Федеральное агентство по образованию

Московский инженерно-физический институт
(государственный университет)

А.М. Загребаев, Н.А. Крицына
Ю.П. Кулябичев, Ю.Ю. Шумилов

МЕТОДЫ МАТЕМАТИЧЕСКОГО ПРОГРАММИРОВАНИЯ В ЗАДАЧАХ ОПТИМИЗАЦИИ СЛОЖНЫХ ТЕХНИЧЕСКИХ СИСТЕМ

Рекомендовано УМО «Ядерные физика и технологии»
в качестве учебного пособия
для студентов высших учебных заведений

Москва 2007

УДК 519.85(075)
ББК 22.18я7
М 54

Методы математического программирования в задачах оптимизации сложных технических систем: *учебное пособие* / А.М. Загребаев, Н.А. Крицына, Ю.П. Кулябичев, Ю.Ю. Шумилов. – М.: МИФИ, 2007. – 332 с.

Приведены теоретические основы методов оптимизации. Рассмотрены методы линейного, целочисленного и нелинейного программирования. Представлено большое количество практических задач, решение которых основано на использовании методов оптимизации.

Предназначено для студентов, обучающихся по специальности «Прикладная математика и информатика» и практикантов, а также будет полезно инженерам и аспирантам, работающим в области оптимизации параметров технических систем различного назначения.

Пособие подготовлено в рамках Инновационной образовательной программы.

Рецензент д-р техн. наук, проф. А.Д. Модяев

ISBN 978-5-7262-0807-7

О Г Л А В Л Е Н И Е

ПРЕДИСЛОВИЕ

Первооснову математического программирования составляют методы нахождения экстремума (максимума или минимума) функций, заданных на некоторой области изменения переменных. Задачи такого рода встречаются в самых разных областях человеческой деятельности: в экономике, технике, военном деле и т.д. Математическое программирование – один из разделов науки, имеющей название «исследование операций».

Задачи, которые решаются с помощью методов математического программирования возникают там, где необходим выбор одного из возможных образов действий (или программ действий). Именно отсюда название «программирование» – в смысле выбор программы действий. Название не имеет ни какого отношения к программированию в смысле написания программ для ЭВМ и возникло раньше, чем ЭВМ. Хотя, надо отметить, что все методы, в конечном счете, рассчитаны на использование ЭВМ. Первые задачи линейного программирования, являющегося основой математического программирования, были рассмотрены русским математиком Канторовичем ещё в 1939 г. (задача выбора наилучшей программы загрузки группы лущильных станков). Однако их практическое решение требовало большого объема вычислений. Создание высоко производительных ЭВМ послужило стимулом развития теории и практики как математического программирования, так и других разделов исследования операций. Суть у всех этих разделов одна – выбор наилучшей программы действий в задачах оптимизации с различными математическими моделями.

Так, в математическом программировании дается функция, которую нужно оптимизировать (т.е. найти экстремум), и ограничения на область изменения переменных; а в оптимальном управления задаётся оптимизируемый функционал и процесс, описываемый с помощью дифференциальных уравнений и т.д.

Математическое программирование – один из важнейших разделов, представляющих интерес для специалистов, занимающихся прикладной математикой, и изложено в той или иной мере в различных изданиях как в России, так и за рубежом, ряд из которых приведен в списке литературы.

Данная книга представляет собой результат многолетнего чтения курса «Математическое программирование» на кафедре «Математическое обеспечение систем» в Московском инженерно-физическом институте и помимо теоретических соображений, охватывающих разделы линейного, целочисленного и нелинейного программирования, содержит многоплановые примеры решений задач, иллюстрирующих применение методов математического программирования на практике.

В книге рассматривается многообразие прикладных задач оптимизации, решение которых обеспечивается методами математического программирования, независимо от их физического содержания. Так, особенностью книги является единый подход к оптимизации игровых задач, задач распределения запасов реактивности в ядерных реакторах, а также задач, связанных с оптимальной кластеризацией и т.д.

Каждая глава книги содержит большое количество доведенных до конечного результата примеров.

Книга предназначена для специалистов по прикладной математике, а также может быть полезна студентам вузов соответствующих специальностей.

ОСНОВНЫЕ ОБОЗНАЧЕНИЯ И ПОНЯТИЯ

Приведем ряд общепринятых обозначений и понятий, используемых при рассмотрении материала последующих глав.

1. Пусть $\bar{x}$ и $\bar{y}$ – n-мерные векторы

$$\bar{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, \bar{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix},$$

$$\bar{x}^{\mathrm{T}} = (x_1, \ldots, x_n) \text{ – транспонированный вектор,}$$

тогда $<\bar{x}, \bar{y}> = \bar{x}^{\mathrm{T}}\bar{y} = \sum_{i=1}^{n} x_i y_i$ – скалярное произведение векторов $\bar{x}$ и $\bar{y}$.

2. $A = \begin{pmatrix} a_{11} & \ldots & a_{1n} \\ \ldots & \ldots & \ldots \\ a_{m1} & \ldots & a_{mn} \end{pmatrix}$ – матрица размерности $m \times n$.

3. $f(\bar{x}) = f(x_1, \ldots, x_{1n})$ – функция n переменных.

Функция $f(\bar{x})$ имеет 1-ю и 2-ю непрерывные производные на множестве D в n-мерном пространстве R^n.

Пусть некоторая точка $\bar{x}^0 \in D$. Тогда $f(\bar{x})$ может быть представлена в виде ряда Тейлора относительно этой точки:

$$f(x_1, \ldots, x_n) = f({x_1}^0, \ldots, {x_n}^0) + \sum_{i=1}^{n} \left.\frac{\partial f}{\partial x}\right|_{x_0} (x_i - x_i^0) +$$

$$+ \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \left.\frac{\partial^2 f}{\partial x_i \partial x_j}\right|_{x_0} (x_i - x_i^0)(x_j - x_j^0) + 0(\left\|\bar{x} - \bar{x}^0\right\|^2).$$

4. $\nabla f(\bar{x}^0) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n} \right)^{\mathrm{T}} \Bigg|_{x_0}$ – градиент функции в точке $\bar{x}^0 \in D$.

Градиент – вектор, который направлен в сторону наискорейшего роста функции и ортогонален в точке x^0 к поверхности (или линии) уровня $f(\vec{x}) = \text{const}$ (рис. В.1).

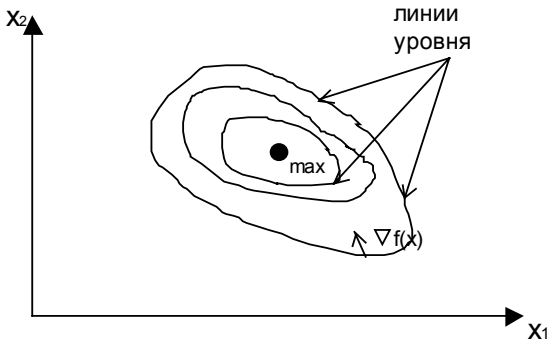


Рис. В.1. Линии уровня $f(\bar{x}) = \text{const}$ для функции двух переменных

5. $H(\bar{x}^0) = \begin{vmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \ldots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \ldots & \ldots & \ldots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \ldots & \frac{\partial^2 f}{\partial x_n \partial x_n} \end{vmatrix}$ – вещественная симметричная матрица (матрица Гессе).

Важность градиента и матрицы Гессе определяется их использованием во многих алгоритмах поиска экстремума функций.

Можно записать

$$f(\bar{x}) = f(\bar{x}^0) + < \nabla f(\bar{x}^0), \ (\bar{x} - \bar{x}^0) > + \frac{1}{2} < (\bar{x} - \bar{x}^0),$$

$$H(\bar{x}^0)(\bar{x} - \bar{x}^0) > +0(\left\| \bar{x} - \bar{x}^0 \right\|^2) .$$

6. Пусть

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n = b_1; \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2j}x_j + \ldots + a_{2n}x_n = b_2; \\ \ldots \\ .a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n = b_i; \\ \ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{nj}x_j + \ldots + a_{mn}x_n = b_m \end{cases} \quad -$$

система линейных уравнений в скалярном виде, тогда $A\overline{x} = \overline{b}$ – система уравнений в матричной форме, где A – матрица коэффициентов a_{ij} $(i = 1, \ldots, m;\ j = 1, \ldots, n)$, $\overline{b}$ – m -мерный вектор.

Система линейных уравнений называется совместной, если она имеет хотя бы одно решение.

Пусть r – ранг матрицы и представляет собой наибольший порядок отличного от нуля определителя матрицы A (число линейно независимых столбцов матрицы A).

$r(A) = r(Ab)$ – необходимое и достаточное условие совместности системы линейных уравнений, т.е. ранг матрицы A должен быть равен рангу распределенной матрицы системы.

Ранг не превосходит числа неизвестных n, $r \le n$.

Тогда:

если $r = n$, то решение системы единственно;

если $r < n$, то система имеет бесчисленное множество решений.

7. $a_0 + a_1x_1 + \ldots + a_nx_n \ge 0$ – линейное неравенство.

Совокупность точек пространства, координаты которых удовлетворяют этому неравенству, представляют собой полупространство; совокупность нескольких линейных неравенств определяет область решения, представляющую собой выпуклый многогранник.

8. Общая задача математического программирования формулируется так:

найти максимум (минимум) функции $f(x)$ при ограничениях

$$g_i(x_1, \ldots, x_j, \ldots, x_n) \le b_i, \quad i = \overline{1,m}; \tag{B.1}$$

$$x_j \ge 0, \quad j = \overline{1,n}. \tag{B.2}$$

При этом:

функция $f(\overline{x})$ называется целевой функцией;

система неравенств (В.1) и условия неотрицательности переменных (В.2) называются системой ограничений задачи;

всякое решение задачи с учетом системы ограничений, т.е. совокупность значений переменных $x_1, \ldots, x_n$, удовлетворяющих условиям (В.1) и (В.2), называется допустимым решением;

совокупность точек n-мерного пространства, удовлетворяющих системе ограничений, образует так называемую допустимую область;

допустимое решение, максимизирующее (минимизирующее) целевую функцию, называется оптимальным.

Итак, задача математического программирования заключается в нахождении оптимального решения, обеспечивающего максимальное (минимальное) значение целевой функции с учетом системы ограничений на переменные.

Замечания. Постановка задачи математического программирования является достаточно общей. Если какая-то практическая задача на первый взгляд кажется иной, то, как правило, с помощью простых математических преобразований её можно свести к задаче рассматриваемого вида.

1. Если требуется найти минимум функции $f(\overline{x})$, то это эквивалентно поиску максимума функции $-f(\overline{x})$, т.е. $\min\limits_{x} f(\overline{x}) = -\max\limits_{x}\left[-f(\overline{x})\right]$.

2. Если заданны неравенства вида

$$g(\overline{x}) \geq \overline{b} \ ,$$

то простой переменой знака можно прийти к виду (В.1), а именно

$$-g(\overline{x}) \leq -\overline{b} \ .$$

3. Если на переменные x_j не наложено условие неотрицательности, а заданно ограничение

$$x_j \geq x_{j\_\min} \ ,$$

то, вводя замену переменных

$$x'_j = x_j - x_{j\_\min} \ ,$$

получаем для новой переменной условие (В.1), т.е. $x'_j \geq 0$.

4. Функции $f(\overline{x})$ может иметь несколько экстремумов, а именно:

локальный и глобальный экстремумы (пусть это будет максимум);

функция $f(\bar{x})$, определенная на области D, достигает на ней глобального максимума в точке $\bar{x}^0 \in D$, если неравенство $f(\bar{x}) \le f(\bar{x}^0)$ справедливо для любой точки $\bar{x} \in D$;

функция $f(\bar{x})$ достигает локального максимума в точке $\bar{x}^0 \in D$, если неравенство $f(\bar{x}) \le f(\bar{x}^0)$ справедливо для любой точки x из некоторой окрестности точки $\bar{x}^0$.

5. В математическом анализе для нахождения экстремума функций используются производные. Эти методы называются классическими методами оптимизации.

Различают задачи безусловного и условного экстремумов.

Задача на безусловный экстремум – задача максимизации функции $f(\bar{x})$ при отсутствии ограничений. Необходимые условия экстремума в этом случае записываются в виде системы уравнений:

$$\frac{\partial f(x_1, \ldots, x_n)}{\partial x_j} = 0, \quad j = \overline{1, n}.$$

Решая данную систему уравнений, находят стационарные точки, а затем после их анализа – оптимальную точку.

Задача на условный экстремум предусматривает ограничения в виде равенств: т.е. минимизировать $f(\bar{x})$ при ограничениях $g(\bar{x}) = \bar{b}$.

В векторной форме g и $\bar{b}$ – m-мерные векторы.

Эта задача сводится к задаче на безусловный экстремум с помощью множителей Лагранжа. С этой целью образуют функцию Лагранжа:

$$L(\bar{x}, \lambda) = f(\bar{x}) + \sum_{i=1}^{m} \lambda_i \left[b_i - g_i(x) \right],$$

где λ_i – множители Лагранжа.

Функция $L(\bar{x}, \lambda)$ зависит от $m + n$ переменных, на которые наложены ограничения.

Оптимальную точку находят в результате решения и последующего анализа системы уравнений:

$$\begin{cases} \dfrac{\partial L}{\partial x_j} = 0, \; i = \overline{1, n}, & \dfrac{\partial f}{\partial x_j} - \sum\limits_{i=1}^{m} \lambda_i \dfrac{\partial g_i}{\partial x_j} = 0; \\ \dfrac{\partial L}{\partial \lambda_i} = 0, \; j = \overline{1, m}, & b_i - g_i(x) = 0. \end{cases}$$

6. Классические методы оптимизации применяют лишь для решения сравнительно простых задач математического программирования, так как они имеют ряд недостатков.

- Для использования методов нужно, чтобы функции $f(\overline{x})$ и $g(\overline{x})$ были непрерывны и имели частные производные, по крайней мере, второго порядка. Поскольку на практике функции не редко задаются не аналитически, а таблично или иными, более сложными способами, то это затрудняет вычисление и анализ производных.
- Применение классических методов связано с большим объемом вычислений. При этом, во-первых, необходимо решить систему алгебраических уравнений, которая в большинстве случаев нелинейна. Это в свою очередь достаточно сложная задача. Во-вторых, для анализа стационарных точек требуется вычислять вторые производные целевой функции, что часто затруднительно из-за громоздкости выражений.
- С помощью классических методов можно найти экстремум функции только внутри области.

Если оптимальная точка находится на границе области, то методы эти бессильны.

Итак, поскольку классические методы оптимизации не всегда приводят к желаемому результату, необходимы принципиально новые методы, составляющие предмет математического программирования.

Глава 1

ОСНОВЫ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

1.1. Задача линейного программирования и ее геометрический смысл

Постановка задачи. Найти максимум функции

$$f(\overline{x}) = < c, \overline{x} > + c_0$$

при ограничениях

$$\begin{aligned} A\overline{x} &\le \overline{b}; \\ \overline{x} &\ge 0, \end{aligned} \qquad (1.1)$$

где $\overline{x}$ – n-мерный вектор; $\overline{c}$ – n-мерный вектор; A – матрица $m \times n$; $\overline{b}$ – m-мерный вектор; c_0 – вещественное число.

