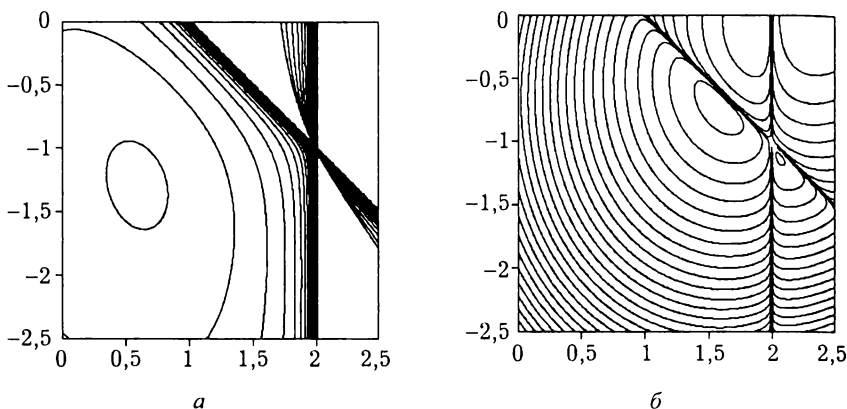


Рис. 5.25

примере 5.15, их результаты представлены в табл. 5.5 (обозначения в ней те же, что и в табл. 5.4).

Таблица 5.5

Штрафная функция	N	x_N^*	$f_N(x_N^*)$	$f(x_N^*)$
$f_k(x)$	25	(1,706, -0,706)	-61,193	-61,195
$\tilde{f}_k(x)$	11	(1,704, -0,704)	-60,796	-61,198

В данном случае способ задания штрафной функции оказывает существенное влияние на эффективность реализуемого алгоритма. Предпочтительнее штрафная функция $\tilde{f}(x)$, применение которой позволяет найти точку x^* с заданной точностью путем решения значительно меньшего числа вспомогательных задач минимизации штрафной функции.

Метод внешних штрафных функций. В этом методе функция штрафа равна нулю на допустимом множестве Ω и положительна за пределами допустимого множества, т.е.

$$\delta_k(x) = \begin{cases} 0, & x \in \Omega; \\ > 0, & x \notin \Omega. \end{cases}$$

Обычно функции штрафа строят в соответствии с формулой $\delta_k(x) = r_k \Phi(x)$, где $r_k > 0$ — параметр штрафа, а $\Phi(x)$ — функция, определенная на всем множестве \mathbb{R}^n , равная нулю на допустимом множестве Ω и положительная за его пределами. Последовательность параметров штрафа $\{r_k\}$ выбирают так, чтобы она стремилась

$k + \infty$ при $k \rightarrow \infty$. Это требование гарантирует сходимость последовательности функций штрафа к индикаторной функции допустимого множества Ω .

Если множество Ω задано ограничениями типа неравенства, т.е. имеет вид (5.74), то в качестве функции штрафа обычно используют квадратичную функцию

$$\delta_k(x) = r_k \sum_{i=1}^m (g_i^+(x))^2, \quad (5.78)$$

где $g_i^+(x) = \max\{0, g_i(x)\}$ — так называемая „срезка“ функции $g_i(x)$:

$$g_i^+(x) = \begin{cases} 0, & g_i(x) \leq 0; \\ g_i(x), & g_i(x) > 0. \end{cases}$$

Если допустимое множество Ω задано ограничениями, среди которых не только неравенства $g_i(x) \leq 0$, $i = \overline{1, m}$, но и равенства $g_i(x) = 0$, $i = \overline{m+1, p}$, то функцию штрафа задают в виде

$$\delta_k(x) = r_k \sum_{i=1}^m (g_i^+(x))^2 + r_k \sum_{i=m+1}^p (g_i(x))^2.$$

В остальном метод внешних штрафных функций не отличается от метода внутренних штрафных функций. В частности, в методе внешних штрафных функций используют те же условия останова (5.77), что и в методе внутренних штрафных функций.

Пример 5.17. Решим методом внешних штрафных функций задачу из примера 5.15, выбрав в качестве начальной точку $x^0 = (0, -\sqrt{5})$ и условие останова $\|x_k^* - x_{k-1}^*\| < \varepsilon$ с параметром точности поиска $\varepsilon = 10^{-3}$. В данном случае штрафные функции имеют вид

$$f_k(x_1, x_2) = f(x_1, x_2) + r_k (g^+(x_1, x_2))^2,$$

где $g^+(x_1, x_2) = \frac{1}{2}(g(x_1, x_2) + |g(x_1, x_2)|)$. Минимизация этих функций проведена численно с применением того же алгоритма метода Ньютона, что и в примере 5.15. Параметры r_k определены рекуррентным соотношением $r_k = r_{k-1} + 2$, $k \in \mathbb{N}$, $r_0 = 2$. Результаты расчетов приведены в табл. 5.6, а траектория поиска точки минимума целевой функции изображена на рис. 5.26 сплошной линией. #

Таблица 5.6

k	x_k^*	$f(x_k^*)$	k	x_k^*	$f(x_k^*)$
1	(0,386, -0,672)	-33,568	8	(1,974, 1,230)	-53,434
2	(0,652, 0,270)	-38,698	9	(1,976, 1,234)	-53,384
3	(0,899, 0,755)	-40,109	10	(1,976, 1,236)	-53,348
4	(1,263, 0,928)	-46,877	11	(1,975, 1,237)	-53,318
5	(1,694, 1,041)	-53,222	12	(1,975, 1,238)	-53,292
6	(1,852, 1,150)	-53,501	13	(1,975, 1,239)	-53,271
7	(1,954, 1,212)	-53,551	14	(1,974, 1,240)	-53,252

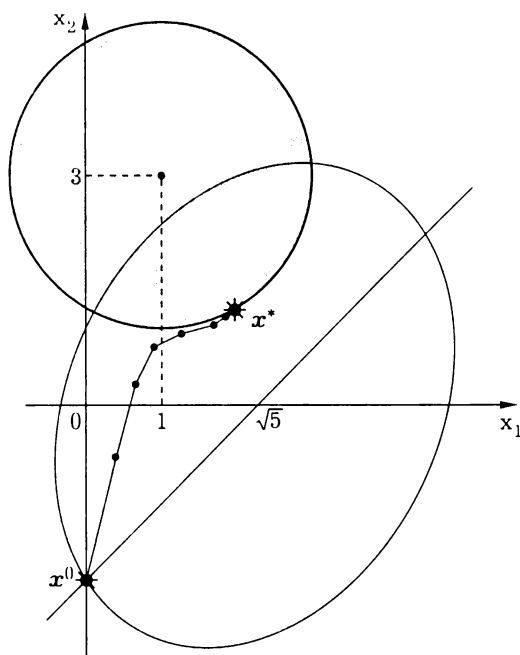


Рис. 5.26

Методы штрафных функций выгодно отличаются от рассмотренных выше методов проектирования и спуска простотой реализации и „сильными“ свойствами сходимости, на их разработку и совершенствование до сих пор затрачиваются значительные усилия. В то же время практическая реализация методов штрафных функций может приводить к трудностям принципиального характера. В основном

они связаны с топографией поверхностей уровня штрафных функций $f_k(x)$: при увеличении номера k , когда функция штрафа все больше приближается к индикаторной функции, графики штрафных функций приобретают все более выраженную овражную структуру. Это может приводить к большим вычислительным затратам при решении вспомогательных задач безусловной минимизации и даже к прерыванию процесса вычислений ввиду потери точности. Для преодоления этих трудностей развиты модификации метода штрафных функций, свободные от указанных недостатков: метод штрафных оценок, метод модифицированных функций Лагранжа, метод штрафов с регуляризацией и др.

Вопросы для самопроверки

1. Опишите известные вам численные методы нелинейного программирования. Укажите их достоинства и недостатки, целесообразность применения каждого конкретного метода в зависимости от вида допустимого множества и вида ограничений задачи.

2. Сформулируйте определение проекции точки на множество. Перечислите основные свойства операции проектирования на множество. Как найти проекцию точки на замкнутое выпуклое множество?

3. Дайте определение идемпотентной матрицы и проекционной матрицы. Какую роль проекционная матрица играет в операциях проектирования на множество?

4. Что такое приведенный градиент? В каких задачах нелинейного программирования используется метод приведенного градиента? В чем различие между методом приведенного градиента и методом проекции антиградиента?

5. Какие ограничения задачи нелинейного программирования называют активными, а какие — пассивными? Что такое ε -активные ограничения?

6. Что означают термины „штрафная функция“, „функция штрафа“, „параметр штрафа“? Чем функция внутреннего штрафа отличается от функции внешнего штрафа? Укажите способы задания функций штрафа.

МЕТОДЫ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

Напомним, что задачей линейного программирования называют задачу оптимизации, в которой и целевая функция, и ограничения являются линейными. В такой задаче могут быть ограничения и типа равенства, и типа неравенства. Ее можно записать следующим образом*:

$$\left\{ \begin{array}{l} \sum_{j=1}^n c_j x_j \rightarrow \max; \\ \sum_{i=1}^n a_{ij} x_j \leq b_i, \quad i = \overline{1, m_1}; \\ \sum_{i=1}^n a_{ij} x_j = b_i, \quad i = \overline{m_1+1, m_2}; \\ \sum_{i=1}^n a_{ij} x_j \geq b_i, \quad i = \overline{m_2+1, m}. \end{array} \right. \quad (6.1)$$

В задачах линейного программирования нельзя использовать классические методы поиска локального экстремума, поскольку линейная функция не имеет точек локального экстремума. Такие задачи можно решать методами нелинейного программирования, но оказывается, что для них есть специальные и гораздо более эффективные методы решения.

Хотя класс задач линейного программирования довольно узкий, такие задачи играют в прикладных областях важную роль. Дело в том,

*Мы рассматриваем задачу линейного программирования как задачу максимизации, отдавая дань сложившейся традиции. Напомним, что задачу минимизации легко преобразовать в задачу максимизации, изменив знак целевой функции.

что линейные математические модели представляют собой модели первого приближения, в которых связь различных факторов выражена самой простой функциональной зависимостью — линейной. В ряде областей (в частности, в экономике) реальные процессы определяются большим количеством факторов. В этом случае на первое место выходит не вид зависимости между какими-либо параметрами модели, а общая структура таких зависимостей. Для описания каждой конкретной зависимости достаточно линейной функции, а линейные модели становятся ключевыми.

6.1. Виды задач линейного программирования

Выделим три вида задач линейного программирования. Задачу, представленную системой уравнений (6.1), назовем *задачей линейного программирования общего вида*.

Во многих прикладных задачах переменные могут принимать только неотрицательные значения (см. пример 1.6). Условие неотрицательности переменного (хотя это частный случай ограничения типа неравенства) часто рассматривают как самостоятельный вид ограничений. Ограничение типа „больше или равно“ всегда можно преобразовать в ограничение типа „меньше или равно“, умножив неравенство на -1 . Поэтому задачу общего вида можно записать следующим образом:

$$\left\{ \begin{array}{l} \sum_{j=1}^n c_j x_j \rightarrow \max; \\ \sum_{j=1}^n a_{ij} x_j \leq b_i, \quad i = \overline{1, m_1}; \\ \sum_{j=1}^n a_{ij} x_j = b_i, \quad i = \overline{m_1+1, m}; \\ x_{i_1} \geq 0, \quad \dots, \quad x_{i_r} \geq 0. \end{array} \right. \quad (6.2)$$

Среди прикладных задач распространен частный случай задачи линейного программирования, в котором все переменные удовлетворяют условию неотрицательности и нет ограничений типа равенства. Таковую задачу можно записать в виде

$$\begin{cases} \sum_{j=1}^n c_j x_j \rightarrow \max; \\ \sum_{i=1}^n a_{ij} x_j \leq b_i, \quad i = \overline{1, m}; \\ x_1 \geq 0, \quad \dots, \quad x_n \geq 0. \end{cases}$$

Эту задачу будем называть *задачей линейного программирования в стандартной форме*.

Задачу линейного программирования, в которой все переменные удовлетворяют условию неотрицательности, а других ограничений типа неравенства нет, т.е. задачу вида

$$\begin{cases} \sum_{j=1}^n c_j x_j \rightarrow \max; \\ \sum_{i=1}^n a_{ij} x_j = b_i, \quad i = \overline{1, m}; \\ x_1 \geq 0, \quad \dots, \quad x_n \geq 0, \end{cases}$$

назовем *задачей линейного программирования в канонической форме*. Такая форма задачи удобна для построения методов оптимизации.

В линейном программировании важную роль играют геометрические представления и интерпретации (точнее, представления, вытекающие из линейной алгебры). Так, вполне естественно набор переменных x_1, \dots, x_n задачи линейного программирования рассматривать как n -мерный арифметический вектор x , записываемый в виде матрицы-столбца: $x = (x_1, \dots, x_n)^T$. Введем *отношение порядка в \mathbb{R}^n* , полагая $x \leq y$, если $x_i \leq y_i, i = \overline{1, n}$ (т.е. каждая компонента вектора x не превышает соответствующей компоненты вектора y). Тогда задачу линейного программирования в стандартной форме можно за-

писать с помощью матричных соотношений:

$$\begin{cases} (c, x) \rightarrow \max; \\ Ax \leq b; \\ x \geq 0, \end{cases}$$

где $A = (a_{ij})$ — матрица коэффициентов левых частей ограничений, а $b = (b_1, b_2, \dots, b_m)^T$ — вектор-столбец правых частей ограничений.