Суть задачи состоит в том, что необходимо найти неотрицательные значения компонентов вектора $\overline{x}$, удовлетворяющие системе линейных неравенств, при которых линейная целевая функция достигает максимума.

Рассмотрим упрощенные модели некоторых прикладных задач, приводящих к задачам линейного программирования.

Пример 1. Пусть некоторая фирма выпускает n видов продукции и использует при этом m видов сырья. Причем ресурсы сырья ограничены: i-го вида сырья не более b_i единиц. На изготовление одной единицы продукции j-го вида (всего изготавливается $x_j \ge 0, j = 1, n$) тратится a_{ij} единиц i-го сырья, т.е. по i-му виду сырья должно выполняться условие:

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, \quad i = \overline{1, m}.$$

Известно также, что на каждой единице продукции j-го вида фирма получает c_j единиц прибыли, т.е. общая прибыль составляет:

$$\sum_{j=1}^{n} c_j x_j .$$

Требуется определить, сколько единиц x_j каждого вида продукции должна изготовлять фирма, чтобы получить максимальную прибыль.

Пример 2. У фермера имеется m участков земли, площадь каждого составляет b_i га ($i = \overline{1, m}$). Фермер собирается посеять n культур – x_{ij} ($j = \overline{1, n}$) – площадь j-й культуры на i-м участке.

Прибыль с одного центнера j-й культуры составляет c_j рублей.

Фермер хочет получить максимальную прибыль.

Известно, что средняя урожайность j-й культуры на i-м участке равна a_{ij} центнеров с гектара.

Кроме того, по каким-то причинам фермер должен получить каждой культуры не менее d_j центнеров (или не более).

Сформулируем задачу:

определить максимальную суммарную прибыль

$$\sum_{j=1}^{n} \left(c_j \sum_{i=1}^{m} a_{ij} x_{ij} \right)$$

при ограничениях на переменные x_{ij}

$$x_{ij} \geq 0, \quad i = \overline{1, m}, \quad j = \overline{1, n} ,$$

площадь i-го участка

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

и урожай j-й культуры

$$\sum_{i=1}^{m} x_{ij} a_{ij} \le d_j, \quad j = \overline{1, n};$$

$$\left(\sum_{i=1}^{m} x_{ij} a_{ij} \ge d_j \right).$$

К задачам линейного программирования также сводятся и многие другие прикладные задачи технико-экономического содержания.

Следует заметить, что приведенные примеры представляют лишь приближенную, упрощенную математическую модель реальных задач, и некритическое использование получаемых на основе анализа этих моделей результатов может привести иногда к парадоксам.

Это относится не только к задачам линейного программирования, но и к другим математическим моделям. Вполне может оказаться, что принятая математическая модель, обычно составленная на основе приближенных данных о реальном моделируемом явлении, не охватывает какие-либо важные существенные стороны исследуемого явления, что приводит к результатам, существенно расходящимся с реальностью. Однако не нужно делать вывод, что линейные модели слишком просты и вовсе непригодны для исследования реальных задач. (Следует помнить, что существует противоречие: сложная адекватная математическая модель, учитывающая много факторов, с одной стороны, и возможность решения задачи, с другой.) Наоборот, практика показывает, что линейное программирование может быть успешно применено для исследования широкого класса реальных технических и экономических задач. Кроме того, задачи линейного программирования часто используются в качестве вспомогательных во многих методах решения более сложных нелинейных задач оптимизации.

Рассмотрим *геометрический смысл* задачи линейного программирования (на примере функции двух переменных).

Как отмечено выше, допустимая область задачи линейного программирования геометрически изображается пересечением полуплоскостей, определенных линейными неравенствами системы ограничений и представляет собой *выпуклый многоугольник* (рис. 1.1).

Линии уровня линейной целевой функции – это прямые линии $c_1 x_1 + c_2 x_2 + c_0 = \text{const}$, параллельные друг другу, они перпендикулярны вектору $\overline{c}$, который равен градиенту линейной функции.

Оптимальной является точка, лежащая на линии уровня, принимающей наибольшее значение.

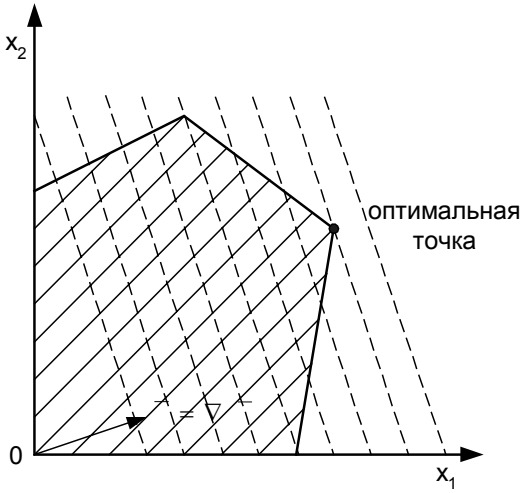


Рис. 1.1. Допустимая область – выпуклый многоугольник (оптимальная точка на пересечении ребер)

Такая точка не может быть внутренней, она всегда лежит на границе, причем соответствует именно вершине многоугольника, за исключением случая, когда линии уровня параллельны некоторому ребру многоугольника (рис. 1.2).

В этом случае задача имеет бесчисленное множество оптимальных точек, принадлежащих указанному ребру.

Задача линейного программирования не имеет решения, если система ограничений несовместна, т.е. допустимое множество пусто (рис. 1.3).

Задача не имеет решения также, если в направлении увеличения функции (рис. 1.4) допустимая область не замкнута.

Если же в направлении увеличения функции (см. рис. 1.4) допустимая область замкнута, то задача решение имеет.

Итак, из геометрического смысла задачи линейного программирования следует, что, в общем случае, оптимальное решение находится в одной из вершин допустимой многогранной области. При решении задачи осуществляется переход от одной вершины допус-

тимой области к другой, соседней с ней вершине, причем такой, в которой значение целевой функции больше. Такое движение составляет сущность симплекс-метода – основного метода решения задач линейного программирования.

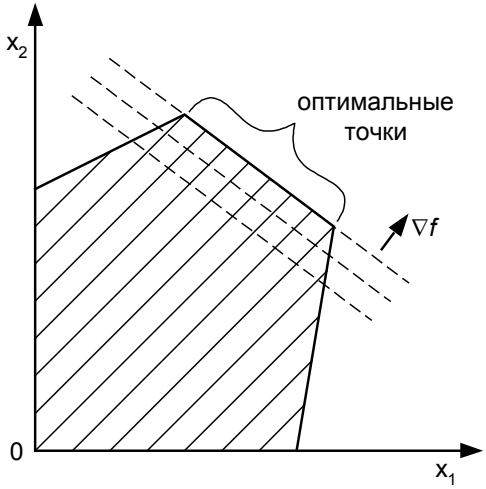


Рис. 1.2. Частный случай допустимой области (оптимальная точка на ребре)

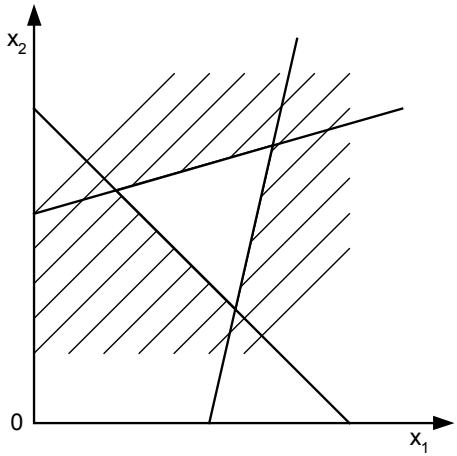


Рис. 1.3. Допустимая область – множество пусто

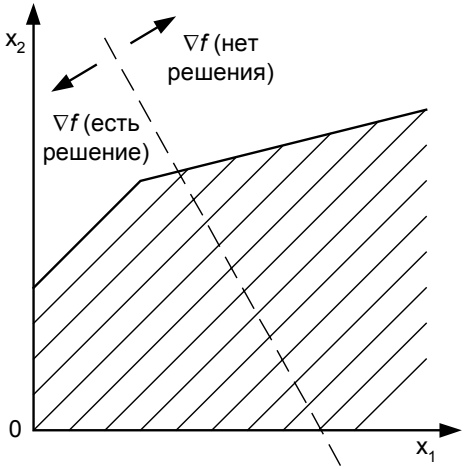


Рис. 1.4. Допустимая область не замкнута

1.2. Симплекс-метод

1.2.1. Порядок решения задач линейного программирования симплекс-методом

Для решения симплекс-методом задача (1.1) должна быть приведена к *стандартному* (*каноническому*) виду:

найти $\max f(\overline{x}) = <\overline{c}, \overline{x}> + c_0$ при ограничениях

$$\begin{gathered} A\overline{x} = \overline{b}, \\ x_{ij} \geq 0, \quad i = \overline{1, n}, \quad j = \overline{i, m}. \end{gathered} \tag{1.2}$$

Приведение задачи к стандартному виду осуществляется с помощью введения дополнительных (ослабляющих) переменных:

1) $A\overline{x} \leq \overline{b} \;\rightarrow\; A\overline{x} + \overline{x}' = \overline{b}$;

2) $A\overline{x} \geq \overline{b} \;\rightarrow\; A\overline{x} - \overline{x}' = \overline{b}$,

где $\overline{x}'$ – неотрицательный вектор.

Пример.

$$f(\overline{x}) = 3x_1 - 2x_2 + x_3; \quad f(\overline{x}) = 3x_1 - 2x_2 + x_3 + 0x_4 + 0x_5;$$

$$\begin{cases} 2x_1 - 3x_3 \le 1; \\ 3x_1 + 4x_2 + x_3 = 6; \\ x_1 + 2x_2 - x_3 - x_5 \ge 2; \\ x_j \ge 0, \quad j = \overline{1,3}. \end{cases} \rightarrow \begin{cases} 2x_1 - 3x_3 + x_4 = 1; \\ 3x_1 + 4x_2 + x_3 = 6; \\ x_1 + 2x_2 - x_3 - x_5 = 2; \\ x_j \ge 0, \quad j = \overline{1,5}. \end{cases}$$

В результате введения ослабляющих переменных получается задача, эквивалентная исходной, так как эти переменные не влияют на значение целевой функции – они входят в нее с нулевыми коэффициентами.

С практической точки зрения можно считать, что ранг системы ограничений (1.2) равен числу уравнений m, т.е. $r = m$. В противном случае хотя бы одно из уравнений представляет собой линейную комбинацию остальных и поэтому является лишним.

Кроме того, как было отмечено выше, ранг $r \le n$ (чаще всего $r < n$) и система имеет бесчисленное множество решений (если, конечно, она вообще совместна).

Выделим среди n переменных первые r и назовем их *базисными переменными*. Остальные $n - r$ переменные назовем *свободными*.

Итак, $x_1, x_2, \ldots, x_r$ – базисные, $x_{r+1}, x_{r+2} \ldots, x_n$ – свободные переменные.

Систему ограничений можно разрешить относительно базисных переменных:

$$\begin{cases} x_1 = \beta_1 - \left(\alpha_{1r+1}x_{r+1} + \alpha_{1r+2}x_{r+2} + \ldots + \alpha_{1n}x_n\right), \\ \qquad \ldots \\ x_i = \beta_i - \left(\alpha_{ir+1}x_{r+1} + \alpha_{ir+2}x_{r+2} + \ldots + \alpha_{in}x_n\right), \\ \qquad \ldots \\ x_r = \beta_r - \left(\alpha_{rr+1}x_{r+1} + \alpha_{rr+2}x_{r+2} + \ldots + \alpha_{rn}x_n\right). \end{cases} \tag{1.3}$$

Свободные переменные в выражении (1.3) могут принимать любые значения. Следовательно, система ограничений имеет бесчисленное множество решений. Задача заключается в том, чтобы среди этого бесчисленного множества выбрать такое решение, которое доставляет максимум целевой функции.

Следует отметить, что свободные переменные выбираются таким образом, чтобы коэффициенты β_i были неотрицательны, т.е. $\beta_i \geq 0$.

Во многих задачах разрешение системы ограничений относительно базисных переменных до вида (1.3) является тривиальным. В некоторых задачах это сделать сложно, но существует, так называемый, метод искусственного базиса, который помогает это сделать (будет рассмотрен ниже).

Выразим также целевую функцию через свободные переменные:

$$f(\bar{x}) = \gamma_0 - \left(\gamma_{r+1}x_{r+1} + \gamma_{r+2}x_{r+2} + \ldots + \gamma_n x_n\right). \qquad (1.4)$$

Возьмем все свободные переменные равными нулю:

$$x_{r+1} = x_{r+2} = \ldots = x_n = 0,$$

тогда из выражения (1.3) следует, что значения базисных переменных соответственно равны:

$$x_1 = \beta_1, \ldots, x_i = \beta_i, \ldots, x_r = \beta_r.$$

Полученное таким образом решение системы ограничений

$$\bar{x}_{б1} = \left(\beta_1, \beta_2, \ldots, \beta_r, 0, \ldots, 0\right)^{т}$$

является допустимым, так как все его компоненты неотрицательны ($\beta_i \geq 0$).

Такое решение называют базисным (в базисном решении отличны от нуля ровно r переменных). В этом случае значение целевой функции

$$f(\bar{x}_{б1}) = \gamma_0.$$

При таком подходе решение задачи линейного программирования заключается в последовательном переходе от одного базисного решения к другому (базисные решения соответствуют вершинам допустимого многогранника), при котором значение целевой функции монотонно возрастает.

Если проанализировать выражение (1.4), то видно, что значение целевой функции можно увеличить только в случае, если имеются отрицательные коэффициенты γ_j. Если все коэффициенты $\gamma_j \geq 0$, то значение целевой функции увеличить нельзя. Поэтому *признаком оптимальности решения* поставленной задачи максимизации

является неотрицательность всех коэффициентов при свободных переменных в выражении (1.4).

Если среди коэффициентов есть хотя бы один отрицательный, например γ_j, тогда за счет увеличения x_j можно увеличить значение $f(\overline{x})$. Однако это возможно в определенных пределах, так как выбор x_j влияет на базисные переменные, которые могут стать недопустимыми, т.е. отрицательными.

Выразим базисные переменные из системы (1.3), считая все свободные переменные за исключением x_j равными нулю:

$$\begin{cases} x_1 = \beta_1 - \alpha_{1j} x_j; \\ \dots \\ x_i = \beta_i - \alpha_{ij} x_j; \\ \dots \\ x_r = \beta_r - \alpha_{rj} x_j. \end{cases}$$

При этом $f(\overline{x}) = \gamma_0 - \gamma_j x_j$.

Если все коэффициенты $\alpha_{ij} \le 0$, $i = \overline{1, r}$, то увеличение x_j может быть неограниченным, это приводит к неограниченному увеличению базисных переменных и они никогда не станут отрицательными. Но это приводит и к бесконечному возрастанию целевой функции, т.е. в этом случае задача не имеет решения.

Итак, *признаком того, что задача не имеет решения* является условие

$$\alpha_{ij} \le 0, \quad i = \overline{1, r},$$

где индекс j соответствует $\gamma_j < 0$.