Задачу линейного программирования в канонической форме также можно записать с помощью матричных соотношений:

$$\begin{cases} (c, x) \rightarrow \max; \\ Ax = b; \\ x \geq 0. \end{cases} \quad (6.3)$$

Формально задача в стандартной форме и задача в канонической форме являются частными случаями задачи линейного программирования в общей форме. Однако любую задачу линейного программирования можно привести к стандартной или канонической форме. Покажем это.

Пусть дана задача линейного программирования общего вида. Сначала преобразуем ее так, чтобы все переменные удовлетворяли условию неотрицательности. Здесь можно использовать два приема. Первый прием состоит в использовании имеющихся в задаче ограничений типа неравенства. Например, пусть переменное x_1 не связано условием неотрицательности, а в ограничении $a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n \leq b_1$ коэффициент a_{11} при переменном x_1 отличен от нуля. Тогда, введя взамен x_1 новое переменное $x'_1 = b_1 - a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n$, вместо рассматриваемого ограничения получим условие неотрицательности $x'_1 \geq 0$.

Второй прием более прост: любое переменное x_i , не связанное условием неотрицательности, можно заменить разностью двух новых переменных $x_i = x'_i - x''_i$, каждое из которых удовлетворяет условию неотрицательности.

Отметим, что первый прием не повышает размерность задачи (иначе говоря, количество переменных) и с этой точки зрения более удобен на практике. Однако неочевидно, что использование этого приема позволяет получить условия неотрицательности для всех переменных.

Второй прием не только увеличивает размерность задачи, но и приводит к задаче, имеющей неединственное решение, так как определенное значение x_i может быть реализовано парой значений x'_i и x''_i бесконечным числом способов. В то же время второй прием позволяет, хотя и формально, но легко показать, что условия неотрицательности переменных по существу не ограничивают общности задачи линейного программирования.

Если в задаче линейного программирования все переменные неотрицательные, то для ее преобразования в стандартную форму нужно избавиться от ограничений типа равенства. Ограничения типа равенства в совокупности образуют систему линейных уравнений $Ax = b$. Разделим по отношению к этой системе все переменные на базисные и свободные. Базисные переменные единственным образом выражаются через свободные. Это значит, что базисные переменные можно исключить из задачи линейного программирования, выразив их через свободные переменные и выполнив в целевой функции и ограничениях соответствующие подстановки. В результате получим задачу, в которой оставшиеся переменные будут связаны условием неотрицательности и которая не имеет ограничений типа равенства, т.е. задачу в стандартной форме.

Отметим, что от ограничений типа равенства можно формально избавиться, заменив каждое двумя ограничениями типа неравенства. Например, ограничение

$$a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n = b_i$$

можно заменить двумя неравенствами

$$\begin{aligned} a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n &\leq b_i, \\ a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n &\geq b_i. \end{aligned}$$

Задачу линейного программирования в общей форме можно преобразовать и в каноническую форму. Для этого, как описано выше, обеспечим условие неотрицательности для каждого переменного, а затем избавимся от ограничений типа неравенства. Для ограничения вида

$$a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n \leq b_i$$

введем новое переменное $z = b_i - (a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n)$. Тогда рассматриваемое ограничение естественным образом превращается в условие неотрицательности нового переменного. Повторив этот прием для каждого ограничения типа неравенства, придем к задаче линейного программирования в канонической форме.

6.2. Графический метод решения задач линейного программирования

Если в задаче линейного программирования всего два переменных, то для решения задачи можно использовать геометрическую интерпретацию задачи.

В этом случае ограничение типа равенства $a_{i1}x_1 + a_{i2}x_2 = b_i$ на плоскости Ox_1x_2 представляет собой прямую, а каждое ограничение типа неравенства $a_{i1}x_1 + a_{i2}x_2 \leq b_i$ — полуплоскость. Если задача записана в стандартной форме

$$\begin{cases} c_1x_1 + c_2x_2 \rightarrow \max; \\ a_{i1}x_1 + a_{i2}x_2 \leq b_i, \quad i = \overline{1, m}; \\ x_1 \geq 0, \quad x_2 \geq 0, \end{cases}$$

то допустимое множество G будет пересечением некоторого множества полуплоскостей. Такое множество на плоскости представляет собой либо выпуклый многоугольник, либо отрезок (в вырожденном случае), расположенный в первом квадранте (рис. 6.1).

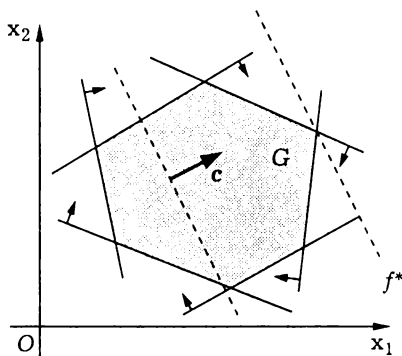


Рис. 6.1

Линии уровня целевой функции составляют семейство параллельных прямых, общим нормальным вектором которых является вектор $c = (c_1, c_2)^T$ коэффициентов целевой функции. Перемещая прямую с таким нормальным вектором в направлении этого вектора, найдем ее крайнее положение, при котором она еще пересекает многоугольник (прямая f^* на рис. 6.1). Каждая точка пересечения этой прямой

с многоугольником изображает оптимальное решение задачи, а значение функции, соответствующее этой линии уровня, есть максимальное значение целевой функции на допустимом множестве.

Пример 6.1. Рассмотрим задачу

$$\begin{cases} 3x + 2y \rightarrow \max; \\ 2x + 3y \leq 8, & 2x + y \leq 4, & -2x + y \leq 1; \\ x \geq 0, & y \geq 0. \end{cases}$$

Это задача в стандартной форме с двумя параметрами оптимизации. Ее решение представлено на рис. 6.2. Оптимальное решение изображено точкой M . Координаты точки M можно найти, решив систему уравнений

$$\begin{cases} 2x + 3y = 8, \\ 2x + y = 4. \end{cases}$$

В результате получим оптимальное решение задачи $x_* = 1$, $y_* = 2$. Оптимальное значение целевой функции равно $f^* = 7$. #

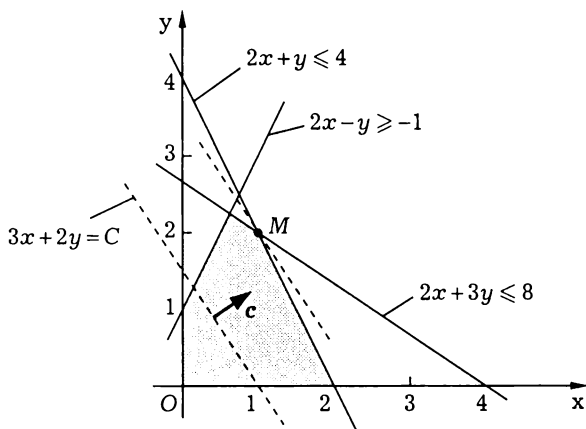


Рис. 6.2

Графический метод полезен и в задачах линейного программирования с большим числом переменных. Предположим, задача линейного программирования имеет n параметров оптимизации и $n - 2$ независимых ограничения типа равенства. Эти ограничения в совокупности составляют систему линейных уравнений $Ax = b$. В силу предположения ранг матрицы системы равен количеству уравнений $n - 2$.

Следовательно, система имеет $n - 2$ базисных переменных и 2 свободных. Выразив с помощью системы базисные переменные через свободные, получим задачу линейного программирования без ограничений типа равенства, имеющую всего два параметра оптимизации. Ее можно решить, применяя графический метод.

Пример 6.2. Рассмотрим задачу линейного программирования

$$\begin{cases} x_1 + x_2 + x_3 + x_4 \rightarrow \max; \\ 2x_1 + 3x_2 + x_3 - x_4 = 4, & x_1 - x_2 + 2x_3 + x_4 = 5; \\ x_1 \geq 0, & x_2 \geq 0, & x_3 \geq 0, & x_4 \geq 0. \end{cases}$$

В этой задаче, записанной в канонической форме, четыре параметра оптимизации и два ограничения типа равенства. В качестве базисных выберем переменные x_1, x_2 , а в качестве свободных — переменные x_3, x_4 . Решая систему двух ограничений типа равенства относительно x_1, x_2 , получим $x_1 = -\frac{7}{5}x_3 - \frac{2}{5}x_4 + \frac{19}{5}$, $x_2 = \frac{3}{5}x_3 + \frac{3}{5}x_4 - \frac{6}{5}$. Исключая переменные x_1 и x_2 из задачи, приходим к следующей задаче в стандартной форме:

$$\begin{cases} \frac{1}{5}x_3 + \frac{6}{5}x_4 + \frac{13}{5} \rightarrow \max; \\ \frac{7}{5}x_3 + \frac{2}{5}x_4 \leq \frac{19}{5}, \\ -\frac{3}{5}x_3 - \frac{3}{5}x_4 \leq -\frac{6}{5}; \\ x_3 \geq 0, & x_4 \geq 0. \end{cases}$$

Последнюю задачу можно упростить, удалив из целевой функции постоянное слагаемое, а также умножив целевую функцию и ограничения на 5 и сократив второе ограничение на 2:

$$\begin{cases} x_3 + 6x_4 \rightarrow \max; \\ 7x_3 + 2x_4 \leq 19, & -x_3 - x_4 \leq -2; \\ x_3 \geq 0, & x_4 \geq 0. \end{cases}$$

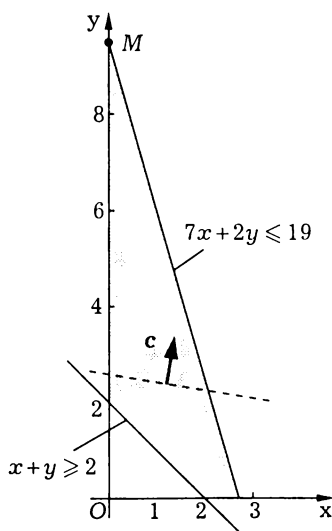


Рис. 6.3

Решение задачи представлено на рис. 6.3. Оптимальному решению соответствует точка M с координатами $x = 0, y = 9,5$.

6.3. Основы теории линейного программирования

Пусть дана задача линейного программирования в канонической форме. Ее можно представить в матричной записи (6.3). При этом полагаем, что матрица A системы ограничений имеет **максимальный строчный ранг**, т.е. ее ранг равен количеству строк m . Предположим, что базисный минор в матрице A можно выбрать в первых m столбцах. Представив матрицу A в блочном виде $A = (A_b \ A_s)$, где блок A_b состоит из первых m столбцов матрицы A , а блок A_s — из оставшихся столбцов, запишем систему ограничений задачи в виде

$$A_b x_b + A_s x_s = b, \quad (6.4)$$

где x_b — вектор-столбец, составленный из базисных переменных x_1, \dots, x_m , а x_s — вектор-столбец, составленный из свободных переменных x_{m+1}, \dots, x_n . Матрица A_b представляет собой базисный минор матрицы A и потому невырождена. Значит, матричное равенство (6.4) можно умножить на матрицу A_b^{-1} , что позволяет выразить вектор x_b через вектор x_s :

$$x_b = A_b^{-1}(b - A_s x_s).$$

Таким образом, вектор

$$x = \begin{pmatrix} x_b \\ x_s \end{pmatrix} = \begin{pmatrix} A_b^{-1}(b - A_s x_s) \\ x_s \end{pmatrix}$$

удовлетворяет системе ограничений, а при $x \geq 0$ является допустимым решением задачи линейного программирования. Особо выделим случай $x_s = 0$. В этом случае вектор x имеет вид

$$x = \begin{pmatrix} x_b \\ 0 \end{pmatrix} = \begin{pmatrix} A_b^{-1}b \\ 0 \end{pmatrix}$$

и называется **базисным решением**. Если базисное решение удовлетворяет условию неотрицательности, т.е. $A_b^{-1}b \geq 0$, то его называют **допустимым базисным решением**. Отметим, что базисное решение в общем случае не является единственным, так как базисный минор в

матрице можно выбирать по-разному. Однако таких решений конечное число, определяемое количеством базисных миноров в матрице ограничений задачи.

Оказывается, что оптимальное решение задачи линейного программирования в канонической форме следует искать среди базисных решений. Открывается возможность найти оптимальное решение задачи путем перебора всех базисных решений. Для этого нужно проверить C_n^m миноров матрицы A порядка m , выбрав среди них базисные, а затем для каждого базисного минора построить соответствующее базисное решение. Но этот подход неэффективен, а при больших размерностях задачи может оказаться практически неосуществимым: величина C_n^m может оказаться слишком большой. Остановимся на свойствах базисных решений задачи линейного программирования.

Выберем две точки x_1 и x_2 и рассмотрим множество точек вида $x = \alpha x_1 + (1 - \alpha)x_2$, $0 \leq \alpha \leq 1$. Это множество на плоскости или в пространстве представляет собой отрезок. В общем случае оно также называется *отрезком в \mathbb{R}^n* . Точки x_1, x_2 называются концами отрезка, остальные точки отрезка — внутренними.