Рассмотрим случай, когда имеются $\alpha_{ij} > 0$. Тогда увеличение x_j приводит к тому, что базовая переменная x_i будет уменьшаться до тех пор, пока не станет равной нулю при некотором значении x_j. Это значение определяется уравнением

$$x_i = \beta_i - \alpha_{ij} x_j = 0.$$

Отсюда $x_j = \dfrac{\beta_i}{\alpha_{ij}}$.

Если несколько $\alpha_{ij} > 0$, то первой обратится в ноль переменная x_l, для которой отношение $\dfrac{\beta_l}{\alpha_{lj}}$ минимально, т.е. $\dfrac{\beta_l}{\alpha_{lj}} = \min\limits_i \left\{ \dfrac{\beta_i}{\alpha_{ij}} \right\} = \rho$ (минимум берется по всем i, для которых $\alpha_{ij} > 0$).

Когда x_l обратится в нуль, остальные базовые переменные будут еще принимать неотрицательные значения.

Элемент α_{lj} называют *разрешающим*. Он определяет переменную x_l, которую выводят из базиса $\bar{x}_{б1}$, и указывает свободную переменную x_j, вводимую в новый базис $\bar{x}_{б2}$:

$$\begin{cases} x_{r+1} = \ldots = x_n = 0, \quad \text{кроме} \quad x_j = \rho, \\ x_1 = \beta_1 - \alpha_{1j} x_j, \\ \ldots \\ x_l = 0, \\ \ldots \\ x_r = \beta_r - \alpha_{rj} x_j. \end{cases}$$

Значение целевой функции $f(\bar{x}_{б2}) = \gamma_0 - \gamma_j \rho > f(x_{б1})$, так как $\gamma_j < 0$, а $\rho > 0$.

Может оказаться, что $\rho = 0$, тогда целевая функция сохраняет предыдущее значение, хотя переменные базиса меняются. Такой случай называется *вырожденным* и встречается редко, для него существуют специальные методы решения задачи.

Поскольку в базис введена новая переменная x_j и выведена x_l, выразим как и прежде базисную переменную x_j через свободные переменные. Для этого понадобится уравнение из системы (1.3)

$$x_l = \beta_l - \left(\alpha_{lr+1}x_{r+1} + \ldots + \alpha_{lj}x_j + \ldots + \alpha_{ln}x_n\right).$$

Из этого уравнения получим

$$x_j = \frac{\beta_l}{\alpha_{lj}} - \left(\frac{\alpha_{lr+1}}{\alpha_{lj}}x_{r+1} + \ldots + \frac{1}{\alpha_{lj}}x_l + \ldots + \frac{\alpha_{ln}}{\alpha_{lj}}x_n\right).$$

Подставив x_j в остальные уравнения системы (1.3), получим выражения для нового базиса:

$$x_i = \beta'_i - \left(\alpha'_{ir+1}x_{r+1} + \ldots + \alpha'_{ij}x_l + \ldots + \alpha'_{in}x_n\right),$$

где на j-м месте стоит x_l – новая свободная переменная.

Целевая функция через новые свободные переменные запишется в виде

$$f(\overline{x}) = \gamma'_0 - \left(\gamma'_{r+1}x_{r+1} + \ldots + \gamma'_j x_l + \ldots + \gamma'_n x_n\right).$$

Далее все рассуждения повторяются снова, т.е. исследуются знаки коэффициентов γ'_j и проводятся соответствующие преобразования до получения окончательного результата.

Для облегчения процесса поиска оптимального решения используются специальные таблицы, называемые *симплекс-таблицами*.

1.2.2. Алгоритм решения задач линейного программирования с использованием симплекс-таблиц

1. Каноническая система ограничений $A\overline{x} = \overline{b}$ приводится к виду, в котором базисные переменные выражены через свободные, т.е. к виду

$$\begin{cases} x_1 = \beta_1 - \left(\alpha_{1r+1}x_{r+1} + \alpha_{1r+2}x_{r+2} + \ldots + \alpha_{1n}x_n\right); \\ \ldots \\ x_r = \beta_r - \left(\alpha_{rr+1}x_{r+1} + \alpha_{rr+2}x_{r+2} + \ldots + \alpha_{rn}x_n\right), \end{cases}$$

а целевая функция к виду (1.4):

$$f(\overline{x}) = \gamma_0 - \left(\gamma_{r+1}x_{r+1} + \gamma_{r+2}x_{r+2} + \ldots + \gamma_n x_n\right).$$

2. Заполняется симплекс-таблица (табл. 1.1).

Таблица 1.1

Базис	Свободные члены	Переменные									
		x_1	…	x_i	…	x_r	x_{r+1}	…	x_j	…	x_n
x_1	β_1	1	…	0	…	0	α_{1r+1}	…	α_{1j}	…	α_{1n}
…	…	…	…	…	…	…	…	…	…	…	…
x_i	β_i	0	…	1	…	0	α_{ir+1}	…	α_{ij}	…	α_{in}
…	…	…	…	…	…	…	…	…	…	…	…
x_r	β_r	0	…	0	…	1	α_{rr+1}	…	α_{rj}	…	α_{rn}
$f(\overline{x})$	γ_0	0	…	0	…	0	γ_{r+1}	…	γ_j	…	γ_n

Каждая строка табл. 1.1 соответствует уравнению, а последняя строка соответствует целевой функции.

3. В строке коэффициентов целевой функции (не считая γ_0) выбирается $\gamma_j < 0$, обычно максимальным по модулю.

Если все $\gamma_j \geq 0$, то оптимальное решение достигнуто, причем значения переменных определяются столбцом свободных членов, а оптимальное значение функции – клеткой, соответствующей свободному члену.

4. В j-м столбце среди положительных коэффициентов α_{ij} выбирается разрешающий элемент α_{lj}, т.е. элемент для которого минимально отношение $\dfrac{\beta_l}{\alpha_{lj}}$.

Если положительных коэффициентов нет, то задача не имеет решения.

5. Делятся все члены строки, содержащей разрешающий элемент, на α_{lj}. Полученная строка вносится на то же место в новой таблице.

6. Из каждой оставшейся i-й $(i \neq l)$ строки вычитается получившаяся строка, умноженная на коэффициент при x_j в l-й стро-

ке. В результате в клетках, соответствующих j-му столбцу появляются нули. Преобразованные строки записываются в новой таблице на место прежних.

Переменные x_j и x_l в новой таблице меняются местами, т.е. x_j вводится в базис вместо x_l.

Далее идет переход на шаг 3.

1.2.3. Примеры решения практических задач линейного программирования

Пример 1.1. Составить математическое описание следующих задач в виде задач линейного программирования.

Задача 1. Из одного города в другой ежедневно отправляются пассажирские и скорые поезда. В табл. 1.2 указан состав поездов каждого типа, количество вагонов в парке и максимальное число пассажиров, на которое рассчитан вагон каждого вида.

Таблица 1.2

Поезда	Вагоны				
	Багажный	Почтовый	Плацкартный	Купейный	Спальный
Скорый	1	1	5	6	3
Пассажирский	1	–	8	4	1
Число вагонов	12	8	81	70	26
Число пассажиров в вагоне	$\alpha_1 = 0$	$\alpha_2 = 0$	$\alpha_3 = 58$	$\alpha_4 = 40$	$\alpha_5 = 32$

Определить число скорых x_1 и пассажирских x_2 поездов, которые необходимо формировать ежедневно из имеющихся вагонов, чтобы число перевозимых пассажиров было максимально.

Решение. Целевая функция:

$$f(\overline{x}) = c_1 x_1 + c_2 x_2,$$

где c_1 – число пассажиров в скором вагоне; c_2 – число пассажиров в пассажирском вагоне.

Найти $\max f(\overline{x})$ при ограничениях:

$1x_1 + 1x_2 \le 12$ – ограничения на число багажных вагонов;

$1x_1 + 0x_2 \le 8$ – ограничения на число почтовых вагонов;

$5x_1 + 8x_2 \le 81$ – ограничения на число плацкартных вагонов;

$6x_1 + 4x_2 \le 70$ – ограничения на число купейных вагонов;

$3x_1 + 1x_2 \le 26$ – ограничения на число спальных вагонов.

При этом

$$c_1 = 1 \cdot \alpha_1 + 1 \cdot \alpha_2 + 5 \cdot \alpha_3 + 6 \cdot \alpha_4 + 3 \cdot \alpha_5;$$

$$c_2 = 1 \cdot \alpha_1 + 0 \cdot \alpha_2 + 8 \cdot \alpha_3 + 4 \cdot \alpha_4 + 1 \cdot \alpha_5.$$

Задача 2. В начале рабочего дня из автобусного парка выходят на линию x_1 автобусов, через час еще x_2 автобусов, еще через час – еще x_3 автобусов.

Каждый автобус работает непрерывно 8 ч. Рабочий день составляет 10 ч.

Минимально необходимое число машин на линии в i -й час рабочего дня $(i = \overline{1,10})$ равно b_i. Превышение этого числа приводит к дополнительным издержкам: c_i рублей в течение i -го часа на каждый дополнительный автобус.

Определить количество автобусов x_1, x_2, x_3, выходящих на линию в первые часы рабочего дня, так, чтобы минимизировать дополнительные издержки в течение всего рабочего дня.

Решение. Составим табл. 1.3.

Таблица 1.3

Часы	1	2	3	…	8	9	10
Минимальное количество автобусов	b_1	b_2	b_3	…	b_8	b_9	b_{10}
Количество автобусов	x_1	x_1 + x_2	x_1 + x_2 + x_3	…		x_2 + x_3	x_3

С учетом табл. 1.3 целевая функция может быть представлена в виде:

$$f(x) = (x_1 - b_1)c_1 + (x_1 + x_2 - b_2)c_3 + (x_1 + x_2 + x_3 - b_3)c_3 + \ldots +$$
$$+ (x_1 + x_2 + x_3 - b_8)c_8 + (x_2 + x_3 - b_9)c_9 + (x_3 - b_{10})c_{10} =$$
$$= x_1 \sum_{i=1}^{8} c_i + x_2 \sum_{i=2}^{9} c_i + \sum_{i=3}^{10} c_i - \sum_{i=1}^{10} b_i c_i \,.$$

Необходимо найти $\min f(\overline{x})$ при ограничениях:

$$\begin{cases} x_1 \ge b_1; \\ x_1 + x_2 \ge b_2; \\ x_1 + x_2 + x_3 \ge \max\limits_{3 \le i \le 8} b_i; \\ x_2 + x_3 \ge b_9; \\ x_3 \ge b_{10}. \end{cases}$$

Пример 1.2. Решение задач симплекс-методом.

Задача 1. Найти $\max f(\overline{x}) = 21x_1 + 11x_2$ при ограничениях:

$$\begin{cases} 7x_1 + 4x_2 \le 13; \\ x_1 \le 1; \\ 2x_1 + x_2 \le 3; \\ x_1 \ge 0, x_2 \ge 0; \end{cases}$$

Решение. 1. Приведем систему ограничений к каноническому виду путем добавления ослабляющих переменных x_3, x_4:

$$\begin{cases} 7x_1 + 4x_2 + x_3 = 13; \\ x_1 + x_4 = 1; \\ 2x_1 + x_2 + x_5 = 3. \end{cases}$$

2. Разрешим систему ограничений относительно базисных переменных. Ранг системы $r = 3$. В качестве базисных переменных примем переменные x_3, x_4, x_5. Тогда переменные x_1, x_2 будут свободными.

$$\begin{cases} x_3 = 13 - (7x_1 + 4x_2); \\ x_4 = 1 - x_1; \\ x_5 = 3 - (2x_1 + x_2). \end{cases}$$

Целевая функция: $f(\overline{x}) = 0 - (-21x_1 - 11x_2)$.

3. Составим табл. 1.4.

Таблица 1.4

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5
x_3	13	7	4	1	0	0
x_4	1	1	0	0	1	0
x_5	3	2	1	0	0	1
$f(\overline{x})$	0	–21	–11	0	0	0

$\min\left\{\frac{13}{7}, \frac{1}{1}, \frac{3}{2}\right\} = \frac{1}{1}$, $\alpha_{ij} = 1$ – разрешающий элемент.

4. Составим табл. 1.5. Для чего выведем элемент x_4 из базиса, а элемент x_1 введем в базис.

Таблица 1.5

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5
x_3	6	0	4	1	–7	0
x_1	1	1	0	0	1	0
x_5	3	0	1	0	–2	1
$f(\overline{x})$	21	0	–11	0	21	0

$\min\left\{\frac{6}{4}, \frac{1}{1}\right\} = \frac{1}{1}$.

5. Составим табл. 1.6. Для этого удалим из базиса элемент x_5 и введем в базис – x_2.

Таблица 1.6

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5
x_3	2	0	0	1	1	–4
x_1	1	1	0	0	1	0
x_2	1	0	1	0	–2	1
$f(\overline{x})$	32	0	0	0	–1	11

6. Составим табл. 1.7, которая образуется при удалении из базиса элемента x_1 и ввода в базис элемента x_4.

Таблица 1.7

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5
x_3	1	–1	0	1	0	–4
x_4	1	1	0	0	1	0
x_2	3	2	1	0	0	1
$f(\overline{x})$	33	1	0	0	0	11

Поскольку в последней строке табл. 1.7 отрицательных коэффициентов нет, то процесс решения задач закончен. При этом $x_1 = 0$, $x_2 = 3$, $x_3 = 1$, $x_4 = 1$, $x_5 = 0$ (x_1, x_5 – свободные переменные).

Оптимальное значение целевой функции – $f(\overline{x}) = 33$.

Геометрическая интерпретация решения задачи представлена на рис. 1.5.

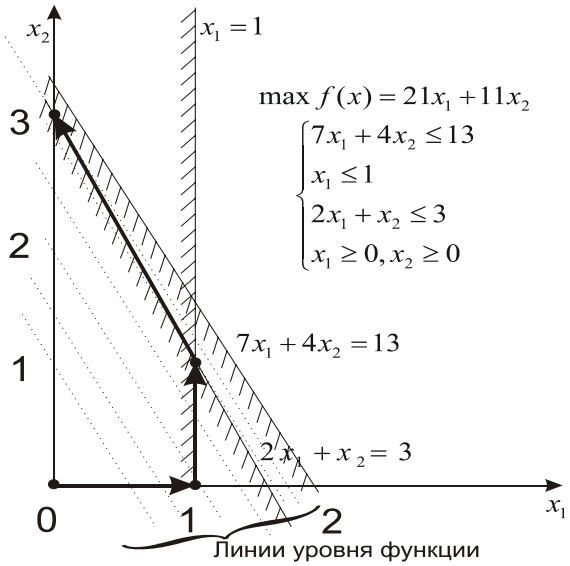


Рис. 1.5. Геометрическое представление задачи

Задача 2. Найти max функции $f(\overline{x}) = 3 + 4x_1 + 6x_2$ при ограничениях:

$$\begin{cases} 2x_1 - 3x_2 \le 0; \\ 2x_1 - 2x_2 \ge -4; \\ 4x_1 - 5x_2 \ge -20; \\ x_1 \ge 0, x_2 \ge 0. \end{cases}$$

Геометрическая интерпретация задачи 2 приведена на рис. 1.6.

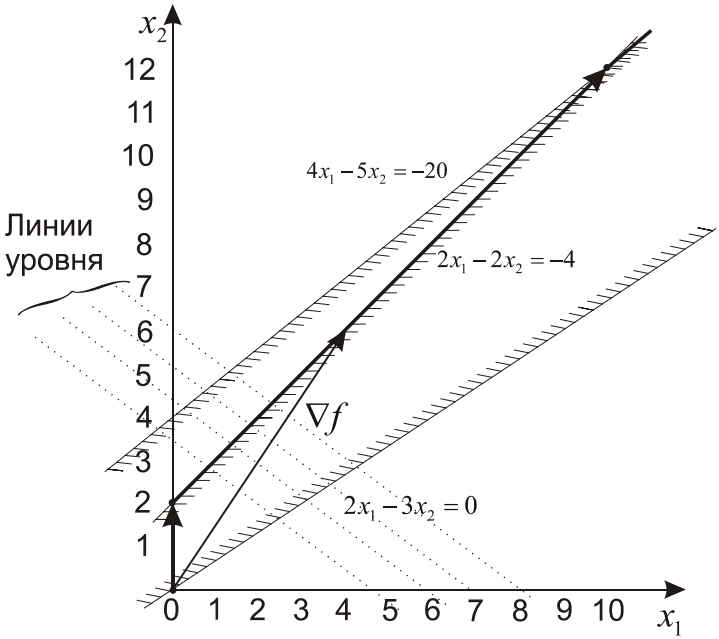


Рис. 1.6. Геометрическое представление задачи 2

Решение будем проводить в соответствии с порядком, изложенным при решении предыдущей задачи.

1. $$\begin{cases} 2x_1 - 3x_2 + x_3 = 0, \\ 2x_1 - 2x_2 - x_4 = -4, \\ 4x_1 - 5x_2 - x_5 = -20. \end{cases} \quad r = 3.$$

2. x_3, x_4, x_5 – базисные; x_1, x_2 – свободные

$$\begin{cases} x_3 = 0 - 2x_1 + 3x_2; \\ x_4 = 4 + 2x_1 - 2x_2; \\ x_5 = 20 + 4x_1 - 5x_2; \end{cases} \quad f(\overline{x}) = 3 + 4x_1 + 6x_2 \text{ – целевая функция.}$$

Составим соответствующие табл. 1.8 – 1.10.

Таблица 1.8

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5
x_3	0	2	–3	1	0	0
x_4	4	–2	2	0	1	0
x_5	20	–4	5	0	0	1
$f(\overline{x})$	3	–4	–6	0	0	0

Таблица 1.9

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5
x_3	6	–1	0	1	3/2	0
x_2	2	–1	1	0	1/2	0
x_5	10	1	0	0	–5/2	1
$f(\overline{x})$	15	–10	0	0	3	0

Таблица 1.10

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5
x_3	16	0	0	1	–1	1
x_2	12	0	1	0	–2	1
x_1	10	1	0	0	–5/2	1
$f(\overline{x})$	115	0	0	0	–-28	10

Как видно, в табл. 1.10 все $\alpha_{ij} < 0$ $(j = 4)$. Следовательно, *задача не имеет решения*.

Задача 3. Найти max функции $f(\overline{x}) = 2 + 4x_1 + 2x_2 + 2x_3$ при ограничениях:

$$\begin{cases} 2x_1 - 3x_2 + 15x_3 \le 3; \\ 2x_1 + 2x_2 + 8x_3 \le 32; \\ 2x_1 - 4x_2 + 16x_3 \le 2; \\ x_1 \ge 0, x_2 \ge 0, x_3 \ge 0. \end{cases}$$

Решение будем проводить в соответствии с порядком, изложенным при решении задач 1 и 2.

1. $$\begin{cases} 2x_1 - 3x_2 + 15x_3 + x_4 = 3; \\ 2x_1 + 2x_2 + 8x_3 + x_5 = 32; \\ 2x_1 - 4x_2 + 16x_3 + x_6 = 2, \end{cases} \quad r = 3.$$

2. $$\begin{cases} x_4 = 3 - (2x_1 - 3x_2 + 15x_3); \\ x_5 = 32 - (2x_1 + 2x_2 + 8x_3); \\ x_6 = 2 - (2x_1 - 4x_2 + 16x_3), \end{cases} \quad f(\overline{x}) = 2 - (-4x_1 - 2x_2 - 2x_3).$$

Составим соответствующие табл. 1.11 – 1.14.

Таблица 1.11

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5	x_6
x_4	3	2	–3	15	1	0	0
x_5	32	2	2	8	0	1	0
x_6	2	2	–4	16	0	0	1
$f(\overline{x})$	2	–4	–2	–2	0	0	0

Таблица 1.12

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5	x_6
x_4	1	0	1	–1	1	0	–1
x_5	30	0	6	-8	0	1	–1
x_1	1	1	–2	8	0	0	1/2
$f(\overline{x})$	6	0	–10	30	0	0	2

Таблица 1.13

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5	x_6
x_2	1	0	1	–1	1	0	–1
x_5	24	0	0	–2	–6	1	–5
x_1	3	1	0	6	2	0	–3/2
$f(\overline{x})$	16	0	0	20	10	0	–8

Таблица 1.14

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5	x_6
x_2	19/5	0	1	–7/5	–1/5	1/5	0
x_6	24/5	0	0	–2/5	–6/5	1/5	1
x_1	51/5	1	0	32/5	1/5	3/10	0
$f(\overline{x})$	272/5	0	0	84/5	2/5	8/5	0

В соответствии с пунктом 3 алгоритма все $\gamma_j > 0$, значит, достигнуто оптимальное решение.

Таким образом, имеем:

$$x_1 = \frac{51}{5},\ x_2 = \frac{19}{5},\ x_3 = 0,\ x_4 = 0,\ x_5 = 0,\ x_6 = \frac{24}{5}$$

(x_3, x_4, x_5 – свободные переменные).

Оптимальное значение целевой функции – $f(\overline{x}) = 54\frac{2}{5}$.

1.3. Вырожденные задачи линейного программирования

1.3.1. Понятия вырожденности и зацикливания решения задач линейного программирования

Определение 1. Если хотя бы одно базисное решение $\overline{x}_{б_i}$ вырожденное, то задача линейного программирования называется *вырожденной*, если все базисные решения $\overline{x}_{б_i}$ – невырожденные, то задачу называют *невырожденной*.

Определение 2. Базисное решение $\overline{x}_{б_i} = (x_1, \ldots, x_r, \ldots, x_n)$ называется *невырожденным*, если оно имеет точно r ($r = m$ – ранг системы ограничений) положительных координат. Если число положительных координат базисного решения меньше r, то решение называется *вырожденным*.

Пример. Рассмотрим задачу 2 из примера 1.2, которая не имеет решения.

Найти max $f(\overline{x}) = 3 + 4x_1 + 6x_2$.

Система ограничений имеет вид:

$$x_3 = 0 - (2x_1 - 3x_2);$$
$$x_4 = 4 - (2x_1 - 3x_2);$$
$$x_5 = 20 - (2x_1 - 3x_2); \qquad r = 3.$$

Первое базисное решение – вырожденное.

$$\left.\begin{aligned} x_3 &= 0; \\ x_4 &= 4; \\ x_5 &= 20 \end{aligned}\right\} \text{ – базисное решение;}$$

$$\left.\begin{aligned} x_1 &= 0; \\ x_2 &= 0; \\ x_3 &= 0 \end{aligned}\right\} \text{ – свободные переменные.}$$

Следовательно, по определению 1, задача – вырожденная (при решении этой задачи симплекс-методом, мы не столкнулись ни с какими особенностями; то что у задачи нет решения с вырожденностью никак не связано).

Рассмотрим ситуации, когда задача может обладать вырожденными базисными решениями.

1. Если ранг матрицы A меньше m $(r < m)$, то все базисные решения задачи – вырожденные. Это не страшно, так как вырожденные задачи линейного программирования решаются не сложнее невырожденных. Такую ситуацию легко исключить, отбросив «лишние» уравнения.

2. Если ранг матрицы A равен m $(r = m)$,то задача также может обладать вырожденными базисными решениями (см. пример), при-

чем это может обнаружиться не на первой итерации симплекс-метода, а позже.

К чему может привести применение симплекс-метода, если он попал в вырожденную точку?

Возможна (но необязательна) ситуация, когда симплекс-таблица принимает вид табл. 1.15.

Таблица 1.15

Базис	Свободный член	x_1	...	x_j	…	x_n	Примечание
x_1	$\beta_1 > 0$	α_{11}	...	α_{1j}	…	α_{1n}	
.	.	.	…	.	…	.	
.	.	.	…	.	…	.	
.	.	.	…	.	…	.	
x_i	$\beta_i = 0$	. . .	… … …	$\alpha_{ij} > 0$	… … …	α_{in}	Коэффициент α_{ij} будет разрешающим элементом, так как отношение $(\beta_i / \alpha_{ij} = 0)$ – минимально
.	.	.	…	.	…	.	
.	.	.	…	.	…	.	
.	.	.	…	.	…	.	
x_r	$\beta_r > 0$	.	…	α_{rj}	…	α_{rn}	
$f(\overline{x})$	γ_0	γ_j	…	γ_j	…	γ_n	$\gamma_j < 0$ – коэффициент, который выбираем

В результате получим следующее текущее базисное решение:

$$x_1 = \beta_1, ..., x_i = 0, ..., x_r = \beta_r, x_{r+1} = x_j = ... = x_n = 0.$$

После изменения таблицы в соответствии с симплекс-методом базисное решение имеет вид.

$$x_1 = \beta_1, ..., x_j = 0, ..., x_r = \beta_r, x_{r+1} = 0, ..., x_i = 0, ..., x_n = 0.$$

Таким образом, новое базисное решение совпадает с предыдущим (в столбце «свободный член» ничего не изменилось). Резуль-

татом итерации явилось то, что изменился базис точки. При этом значение целевой функции ($f(\bar{x})=\gamma_0$), естественно, не увеличилось. (Коэффициенты α_{ij} и γ_j изменились.)

Вычисления можно продолжать, надеясь, что в ходе очередной итерации удастся увеличить значение целевой функции, при этом возможно повторение нескольких «холостых» итераций.

Поскольку число различных базисов любого базисного решения – конечно, то после некоторого числа «холостых» итераций либо:

1) выяснится, что базисное решение, на котором мы «застряли» – оптимально;

2) обнаружится, что целевая функция задачи не ограничена сверху на допустимом множестве;

3) получится новое базисное решение;

4) мы придем к базису текущей точки, который уже фигурировал на одной из предыдущих итераций.

В отличие от благополучных случаев 1, 2 и 3, случай 4 означает, что мы вернемся к состоянию (с той же самой симплекс-таблицей), в котором уже были, и, следовательно, проходя снова и снова через тот же самый ряд пустых итераций, будем возвращаться в это же положение. Такое явление называют *зацикливанием* процесса решения задачи.

Из опыта применения симплекс-метода для исследования моделей реальных явлений сложилось убеждение, что вероятность зацикливания ничтожно мала. Известно лишь несколько специально разработанных задач линейного программирования, в процессе решения которых возникает зацикливание. Поэтому, поскольку зацикливание хотя и маловероятно, но все же возможно, необходимо уметь его устранять.

Оказывается, зацикливание можно предупредить, если внести в симплекс-алгоритм определенные уточнения относительно правила выбора разрешающего элемента. Любое правило выбора разрешающего элемента, с помощью которого можно избежать зацикливания, называют *антициклином*. Имеется несколько причем достаточно простых антициклинов. Остановимся на одном из них.

1.3.2. Антициклин

Легко доказывается тот факт, что в невырожденной задаче минимум отношения $\frac{\beta_i}{\alpha_{lj}}$ достигается на единственном номере l. Следовательно, номер переменной, которая выводится из числа базисных переменных – x_l, и разрешающий элемент – α_{lj} в невырожденных задачах определяется однозначно. В вырожденных задачах это не так, т.е. может быть несколько минимальных отношений $\frac{\beta_i}{\alpha_{ij}}$. Предположим, что на очередной итерации симплекс-таблица имеет вид табл. 1.16.

Таблица 1.16

Базис	Свободный член	x_1	…	x_r	x_{r+1}	…	x_j	…	x_n
x_1	$\beta_1 > 0$	1	…	α_{1r}	α_{1r+1}	…	α_{1j}	…	α_{1n}
.	.	.	…	.	.	…	.	…	.
.	.	.	…	.	.	…	.	…	.
.	.	.	…	.	.	…	.	…	.
x_{l_1}	$\beta_{l_1} > 0$	0	…	0	$\alpha_{l_1 r+1}$	…	$\alpha_{l_1 j} > 0$	…	$\alpha_{l_1 n}$
.	.	.	…	.	.	…	.	…	.
.	.	.	…	.	.	…	.	…	.
.	.	.	…	.	.	…	.	…	.
x_{l_2}	$\beta_{l_2} > 0$	0	…	0	$\alpha_{l_2 r+1}$	…	$\alpha_{l_2 j} > 0$	…	$\alpha_{l_2 n}$
.	.	.	…	.	.	…	.	…	.
.	.	.	…	.	.	…	.	…	.
.	.	.	…	.	.	…	.	…	..
x_r	$\beta_r > 0$	0	…	1	α_{rr+1}	…	α_{rj}	…	α_{rn}
$f(\bar{x})$	γ_0	0	…	0	γ_{r+1}	…	$\gamma_j < 0$	…	γ_n

Предположим, что

$$\frac{\beta_{l_1}}{\alpha_{l_1 j}} = \frac{\beta_{l_2}}{\alpha_{l_2 j}} = \min_{\alpha_{ij} > 0} \frac{\beta_i}{\alpha_{ij}}.$$

Следовательно, нет однозначного выбора разрешающего элемента. Предположим, что мы произвольно выбрали $l = l_1$. Тогда после преобразования таблицы на месте свободного члена β_{l_2} появится нуль:

$$\beta_{l_2} - \frac{\beta_{l_1}}{\alpha_{l_1 j}}\alpha_{l_2 j} = \left(\frac{\beta_{l_2}}{\alpha_{l_2 j}} - \frac{\beta_{l_1}}{\alpha_{l_1 j}}\right)\alpha_{l_2 j} = 0 .$$

Таким образом, появится вырожденное базисное решение и станет возможным зацикливание (Теперь ясно, почему в невырожденной задаче может быть только одно минимальное отношение.)

Доказано [ ], что зацикливание не может возникнуть, если на каждой итерации номер l-й переменной, выводимой из базиса, в случае неоднозначности, выбирать по следующему правилу.

Пусть имеем:

$$Q^0 = \min_{\alpha_{ij}>0} \frac{\beta_i}{\alpha_{ij}} = \frac{\beta_{l_1}}{\alpha_{l_1 j}} = ... = \frac{\beta_{lk}}{\alpha_{l_k j}}$$

(т.е. минимум достигается на номерах $k > 1$).

Надо вычислить отношения $\frac{\alpha_{i1}}{\alpha_{ij}}$ для $i = l_1, ..., l_k$ (т.е. вместо β_i из столбца свободных членов брать α_{i1} из первого столбца, который соответствует x_1) и найти среди них $\min Q^1 = \min\limits_{i=l_1...l_k} \frac{\alpha_{i1}}{\alpha_{ij}}$.