Точка x выпуклого множества $M \subset \mathbb{R}^n$ называется *крайней точкой* этого множества, если она не является внутренней точкой какого-либо отрезка, целиком содержащегося в M . Любая крайняя точка является граничной точкой выпуклого множества, но очевидно, что не все граничные точки являются крайними.

Множество может не иметь крайних точек (например, полуплоскость в \mathbb{R}^2). Однако известно, что любое замкнутое ограниченное выпуклое множество $M \subset \mathbb{R}^n$ имеет крайние точки, причем любая точка $x \in M$ может быть представлена в виде выпуклой комбинации крайних точек x_1, \dots, x_k множества M , т.е. в виде

$$x = \alpha_1 x_1 + \alpha_2 x_2 + \dots + \alpha_k x_k,$$

где все коэффициенты α_i неотрицательны и в сумме дают 1. На таком множестве линейная функция достигает минимума в одной из крайних точек, хотя, возможно, есть точки минимума, не являющиеся крайними.

Отметим, что допустимое множество задачи линейного программирования, в частности задачи в канонической форме, является выпуклым множеством, как пересечение выпуклых множеств $Ax = b$

и $x \geq 0$. Отсюда вытекает важная роль крайних точек допустимого множества в задаче линейного программирования.

Лемма 6.1. Допустимое решение $x = (x_1 \ x_2 \ \dots \ x_n)^T$ задачи линейного программирования в канонической форме (6.3) является крайней точкой множества допустимых решений тогда и только тогда, когда система всех столбцов матрицы A , отвечающих ненулевым значениям x_i , $i = \overline{1, n}$, линейно независима.

◀ Можно считать, не теряя общности, что вектор x имеет вид

$$x = (x_1 \ x_2 \ \dots \ x_r \ 0 \ \dots \ 0)^T, \quad x_i \neq 0, \quad i = \overline{1, r}. \quad (6.5)$$

Если вектор x не является крайней точкой допустимого множества, то существует такой ненулевой вектор u и такое достаточно малое число $\delta > 0$, что вектор $x + tu$, $|t| < \delta$, принадлежит допустимому множеству, т.е. $A(x + tu) = b$, $x + tu \geq 0$. Последнее равенство, записанное в виде $x \geq -tu$, позволяет сделать вывод, что все компоненты u_i вектора u с номерами $i = r+1, \overline{n}$ равны нулю, так как для таких компонент имеем неравенства $u_i \leq 0$ (при положительных значениях t) и $-u_i \leq 0$ (при отрицательных значениях t).

Поскольку x принадлежит допустимому множеству, выполняется равенство $Ax = b$. С учетом равенства $A(x + tu) = b$ заключаем, что $Au = 0$. Последнее равенство можно записать в виде

$$a_1 u_1 + a_2 u_2 + a_r u_r + a_{r+1} u_{r+1} \dots a_n u_n = 0,$$

а с учетом доказанных соотношений $u_i = 0$, $i = \overline{r+1, n}$,

$$a_1 u_1 + a_2 u_2 + a_r u_r = 0. \quad (6.6)$$

Из соотношения $u \neq 0$ вытекает, что одно из чисел u_i , $i = \overline{1, r}$, не равно нулю. Тем самым показано, что если вектор x вида (6.5) не является крайней точкой допустимого множества, то столбцы a_1, a_2, \dots, a_r линейно зависимы.

Предположим, что столбцы a_1, a_2, \dots, a_r линейно зависимы. Тогда существуют коэффициенты u_1, u_2, \dots, u_r , одновременно не равные нулю, для которых выполняется равенство (6.6). Рассмотрим вектор $u = (u_1 \ \dots \ u_r \ 0 \ \dots \ 0)^T$. Равенство (6.6) означает, что $Au = 0$. Следовательно, $A(x + tu) = Ax = b$ для любого числа t , т.е. вектор $x + tu$ удовлетворяет ограничениям типа равенства задачи

(6.3). Покажем, что при малых значениях t этот вектор удовлетворяет и условию неотрицательности. Пусть $\rho = \max_{i=1, r} |u_i|$. Тогда $\rho > 0$.

Положим $\delta = \min_{i=1, r} \frac{|x_i|}{\rho}$. Так как $x_i > 0, i = \overline{1, r}$, то $\delta > 0$. При этом при $|t| < \delta$ выполняются неравенства

$$x_i + tu_i \geq x_i - |t||u_i| > x_i - \delta|u_i| \geq x_i - \delta\rho \geq 0.$$

Следовательно, $x + u \geq 0$, $|t| < \delta$, и вектор x не является крайней точкой допустимого множества задачи (6.3). ►

Теорема 6.1. Допустимое решение задачи линейного программирования в канонической форме является допустимым базисным тогда и только тогда, когда оно представляет собой крайнюю точку допустимого множества задачи.

◄ У допустимого базисного решения ненулевыми могут быть только те компоненты, которые отвечают базисным переменным, т.е. система столбцов матрицы A ограничений, отвечающих ненулевым компонентам вектора, есть часть системы базисных столбцов, а потому линейно независима. Следовательно, допустимое базисное решение есть крайняя точка допустимого множества.

У крайней точки x допустимого множества количество ненулевых значений не превышает ранга матрицы ограничений A , а значит, количества ограничений m . Соответствующие столбцы матрицы A линейно независимы, и их можно дополнить до набора базисных столбцов матрицы A . Всем оставшимся столбцам будут соответствовать нулевые компоненты рассматриваемого вектора x . Следовательно, вектор x является базисным решением. Поскольку он принадлежит допустимому множеству, то он есть допустимое базисное решение. ►

6.4. Симплекс-метод

Согласно результатам, изложенным выше (см. разд. 6.3), поиск оптимального решения задачи линейного программирования в канонической форме следует искать среди допустимых базисных решений, или, что то же самое, среди крайних точек допустимого множества задачи. Полный перебор допустимых базисных решений неэффективен, но можно процесс поиска оптимального решения строить, переходя

от одного допустимого базисного решения к другому. В этом состоит суть *симплекс-метода*.

Симплекс-метод можно проиллюстрировать следующим образом. Пусть требуется максимизировать линейную функцию в области, ограниченной выпуклым многогранником в \mathbb{R}^3 . Систему координат можно выбрать так, что целевая функция будет иметь вид $f(x_1, x_2, x_3) = \alpha x_3$, $\alpha > 0$. При таком расположении системы координат задача заключается в поиске самой высокой точки многогранника (имеется в виду, что высота определяется третьей координатой x_3). Ясно, что такая точка является вершиной многогранника (правда, возможен случай, когда точками максимума будут все точки некоторого ребра или некоторой грани; но и в этом случае в качестве решения можно выбрать вершину многогранника).

Процесс поиска можно построить так. Выбираем одну из вершин многогранника в качестве начальной. На каждом шаге мы из текущей вершины переходим в одну из соседних (т.е. вершин, соединенных с текущей ребром многогранника), выбирая среди таких вершин самую высокую. Если текущая вершина расположена выше всех своих соседей, то эта вершина будет оптимальной.

Рассмотренная задача — частный случай задачи линейного программирования, в которой допустимое множество — внутренность многогранника — описывается системой линейных неравенств. Каждая вершина многогранника — это крайняя точка допустимого множества. Вершину многогранника можно определить как точку пересечения трех примыкающих к ней граней; алгебраически это означает, что в системе ограничений типа неравенства по крайней мере три выполняются как равенства. Если ограничение типа неравенства удовлетворяется в данной точке как равенство, его называют *активным ограничением*. А если оно удовлетворяется как строгое неравенство, то его называют *неактивным (пассивным) ограничением*.

Итак, вершина многогранника — это точка допустимого множества, в котором по крайней мере три ограничения активны. Переход по ребру в соседнюю вершину состоит в замене одной из трех определяющих граней новой (две сохраняющиеся грани определяют прямую, на которой расположено ребро многогранника, соединяющее текущую вершину с новой). С алгебраической точки зрения замена грани представляет собой смену активного ограничения.

Все описанные представления относятся к любой задаче линейного программирования, в том числе и к задачам в канонической

форме. В таких задачах есть m ограничений типа равенства и n условий неотрицательности, являющихся ограничениями типа неравенства. Конкретная крайняя точка (допустимое базисное решение) определяется системой из $n - m$ активных ограничений, или, другими словами, нулевыми значениями $n - m$ переменных. Отметим, что приведенные рассуждения нестрогие и обходят стороной некоторые особые ситуации. Так, допустимое базисное решение — это не только $n - m$ нулевых значений переменных, но и условие базисности остальных переменных (соответствующие столбцы матрицы A системы ограничений должны быть линейно независимыми).

В свете изложенного процесс поиска оптимального решения можно представить себе как последовательный переход от одного допустимого базисного решения к „соседнему“ допустимому базисному решению, дающему большее значение целевой функции. „Соседство“ состоит в том, что новое допустимое базисное решение получается из исходного заменой одного из свободных переменных; иначе говоря, из базиса выводится одно переменное и в него включается другое, которое до этого было свободным.

Остановимся подробнее на указанном переходе. Полагаем, что текущее допустимое базисное решение x имеет вид

$$x^0 = (x_1^0 \ x_2^0 \ \dots \ x_m^0 \ 0 \ \dots \ 0)^T,$$

т.е. базисными являются первые m (по количеству ограничений) переменных x_1, \dots, x_m . Совокупность базисных переменных будем называть *базисом задачи линейного программирования*.

Выводимое из базиса переменное. Предположим, что переменное $x_i, i \leq m$, выводится из набора базисных, а свободное переменное $x_j, j > m$, вводится в набор базисных.

Заданное допустимое базисное решение (точнее, заданное разделение переменных на свободные и базисные) позволяет исключить базисные переменные из задачи следующим образом. Пусть, как и выше (см. разд. 6.3), $A_b = (a_1, \dots, a_m)$ — матрица из базисных столбцов матрицы A ограничений задачи, $A_s = (a_{m+1}, \dots, a_n)$ — матрица свободных столбцов этой матрицы, $x_b = (x_1 \ \dots \ x_m)^T$ — вектор-столбец базисных переменных, $x_s = (x_{m+1} \ \dots \ x_n)^T$ — вектор-столбец свободных переменных. Тогда систему ограничений задачи можно записать в виде (6.4). Из этого представления получим

$$x_b + A_b^{-1} A_s x_s = A_b^{-1} b.$$

Введем обозначения: $\hat{A}_s = A_b^{-1}A_s$, $\hat{b} = A_b^{-1}b$. Это позволяет несколько упростить систему ограничений:

$$x_b + \hat{A}_s x_s = \hat{b}. \quad (6.7)$$

Пусть \hat{a}_{m+1} , \hat{a}_{m+2} , ..., \hat{a}_n — столбцы матрицы \hat{A}_s (номера столбцов совпадают с номерами соответствующих переменных). При замене в базисе переменного x_i на переменное x_j нужно использовать нулевое значение переменного x_i и неотрицательное значение x_j . Из i -й строки соотношений (6.7), учитывая, что все свободные переменные, кроме x_j , нулевые, получаем $x_i + \hat{a}_{ij}x_j = \hat{b}_i$, где \hat{a}_{ij} — элемент матрицы \hat{A} в i -й строке и $(j - m)$ -м столбце. Нулевое значение x_i порождается значением $x_j = \hat{b}_i/\hat{a}_{ij}$. Отметим, что \hat{b}_i совпадает со значением x_i при нулевых значениях всех свободных переменных, т.е. с компонентой x_i^0 исходного допустимого базисного значения. Значит, $\hat{b}_i \geq 0$ и условие неотрицательности x_j равносильно неравенству $\hat{a}_{ij} > 0$. Впрочем, возможен случай $\hat{b}_i = 0$. Тогда при $x_j = 0$ и $x_i = 0$. В этом случае происходит замена в базисе одного нулевого переменного другим, а допустимое базисное решение на самом деле не изменяется. Чтобы замена в базисе переменного x_i на переменное x_j снова приводила к набору базисных переменных, должно выполняться условие $\hat{a}_{ij} \neq 0$.

Случай, когда хотя бы одно из базисных переменных x_s , $s = \overline{1, m}$, имеет нулевое значение \hat{b}_s , является с точки зрения симплекс-метода особым. В этом случае говорят о **вырожденном допустимом базисном решении**. Если базисное решение вырождено, то этому решению соответствуют несколько наборов базисных переменных. Если задача линейного программирования имеет хотя бы одно вырожденное допустимое базисное решение, ее называют **вырожденной**.

При изменении значения x_j все базисные переменные изменяют значения. Исходное значение переменного x_s есть x_s^0 и совпадает с \hat{b}_s . Согласно равенству (6.7) при $x_j = \hat{b}_i/\hat{a}_{ij}$ получаем

$$x_s = \hat{b}_s - \hat{a}_{sj}x_j = \hat{b}_s - \hat{a}_{sj}\frac{\hat{b}_i}{\hat{a}_{ij}}.$$

Условие неотрицательности переменных в новом векторе таково:

$$\hat{b}_s - \hat{a}_{sj}\frac{\hat{b}_i}{\hat{a}_{ij}} \geq 0, \quad s = \overline{1, m}. \quad (6.8)$$

Это условие автоматически выполняется для тех номеров s , для которых $\hat{a}_{sj} \leq 0$. Поэтому можно ограничиться множеством S_j номеров s , для которых $\hat{a}_{sj} > 0$. Для таких номеров неравенство (6.8) равносильно неравенству

$$\frac{\hat{b}_s}{\hat{a}_{sj}} \geq \frac{\hat{b}_i}{\hat{a}_{ij}}, \quad s = \overline{1, m}. \quad (6.9)$$

Итак, среди всех базисных переменных x_s , $s = \overline{1, m}$, выводимое переменное x_i отличается тем, что, во-первых, величина \hat{a}_{ij} положительна, а во-вторых, среди всех таких переменных имеет наименьшее отношение \hat{b}_i/\hat{a}_{ij} . Эти условия позволяют легко определить выводимое переменное x_i , если уже известно вводимое переменное x_j . Как правило, переменное x_i однозначно определяется переменным x_j : для этого достаточно, чтобы минимальное значение отношения \hat{b}_s/\hat{a}_{sj} , $s \in S_j$, достигалось только для одного номера s .

Вводимое в базис переменное. Теперь обсудим, каким образом следует выбирать вводимое в базис переменное x_j . Есть очевидное ограничение: должен быть хотя бы один индекс $i \leq m$, для которого $\hat{a}_{ij} > 0$. Но еще не учтено условие, что при переходе к новому базисному решению целевая функция не должна убывать. Целевую функцию $f(x) = (c, x)$ представим в виде $f(x) = c^T x$, используя для записи скалярного произведения матричную запись. Разделив вектор-столбец c на блоки c_b и c_s так же, как и вектор x , можем записать

$$f(x) = c_b^T x_b + c_s^T x_s.$$

Используя равенство (6.7), из записи целевой функции можем исключить базисные переменные:

$$\begin{aligned} f(x) &= c_b^T (\hat{b} - \hat{A}_s x_s) + c_s^T x_s = c_b^T \hat{b} + (c_s^T - c_b^T \hat{A}_s) x_s = \\ &= c_b^T \hat{b} + (c_s - \hat{A}_s^T c_b)^T x_s = \hat{f} + d^T x_s, \end{aligned}$$

где $\hat{f} = c_b^T \hat{b}$ — значение целевой функции на допустимом базисном решении x^0 ; $d = c_s - \hat{A}_s^T c_b$ — вектор *симплекс-разностей*, соответствующих свободным переменным.

Если переменное x_j получает значение $x_j = \hat{b}_i / \hat{a}_{ij}$, а остальные свободные переменные остаются нулевыми, то целевая функция примет значение

$$f(x) = \hat{f} + d_j x_j = \hat{f} + d_j \frac{\hat{b}_i}{\hat{a}_{ij}}.$$

Изменение значения целевой функции

$$f(x) - f(x^0) = d_j x_j = d_j \frac{\hat{b}_i}{\hat{a}_{ij}}$$

положительно, если $d_j > 0$.

В принципе можно выбирать в качестве вводимого в базис переменного x_j любое свободное переменное, у которого симплекс-разность положительна. Наибольшее приращение целевой функции достигается при введении переменного, для которого максимальна величина $d_j \frac{\hat{b}_i}{\hat{a}_{ij}}$. Но чтобы вычислить эту величину, нужно определиться и с выводимым переменным, т.е. фактически построить новое допустимое базисное решение. Но если уже имеется лучшее допустимое базисное решение, то зачем возвращаться к прежнему? Можно использовать более простой подход, выбирая среди всех симплекс-разностей наибольшую.

Замечание 6.1. Если все симплекс-разности d_j неположительны, то для любых значений $x_{m+1}, x_{m+2}, \dots, x_n$

$$f(x) = \hat{f} + d^T x_s = \hat{f} + \sum_{j=m+1}^n d_j x_j \leq \hat{f} = f(x^0).$$

Следовательно, допустимое базисное решение x^0 — оптимальное решение задачи. Если же есть положительные симплекс-разности d_j , а допустимое базисное решение x^0 невырождено, то, выбрав номер j с $d_j > 0$, а затем выбрав номер i выводимого из базиса переменного, получим новое допустимое базисное решение x^1 , для которого $f(x^1) > f(x^0)$. Значит, невырожденное допустимое базисное решение, дающее хотя бы одну положительную симплекс-разность, не является оптимальным решением.

Условие неположительности симплекс-разностей можно записать в виде $c_s \leq \hat{A}_s^T c_b$, или $c_s \leq A_s^T (A_b^{-1})^T c_b$. Введя обозначение $\gamma = (A_b^{-1})^T c_b$, придем к неравенству $c_s \leq A_s^T \gamma$. Поскольку $c_b = A_b^T \gamma$, заключаем, что

$$\begin{pmatrix} c_b \\ c_s \end{pmatrix} \leq \begin{pmatrix} A_b^T \gamma \\ A_s^T c_b \end{pmatrix} = \begin{pmatrix} A_b^T \\ A_s^T \end{pmatrix} \gamma.$$

Таким образом, допустимое базисное решение оптимально тогда и только тогда, когда

$$c \leq A^T \gamma. \quad (6.10)$$

Вектор $\gamma = (A_b^{-1})^T c_b$ будем называть **вектором оценок**.

Особо отметим случай вырожденного допустимого базисного решения. В этом случае смена базиса не вызывает изменения допустимого базисного решения и фактического продвижения к оптимальному решению не происходит. Избегая этой ситуации, можно попробовать изменить вводимое в базис переменное, выбрав другую положительную симплекс-разность. Но это не всегда возможно. Сама по себе смена базиса для данного допустимого базисного решения не страшна, однако может случиться, что после перебора нескольких базисов данного допустимого базисного решения произойдет возврат к уже рассматривавшемуся базису. Произойдет заикливание, которое не позволит завершить процесс решения задачи несмотря на то, что количество допустимых базисных решений и количество базисов в задаче конечно.

На практике такое происходит редко, но можно исключить и эту редкую ситуацию, если все базисы определенным образом упорядочить, используя для этого в первую очередь значение целевой функции на соответствующем допустимом базисном решении, а во вторую очередь состав переменных, входящих в базис. Тогда базисы не будут повторяться. Можно показать, что при наличии положительных симплекс-разностей уточненный итерационный процесс перехода от одного допустимого базисного решения к другому приведет к новому „большому“ базису. Следовательно, итерационный процесс рано или поздно остановится, и произойдет это в момент, когда для очередного базиса все симплекс-разности окажутся неположительными. Это будет означать получение оптимального решения.

Алгоритм применения симплекс-метода. Применение симплекс-метода предполагает, что изначально известно какое-либо допустимое базисное решение x^0 задачи, которое называют *начальным допустимым базисным решением*. Для этого решения выбирают начальный базис x_b^0 , преобразуют ограничения задачи в форму (6.7), разрешенную относительно базисных переменных, и исключают базисные переменные из целевой функции, вычисляя тем самым симплекс-разности d_j^0 для начального базиса. Далее на каждой итерации процесса производят смену базиса в соответствии со следующей процедурой:

1) среди симплекс-разностей выбирают положительную d_j (если таких несколько, можно выбрать максимальную); номер j симплекс-разности указывает переменное, которое будет вводиться в базис;

2) отбирают все номера s базисных переменных, для которых элементы \hat{a}_{sj} положительны, и среди них выбирают номер i , для которого отношение \hat{b}_s/\hat{a}_{sj} минимально. Этот номер определит выводимое из базиса переменное;

3) корректируют систему ограничений задачи и целевую функцию, исключая из уравнений переменное x_j , уже не являющееся свободным; при этом в уравнениях появляется новое свободное переменное x_i ; корректировка приводит к новым параметрам \hat{a}_{sj} , \hat{b}_s , d_j .

Процесс вычислений удобно реализовать в матричной форме, записывая параметры вычислений в специальную таблицу, называемую *симплекс-таблицей*. В симплекс-таблице $m + 1$ строка и $n + 1$ столбец. Строки симплекс-таблицы соответствуют базисным переменным и целевой функции, столбцы — всем переменным задачи и правым частям ограничений (они же — значения базисных переменных текущего базиса). Порядок строк таблицы не является существенным, но каждая строка соответствует определенному базисному переменному или целевой функции. Поэтому в записи таблицы используется столбец с заголовками строк. Строка, соответствующая целевой функции, располагается либо первой, либо последней. Для удобства используют и заголовки столбцов, хотя столбцы занимают фиксированное положение, и можно было бы обойтись без заголовков. Пример симплекс-таблицы для набора базисных переменных x_1, \dots, x_m представлен в табл. 6.1.

Рассмотрим использование симплекс-метода на конкретном примере.

Таблица 6.1

	b	x_1	x_2	\dots	x_m	x_{m+1}	\dots	x_n
x_1	\hat{b}_1	1	0	\dots	0	$\hat{a}_{1,m+1}$	\dots	\hat{a}_{1n}
x_2	\hat{b}_2	0	1	\dots	0	$\hat{a}_{2,m+1}$	\dots	\hat{a}_{2n}
\dots								
x_m	\hat{b}_m	0	0	\dots	1	$\hat{a}_{m,m+1}$	\dots	\hat{a}_{mn}
$-f$	0	0	0	\dots	0	d_{m+1}	\dots	d_n

Пример 6.3. Рассмотрим задачу линейного программирования

$$\begin{cases} f(x_1, x_2) = 3x_1 - x_2 \rightarrow \max; \\ 2x_1 - x_2 \leq 4, \quad x_1 - 2x_2 \leq 2, \quad x_1 + x_2 \leq 5; \\ x_1 \geq 0, \quad x_2 \geq 0. \end{cases}$$

Запишем задачу в стандартной форме ($y_1 = x_1, y_2 = x_2, y_3, y_4, y_5$ — новые переменные):

$$\begin{cases} 3y_1 - y_2 \rightarrow \max; \\ 2y_1 - y_2 + y_3 = 4, \quad y_1 - 2y_2 + y_4 = 2, \quad y_1 + y_2 + y_5 = 5; \\ y_k \geq 0, \quad k = \overline{1, 5}. \end{cases}$$

В качестве начального можно использовать допустимое базисное решение $Y = (0 \ 0 \ 4 \ 2 \ 5)^T$, на котором целевая функция принимает нулевое значение. Базисными переменными являются y_3, y_4, y_5 . Заполним начальную симплекс-таблицу (табл. 6.2).

Таблица 6.2

	b	y_1	y_2	y_3	y_4	y_5
y_3	4	2	-1	1	0	0
y_4	2	1	-2	0	1	0
y_5	5	1	1	0	0	1
$-f$	0	3	-1	0	0	0

Наибольшая положительная симплекс-разность равна 3 и расположена в столбце y_1 . Значит, y_1 — вводимое в базис переменное. Все

три элемента столбца y_1 , соответствующие базисным переменным, положительны, для поиска выводимого из базиса переменного нужно сравнить три отношения $\frac{4}{2}, \frac{2}{1}, \frac{5}{1}$. Наименьшее значение имеют первые два отношения. Значит, из базиса можно выводить переменное y_3 или переменное y_4 . Остановимся на втором варианте.

Итак, вводимое переменное y_1 , выводимое y_4 . На пересечении строки y_4 и столбца y_1 находится число 1 (в табл. 6.2 оно выделено полужирным шрифтом), которое называется *ведущим элементом* симплекс-таблицы. Строка y_4 , в которой находится ведущий элемент, называется *ведущей строкой*.

Для преобразования симплекс-таблицы на 1-й итерации необходимо, используя ведущую строку, обнулить все элементы столбца y_1 , кроме ведущего. Ведущий элемент должен преобразоваться в 1, для чего ведущую строку достаточно разделить на ведущий элемент. Результаты преобразований представлены в табл. 6.3.

Таблица 6.3

	b	y_1	y_2	y_3	y_4	y_5
y_3	0	0	3	1	-2	0
y_1	2	1	-2	0	1	0
y_5	3	0	3	0	-1	1
$-f$	-6	0	5	0	-3	0

После 1-й итерации осталась одна положительная симплекс-разность в столбце y_2 . Ведущей является строка y_3 (ведущий элемент в ней выделен). Обращаем внимание на то, что в результате 1-й итерации получено вырожденное допустимое базисное решение, поскольку в столбце b присутствует нулевой элемент (естественно, он оказывается в ведущей строке). Выбрав выводимое из базиса и вводимое в базис переменные, преобразуем симплекс-таблицу, обнуляя в столбце выводимого переменного все элементы, кроме ведущего. Результаты 2-й итерации представлены в табл. 6.4.

После 2-й итерации имеем единственную положительную симплекс-разность, расположенную в столбце y_4 . Ведущей строкой является y_5 . Результаты 3-й итерации представлены в табл. 6.5.

В табл. 6.5 все симплекс-разности отрицательны. Значит, эта симплекс-таблица соответствует оптимальному решению. Значения базисных переменных указаны в столбце b , их имена — в первом

Таблица 6.4

	b	y_1	y_2	y_3	y_4	y_5
y_2	0	0	1	$1/3$	$-2/3$	0
y_1	2	1	0	$2/3$	$-1/3$	0
y_5	3	0	0	-1	1	1
$-f$	-6	0	0	$-5/3$	$1/3$	0

Таблица 6.5

	b	y_1	y_2	y_3	y_4	y_5
y_2	2	0	1	$-1/3$	0	$2/3$
y_1	3	1	0	$1/3$	0	$1/3$
y_4	3	0	0	-1	1	1
$-f$	-7	0	0	$-4/3$	0	$-1/3$

столбце. Свободные переменные имеют нулевые значения. Таким образом, с помощью симплекс-метода получено оптимальное решение $y_* = (3 \ 2 \ 0 \ 3 \ 0)^T$. В столбце b также присутствует оптимальное значение $f_* = 7$ целевой функции с противоположным знаком.