Если минимум Q^1 достигается только для одного номера из $l_1, ..., l_k$ (например, l_2), то вопрос решен – x_{l_2} выводится из базиса, $\alpha_{l_2 j}$ – разрешающий элемент.

Если минимум Q^1 достигается более чем для одного номера, то для всех этих номеров $l_1...l_s$ $(s \le k)$ находится $Q^2 = \min\limits_{i=l_1...l_s} \frac{\alpha_{i2}}{\alpha_{ij}}$ (коэффициенты из второго столбца). И так далее.

Гарантируется, что найдется такое $q\,(1 \le q \le n)$, для которого минимальное значение Q^q достигается только для одного номера l.

Следовательно, приведенное правило всегда однозначно определяет переменную, выводимую из базиса.

Необходимо заметить, что хотя среди прикладных задач линейного программирования вырожденные задачи встречаются довольно часто, тем не менее зацикливание бывает крайне редко. Использование антициклина на каждом шаге симплекс-метода приводит к заметному увеличению времени, которое требуется для решения задачи на ЭВМ. Поэтому на практике чаще всего пользуются *упрощенным правилом* выбора разрешающего элемента в случае неоднозначности, т.е. если минимум достигается на нескольких отношениях $\frac{\beta_i}{\alpha_{lj}}$, то берут в качестве l *наименьший* из номеров i.

В том случае, если обнаружится зацикливание, принимают описанный или какой-нибудь другой антициклин, а после того, как зацикливание преодолено, снова возвращаются к упрощенному правилу выбора разрешающего элемента.

1.4. Метод искусственного базиса

1.4.1. Особенности метода

Для решения задачи линейного программирования симплекс-методом, как было отмечено в п. 1.2, необходимо выразить базисные переменные через свободные, т.е. получить выражение (1.3). *Причем в этом выражении свободные члены должны быть неотрицательны* $(\beta_j \geq 0,\ j = \overline{1, r})$. Во многих задачах эта проблема решается тривиально, но не во всех. В случае, если проблема не решается тривиально, используют метод искусственного базиса. Метод позволяет либо выразить базисные переменные через свободные, либо установить, что система ограничений несовместна. (Процедура Жордана – Гаусса [3] гарантирует $\beta_j \geq 0$ и отличается от симплекс-метода, а метод искусственного базиса сводится к тому же симплекс-методу.)

Запишем основную задачу линейного программирования:

найти $\max f(\overline{x}) = c_1 x_1 + \ldots + c_n x_n + c_0$ при ограничениях

$$\begin{cases} a_{11}x_1 + ... + a_{1n}x_n = b_1, \\ a_{21}x_1 + ... + a_{2n}x_n = b_2, \\ \dots \\ a_{r1}x_1 + ... + a_{rn}x_n = b_r. \end{cases} \qquad (1.5)$$

Предположим, что все коэффициенты $(b_i \geq 0,\ i = \overline{1, r})$. Если какое-то $(b_i < 0,\ )$, то умножением *i*-го уравнения на –1 можно обеспечить неотрицательность b_i.

Введем вспомогательные переменные $(y_i \geq 0,\ i = \overline{1, n})$:

$$\begin{cases} y_1 = b_1 - a_{11}x_1 - ... - a_{1n}x_n; \\ y_2 = b_2 - a_{21}x_1 - ... - a_{2n}x_n; \\ \dots \\ y_r = b_r - a_{r1}x_1 - ... - a_{rn}x_n. \end{cases} \qquad (1.6)$$

Решение системы линейных уравнений (1.5) эквивалентно решению системы (1.6) при дополнительных условиях

$$y_1 = y_2 = \ldots = y_r = 0.$$

Введем также дополнительную целевую функцию:

$$\Phi = y_1 + y_2 + \ldots + y_r.$$

Итак, поставлена вспомогательная задача линейного программирования, которая принимает следующий вид:

найти $\min \Phi$ при ограничениях (1.6) и условиях:

$$x_1 \geq 0, \ldots, x_n \geq 0, \quad y_1 \geq 0, \ldots, y_r \geq 0.$$

Для ее решения можно воспользоваться симплекс-методом. При этом начальный базис уже найден. Действительно, если за базисные переменные принять $y_1, \ldots, y_r$, а за свободные – $x_1, \ldots, x_n$, то в системе (1.6) базисные переменные уже выражены через свободные. Причем, $(b_i \geq 0,\ i = \overline{1, r})$.

Предположим, что вспомогательная задача решена и найден минимум целевой функции Ф. Возможны два варианта:

1) $\min \Phi = 0$. Очевидно, что это возможно, лишь когда все вспомогательные переменные равны нулю, а именно, когда $y_1^* = \ldots = y_r^* = 0$ (* – означает оптимальное решение).

Если выписать из последней симплекс-таблицы оптимальные значения переменных $x_1^*, \ldots, x_n^*$, то очевидно, что они являются решениями системы уравнений (1.5);

2) $\min \Phi > 0$. Это означает, что система (1.6) не имеет решений, для которых $y_1^* = \ldots = y_r^* = 0$.

Отсюда следует, что система (1.5) несовместна.

Рассмотрим более подробно первый случай, когда $\min \Phi = 0$ Предположим, что в последней симплекс-таблице вспомогательной задачи все вспомогательные переменные y_i находятся в группе свободных переменных (табл. 1.17).

Таблица 1.17

Базис	Свободные члены	x_1	x_2	x_3	x_4	y_1	y_2	y_3
x_1	2	1	0	0	1	–2	3	1
x_2	1	0	1	0	2	3	–2	4
x_3	3	0	0	1	–3	1	3	2
–Ф	0	0	0	0	5	7	1	4

Из табл. 1.17 следует, что базисные переменные x_1, x_2, x_3 можно выразить через свободные – x_4, y_1, y_2, y_3 следующим образом:

$$\begin{cases} x_1 = 2 - (x_4 - 2y_1 + 3y_2 + y_3); \\ x_2 = 1 - (2x_4 + 3y_1 - 2y_2 + 4y_3); \\ x_3 = 3 - (-3x_4 + y_1 + y_3 + y_2). \end{cases}$$

В общем виде это можно записать так:

$$\begin{cases} x_j = \beta_j - (\alpha_{jr+1} x_{r+1} + \ldots + \alpha_{jn} x_n + \alpha_{j1} y_1 + \ldots + \alpha_{jr} y_r); \\ j = \overline{1, r}. \end{cases} \quad (1.7)$$

Обратим внимание на то, что все $\beta_j \geq 0,\ \ j = \overline{1,r}.$ Это справедливо, так как в столбце свободных членов находятся оптимальные значения базисных переменных, а они всегда неотрицательны.

Полагая теперь в выражении (1.7) $y_1 = \ldots = y_r = 0,$ получаем искомый начальный базис:

$$\begin{cases} x_j = \beta_j - (\alpha_{jr+1} x_{r+1} + \ldots + \alpha_{jn} x_n); \\ \beta_j \geq 0. \end{cases}$$

Далее необходимо выразить целевую функцию через свободные переменные $x_{r+1}, \ldots, x_n$ и решать основную задачу максимизации функции $f(\overline{x})$ с системой ограничений (1.5). Начальная симплекс-таблица для этой задачи будет лишь немного отличаться от последней симплекс-таблицы для вспомогательной задачи, т.е. будут отсутствовать столбцы, соответствующие переменным y_j, и будет другой строка целевой функции, а основная часть таблицы будет та же.

Однако может сложиться такая ситуация, что в табл. 1.17 вспомогательной задачи не все переменные y_j окажутся в числе свободных, некоторые из них могут оказаться в числе базисных (табл. 1.18).

Таблица 1.18

Базис	Свободные члены	x_1	x_2	x_3	x_4	y_1	y_2	y_3
x_1	2	1	0	1	0	0	1	2
x_2	1	0	1	–1	2	0	0	1
y_1	0	0	0	1	0	1	–1	0
–Ф	0	0	0	2	1	0	0	3

Здесь базисные переменные – $x_1, x_2, y_1,$ а, x_3, x_4, y_2, y_3 – свободные. Причем свободный член, соответствующий y_1, равен нулю – это обязательно, так как в оптимальном решении $y_1^* = 0$ (т.е. имеем дело с вырожденной задачей). В этом случае необходимо

произвести дополнительные действия и перевести вспомогательные переменные y_j из базисных в число свободных.

Возможны два случая.

1. Если ранг матрицы A в выражении (1.5) равен m, т.е. в исходной системе ограничений нет «лишних» (линейно-зависимых) уравнений, то *всегда* можно вывести *все* переменные y_j из числа базисных. Для этого в строке, соответствующей y_j, надо найти положительный коэффициент при свободной переменной x_k и, условно приняв его за разрешающий элемент, осуществить переход к следующей симплекс-таблице обычным способом. Решение вспомогательной задачи при этом не изменится, так как свободный член в строке y_j равен нулю и, следовательно, элементы столбца свободных членов при преобразовании таблицы не изменятся. В последнем примере «1» в заштрихованном прямоугольнике таблицы является разрешающим элементом, поэтому y_1 выводится из базиса в число свободных переменных, а x_3 вводится в базис.

2. Если ранг матрицы A ($r < m$), то оказывается, что $(m - r)$ переменных y_j не удается вывести из числа базисных способом, описанным выше. Это означает, что ограничения задачи, номера которых соответствуют индексам оставшихся векторов y_j можно отбросить – они «лишние».

Таким образом, решение вспомогательной задачи не только позволяет найти начальный базис исходной задачи, но и дает возможность выявить линейно-независимые ограничения данной задачи.

1.4.2. Примеры решения задач с использованием метода искусственного базиса

Пример 1. Найти $\max f(\overline{x}) = 2x_1 + x_2 - 3x_3 + x_4$ при ограничениях:

$$\begin{cases} -2x_1 + 4x_2 + 3x_3 = -4; \\ -x_1 + 2x_2 - 2x_4 = 2 \end{cases}$$

и условиях: $x_1 \geq 0,\ x_2 \geq 0,\ x_3 \geq 0,\ x_4 \geq 0.$

Решение. Сформулируем вспомогательную задачу. Для этого введем две вспомогательные переменные y_1 и y_2.

Необходимо найти $\min \Phi(\bar{y}) = y_1 + y_2$ при ограничениях:

$$\begin{cases} y_1 = 4 - (2x_1 - 4x_2 - 3x_3); \\ y_2 = 2 - (-x_1 + 2x_2 - 2x_4) \end{cases}$$

и условиях неотрицательности переменных $x_1 \geq 0$, $x_2 \geq 0$, $x_3 \geq 0$, $x_4 \geq 0$, $y_1 \geq 0$, $y_2 \geq 0$.

Выразим функцию $\Phi(\bar{y})$ через свободные переменные x_1, x_2, x_3, x_4 (y_1, y_2 – базисные):

$$\Phi = 6 - x_1 + 2x_2 + 3x_3 + 2x_4.$$

Будем искать $\max(-\Phi)$ симплекс-методом, при этом целевую функцию приведем к виду

$$-\Phi = -6 - (-x_1 + 2x_2 + 3x_3 + 2x_4).$$

Составим симплекс-таблицу (табл. 1.19).

Таблица 1.19

Базис	Свободные члены	x_1	x_2	x_3	x_4	y_1	y_2
y_2	4	2	–4	–3	0	1	0
y_2	2	–1	2	0	–2	0	1
–Ф	–6	–1	2	3	2	0	0

В соответствии с симплекс-методом получим табл. 1.20.

Таблица 1.20

Базис	Свободные члены	x_1	x_2	x_3	x_4	y_1	y_2
x_1	2	1	–2	$-3/2$	0	$1/2$	0
y_2	4	0	0	$-3/2$	2	1/2	1
–Ф	-4	0	0	$3/2$	2	1/2	0

Из последней таблицы следует, что решением вспомогательной задачи являются:

$$\Phi = 4\,,\ y_1 = 0\,,\ y_2 = 4\,.$$

Отсюда вывод – система ограничений основной задачи несовместна.

Пример 2. Этот пример дает возможность рассмотреть особенности использования метода искусственного базиса в вырожденных задачах линейного программирования.

Итак, необходимо найти $\max f(\overline{x}) = x_1 + 3x_2 + 2x_3 + 4x_4 + x_5$ при ограничениях:

$$\begin{cases} -x_1 + 4x_3 + 3x_4 = 2; \\ 2x_1 + 3x_2 + 3x_3 + 5x_4 - x_5 = 3; \\ x_1 + 3x_2 + x_3 + 2x_4 + x_5 = 2; \\ 2x_1 + 6x_2 + 8x_3 + 10x_4 = 7; \\ x_j \geq 0,\ j = \overline{1,5}. \end{cases}$$

Как видно, в данной системе ограничений последнее уравнение является суммой трех предыдущих.

Вспомогательная задача имеет вид:

найти $\min \Phi(\overline{y}) = y_1 + y_2 + y_3 + y_4$ при ограничениях:

$$\begin{cases} y_1 = 2 - (-x_1 + 4x_3 + 3x_4); \\ y_2 = 3 - (2x_1 + 3x_2 + 3x_3 + 5x_4 - x_5); \\ y_3 = 2 - (x_1 + 3x_2 + x_3 + 2x_4 + x_5); \\ y_4 = 7 - (2x_1 + 6x_2 + 8x_3 + 10x_4). \end{cases}$$

При этом целевая функция описывается выражением:

$$\Phi = 14 - (4x_1 + 12x_2 + 16x_3 + 20x_4)\,,$$

или

$$-\Phi = -14 - (-4x_1 - 12x_2 - 16x_3 - 20x_4).$$

Составим симплекс-таблицу (табл. 1.21).

Таблица 1.21

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3	y_4
y_1	2	–1	0	4	3	0	1	0	0	0
y_2	3	2	3	3	5	–1	0	1	0	0
y_3	2	1	3	1	2	1	0	0	1	0
y_4	7	2	6	8	10	0	0	0	0	1
–Ф	–14	–4	–12	–16	–20	0	0	0	0	0

Для нахождения разрешающего элемента используем другое правило выбора столбца, а именно, берем первый столбец, у которого $\gamma_j < 0$, а не максимально по модулю, и преобразуем табл. 1.21 в соответствии с симплекс-методом, на основе которого вводим в базис переменную x_1 вместо y_2.

Таблица 1.22

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3	y_4
y_1	7/2	0	3/2	11/2	11/2	–1/2	1	1/2	0	0
x_1	3/2	1	3/2	3/2	5/2	–1/2	0	1/2	0	0
y_3	1/2	0	3/2	–1/2	–1/2	3/2	0	–1/2	1	0
y_4	4	0	3	5	5	1	0	–1	0	1
–Ф	-8	0	–6	–10	–10	–2	0	2	0	0

Исходя из указанного выше правила, в табл. 1.22 выбираем разрешающий элемент в столбце x_2.

На основании симплекс-метода вводим в базис переменную x_2 вместо y_3. При этом получим табл. 1.23.

При выборе разрешающего элемента в табл. 1.23 (столбец x_3) имеем минимальные отношения

$$\min\left\{\frac{3}{6},\frac{1}{2},\frac{3}{6}\right\}=\frac{1}{2},$$

в трех строках, соответствующих элементам y_1, x_1, y_4. Следовательно, задача вырожденная, однозначного выбора разрешающего элемента нет.

Таблица 1.23

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3	y_4
y_1	3	0	0	6	6	–2	1	1	–1	0
x_1	1	1	0	2	3	–2	0	1	–1	0
x_2	1/3	0	1	–1/3	–1/3	1	0	–1/3	2/3	0
y_4	3	0	0	6	6	–2	0	0	–2	1
–Ф	–6	0	0	–12	–12	–4	0	0	4	0

Применяем антициклин. Для этого составим таблицу, элементами которой являются отношения элементов табл. 1.23, указанных выше строк к коэффициентам $\alpha_{13}, \alpha_{23}, \alpha_{43}$ соответственно.

Таблица 1.23а

α_{y_1}/α_{13}	1/2	0	0	1	1	–1/3	1/6	1/6	–1/6	0
α_{x_1}/α_{23}	1/2	1/2	0	1	3/2	–1	0	1/2	–1/2	0
α_{y_4}/α_{43}	1/2	0	0	1	1	–1/3	0	0	–1/3	1/6

Анализируя каждый из столбцов табл. 1.23а в соответствии с подходом, изложенным в п. 1.3.2, получим, что из базиса необходимо вывести переменную y_4.

В результате получим табл. 1.24.

Таблица 1.24

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3	y_4
y_1	0	0	0	0	0	0	1	1	1	–1
x_1	0	1	0	0	1	–4/3	0	1	–1/3	–1/3
x_2	1/2	0	1	0	0	8/9	0	–1/3	5/9	1/18
x_3	1/2	0	0	1	1	–1/3	0	0	–1/3	1/6
–Ф	0	0	0	0	0	0	0	0	0	2

Табл. 1.24 – оптимальная и $\Phi=0$. Следовательно, задача имеет непустое допустимое множество.

Удалить y_1 из базисных переменных невозможно, так как коэффициенты этой строки при свободных переменных x_4 ,x_5 равны нулю. Это значит, что ограничение первое – лишнее, его можно отбросить и ранг матрицы ограничений станет равным числу строк (r = 3).

При этом начальный базис для основной задачи запишется в виде:

$$\begin{cases} x_1 = 0 - \left(x_4 - \dfrac{4}{3} x_5 \right); \\ x_2 = \dfrac{1}{2} - \left(\dfrac{8}{9} x_5 \right); \\ x_3 = \dfrac{1}{2} - \left(x_4 - \dfrac{1}{3} x_5 \right). \end{cases}$$

Выразим целевую функцию $f(\overline{x})$ через свободные переменные x_4, x_5:

$$f(\overline{x}) = x_1 + 3x_2 + 2x_3 + 4x_4 + x_5 = \left(-x_4 + \frac{4}{3} x_5 \right) +$$

$$+ \left(\frac{3}{2} - \frac{8}{3} x_5 \right) + \left(1 - 2x_4 + \frac{2}{3} x_5 \right) + 4x_1 + x_5 = \frac{5}{2} - \left(-x_4 - \frac{1}{3} x_5 \right).$$

Тогда начальная симплекс-таблица для основной задачи принимает вид табл. 1.25.

Таблица 1.25

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5
x_1	0	1	0	0	1	–4/3
x_2	1/2	0	1	0	0	8/9
x_3	1/2	0	0	1	1	–1/3
$f(\bar{x})$	5/2	0	0	0	–1	–1/3

В первой строке табл. 1.25 есть вырожденность. Используя симплекс-метод, вводим в базис переменную x_4, при этом переменную x_1 исключаем из базиса. В результате табл. 1.25 приводится к табл. 1.25а.

Таблица 1.25а

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5
x_4	0	1	0	0	1	–4/3
x_2	1/2	0	1	0	0	8/9
x_3	1/2	–1	0	1	0	1
$f(\bar{x})$	5/2	1	0	0	0	5/3

Поскольку решение задачи на данном этапе не получено, проведем дальнейшие преобразования последней таблицы. Выбираем в качестве разрешающего элемента коэффициент, расположенный в третьей строке таблицы, в соответствии с указанным выше правилом. Получим табл. 1.26, которая требует дальнейших преобразований.

В соответствии с симплекс-методом, выбирая разрешающий элемент в столбце x_1, табл. 1.26 приводим к табл. 1.26а.

Таблица 1.26

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5
x_4	2/3	–1/3	0	4/3	1	0
x_2	1/18	8/9	1	–8/9	0	0
x_5	1/2	–1	0	1	0	1
$f(\bar{x})$	10/3	–2/3	0	5/3	0	0

Таблица 1.26а

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5
x_4	11/16	0	3/8	1	$\underline{1}$	0
x_1	1/16	1	9/8	–1	0	0
x_5	9/16	0	9/8	–1	1	1
$f(\bar{x})$	27/8	0	3/4	1	0	0

Таким образом, как следует из последней таблицы, оптимальным решением исходной задачи будут:

$$x_1 = 1/16,\ x_2 = 0,\ x_3 = 0,\ x_4 = 11/16,\ x_5 = 9/16,\ f(\bar{x}) = 27/8.$$

Пример 3. Найти $\max f(\bar{x}) = 2x_1 + x_3 - x_4 + x_5$ при ограничениях:

$$\begin{cases} -x_1 + 2x_2 + x_3 = 2; \\ 3x_1 + 5x_2 + x_3 + x_4 + 2x_5 = 14; \\ x_1 - x_2 + x_5 = 1. \end{cases}$$

Запишем вспомогательную задачу:
найти $\min\{\Phi(\bar{y}) = y_1 + y_2 + y_3\}$ при ограничениях:

$$\begin{cases} y_1 = 2 - (-x_1 + 2x_2 + x_3); \\ y_2 = 14 - (3x_1 + 5x_2 + x_3 + x_4 + 2x_5); \\ y_3 = 1 - (x_1 - x_2 + x_5). \end{cases}$$

Подставляя значения $\overline{y}$ в формулу для целевой функции вспомогательной задачи, получим:

$$\Phi(\overline{x}) = 17 - (3x_1 + 6x_2 + 2x_3 + x_4 + 3x_5)\,.$$

Будем искать $\max\{-\Phi(x)\}$.

Исходная симплекс-таблица имеет вид табл. 1.27.

Таблица 1.27

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_5
y_1	2	–1	2	1	0	0	1	0	0
y_2	14	3	5	1	1	2	0	1	0
y_3	1	1	–1	0	0	1	0	0	1
–Ф	-17	–3	-6	–2	–1	–3	0	0	0

После ряда последовательных преобразований, в соответствии с симплекс-методом, получим последовательность симплекс-таблиц, отражающих процесс нахождения решения исходной задачи (табл. 1.27а – 1.27.в).

Таблица 1.27а

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_5
1	–1/2	1	1/2	0	0	1/2	0	0	
9	11/2	0	–3/2	1	2	–5/2	1	0	
2	1/2	0	1/2	0	1	1/2	0	1	
–11	–6	0	1	–1	–3	3	0	0	

Если в табл. 1.27б взять элемент 7/11 в качестве разрешающего, то процесс сойдется быстрее и окончательная симплекс-таблица примет вид табл. 1.27в.

Таблица 1.27б

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_5
x_2	20/11	0	1	4/11	1/11	2/11	3/11	1/11	0
x_1	18/11	1	0	–3/11	2/11	4/11	–5/11	2/11	0
y_3	13/11	0	0	7/11	–1/11	9/11	8/11	–1/11	1
–Ф	–13/11	0	0	–7/11	1/11	–9/11	3/11	12/11	0

Таблица 1.27в

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_5
x_2	8/7	0	1	0	1/7	–2/7	–-1/7	1/7	–4/7
x_1	15/7	1	0	0	1/7	5/7	1/7	1/7	3/7
x_3	13/7	0	0	1	–1/7	9/7	8/7	–1/7	11/7
–Ф	0	0	0	0	0	0	1	1	1

В результате найдено оптимальное решение вспомогательной задачи.

Поскольку $\min \Phi = 0$ и $y_1 = y_2 = y_3 = 0$, то начальный базис для основной задачи запишется следующим образом:

$$\begin{cases} x_2 = \dfrac{8}{7} - \left(\dfrac{1}{7}x_4 - \dfrac{2}{7}x_5\right); \\ x_1 = \dfrac{15}{7} - \left(\dfrac{1}{7}x_4 + \dfrac{5}{7}x_5\right); \\ x_3 = \dfrac{13}{7} - \left(-\dfrac{1}{7}x_4 + \dfrac{9}{7}x_5\right). \end{cases}$$

Выразим целевую функцию через x_4 и x_5:

$$f(\overline{x}) = 2x_1 + x_3 - x_4 + x_5 =$$

$$= \frac{30}{7} - \frac{2}{7}x_4 - \frac{10}{7}x_5 + \frac{13}{7} + \frac{1}{7}x_4 - \frac{9}{7}x_5 - x_4 + x_5 = \frac{43}{7} - \frac{8}{7}x_4 - \frac{12}{7}x_5 .$$

С учетом полученного результата составим исходную симплекс-таблицу для основной задачи (табл. 1.28).

Таблица 1.28

Базис	Свободные члены	x_1	x_2	x_3	x_4	x_5
x_2	8/7	0	1	0	1/7	–2/7
x_1	15/7	1	0	0	1/7	5/7
x_3	13/7	0	0	1	–1/7	9/7
$f(\bar{x})$	43/7	0	0	0	8/7	12/7

В нижней строке табл. 1.28, соответствующей целевой функции, нет отрицательных коэффициентов, следовательно, получено оптимальное решение поставленной вначале задачи, а именно:

$$x_1 = \frac{15}{7}, \quad x_2 = \frac{8}{7}, \quad x_3 = \frac{13}{7}, \quad x_4 = 0, \quad x_5 = 0, \quad f(\bar{x}) = \frac{43}{7}.$$

1.5. Двойственные задачи линейного программирования

1.5.1. Основные положения

Понятие двойственности в задачах линейного программирования расширяет границы применения симплексных алгоритмов.

Пусть дана задача линейного программирования, которую назовем исходной или первой задачей:

Задача 1 (исходная). Найти

$$\max f(\bar{x}) = \sum_{j=1}^{n} c_j x_j + c_0$$

при ограничениях

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, \quad i = \overline{1,m};$$

$$x_j \ge 0, \quad j = \overline{1,n}.$$

Сформулируем другую задачу, которую назовем двойственной по отношению к исходной или задачей два:

Задача 2 (двойственная). Найти

$$\min \varphi(\overline{y}) = \sum_{i=1}^{m} b_i y_i + c_0$$

при ограничениях

$$\sum_{i=1}^{m} a_{ij} y_i \geq c_j\,, \quad j = \overline{1,n};$$

$$y_i \geq 0\,, \quad i = \overline{1,m}.$$

В матричном виде эти задачи можно записать так.

Задача 1. Найти $\max f(\overline{x}) = \langle \overline{c}, \overline{x} \rangle + c_0$ при ограничениях

$$A\overline{x} \leq \overline{b}; \quad \overline{x} \geq 0\,.$$

Задача 2. Найти $\min \varphi(\overline{y}) = \langle \overline{b}, \overline{y} \rangle + c_0$ при ограничениях

$$A^{\text{т}} \overline{y} \geq \overline{c};$$

$$\overline{y} \geq 0.$$

Итак, при переходе от исходной задачи к соответствующей двойственной задаче производят следующие преобразования:

1) заменяют максимизацию целевой функции минимизацией;

2) смысл неравенств – ограничений меняется на противоположный. Если в исходной задаче неравенство меньше или равно нулю, то в двойственной задаче – больше или равно нулю;

3) число переменных x в исходной задаче равно числу ограничений в двойственной, и наоборот;

4) матрица A системы ограничений исходной задачи транспонируется;

5) векторы $\overline{b}$ и $\overline{c}$ меняются местами, т.е. коэффициенты целевой функции исходной задачи $(\overline{c})$ становятся столбцом свободных членов системы ограничений двойственной задачи, а столбец свободных членов системы ограничений исходной задачи $(\overline{b})$ становится коэффициентами целевой функции двойственной задачи.

Пример. *Исходная задача:*
найти max $f(\overline{x}) = 12 + 8x_1 + 36x_2 + 4x_3 + 12x_4$ при ограничениях

$$\begin{cases} 88x_1 - 20x_2 + 32x_3 + 14x_4 \le 12; \\ -16x_1 + 4x_2 + 48x_3 + 4x_4 \le 10; \\ -24x_1 - 10x_2 + 6x_3 + 8x_4 \le 44; \\ x_i \ge 0, \ i = \overline{1,4}. \end{cases}$$

Двойственная задача:
найти $\min \varphi(\overline{y}) = 12 + 12y_1 + 10y_2 + 44y_3$ при ограничениях

$$\begin{cases} 88y_1 - 16y_2 - 24y_3 \ge 8; \\ -20y_1 + 4y_2 - 10y_3 \ge 36; \\ 32y_1 + 48y_2 + 6y_3 \ge 4; \\ 14y_1 + 4y_2 + 8y_3 \ge 12; \\ y_j \ge 0, \ j = \overline{1,3}. \end{cases}$$

Легко показать, что если за исходную принять задачу два и применить к ней аналогичные преобразования, то получим задачу 1. Поэтому рассмотренные задачи называются взаимно двойственными.

Взаимозависимость исходной и двойственной задач определяется рядом теорем, которые мы рассмотрим без доказательств.

Лемма. Если $\overline{x}^0$ – произвольное допустимое решение задачи 1, а $\overline{y}^0$ – произвольное допустимое решение задачи 2, то справедливо соотношение $f(\overline{x}^0) \le \varphi(\overline{y}^0)$. Если для $\overline{x}^0$ и $\overline{y}^0$ значения целевых функций совпадают, т.е. $f(\overline{x}^0) = \varphi(\overline{y}^0)$, то $\overline{x}^0$ – оптимальное решение задачи 1, а $\overline{y}^0$ – оптимальное решение задачи 2.

Доказательство. 1. Так как $\overline{x}^0$ является допустимым решением задачи 1, то имеет место система неравенств

$$\sum_{j=1}^{n} a_{ij} x_j^0 \le b_i, \quad i = \overline{1, m}.$$

Умножим i-е неравенство на i-ю компоненту вектора $\overline{y}^0$, т.е. на y_i^0 (так как $y_i^0 \geq 0$, то знак неравенства не изменится) и просуммируем по всем i.

В результате получим:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij} x_j^0 y_i^0 \leq \sum_{i=1}^{m} b_i y_i^0 \ .$$

Аналогично, для вектора $\overline{y}^0$ справедливо неравенство

$$\sum_{i=1}^{m} a_{ij} y_i^0 \geq c_j \ , \quad j = \overline{1, n} \, .$$

Умножим j-е неравенство на $x_j^0 \geq 0$ и просуммируем по всем j:

$$\sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij} y_i^0 x_j^0 \geq \sum_{j=1}^{n} c_j x_j^0 \ .$$

Объединяя полученные неравенства, получим

$$\sum_{j=1}^{n} c_j x_j^0 \leq \sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij} x_j^0 y_i^0 \leq \sum_{i=1}^{m} b_i y_i^0 \, .$$

Отсюда

$$\sum_{j=1}^{n} c_j x_j^0 + c_0 \leq \sum_{i=1}^{m} b_i y_i^0 + c_0 \ ,$$

т.е. $f(\overline{x}^0) \leq \varphi(\overline{y}^0)$.