Теперь нетрудно вернуться к исходной задаче линейного программирования, учитывая, что y_3, y_4, y_5 — это дополнительные переменные, а y_1 и y_2 — переменные исходной задачи. Итак, $x_1 = y_1 = 3$, $x_2 = y_2 = 2$. Решение задачи графическим методом представлено на рис. 6.4.

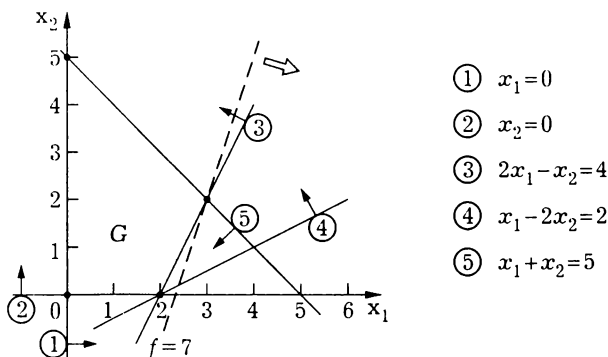


Рис. 6.4

Отметим, что крайняя точка $(2, 0)$ допустимого множества задачи расположена на пересечении трех прямых, отвечающих трем активным ограничениям. Именно эта крайняя точка и привела в задаче канонического вида к вырожденному допустимому базисному решению, которому соответствуют три пары свободных переменных: y_2 и y_3 , y_2 и y_4 , y_3 и y_4 . Иначе говоря, указанное допустимое базисное решение имеет три базиса.

Попав в точку $(2, 0)$ после 1-й итерации, мы имели свободные переменные y_2 и y_4 , что на рис. 6.4 соответствует прямым с теми же номерами. Выбор переменного y_2 для ввода в базис означает перемещение в новую крайнюю точку по ребру, определяемому оставшимися свободными переменными, т.е. по прямой с номером 4. Однако на этой прямой нет ни одной стороны многоугольника. В результате мы остались в этой крайней точке, но сменили пару прямых, ее определяющих. Замена 2-й прямой на 3-ю позволила на следующей итерации по 3-й прямой попасть в вершину, соответствующую оптимальному решению. #

6.5. Построение начального допустимого базисного решения

В некоторых случаях начальное допустимое базисное решение можно получить путем непосредственного анализа ограничений задачи. Например, пусть задача линейного программирования имеет вид

$$\begin{cases} (c, x) \rightarrow \max; \\ Ax \leq b; \\ x \geq 0, \end{cases}$$

где $b \geq 0$, т.е. в задаче нет ограничений типа равенства, все ограничения типа неравенства имеют вид „меньше или равно“, причем их правые части неотрицательны. После приведения к каноническому виду задача будет иметь вид

$$\begin{cases} (c, x) \rightarrow \max; \\ Ax + y = b; \\ x \geq 0, \quad y \geq 0, \end{cases}$$

где y — вектор-столбец дополнительных переменных. Нетрудно увидеть, что вектор $(x^T \ y^T)^T$, для которого $x = 0$, $y = b - Ax$, есть допустимое базисное решение. Именно таким способом получено начальное допустимое базисное решение в примере 6.3.

На практике, когда задача содержит сотни ограничений и тысячи переменных, таким способом построить начальное допустимое базисное решение сложно. Существует общий метод решения этой задачи. Он состоит в решении некоторой вспомогательной задачи линейного программирования.

Пусть дана задача линейного программирования в канонической форме

$$\begin{cases} (c, x) \rightarrow \max; \\ Ax = b; \\ x \geq 0. \end{cases}$$

Полагаем, что в этой задаче матрица A типа $m \times n$ имеет максимальный строчный ранг m и все компоненты b_i вектора b правых частей ограничений неотрицательны. Рассмотрим вспомогательную задачу

$$\begin{cases} -y_1 - y_2 - \dots - y_m \rightarrow \max; \\ Ax + y = b; \\ x \geq 0, \quad y \geq 0, \end{cases}$$

где $y = (y_1 \ y_2 \ \dots \ y_m)^T$ — вектор-столбец *искусственных переменных*.

Как было отмечено, можно сразу указать допустимое базисное решение вспомогательной задачи: $x = 0$, $y = b$. Целевая функция вспомогательной задачи не превышает нуля. Можно показать, что в силу этого рассматриваемая задача достигает максимума. Если максимальное значение целевой функции вспомогательной задачи равно нулю, то, как легко увидеть, для любого оптимального решения вспомогательной задачи искусственные переменные равны нулю. Значит, любое оптимальное решение вспомогательной задачи дает (после удаления искусственных переменных) допустимое решение основной задачи.

Зная начальное допустимое базисное решение вспомогательной задачи, симплекс-методом найдем ее оптимальное базисное решение. В этом решении из $n + m$ переменных в нуль обращаются n переменных, включая все искусственные переменные. Следовательно, совокупность основных переменных содержит $n - m$ нулевых переменных, а столбцы матрицы A , соответствующие ненулевым переменным, линейно независимы. Это означает, что, удалив из оптимального базисного решения вспомогательной задачи искусственные переменные, получим допустимое базисное решение основной задачи.

Отметим, что во многих случаях задача линейного программирования в канонической форме получается из поставленной задачи введением дополнительных переменных на каждое ограничение типа неравенства. Вспомогательная задача формируется дополнительным введением искусственных переменных. Общее количество фиктивных переменных, отсутствующих в исходной задаче, может быть довольно значительным, а это усложняет вычисления.

Количество фиктивных переменных можно сократить, введя искусственные переменные только для части ограничений. Пусть дана задача линейного программирования общего вида (6.1), в которой все правые части неотрицательны. Выполнения этого условия можно добиться, умножая, если необходимо, неравенства и уравнения на -1 и соответствующим образом упорядочивая их. Преобразуем эту задачу в каноническую форму, добавив в ограничения типа неравенства дополнительные переменные:

$$\left\{ \begin{array}{l} \sum_{j=1}^n c_j x_j \rightarrow \max; \\ \sum_{j=1}^n a_{ij} x_j + y_i = b_i, \quad i = \overline{1, m_1}; \\ \sum_{j=1}^n a_{ij} x_j = b_i, \quad i = \overline{m_1+1, m_2}; \\ \sum_{j=1}^n a_{ij} x_j - y_i = b_i, \quad i = \overline{m_2+1, m}. \end{array} \right. \quad (6.11)$$

Формируем вспомогательную задачу, при этом искусственные переменные вводим только во второй и третий блоки ограничений:

$$\left\{ \begin{array}{l} - \sum_{i=m_1+1}^m z_i \rightarrow \max; \\ \sum_{i=1}^n a_{ij}x_j + y_i = b_i, \quad i = \overline{1, m_1}; \\ \sum_{i=1}^n a_{ij}x_j + z_i = b_i, \quad i = \overline{m_1+1, m_2}; \\ \sum_{i=1}^n a_{ij}x_j - y_i + z_i = b_i, \quad i = \overline{m_2+1, m}. \end{array} \right.$$

В этой задаче допустимое базисное решение имеет вид

$$\begin{aligned} x_i &= 0, \quad i = \overline{1, m}; \\ y_i &= \begin{cases} b_i, & i = \overline{1, m_1}; \\ 0, & i = \overline{m_1+1, m}; \end{cases} \\ z_i &= b_i, \quad i = \overline{m_1+1, m}. \end{aligned}$$

Найдя оптимальное базисное решение этой задачи и удалив из него искусственные переменные, получим допустимое базисное решение задачи (6.11) канонического вида.

Пример 6.4. Рассмотрим задачу линейного программирования

$$\left\{ \begin{array}{l} 3x_1 + 4x_2 \rightarrow \max; \\ x_1 + x_2 \leq 20, \quad -x_1 + 4x_2 \leq 20, \quad x_1 \geq 10, \quad x_2 \geq 5; \\ x_1 \geq 0, \quad x_2 \geq 0. \end{array} \right. \quad (6.12)$$

Для преобразования задачи в стандартную форму вводим четыре дополнительных переменных:

$$\left\{ \begin{array}{l} 3x_1 + 4x_2 \rightarrow \max; \\ x_1 + x_2 + y_1 = 20, \quad -x_1 + 4x_2 + y_2 = 20, \\ x_1 - y_3 = 10, \quad x_2 - y_4 = 5; \\ x_1, x_2 \geq 0, \quad y_1, y_2, y_3, y_4 \geq 0. \end{array} \right. \quad (6.13)$$

В исходной задаче можно ограничиться введением искусственных переменных во второе и третье ограничения, являющиеся ограничениями вида „больше или равно“ с неотрицательной правой частью. Вспомогательная задача содержит восемь переменных (из них два искусственных) и имеет вид

$$\begin{cases} -z_3 - z_4 \rightarrow \max; \\ x_1 + x_2 + y_1 = 20, & -x_1 + 4x_2 + y_2 = 20, \\ x_1 - y_3 + z_3 = 10, & x_2 - y_4 + z_4 = 5; \\ x_1, x_2 \geq 0, & y_1, y_2, y_3, y_4 \geq 0, \quad z_3, z_4 \geq 0. \end{cases} \quad (6.14)$$

Допустимым базисным решением вспомогательной задачи является вектор со значениями переменных $x_1 = x_2 = 0$, $y_1 = y_2 = 20$, $y_3 = y_4 = 0$, $z_3 = 10$, $z_4 = 5$. Базисными переменными будут y_1 , y_2 , z_3 , z_4 .

Вспомогательную задачу линейного программирования решаем симплекс-методом с использованием симплекс-таблиц. Ограничения уже представлены в нужной форме, остается исключить базисные переменные из целевой функции. Выражаем базисные переменные z_3 , z_4 через свободные:

$$z_3 = 10 - x_1 + y_3, \quad z_4 = 5 - x_2 + y_4.$$

Таким образом, целевая функция имеет вид

$$\varphi = -15 + x_1 + x_2 - y_3 - y_4.$$

Решение вспомогательной задачи представлено в табл. 6.6.

Итак, максимальное значение целевой функции равно нулю. Следовательно, допустимое множество задачи (6.13) непусто. Из вспомогательной задачи получаем допустимое базисное решение задачи (6.13): $x_1 = 10$, $x_2 = 5$, $y_1 = 5$, $y_2 = 10$, $y_3 = y_4 = 0$. Целевая функция после исключения базисных переменных имеет вид $f = 50 + 3y_3 + 4y_4$. Решение задачи (6.13) представлено в табл. 6.7.

Оптимальное решение задачи (6.13) имеет вид $x_1 = 12$, $x_2 = 8$, $y_1 = y_2 = 0$, $y_3 = 2$, $y_4 = 3$. Ему соответствует оптимальное значение целевой функции $f = 68$. Оставляя только основные переменные x_1 и x_2 , получаем оптимальное решение исходной задачи (6.12): $x_1 = 12$, $x_2 = 8$; соответствующее значение целевой функции $f = 68$.

Таблица 6.6

Итерация	Базисные переменные	b	x_1	x_2	y_1	y_2	y_3	y_4	z_3	z_4
0	y_1	20	1	1	1	0	0	0	0	0
	y_2	20	-1	4	0	1	0	0	0	0
	z_3	10	1	0	0	0	-1	0	1	0
	z_4	5	0	1	0	0	0	-1	0	1
	$-\varphi$	15	1	1	0	0	-1	-1	0	0
1	y_1	10	0	1	1	0	1	0	-1	0
	y_2	30	0	4	0	1	-1	0	1	0
	x_1	10	1	0	0	0	-1	0	1	0
	z_4	5	0	1	0	0	0	-1	0	1
	$-\varphi$	5	0	1	0	0	0	-1	-1	0
2	y_1	5	0	0	1	0	1	1	-1	-1
	y_2	10	0	0	0	1	-1	4	1	-4
	x_1	10	1	0	0	0	-1	0	1	0
	x_2	5	0	1	0	0	0	-1	0	1
	$-\varphi$	0	0	0	0	0	0	0	-1	-1

Таблица 6.7

Итерация	Базисные переменные	b	x_1	x_2	y_1	y_2	y_3	y_4
0	y_1	5	0	0	1	0	1	1
	y_2	10	0	0	0	1	-1	4
	x_1	10	1	0	0	0	-1	0
	x_2	5	0	1	0	0	0	-1
	$-f$	-50	0	0	0	0	3	4
1	y_1	5/2	0	0	1	-1/4	5/4	0
	y_4	5/2	0	0	0	1/4	-1/4	1
	x_1	10	1	0	0	0	-1	0
	x_2	15/2	0	1	0	1/4	-1/4	0
	$-f$	-60	0	0	0	-1	4	0
2	y_3	2	0	0	4/5	-1/5	1	0
	y_4	3	0	0	1/5	1/5	0	1
	x_1	12	1	0	4/5	-1/5	0	0
	x_2	8	0	1	1/5	1/5	0	0
	$-f$	-68	0	0	-16/5	-1/5	0	0

6.6. Двойственная задача линейного программирования

Рассмотрим задачу линейного программирования в стандартной форме

$$\begin{cases} (c, x) \rightarrow \max; \\ Ax \leq b; \\ x \geq 0. \end{cases} \quad (6.15)$$

Задача линейного программирования

$$\begin{cases} (b, u) \rightarrow \min; \\ A^T u \geq c; \\ u \geq 0 \end{cases} \quad (6.16)$$

называется *двойственной задачей* по отношению к задаче (6.15). В стандартной форме двойственную задачу можно записать следующим образом:

$$\begin{cases} (-b, u) \rightarrow \max; \\ -A^T u \leq -c; \\ u \geq 0. \end{cases} \quad (6.17)$$

Отметим, что, записав задачу линейного программирования, двойственную к задаче (6.17), приходим к исходной задаче (6.15). Таким образом, отношение двойственности является симметричным, т.е. каждая из двух задач является двойственной по отношению к другой.

Теорема 6.2. Если прямая (6.15) и двойственная (6.16) задачи линейного программирования имеют непустые допустимые множества, то для любых их допустимых решений x и u выполняется неравенство

$$(c, x) \leq (b, u), \quad (6.18)$$

т.е. значение целевой функции прямой задачи не превышает значения целевой функции двойственной задачи.

◄ Отметим очевидное утверждение: если $x \geq 0$, $y \geq 0$, то $(x, y) \geq 0$ (имеется в виду стандартное скалярное произведение в \mathbb{R}^n). Отсюда, в частности, вытекает, что если $x_1 \leq x_2$, $y \geq 0$, то $(x_1, y) \leq (x_2, y)$, т.е. векторные неравенства можно скалярно умножать на неотрицательный вектор.

Пусть x и u — допустимые решения прямой и двойственной задач. Тогда в соответствии с формулировками этих задач имеем $Ax \leq b$ и $A^T u \geq c$. Умножим первое из этих равенств скалярно на вектор u :

$$u^T Ax = (u, Ax) \leq (u, b) = (b, u).$$

Умножая второе равенство скалярно на $x \geq 0$, получаем аналогичные соотношения:

$$u^T Ax = (A^T u)^T x = (A^T u, x) \geq (c, x).$$

Сопоставляя две цепочки неравенств, получаем неравенство (6.18). ►

Следствие 6.1. Если целевые функции прямой (6.15) и двойственной (6.16) задач линейного программирования имеют на допустимых решениях x^* и u^* одинаковые значения, то эти решения являются оптимальными.

◄ Пусть $(c, x^*) = (b, u^*)$. По теореме 6.2 для любого допустимого решения x прямой задачи имеем

$$(c, x) \leq (b, u^*) = (c, x^*).$$

Следовательно, на векторе x^* целевая функция прямой задачи достигает максимума, и этот вектор является оптимальным решением. Аналогично можно показать, что u^* — оптимальное решение двойственной задачи. ►

В силу доказанной теоремы любое значение целевой функции двойственной задачи дает оценку сверху оптимальному значению целевой функции прямой задачи, и наоборот, любое значение целевой функции прямой задачи есть оценка снизу оптимального значения целевой функции двойственной задачи. Следствие указывает на то, что оптимумы в двух задачах могут совпадать. На самом деле такое

совпадение происходит всегда. Более того, оптимальное решение прямой (двойственной) задачи можно легко получить, зная оптимальное решение двойственной (прямой). Отметим, что если допустимые множества прямой и двойственной задач непусты, то возможные значения целевой функции прямой задачи ограничены сверху, а двойственной — снизу. Как отмечено выше, в этом случае и прямая, и двойственная задачи имеют оптимальные решения.

Теорема 6.3. Если допустимые множества прямой (6.15) и двойственной (6.16) задач непусты, то они имеют оптимальные решения x^* и u^* , причем оптимальные значения целевых функций двух задач совпадают:

$$(c, x^*) = (b, u^*).$$

◀ Пусть x^* — оптимальное решение прямой задачи (6.15). Тогда $\tilde{x}^* = (x^* \ y^*)^T$ — оптимальное решение задачи

$$\begin{cases} c^T x \rightarrow \max; \\ Ax + y = b; \\ x \geq 0, \quad y \geq 0. \end{cases}$$

Вводя обозначения $\tilde{c} = (c \ 0)^T$, $\tilde{A} = (A \ E)$, $\tilde{x} = (x \ y)^T$, приходим к задаче

$$\begin{cases} \tilde{c}^T \tilde{x} \rightarrow \max; \\ \tilde{A} \tilde{x} = b; \\ \tilde{x} \geq 0. \end{cases}$$

Введем для этой задачи вектор оценок $\gamma = (\tilde{A}_b^{-1})^T \tilde{c}_b$ (индексы b и c обозначают совокупности базисных и свободных переменных в векторе \tilde{x} , а также наборы соответствующих столбцов матрицы \tilde{A}) и запишем условие (6.10) оптимальности $\tilde{A}^T \gamma \geq \tilde{c}$ решения \tilde{x}^* , равносильное неравенствам $A^T \gamma \geq c$ и $\gamma \geq 0$. Эти неравенства означают, что вектор оценок γ является допустимым решением двойственной задачи. При этом

$$b^T \gamma = b^T (\tilde{A}_b^{-1})^T \tilde{c}_b = (\tilde{A}_b^{-1} b)^T \tilde{c}_b = \tilde{x}_b^T \tilde{c}_b = \tilde{c}^T \tilde{x} = c^T x.$$

Следовательно, допустимые решения x^* и γ прямой и двойственной задач в соответствии со следствием 6.1 являются оптимальными, и для этих решений значения целевых функций двух задач равны. Очевидно, что равенство сохраняется для любых двух оптимальных решений. ►

Замечание 6.2. Из доказательства теоремы 6.3 вытекает, что если известно решение x^* исходной задачи, то решение двойственной задачи можно получить, вычислив вектор оценок γ . Это правило можно конкретизировать следующим образом.

Пусть x^* и u^* — оптимальные решения прямой (6.15) и двойственной (6.16) задач. Тогда выполняется двойное равенство

$$c^T x^* = (u^*)^T A x^* = (u^*)^T b.$$

Следовательно,

$$(A^T u^* - c)^T x^* = 0 \quad \text{и} \quad (b - A x^*)^T u^* = 0.$$

Поскольку $A^T u^* - c \geq 0$, $x^* \geq 0$, $b - A x^* \geq 0$, $u^* \geq 0$, эти равенства выполняются, если имеют место следующие условия:

1) каждой положительной компоненте x_i^* вектора x^* соответствует нулевая компонента $(A^T u^* - c)_i$ вектора $A^T u^* - c$, т.е. положительной переменной прямой задачи соответствует активное ограничение двойственной задачи;

2) каждой положительной компоненте $(b - A x^*)_j$ вектора $b - A x^*$ соответствует нулевая компонента u_j^* вектора u^* , т.е. активному ограничению прямой задачи соответствует нулевое значение переменной в оптимальном решении двойственной задачи.

Если прямая задача невырождена, то после перехода к стандартной форме все базисные переменные (их количество m равно количеству ограничений, т.е. строк матрицы A) будут положительными, а все свободные — нулевыми. Каждой основной базисной переменной соответствует активное ограничение двойственной задачи, а каждой дополнительной базисной переменной — нулевое значение переменной двойственной задачи. В результате имеем m линейных уравнений относительно m переменных двойственной задачи. Систе-

ма уравнений позволяет однозначно определить оптимальное решение двойственной задачи.

Если прямая задача вырождена, какое-то количество базисных переменных будет иметь нулевое значение. Это приводит к сокращению количества уравнений, связывающих переменные двойственной задачи, и к неединственности оптимального решения двойственной задачи.

Пример 6.5. Рассмотрим задачу линейного программирования

$$\begin{cases} x_1 - 4x_2 - 16x_3 \rightarrow \max; \\ x_1 - 4x_3 \leq 2, & x_2 + 2x_3 \geq 1; \\ x_1 \geq 0, & x_2 \geq 0, & x_3 \geq 0. \end{cases}$$

Преобразуем эту задачу в стандартную форму:

$$\begin{cases} x_1 - 4x_2 - 16x_3 \rightarrow \max; \\ x_1 - 4x_3 \leq 2, & -x_2 - 2x_3 \leq -1; \\ x_1 \geq 0, & x_2 \geq 0, & x_3 \geq 0. \end{cases}$$

В матричной форме рассматриваемая задача имеет вид

$$\begin{cases} (1 \quad -4 \quad -6)\mathbf{x} \rightarrow \max; \\ \begin{pmatrix} 1 & 0 & -4 \\ 0 & -1 & -2 \end{pmatrix} \mathbf{x} \leq \begin{pmatrix} 2 \\ -1 \end{pmatrix}; \\ \mathbf{x} \geq 0. \end{cases}$$

где $\mathbf{x} = (x_1 \ x_2 \ x_3)^T$. Используя (6.17), запишем двойственную задачу линейного программирования:

$$\begin{cases} (-2 \ 1)\mathbf{u} \rightarrow \max; \\ \begin{pmatrix} -1 & 0 \\ 0 & 1 \\ 4 & 2 \end{pmatrix} \mathbf{u} \leq \begin{pmatrix} -1 \\ 4 \\ 16 \end{pmatrix}; \\ \mathbf{u} \geq 0, \end{cases}$$

или

$$\begin{cases} -2u_1 + u_2 \rightarrow \max; \\ -u_1 \leq -1, \quad u_2 \leq 4, \quad 4u_1 + 2u_2 \leq 16; \\ u_1 \geq 0, \quad u_2 \geq 0. \end{cases}$$

Полученная задача имеет два параметра оптимизации и может быть решена графическим методом (рис. 6.5). Оптимальное решение представлено точкой M , имеющей координаты $u_1^* = 1$, $u_2^* = 4$. Оптимальное значение целевой функции равно $-2u_1^* + u_2^* = 2$.

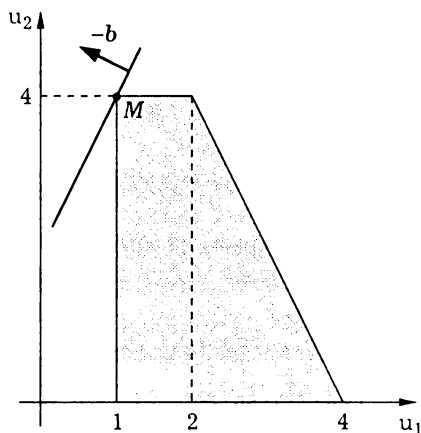


Рис. 6.5

Записывая двойственную задачу в форме (6.16), согласно теореме 6.3 находим оптимальное значение исходной задачи, равное 2. Чтобы получить оптимальное решение исходной задачи, отметим, что обе переменные в оптимальном решении двойственной задачи положительны, а из трех ограничений неактивным является только третье. В соответствии с замечанием 6.2 получаем систему трех уравнений, связывающих оптимальные значения x_1^* , x_2^* , x_3^* переменных прямой задачи:

$$\begin{cases} x_1^* - 4x_3^* = 2, \\ x_2^* + 2x_3^* = 1, \\ x_3^* = 0. \end{cases}$$

Из этой системы находим оптимальное решение исходной задачи: $x_1^* = 2$, $x_2^* = 1$, $x_3^* = 0$. #

Понятие двойственности, введенное для задач линейного программирования в стандартной форме, распространяется на любые задачи линейного программирования. Рассмотрим, например, задачу линейного программирования в канонической форме

$$\begin{cases} \mathbf{c}^T \mathbf{x} \rightarrow \max; \\ A\mathbf{x} = \mathbf{b}; \\ \mathbf{x} \geq \mathbf{0}. \end{cases} \quad (6.19)$$

Используя предположения и обозначения, введенные выше (см. разд. 6.3), запишем нашу задачу в виде

$$\begin{cases} \mathbf{c}_b^T + \mathbf{c}_s^T \mathbf{x}_s \rightarrow \max; \\ A_b \mathbf{x}_b + A_s \mathbf{x}_s = \mathbf{b}; \\ \mathbf{x}_b \geq \mathbf{0}, \quad \mathbf{x}_s \geq \mathbf{0}. \end{cases}$$

Исключим базисные переменные \mathbf{x}_b из ограничений и целевой функции (постоянное слагаемое в целевой функции опускаем):

$$\begin{cases} \mathbf{c}_b^T A_b^{-1} \mathbf{b} + (\mathbf{c}_s - A_s^T (A_b^{-1})^T \mathbf{c}_b)^T \mathbf{x}_s \rightarrow \max; \\ A_b^{-1} A_s \mathbf{x}_s \leq A_b^{-1} \mathbf{b}; \\ \mathbf{x}_s \geq \mathbf{0}. \end{cases}$$

Полученная задача в стандартной форме позволяет сформулировать двойственную задачу:

$$\begin{cases} \mathbf{b}^T (A_b^{-1})^T \mathbf{u} \rightarrow \min; \\ A_s^T (A_b^{-1})^T \mathbf{u} + A_b^{-1} \mathbf{c}_b \geq \mathbf{c}_s; \\ \mathbf{u} \geq \mathbf{0}. \end{cases}$$

Выполним линейную замену переменных $\mathbf{z} = A_b^{-1} \mathbf{u} + A_b^{-1} \mathbf{c}_b$. В результате придем к следующей задаче линейного программирования (постоянное слагаемое в целевой функции опускаем):

$$\begin{cases} \mathbf{b}^T \mathbf{z} \rightarrow \min; \\ A_s^T \mathbf{z} \geq \mathbf{c}_s; \\ A_b \mathbf{z} = \mathbf{c}_b \geq \mathbf{0}. \end{cases}$$

Наконец, объединяя ограничения, получаем задачу

$$\begin{cases} b^T z \rightarrow \min; \\ A^T z \geq c. \end{cases} \quad (6.20)$$

Обратим внимание на то, что в этой задаче отсутствуют условия неотрицательности переменных. Задачу (6.20) назовем двойственной к задаче (6.19).