2. Покажем, что если $f(\overline{x}^0) = \varphi(\overline{y}^0)$, то $\overline{x}^0$ и $\overline{y}^0$ – оптимальные решения.

а) Предположим противное, т.е. что существует такая точка $\overline{x}'$, что $f(\overline{x}') > f(\overline{x}^0)$, но в этом случае получим, что $f(\overline{x}') > \varphi(\overline{y}^0)$, что противоречит доказанному в пункте 1.

б) Аналогично, предположим, что найдется такая точка $\overline{y}'$, что $\varphi(\overline{y}') < \varphi(\overline{y}^0)$, но в этом случае получим $\varphi(\overline{y}') < f(\overline{x}^0)$, что противоречит доказанному условию.

Первая теорема двойственности. Если одна из задач (1-я или 2-я) имеет оптимальное решение, то и другая также имеет оптимальное решение, причем их оптимумы совпадают, т.е. $\max f(\bar{x}) = \min \varphi(\bar{y})$.

Если целевая функция одной из задач не ограничена, то другая задача не имеет допустимых решений.

Вторая теорема двойственности (теорема о дополняющей нежесткости). Для того чтобы допустимые векторы $\bar{x}^*$ и $\bar{y}^*$ были оптимальными, необходимо выполнение следующих условий:

$$y_i^*\left(\sum_{j=1}^{n} a_{ij} x_j^* - b_i\right) = 0, \quad i = \overline{1,m};$$

$$x_j^*\left(\sum_{i=1}^{m} a_{ij} y_i^* - c_j\right) = 0, \quad j = \overline{1,n}.$$

Эту теорему надо понимать так, что если оптимальное решение исходной задачи подставить в систему ограничений и если *i*-я строка обратится в строгое неравенство, то *i*-я компонента оптимального решения двойственной задачи равна нулю.

Или так, если *i*-я компонента оптимального решения двойственной задачи положительна, то *i*-е ограничение исходной задачи выполняется как строгое равенство.

Исходя из рассмотренных свойств взаимно двойственных задач можно сформулировать *правило решения двойственной задачи*. Оказывается, если решена исходная задача, то двойственную задачу решать не нужно. Оптимальным решением двойственной задачи являются коэффициенты при ослабляющих переменных в строке целевой функции в последней симплекс-таблице исходной задачи.

Отметим, что коэффициенты при остальных x_j в строке целевой функции последней симплекс-таблицы равны разности между левой и правой частями *j*-го ограничения двойственной задачи, т.е. величине $\sum_{i=1}^{m} a_{ij} y_i^* - c_j$.

1.5.2. Двойственный симплекс-алгоритм

В связи с постановкой двойственных задач и приведением их к стандартному виду возникает проблема решения линейных задач, когда оптимальный базис является недопустимым.

Например, если привести двойственную задачу последнего примера к стандартному виду, получим:

найти $\max\{-\varphi(\bar{y})) = -12 - (12y_1 + 10y_2 + 44y_3\}$ при ограничениях:

$$\begin{cases} -88y_1 + 16y_2 + 24y_3 + y_4 = -8; \\ 20y_1 - 4y_2 + 10y_3 + y_5 = -36; \\ -32y_1 - 48y_2 - 6y_3 + y_6 = -4; \\ -14y_1 - 4y_2 - 8y_3 + y_7 = -12. \end{cases}$$

Начальный базис для данной задачи имеет вид:

$$\begin{aligned} y_4 &= -8 - (-88y_1 + 16y_2 + 24y_3); \\ y_5 &= -36 - (20y_1 - 4y_2 + 10y_3); \\ y_6 &= -4 - (-32y_1 - 48y_2 - 6y_3); \\ y_7 &= -4 - (-14y_1 - 4y_2 - 8y_3). \end{aligned}$$

Этот базис будет недопустимым, так как $\beta_i < 0$, т.е. свободные члены в системе ограничений – отрицательны.

Подобные задачи могут возникать не только при решении двойственных задач, но и в самом общем случае.

Однако существует метод, позволяющий решать задачи с недопустимым базисом. Этот метод называется двойственным симплекс-алгоритмом.

Алгоритм двойственного симплекс-метода можно получить, проведя рассуждения, аналогичные выводу простого симплекс-метода. Все пункты рассматриваемого алгоритма такие же, как у простого симплекс-метода, но имеется *ряд отличий* (предполагается, что решается задача максимизации):

1) возможность применения обычного симплекс-метода определялась положительностью коэффициентов столбца свободных членов β_i в системе ограничений, разрешенной относительно базиса. Признаком допустимости в двойственном симплекс-алгоритме является неотрицательность коэффициентов целевой функции γ_j (не считая свободного члена);

2) признаком оптимальности полученного решения является неотрицательность всех β_i-коэффициентов столбца свободных членов (не считая значения целевой функции);

3) переменная x_i, выводимая из базиса, определяется максимальным по модулю отрицательным коэффициентом в столбце свободных членов;

4) разрешающий элемент выбирается в *i*-й строке среди отрицательных коэффициентов $\alpha_{ij} < 0$ и соответствует максимальному отношению коэффициентов строки целевой функции к элементам *i*-й строки, т.е. $\dfrac{\gamma_j}{\alpha_{ij}}$ (табл. 1.29).

Таблица 1.29

Базис	Свободные члены	x_1	…	x_i	…	x_r	x_{r+1}	…	x_j	…	x_n
x_1	β_1	1	…	0	…	0	α_{1r+1}	…	α_{1j}	…	α_{1n}
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
x_i	β_i	0	…	1	…	0	α_{ir+1}	…	α_{ij}	…	α_{in}
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
x_r	β_r	0	…	0	…	1	α_{rr+1}	…	α_{rj}	…	α_{rn}
$f(\bar{x})$	γ_0	0	…	0	…	0	γ_{r+1}	…	γ_j	…	γ_n

1.5.3. Примеры решения двойственных задач линейного программирования

Пример 1. *Исходная задача:* найти $\max\left\{f(\overline{x}) = 21x_1 + 11x_2\right\}$ при ограничениях:

$$\begin{cases} 7x_1 + 4x_2 \le 13; \\ x_1 \le 1; \\ 2x_1 + x_2 \le 3; \\ x_1, x_2 \ge 0. \end{cases}$$

Двойственная задача: найти $\min\left\{\varphi(\overline{y}) = 13y_1 + y_2 + 3y_3\right\}$ при ограничениях:

$$\begin{cases} 7y_1 + y_2 + 2y_3 \ge 21; \\ 4y_1 + y_3 \ge 11; \\ y_1, y_2, y_3 \ge 0. \end{cases}$$

Решение двойственной задачи имеет вид:

$$y_1 = 0, \quad y_2 = 0, \quad y_3 = 11;$$
$$\varphi_{\min}(\overline{y}) = 33.$$

Полученное решение находится из оптимальной таблицы исходной задачи

Пример 2. *Исходная задача:* найти $\max\left\{f(\overline{x}) = 3 + 4x_1 + 6x_2\right\}$ при ограничениях:

$$\begin{cases} 2x_1 - 3x_2 \le 0; \\ -2x_1 + 2x_2 \le 4; \\ -4x_1 + 5x_2 \le 20; \\ x_1, x_2 \ge 0. \end{cases}$$

Двойственная задача: найти $\min\left\{\varphi(\overline{y}) = 3 + 4y_2 + 20y_3\right\}$ при ограничениях:

$$\begin{cases} 2y_1 - 2y_2 - 4y_3 \ge 4; \\ -3y_1 + 2y_2 + 5y_3 \ge 6; \\ y_1, y_2, y_3 \ge 0. \end{cases}$$

Так как в исходной задаче целевая функция неограниченно возрастает, двойственная задача также не имеет решения (причем ее допустимое множество пусто).

Пример 3. *Исходная задача:* найти

$$\max\left\{f(\overline{x}) = 2 + 4x_1 + 2x_2 + 2x_3\right\}$$

при ограничениях

$$\begin{cases} 2x_1 - 3x_2 + 15x_3 \le 3; \\ 2x_1 + 2x_2 + 8x_3 \le 32; \\ 2x_1 - 4x_2 + 16x_3 \le 2; \\ x_1, x_2, x_3 \ge 0. \end{cases}$$

Двойственная задача: найти

$$\min\left\{\varphi(\overline{y}) = 2 + 3y_1 + 32y_2 + 2y_3\right\}$$

при ограничениях:

$$\begin{cases} 2y_1 + 2y_2 + 2y_3 \ge 4; \\ 3y_1 + 2y_2 - 4y_3 \ge 2; \\ 15y_1 + 8y_2 + 16y_3 \ge 2; \\ y_1, y_2, y_3 \ge 0. \end{cases}$$

Решение двойственной задачи имеет вид:

$$y_1 = \frac{2}{5}, \quad y_2 = \frac{8}{5}, \quad y_3 = 0;$$

$$\varphi(\overline{y}) = 54\frac{2}{5}.$$

В рассматриваемом примере решением являются коэффициенты при ослабляющих переменных x_4, x_5, x_6 в строке целевой функции оптимальной таблицы исходной задачи.

Пример 4. Решить задачу на основе двойственного симплекс-алгоритма.

Исходная задача: найти

$$\max\left\{f(\overline{x}) = 12 + 8x_1 + 36x_2 + 4x_3 + 12x_4\right\}$$

при ограничениях:

$$\begin{cases} 88x_1 - 20x_2 + 32x_3 + 14x_4 \le 12; \\ -16x_1 + 4x_2 + 48x_3 + 4x_4 \le 10; \\ -24x_1 - 10x_2 + 6x_3 + 8x_4 \le 44; \\ x_1, x_2, x_3, x_4 \ge 0. \end{cases}$$

Двойственная задача: найти

$$\min \{\varphi(\overline{y}) = 12 + 12y_1 + 10y_2 + 44y_3\}$$

при ограничениях:

$$\begin{cases} 88y_1 - 16y_2 - 24y_3 \ge 8; \\ -20y_1 + 4y_2 - 10y_3 \ge 36; \\ 32y_1 + 48y_2 + 6y_3 \ge 4; \\ 14y_1 + 4y_2 + 8y_3 \ge 12; \\ y_1, y_2, y_3 \ge 0. \end{cases}$$

Добавим ослабляющие переменные y_4, y_5, y_6, y_7. В результате получим начальный базис:

$$\begin{cases} y_4 = -8 - (-88y_1 + 16y_2 + 24y_3); \\ y_5 = -36 - (20y_1 - 4y_2 + 10y_3); \\ y_6 = -4 - (-32y_1 - 48y_2 + 6y_3); \\ y_7 = -12 - (-14y_1 - 4y_2 - 8y_3). \end{cases}$$

Это недопустимый базис, так как свободные члены в уравнениях начального базиса отрицательные. Следовательно, необходимо использовать двойственный симплекс-алгоритм.

Будем искать $\max \{(-\varphi(\overline{y})) = -12 - (12y_1 + 10y_2 + 44y_3)\}$.

Соответствующая последовательность симплекс-таблиц имеет вид:

Базис	Свободные члены	y_1	y_2	y_3	y_4	y_5	y_6	y_7
y_4	–8	–88	16	24	1	0	0	0
y_5	–36	20	–4	10	0	1	0	0
y_6	–4	–32	–48	–6	0	0	1	0
y_7	–12	–14	–4	–8	0	0	0	1
$-\varphi$	–12	12	10	44	0	0	0	0

Базис	Свободные члены	y_1	y_2	y_3	y_4	y_5	y_6	y_7
y_4	–152	–8	0	64	1	4	0	0
y_2	9	–5	1	–5/2	0	–1/4	0	0
y_6	428	–272	0	–126	0	–12	1	0
y_7	24	–34	0	–18	0	–1	0	1
$-\varphi$	–102	62	0	69	0	2.5	0	0

Базис	Свободные члены	y_1	y_2	y_3	y_4	y_5	y_6	y_7
y_1	19	1	0	–8	–1/8	–1/2	0	0
y_2	104	0	1	–85/2	–5/8	–11/4	0	0
y_6	5596	0	0	–2302	–34	–148	1	0
y_7	670	0	0	–290	–17/4	–18	0	1
$-\varphi$	–1280	0	0	565	7.75	67/2	0	0

Как следует из последней симплекс-таблицы, оптимальным решением будут:

$$y_1 = 19,\ y_2 = 104,\ y_3 = 0;$$
$$\varphi_{\min}(\bar{y}) = 1280.$$

Пример 5. Решить задачу с использованием двойственного симплекс-алгоритма.

Пусть после приведения некоторой задачи к стандартному виду вместе с ограничениями и замены требования минимизации целевой функции на максимизацию мы пришли к следующей задаче:

найти $\max\left\{-\varphi(\overline{y}) = -8y_1 - 12y_2 - 18y_3\right\}$ при ограничениях:

$$\begin{cases} y_1 + 3y_2 \geq 2; \\ y_1 + 3y_2 \geq 1; \\ y_1, y_2, y_3 \geq 0. \end{cases}$$

Запишем начальный базис:

$$\begin{cases} y_4 = -2 - (y_1 + 3y_2); \\ y_5 = -1 - (y_1 - 3y_2). \end{cases}$$

Отрицательные свободные члены в начальном базисе преобразованной задачи указывают на необходимость применения двойственного симплекс-алгоритм, при этом

$$-\varphi(\overline{y}) = 0 - (8y_1 + 12y_2 + 18y_3).$$

Соответствующая симплекс-таблица имеет вид:

Базис	Свободные члены	y_1	y_2	y_3	y_4	y_5
y_1	2	–1	0	–3	1	0
y_5	–1	–1	–3	0	0	1
$-\varphi$	0	8	12	18	0	0

Как следует из таблицы,

$$\max\left\{\frac{\gamma_k}{\alpha_{ik}}\right\} = \max\left\{-\frac{8}{1}, -\frac{18}{3}\right\} = \max\{-8, -6\} = -6.$$

Соответственно, преобразованная таблица имеет вид:

Базис	Свободные члены	y_1	y_2	y_3	y_4	y_5
y_3	2/3	1/3	0	1	–1/3	0
y_5	–1	–1	–3	0	0	1
$-\varphi$	–12	2	12	0	6	0

Аналогично, из полученной таблицы следует:

$$\max\{-2, -4\} = -2.$$

Тогда, преобразованная таблица представится в виде:

Базис	Свободные члены	y_1	y_2	y_3	y_4	y_5
y_3	1/3	0	–1	1	–1/3	1/3
y_1	1	1	3	0	0	–1
$-\varphi$	–14	0	6	0	6	2

В результате, из последней таблицы следует, что оптимальным решением задачи будут:

$$y_1 = 1, y_2 = 0, y_3 = 1/3;$$
$$\varphi_{\min} = 14.$$

1.6. Транспортная задача линейного программирования

Ранее был рассмотрен симплекс-метод, который является методом решения задач линейного программирования общего вида. Симплекс-метод является конечным, а не итерационным, тем не менее, число его итераций может быть весьма большим. Многие классы широко распространенных на практике задач линейного программирования обладают особенностями, выражающимися в специфическом строении матрицы A коэффициентов системы ограничений, которые позволяют существенно упростить общий метод решения применительно к рассматриваемой в данном разделе задаче. При этом процесс получения оптимального решения ускоряется. В других случаях удается построить принципиально новый метод решения, приспособленный к определенному классу задач.