Рассмотренный пример показывает, как различные ограничения прямой задачи реализуются в двойственной. Каждому ограничению прямой задачи соответствует переменное двойственной задачи, а каждому переменному прямой задачи — ограничение двойственной задачи. При этом:

1) ограничению типа неравенства прямой задачи соответствует неотрицательное переменное двойственной;

2) ограничению типа равенства прямой задачи соответствует неограниченное в знаке переменное двойственной задачи;

3) неотрицательному переменному прямой задачи соответствует ограничение типа неравенства двойственной;

4) неограниченному в знаке переменному прямой задачи соответствует ограничение типа равенства двойственной задачи.

Пользуясь сформулированными правилами, запишем задачу линейного программирования, двойственную к задаче (6.2), полагая, что $i_s = s$, $s = \overline{1, r}$, т.е. переменные прямой задачи перенумерованы так, что вначале идут неотрицательные переменные. Получим следующее:

$$\begin{cases} \sum_{i=1}^m b_i u_i \rightarrow \min; \\ \sum_{i=1}^m a_{ij} u_i \geq c_j, \quad j = \overline{1, r}; \\ \sum_{i=1}^m a_{ij} u_i = c_j, \quad j = \overline{r+1, n}; \\ u_1 \geq 0, \quad u_2 \geq 0, \quad \dots, \quad u_{m_1} \geq 0. \end{cases}$$

Расширение понятия двойственности сохраняет основные свойства:

- любое значение целевой функции прямой задачи не превышает любого значения целевой функции двойственной задачи;
- если допустимые множества прямой и двойственной задач непусты, то каждая из них имеет оптимальное решение, причем оптимальные значения двух задач совпадают.

Вопросы для самопроверки

1. Дайте определения оптимального и допустимого решений задачи линейного программирования.
2. Изменится ли оптимальное решение задачи линейного программирования (6.1), если во всех ограничениях (кроме требования неотрицательности переменных) знаки неравенства заменить знаками равенства?
3. Объясните различие между активными и пассивными ограничениями в задаче линейного программирования.
4. Как связаны значения целевой функции, соответствующие различным оптимальным решениям?
5. Обязательно ли при использовании геометрического метода представлять задачу линейного программирования в стандартной форме?
6. Может ли ведущий элемент симплекс-таблицы быть: а) нулевым; б) отрицательным?
7. Можно ли утверждать, что если на текущей итерации симплекс-метода получено вырожденное базисное решение, то и на следующей итерации будет получено вырожденное базисное решение?
8. Опишите построение двойственной задачи линейного программирования для задачи: а) в стандартной форме; б) общего вида.
9. Покажите, что ограничениям типа равенства исходной задачи линейного программирования соответствуют неограниченные в знаке переменные двойственной задачи.
10. Покажите, что неограниченным в знаке переменным исходной задачи линейного программирования соответствуют ограничения типа равенства двойственной задачи.
11. Может ли двойственная задача линейного программирования иметь допустимое решение, если исходная задача не имеет оптимального решения?

12. Задачу линейного программирования

$$\begin{cases} x_1 + 2x_2 + x_3 + 8x_4 \rightarrow \max; \\ x_1 + 4x_2 - 3x_3 - 4x_4 \leq -1, \\ 2x_1 + 3x_2 + x_3 - 2x_4 \geq 3; \\ x_k \geq 0, \quad k = \overline{1, 4}. \end{cases}$$

решите симплекс-методом, а двойственную ей задачу — геометрическим методом. Сопоставьте полученные результаты.

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

Учебники и учебные пособия

Алексеев В.М. Оптимальное управление / В.М. Алексеев, В.М. Тихомиров, С.В. Фомин. – М.: Наука, 1979. – 429 с.

Аоки М. Введение в методы оптимизации. Основы и приложения нелинейного программирования: пер. с англ. / М. Аоки. – М.: Наука, 1977. – 344 с.

Аттетков А.В. Методы оптимизации: учебник для вузов / А.В. Аттетков; С.В. Галкин, В.С. Зарубин; под ред. В.С. Зарубина, А.П. Крищенко. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2001. – 440 с.

Ашманов С.Л. Линейное программирование / С.Л. Ашманов. – М.: Наука, 1981. – 304 с.

Базара М. Нелинейное программирование. Теория и алгоритмы: пер. с англ. / М. Базара, К. Шетти. – М.: Мир, 1982. – 584 с.

Банди Б. Методы оптимизации. Вводный курс: пер. с англ. / Б. Банди. – М.: Радио и связь, 1988. – 128 с.

Банди Б. Основы линейного программирования: пер. с англ. / Б. Банди. – М.: Радио и связь, 1989. – 176 с.

Васильев Ф.П. Численные методы решения экстремальных задач / Ф.П. Васильев. – М.: Наука, 1988. – 552 с.

Волков И.К. Исследование операций: учебник для вузов / И.К. Волков, Е.А. Загоруйко; под ред. В.С. Зарубина, А.П. Крищенко. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2000. – 436 с.

Волков И.К. Случайные процессы: учебник для вузов / И.К. Волков, С.М. Зуев, Г.М. Цветкова; под ред. В.С. Зарубина, А.П. Крищенко. – М.: Изд-во МГТУ им. Н.Э. Баумана, 1999. – 448 с.

Габасов Р.Ф. Методы оптимизации / Р.Ф. Габасов, Ф.М. Кириллова. – Минск: Изд-во БГУ, 1981. – 352 с.

Гилл Ф. Практическая оптимизация: пер. с англ. / Ф. Гилл, У. Мюррей, М. Райт. – М.: Мир, 1985. – 510 с.

Гроссман К. Нелинейное программирование на основе безусловной минимизации / К. Гроссман, А.А. Каплан. – Новосибирск: Наука. Сиб. отделение, 1981. – 183 с.

Дамбраускас А.П. Симплексный поиск / А.П. Дамбраускас. – М.: Энергия, 1979. – 176 с.

Дэннис Дж. Численные методы безусловной оптимизации и решения нелинейных уравнений: пер. с англ. / Дж. Дэннис, Р. Шнабель. – М.: Мир, 1988. – 440 с.

Евтушенко Ю.Г. Методы решения экстремальных задач и их применение в системах оптимизации / Ю.Г. Евтушенко. – М.: Наука, 1982. – 432 с.

Зуховицкий И.И. Линейное и выпуклое программирование / И.И. Зуховицкий, Л.И. Авдеева. – М.: Наука, 1970. – 264 с.

Измаилов А.Ф. Численные методы оптимизации: учеб. пособие / А.Ф. Измаилов, М.В. Солодов. – М.: Физматлит, 2003. – 304 с.

Канатников А.Н. Линейная алгебра: учебник для вузов / А.Н. Канатников, А.П. Крищенко; под ред. В.С. Зарубина, А.П. Крищенко. – М.: Изд-во МГТУ им. Н.Э. Баумана, 1998. – 336 с.

Канатников А.Н. Дифференциальное исчисление функций многих переменных: учебник для вузов / А.Н. Канатников, А.П. Крищенко, В.Н. Чествериков; под ред. В.С. Зарубина, А.П. Крищенко. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2000. – 456 с.

Карманов В.Г. Математическое программирование: учеб. пособие / В.Г. Карманов. – М.: Физматлит, 2001. – 264 с.

Кузнецов Ю.Н. Математическое программирование / Ю.Н. Кузнецов, В.И. Кузубов, А.Б. Волощенко. – М.: Высш. шк., 1976. – 352 с.

Лесин В.В. Основы методов оптимизации / В.В. Лесин, Ю.П. Лисовец. – М.: Изд-во МАИ, 1995. – 341 с.

Линейное и нелинейное программирование / Под ред. И.Н. Ляшенко. – Киев: Выща шк., 1975. – 372 с.

Моисеев Н.Н. Методы оптимизации / Н.Н. Моисеев, Ю.П. Иванилов, Е.М. Столярова. – М.: Наука, 1978. – 351 с.

Ногин В.Д. Основы теории оптимизации / В.Д. Ногин, И.О. Протодяконов, И.И. Евлампиев. – М.: Высш. шк., 1986. – 384 с.

Пантелеев А.В. Методы оптимизации в примерах и задачах: учеб. пособие / А.В. Пантелеев, Т.А. Легова. – М.: Высш. шк., 2002. – 544 с.

Поляк Б.Т. Введение в оптимизацию / Б.Т. Поляк. – М.: Наука, 1983. – 384 с.

Пшеничный Б.Н. Численные методы в экстремальных задачах / Б.Н. Пшеничный, Ю.М. Данилин. – М.: Наука, 1975. – 320 с.

Растргин Л.А. Статистические методы поиска / Л.А. Растргин. – М.: Наука, 1968. – 376 с.

Реклейтис Г. Оптимизация в технике: в 2 кн.: пер. с англ. / Г. Реклейтис, А. Рейвиндран, К. Рэгсдел. – М.: Мир, 1986.

Рыков А.С. Поисковая оптимизация. Методы деформируемых конфигураций / А.С. Рыков. – М.: Физматлит, 1993. – 216 с.

Сухарев А.Г. Курс методов оптимизации / А.Г. Сухарев, А.В. Тимохов, В.В. Федоров. – М.: Наука, 1986. – 328 с.

Теория вероятностей: учебник для вузов / В.А. Печинкин, О.И. Тескин, Г.М. Цветкова и др.; под ред. В.С. Зарубина, А.П. Крищенко. – М.: Изд-во МГТУ им. Н.Э. Баумана, 1998. – 456 с.

Фиакко А. Нелинейное программирование. Методы последовательной безусловной минимизации: пер. с англ. / А. Фиакко, Г. Мак-Кормик. – М.: Мир, 1972. – 240 с.

Химмельблау Д. Прикладное нелинейное программирование: пер. с англ. / Д. Химмельблау. – М.: Мир, 1975. – 534 с.

Юдин Д.Б. Линейное программирование. Теория, методы, приложения / Д.Б. Юдин, Е.Г. Гольштейн. – М.: Наука, 1969. – 424 с.

Дополнительная литература

Абрамов Л.М. Математическое программирование / Л.М. Абрамов, В.Ф. Капустин. – Л.: Изд-во ЛГУ, 1981. – 328 с.

Амосов А.А. Вычислительные методы для инженеров: учеб. пособие / А.А. Амосов, Ю.А. Дубинский, Н.В. Копченова. – М.: Высш. шк., 1994. – 554 с.

Ашманов С.А. Линейное программирование / С.А. Ашманов. – М.: Наука, 1981. – 304 с.

Белоусов А.И. Элементы линейного программирования: учеб. пособие / А.И. Белоусов. – М.: Изд-во МГТУ им. Н.Э. Баумана, 1993. – 76 с.

Бертсекас Л. Условная оптимизация и методы множителей Лагранжа: пер. с англ. / Д. Бертсекас. – М.: Радио и связь, 1987. – 400 с.

Васютинский Н. Золотая пропорция / Н. Васютинский. – М.: Молодая гвардия, 1990. – 235 с.

Введение в нелинейное программирование / Под ред. К.Х. Эльстера: пер. с англ. – М.: Наука, 1985. – 264 с.

Галеев Э.М. Курс лекций по вариационному исчислению и оптимальному управлению / Э.М. Галеев. – М.: Изд-во механико-математического факультета МГУ, 1996. – 160 с.

Ганшин Г.С. Методы оптимизации и решение уравнений / Г.С. Ганшин. – М.: Наука, 1987. – 128 с.

Гасс С. Линейное программирование / С. Гасс. – М.: Физматгиз, 1961. – 303 с.

Гейл Д. Теория линейных экономических моделей / Д. Гейл. – М.: Изд-во иностр. литер., 1963.

Голуб Дж. Матричные вычисления: пер. с англ. / Дж. Голуб, Ч. Ван Лоун. – М.: Мир, 1999. – 548 с.

Гольштейн Е.Г. Выпуклое программирование (элементы теории) / Е.Г. Гольштейн. – М.: Наука, 1970. – 268 с.

Гольштейн Е.Г. Теория двойственности в математическом программировании и ее приложения / Е.Г. Гольштейн. – М.: Наука, 1971. – 351 с.

Дамбраускас А.П. Алгоритмы и программы симплексного поиска / А.П. Дамбраускас, В.А. Кашеев, Д.В. Кошаев. – Красноярск: Изд-во Краснояр. ун-та, 1993. – 160 с.

Данциг Дж. Линейное программирование: пер. с англ. / Дж. Данциг. – М.: Прогресс, 1966. – 600 с.

Демьянов В.Ф. Введение в минимакс / В.Ф. Демьянов, В.Н. Малоземов. – М.: Наука, 1972. – 368 с.

Демьянов В.Ф. Приближенные методы решения экстремальных задач / В.Ф. Демьянов, А.М. Рубинов. – Л.: Изд-во ЛГУ, 1965. – 180 с.

Еремин И.И. Введение в теорию линейного и выпуклого программирования / И.И. Еремин, Н.Н. Астафьев. – М.: Наука, 1976. – 192 с.

Ермольев Ю.И. Методы стохастического программирования / Ю.И. Ермольев. – М.: Наука, 1976. – 240 с.

Зангвилл У.И. Нелинейное программирование. Единый подход: пер. с англ. / У.И. Зангвилл. – М.: Сов. радио, 1973. – 312 с.

Зойтендейк Г. Методы возможных направлений: пер. с англ. / Г. Зойтендейк. – М.: Изд-во иностр. лит., 1963. – 176 с.

Иоффе А.Д. Теория экстремальных задач / А.Д. Иоффе, В.М. Тихомиров. – М.: Наука, 1974. – 480 с.

Карлин С. Математические методы в теории игр, программировании и экономике / С. Карлин. – М.: Мир, 1964. – 836 с.

Кузнецов Ю.Н. Математическое программирование / Ю.Н. Кузнецов, В.И. Кузубов, А.Б. Волощенко. – М.: Высш. шк., 1980. – 320 с.

Кюнц Г. Нелинейное программирование: пер. с англ. / Г. Кюнц, В. Крелле. – М.: Советское радио, 1965. – 304 с.

Лейхтвейс К. Выпуклые множества / К. Лейхтвейс. – М.: Наука, 1985. – 336 с.

Мину М. Математическое программирование. Теория и алгоритмы: пер. с франц. / М. Мину. – М.: Наука, 1990. – 487 с.

Михалевич В.С. Оптимизационные задачи производственно-транспортного планирования: модели, методы, алгоритмы / В.С. Михалевич, В.А. Трубин, Н.З. Шор. – М.: Наука, 1986. – 259 с.

Моисеев Н.Н. Численные методы в теории оптимальных систем / Н.Н. Моисеев. – М.: Наука, 1971. – 424 с.

Муртаф Б. Современное линейное программирование: пер. с англ. / Б. Муртаф. – М.: Мир, 1984. – 224 с.

Мухачева Э.А. Математическое программирование / Э.А. Мухачева, Г.Ш. Рубинштейн. – Новосибирск: Наука, 1983. – 224 с.

Немировский А.С. Сложность задач и эффективность методов оптимизации / А.С. Немировский, Д.Б. Юдин. – М.: Наука, 1979. – 384 с.

Полак Э. Численные методы оптимизации. Единый подход: пер. с англ. / Э. Полак. – М.: Мир, 1974. – 376 с.

Пшеничный Б.Н. Выпуклый анализ и экстремальные задачи / Б.Н. Пшеничный. – М.: Наука, 1980. – 320 с.

Пшеничный Б.Н. Метод линеаризации / Б.Н. Пшеничный. – М.: Наука, 1983. – 136 с.

Растргин Л.А. Случайный поиск в процессах адаптации / Л.А. Растргин. – Рига: Зинатне, 1973. – 131 с.

Рокафеллар Р.Т. Выпуклый анализ: пер. с англ. / Р.Т. Рокафеллар. – М.: Мир, 1973. – 468 с.

Романовский И.В. Алгоритмы решения экстремальных задач / И.В. Романовский. – М.: Наука, 1977. – 352 с.

Рубинштейн Г.Ш. Конечномерные модели оптимизации / Г.Ш. Рубинштейн. – Новосибирск: Изд-во Новосибирск. ун-та, 1970. – 228 с.

Сea Ж. Оптимизация. Теория и алгоритмы: пер. с англ. / Ж. Сea. – М.: Мир, 1973. – 244 с.

Стронгин Р.Г. Численные методы в многоэкстремальных задачах (информационно-статистические алгоритмы) / Р.Г. Стронгин. – М.: Наука, 1978. – 240 с.

Сухарев А.Г. Курс методов оптимизации / А.Г. Сухарев, А.В. Тимохов, В.В. Федоров. – М.: Наука, 1986. – 328 с.

Сухарев А.Г. Минимаксные алгоритмы в задачах численного анализа / А.Г. Сухарев. – М.: Наука, 1989. – 304 с.

Сухарев А.Г. Оптимальный поиск экстремума / А.Г. Сухарев. – М.: Изд-во МГУ, 1975. – 100 с.

Фаддеев Д.К. Вычислительные методы линейной алгебры / Д.К. Фаддеев, В.Н. Фаддеева. – СПб.: Лань, 2002. – 733 с.

Федоров В.В. Численные методы максимина / В.В. Федоров. – М.: Наука, 1979. – 280 с.

Формалев В.Ф. Численные методы / В.Ф. Формалев, Д.Л. Ревизников. – М.: Физматлит, 2004. – 400 с.

Хедли Дж. Нелинейное и динамическое программирование: пер. с англ. / Дж. Хедли. – М.: Мир, 1967. – 534 с.

Численные методы условной оптимизации / Под ред. Ф. Гилла, У. Мюррея. – М.: Мир, 1977. – 256 с.

Чичинадзе В.К. Решение невыпуклых нелинейных задач оптимизации / В.К. Чичинадзе. – М.: Наука, 1983. – 256 с.

Шор Н.З. Квадратичные экстремальные задачи и недифференцируемая оптимизация / Н.З. Шор, С.И. Стеценко. – Киев: Наукова думка, 1989. – 203 с.

Щедрин Н.И. Математические методы программирования в экономике / Н.И. Щедрин, А.Н. Кархов. – М.: Статистика, 1974. – 144 с.

Задачники

Акулич И.Л. Математическое программирование в примерах и задачах / И.Л. Акулич. – М.: Высш. шк., 1986. – 320 с.

Алексеев В.М. Сборник задач по оптимизации / В.М. Алексеев, Э.М. Галеев, В.М. Тихомиров. – М.: Наука, 1984. – 287 с.

Ашманов С.А. Теория оптимизации в задачах и упражнениях / С.А. Ашманов, А.В. Тимохов. – М.: Наука, 1991. – 448 с.

Калихман И.Л. Сборник задач по математическому программированию / И.Л. Калихман. – М.: Высш. шк., 1975. – 272 с.

Капустин В.Ф. Практические занятия по курсу математического программирования / В.Ф. Капустин. – Л.: Изд-во ЛГУ, 1976. – 192 с.

Летова Т.А. Экстремум функций в примерах и задачах: учеб. пособие / Т.А. Летова, А.В. Пантелеев. – М.: Изд-во МАИ, 1998. – 376 с.

Морозов В.В. Исследование операций в примерах и задачах / В.В. Морозов, А.Г. Сухарев, В.В. Федоров. – М.: Высш. шк., 1986. – 287 с.

Справочные издания

Александрова Н.В. Математические термины: справочник / Н.В. Александрова. – М.: Высш. шк., 1978. – 190 с.

Бронштейн И.Н. Справочник по математике для инженеров и учащихся втузов. 13-е изд., испр. / И.Н. Бронштейн, К.А. Семендяев. – М.: Наука, 1986. – 544 с.

Воднев В.Т. Математический словарь высшей школы / В.Т. Воднев, А.Ф. Наумович, Н.Ф. Наумович; под ред. Ю.С. Богдановича. – Минск: Высшейш. шк., 1984. – 528 с.

Воеводин В.В. Матрицы и вычисления / В.В. Воеводин, Ю.А. Кузнецов. – М.: Наука, 1984. – 320 с.

Выгодский М.Я. Справочник по высшей математике. 13-е изд., стереотип / М.Я. Выгодский. – М.: Физматгиз, 1995. – 872 с.

Иванов В.В. Методы вычислений на ЭВМ: справочное пособие / В.В. Иванов. – Киев: Наукова думка, 1986. – 584 с.

Корн Г. Справочник по математике. Для научных работников и инженеров: пер. с англ. / Г. Корн, Т. Корн. – М.: Наука, 1973. – 832 с.

Математический энциклопедический словарь / гл. ред. Ю.В. Прохоров. – М.: Сов. энцикл., 1988. – 848 с.

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ

Альтернатива 9

Антиградиент 49

Базис задачи линейного программирования 235

Базисное решение 230

— допустимое 230

— вырожденное 236

— начальное 240

БФШ-метод 94

Вектор оценок 239

— перемещений 112

Векторы сопряженные 69

Вершина симплекса 99

Градиент приведенный 160

ДФП-метод 92

Задача безусловной минимизации 20

— выпуклого программирования 22

— геометрического программирования 22

— двойственная 147

— дискретного программирования 23

— Евклида 13

— квадратичного программирования 21

— линейного программирования 20

— в канонической форме 224

— стандартной форме 224

Задача линейного программирования вырожденная 236

— двойственная 250

— общего вида 223

— математического программирования 11

— нелинейного программирования 20

— о диете 19

— оптимального проектирования 15

— оптимизации 9

— сепарабельного программирования 21

— транспортная 18

— условной минимизации 20

— целочисленного программирования 23

Интервал неопределенности 28

Итерация 45

Конус допустимых направлений 129

Коэффициент золотого сечения 33

— отражения 105

— растяжения 106

— редукции 103

— сжатия 107

Критерий оптимальности 9

Масштабирование 63

Матрица идемпотентная 179

— поправочная 91

— проекционная 179

Метод барьерных функций 213

- Бройдена — Флетчера — Шенно 94
- внешней точки 214
- внутренней точки 213
- возможных направлений 201
- второго порядка 45
- градиентного спуска 49
- Давидона — Флетчера — Пауэлла 92
- деформируемых конфигураций 100
- дихотомии 29
- дробления шага 55
- золотого сечения 34
- квадратичной аппроксимации 37
- квазиньютоновский 89
- кубической аппроксимации 40
- Марквардта 84
- наилучшей пробы 125
- наискорейшего спуска 49
- Нелдера — Мида 105
- нулевого порядка 45
- Ньютона 80
- Пауэлла 95
- первого порядка 45
- Полака — Рибьера 77
- последовательной безусловной оптимизации 212
- приведенного градиента 160
- проекции антиградиента 186
- точки на множество 172
- прямого поиска 26, 45
- симметричный 29
- случайного поиска с возвратом при неудачном шаге 125
- сопряженных градиентов 76
- направлений 72
- спуска 45
- статистического градиента 125

Метод условного градиента 154

- Флетчера — Ривса 76
- циклического покоординатного спуска 109
- штрафных функций 213
- внешних 214
- внутренних 213

Методы полиномиальной аппроксимации 37

Минимизация функции 11

Минимум локальный на множестве 130

- строгий 130

Многочлен интерполяционный

кубический Эрмита 40

Множество выпуклое 22

- допустимое 11

Множитель ускоряющий 112

Момент обновления алгоритма 77

Направление допустимое 129

- спуска 48
- возможное 154
- ньютоновское 81

Направления сопряженные 69

Неизвестное базисное 159

- свободное 159

Норма спектральная 64

Нумерация вершин правильная 102

Ограничение активное 140, 234

- линейное 20
- неактивное 140, 234
- нелинейное 20
- пассивное 140, 234
- типа неравенства 20
- равенства 20

Операция восстановления симплекса 108

Определитель Вандермонда 37

Оптимизация бесконечномерная 10
– конечномерная 10
– многокритериальная 10
– одномерная 10

Ортант неотрицательный 158

– положительный 22

Отношение золотого сечения 33

– порядка в \mathbb{R}^n 224

Отображение сжимающее 64

Отражение вершины 102

Отрезок в \mathbb{R}^n 231

Параметр точности поиска 47

– штрафа 214

Параметры оптимизации 11

Переменное базисное 159

– искусственное 245

– свободное 159

Позином 22

Поиск исследующий 112

– пассивный 27

– по образцу 114

– последовательный 27

– прямой управляемый 100

– симплексный 99

– случайный 119

Последовательность итерационная 45

– релаксационная 45

Правило множителей Лагранжа
обобщенное 135

Проекция точки на множество 168

Процедура исключения отрезка 28

Процесс ортогонализации 69

Ранг максимальный строчный 230

Расстояние от точки до множества 168

Редукция симплекса 103

Рестарт 77

Решение допустимое 11

– оптимальное 11

Сечение отрезка золотое 32

Симплекс 99

– регулярный 100

Симплекс-метод 234

– выпуклый 160

Симплекс-разность 237

Симплекс-таблица 240

Скорость сходимости квадратичная 84

– линейная 88

– сверхлинейная 89

Спуск исчерпывающий 53

Строка ведущая 242

Структура овражная 63

Сходимость по вероятности 120

– почти наврное 120

– сильная 47

– слабая 46

Теорема двойственности 147

– Куна — Таккера 140

– о неподвижной точке 64

Точка базовая 98

– крайняя 231

– минимума 44

– седловая 143

– стационарная 23

Уравнение связи 133

Условие квазиньютоновское 91

– останова 47

Условия дополняющей нежесткости 140

Функция барьерная 213

– выпуклая 22

– двойственная 147

– дробно-линейная 21

Функция индикаторная 212

- квадратичная 21
- Лагранжа 140
- сепарабельная 21
- сильно выпуклая 23
- строго выпуклая 22
- унимодальная 26
- целевая 11
- штрафа 213

Функция штрафа внешнего 214

- внутреннего 213
- штрафная 213

Число обусловленности 65**Шаблон 99****Шаг спуска 48****Элемент ведущий 242**