Одним из примеров задач со специальной структурой матрицы ограничений является транспортная задача.

Мы рассмотрим постановку простейшей транспортной задачи и один из наиболее популярных методов ее решения, принципиально отличающийся от симплекс-метода.

1.6.1. Постановка задачи

Пусть на базах $A_1, A_2, ..., A_m$ сосредоточены определенные ресурсы в количестве $a_1, a_2, ..., a_m$. Эти ресурсы должны быть доставлены на объекты $B_1, B_2, ..., B_n$ в количестве $b_1, b_2, ..., b_n$. Стои-

мость перевозки единицы груза из A_i в B_j обозначим через c_{ij}. Через x_{ij} обозначим количество ресурсов, которое доставляются из A_i в B_j.

Задача. Найти матрицу перевозок $\|x_{ij}\|$, $i=\overline{1,m}$, $j=\overline{1,n}$, которая минимизирует целевую функцию $f(\overline{x})=\sum_{i=1}^{m}\sum_{j=1}^{n}c_{ij}x_{ij}$, удовлетворяющую ограничениям

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

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

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

Первое и второе ограничения означают, что суммарное количество груза, вывозимого с базы A_i, точно равно имеющимся там запасам a_i, а ввозимый груз на базу B_j точно равен заявкам этой базы.

Последнее условие называется *балансным условием* и означает равенство всего количества имеющихся запасов и потребностей.

Поставленная транспортная задача – это задача линейного программирования, но записанная особым образом. Для её решения разработаны специальные методы более эффективные, чем симплекс-метод.

В представленной задаче $m\times n$ неизвестных.

Введем ряд понятий.

Матрица $\|x_{ij}\|$ размерности $m\times n$ называется *планом перевозок*, x_{ij} *– перевозкой*, $\|c_{ij}\|$ *– матрицей издержек*.

План называется *допустимым*, если он удовлетворяет указанным выше ограничениям.

План называется *оптимальным*, если он минимизирует функцию $f(\overline{x})$.

Переменные x_{ij} удовлетворяют $m+n$ уравнениям системы ограничений, ранг которой r равен $m+n-1$. Если расположить переменные x_{ij} следующим образом:

$$x_{11}, x_{12}, ..., x_{1n}, x_{21}, ..., x_{2n}, ..., x_{m1}, ..., x_{mn},$$

то матрица ограничений транспортной задачи примет вид

$$A = \left| \begin{array}{cccc} \overbrace{1\ 1\ \dots\ 1}^{n} & \overbrace{0\ 0\ \dots\ 0}^{n} & \dots & \overbrace{0\ 0\ \dots\ 0}^{n} \\ 0\ 0\ \dots\ 0 & 1\ 1\ \dots\ 1 & \dots & 0\ 0\ \dots\ 0 \\ \dots\dots\dots & \dots\dots\dots & \dots & \dots\dots\dots \\ 0\ 0\ \dots\ 0 & 0\ 0\ \dots\ 0 & \dots & 1\ 1\ \dots\ 1 \\ 1\ 0\ \dots\ 0 & 1\ 0\ \dots\ 0 & \dots & 1\ 0\ \dots\ 0 \\ 0\ 1\ \dots\ 0 & 0\ 1\ \dots\ 0 & \dots & 0\ 1\ \dots\ 0 \\ \dots\dots\dots & \dots\dots\dots & \dots & \dots\dots\dots \\ 0\ 0\ \dots\ 1 & 0\ 0\ \dots\ 1 & \dots & 0\ 0\ \dots\ 1 \end{array} \right| \begin{array}{l} \left.\begin{array}{l} \\ \\ \\ \\ \end{array}\right\} m \\ \left.\begin{array}{l} \\ \\ \\ \\ \end{array}\right\} n \end{array}.$$

Нетрудно видеть, что ранг r матрицы A равен $m+n-1$. Действительно, всего в матрице $m+n$ строк и все строки линейно зависимы. Чтобы в этом убедиться, достаточно сложить первые m строк и вычесть сумму последних n, при этом получится нулевой вектор. С другой стороны, нетрудно заметить, что любые $m+n-1$ строк матрицы A линейно независимы.

Итак, ранг системы ограничений равен $m+n-1$. Следовательно, можно выделить $m+n-1$ базисных переменных.

Существует понятие *опорный план* – это план, в котором отличны от нуля не более чем $m+n-1$ переменных x_{ij}.

Опорный план это фактически базис.

1.6.2. Нахождение первого опорного плана

Как и в задаче линейного программирования, в транспортной задаче необходимо сначала найти первый допустимый базис. Для

этого существует несколько методов, на некоторых из них остановимся ниже.

Зададим транспортную задачу табл. 1.30.

Таблица 1.30

A	B				
	B_1	B_2	…	B_n	Запасы a_i
A_1	c_{11} x_{11}	c_{12} x_{12}	…	c_{1n} x_{1n}	a_1
A_2	c_{21} x_{21}	c_{22} x_{22}	…	c_{2n} x_{2n}	a_2
…	…	…	…	…	…
A_m	c_{m1} x_{m1}	c_{m2} x_{m2}	…	c_{mn} x_{mn}	a_m
Заявки b_j	b_1	b_2	…	b_n	$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$

В табл. 1.30 A – пункт отправления, B – пункт назначения.

Диагональный метод нахождения опорного плана состоит в следующем. В верхнюю левую клетку таблицы ставится минимальное из чисел a_1 и b_1, т.е. $\min\{a_1, b_1\}$.

Пусть $a_1 < b_1$, Следовательно, поставщик A_1 полностью использовал свои запасы и при установлении остальных перевозок его можно не учитывать (строка, соответствующая a_1, из таблицы вычеркивается). Теперь потребность 1-го потребителя будет составлять $b_1 - a_1$.

Если наоборот, $a_1 > b_1$, то соответствующий столбец из таблицы вычеркивается.

Далее процесс повторяется.

Пример 1. Рассмотрим табл. 1.31. В этом примере $m = 3, n = 4$.

В результате получили $m + n - 1 = 6$ отличных от нуля переменных, т.е. это опорный план, в соответствии с которым

$$f(\overline{x}) = \sum_{i=1}^{3} \sum_{j=1}^{4} c_{ij} x_{ij} = 1630 .$$

Таблица 1.31

A	B				
	B_1	B_2	B_3	B_4	Запасы a_i
A_1	5 60	7 20	1	3	80 / 20
A_2	4	8 20	4 70	2	90 / 70
A_3	1	5	6 20	9 70	90
Заявки b_j	60	40 / 20	90 / 20	70	260

Можно показать, что диагональный метод всегда приводит к опорному допустимому решению. Однако следует иметь в виду, что поскольку данный метод не учитывает стоимость перевозок, то опорные планы, полученные с его помощью, значительно отличаются от оптимальных.

Метод минимального элемента дает возможность учесть заданные стоимости. От диагонального метода он отличается только тем, что в таблице выбирается не верхняя левая клетка, а клетка с минимальной стоимостью c_{ij}, и как в диагональном методе в эту клетку записывается минимальное из двух чисел a_1 и b_1, т.е. $\min\{a_1, b_1\}$. Затем соответствующий столбец или строка из таблицы вычеркивается и т.д.

Пример 2. Рассмотрим табл. 1.32.

Таблица 1.32

A	B				
	B_1	B_2	B_3	B_4	Запасы a_i
A_1	5	7	1 80	3	80
A_2	4	8 10	4 10	2 70	90 / 20 / 10
A_3	1 60	5 30	6	9	90 / 30
Заявки b_j	60	40 / 10	90 / 10	70	260

В соответствии с опорным планом, полученным данным методом, значение функции

$$f(\overline{x}) = \sum_{i=1}^{3}\sum_{j=1}^{4} c_{ij}x_{ij} = 550.$$

Опыт решения транспортных задач показал, что далеко не всегда использование метода минимального элемента, вместо диагонального, дает лучшее приближение к оптимальному решению.

1.6.3. Метод потенциалов

Это основной метод решения транспортной задачи. Метод базируется на ряде теорем, которые рассмотрим ниже.

Введем понятие *псевдостоимости*.

Пусть за перевозку единицы груза с базы A_i вносится плата α_i, а за перевозку единицы груза на базу B_j – плата β_j. *Псевдостоимостью* перевозки из A_i в B_j называется сумма $\widetilde{c}_{ij} = \alpha_i + \beta_j$.

Совокупность величин (α_i, β_j), $i = \overline{1,m}$, $j = \overline{1,n}$, называется *платежами*, или *потенциалами*.

Теорема о платежах. Для заданной совокупности платежей суммарная псевдостоимость перевозки при любом допустимом плане x_{ij} сохраняет одно и то же значение, т. е.

$$\widetilde{f}(\overline{x}) = \sum_{i=1}^{m}\sum_{j=1}^{n} \widetilde{c}_{ij}x_{ij} = \text{const}.$$

Доказательство.

$$\widetilde{f}(\overline{x}) = \sum_{i=1}^{m}\sum_{j=1}^{n} \widetilde{c}_{ij}x_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{n} (\alpha_i + \beta_j)x_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_i x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n} \beta_j x_{ij} =$$

$$= \sum_{i=1}^{m} \alpha_i \sum_{j=1}^{n} x_{ij} + \sum_{j=1}^{n} \beta_j \sum_{i=1}^{m} x_{ij} = \sum_{i=1}^{m} \alpha_i a_i + \sum_{j=1}^{n} \beta_j b_j = \text{const}.$$

Так как план – допустимый, то выполняются соотношения:

$$\sum_{i=1}^{m} x_{ij} = a_i,$$

$$\sum_{j=1}^{n} x_{ij} = b_j,$$

при этом суммарная псевдостоимость перевозки не зависит от плана x_{ij}, т.е. $\widetilde{f}(\overline{x}) = \text{const}$. Таким образом, теорема доказана.

Теорема об оптимальности. Если для всех базисных клеток $(x_{ij} > 0)$ псевдостоимость равна стоимости, т.е. $\widetilde{c}_{ij} = c_{ij}$, а для свободных клеток $(x_{ij} = 0)$ выполняется неравенство $\widetilde{c}_{ij} \le c_{ij}$, то план является оптимальным и никаким способом не может быть улучшен.

Доказательство. Для базисных клеток имеем

$$f(\overline{x}) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{n} \widetilde{c}_{ij}x_{ij} = \widetilde{f}(\overline{x}) = \text{const}.$$

Заменим план x_{ij} на любой другой допустимый план x'_{ij}. Тогда получим другое значение целевой функции

$$f(\overline{x}') = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x'_{ij}.$$

В базисных клетках, которые сохранились от старого плана, стоимость $c_{ij} = \widetilde{c}_{ij}$, а в базисных клетках нового плана, которые в плане x_{ij} были свободными, стоимость $c_{ij} \ge \widetilde{c}_{ij}$. Следовательно,

$$f(\overline{x}') \ge \sum_{i=1}^{m}\sum_{j=1}^{n} \widetilde{c}_{ij}x'_{ij} = \text{const} = \widetilde{f}(\overline{x}'),$$

т.е. $f(\overline{x}') \ge \widetilde{f}(\overline{x}')$.

Таким образом, при любом плане, отличном от x_{ij}, функция $f(\overline{x}')$ может быть только больше. Значит, x_{ij} – оптимальный план.

Теорема доказана.

Из условия равенства $c_{ij} = \widetilde{c}_{ij}$ в базисных клетках можно определить потенциалы α_i и β_j. В опорном плане $(n+m-1)$ базисных клеток; причем значения c_{ij} известны. Для нахождения потенциалов составим систему уравнений, используя табл. 1.32:

$$\alpha_1 + \beta_3 = 1;$$
$$\alpha_2 + \beta_3 = 4;$$
$$\alpha_2 + \beta_2 = 8;$$
$$\alpha_3 + \beta_2 = 5;$$
$$\alpha_3 + \beta_1 = 1;$$
$$\alpha_2 + \beta_4 = 2.$$

Система имеет $(n+m-1)$ уравнений и $(n+m)$ неизвестных потенциалов. Для получения решения необходимо произвольно выбрать один из потенциалов, например $\alpha_1 = 1$. Остальные потенциалы однозначно определяются при решении системы уравнений: $\beta_3 = 0$, $\alpha_2 = 4$, $\beta_2 = 4$, $\alpha_3 = 1$, $\beta_1 = 0$, $\beta_4 = -2$.

Следует отметить, что в системе потенциалов могут быть и отрицательные элементы.

Определив потенциалы, находят псевдостоимости $\widetilde{c}_{ij} = \alpha_i + \beta_j$ в остальных небазисных клетках и проставляют их в левом углу небазисных клеток (табл. 1.33).

Таблица 1.33

A	B				
	B_1	B_2	B_3	B_4	α_i
A_1	1 < 5	6 < 7	1 80	–1 < 3	1
A_2	4 ≤ 4	8 10	4 10	2 70	4
A_3	1 60	5 30	1 < 6	–1 < 9	1
β_j	0	4	0	–2	

Далее проверяют, является ли опорный план оптимальным.

В рассматриваемом случае план является оптимальным, так как во всех небазисных клетках $\widetilde{c}_{ij} \le c_{ij}$.

Возьмем другую таблицу, построенную диагональным методом (табл. 1.34).

Таблица 1.34

A	B				
	B_1	B_2	B_3	B_4	α_i
A_1	5 60	7 20	3 > 1	6 > 3	1
A_2	6 > 4	8 20	4 70	7 > 2	2
A_3	8 > 1	10 > 5	6 20	9 70	4
β_j	4	6	2	5	

Опорный план не является оптимальным, так как имеются клетки, где $\widetilde{c}_{ij} > c_{ij}$.

Выбирается клетка с максимальной разностью ($\widetilde{c}_{31} - c_{31} = 7$). Эта клетка берется за начальную для построения, так называемого, цикла пересчета, с помощью которого можно найти новый опорный план с меньшим значением целевой функции.

Циклом пересчета для данной свободной клетки называется замкнутая линия, все вершины которой, за исключением данной, лежат в базисных клетках. Вершинам цикла пересчета присваиваются знаки «+» или «–». Знаки последовательных вершин чередуются, причем свободная клетка имеет положительный знак. Доказано, что для каждой свободной клетки существует и притом единственный цикл пересчета.

Ценой цикла называется приращение стоимости перевозок при перемещении единицы груза по циклу, она равна алгебраической сумме стоимостей, стоящих в вершинах цикла, а, в конечном счете, равна:

$$\gamma_{ij} = c_{ij} - \widetilde{c}_{ij}.$$

Цена может быть как положительной, так и отрицательной. Очевидно, $\gamma_{ij} < 0$ при $\widetilde{c}_{ij} > c_{ij}$ и в этом случае стоимость перевозок уменьшается.

Некоторые циклы пересчета представлены в табл. 1.35.

Таблица 1.35

A	B					
	B_1	B_2	B_3	B_4	B_5	B_6
A_1	c_{11}	$-\ c_{12}$	c_{13}	c_{14}	$+\ +\ c_{15}$	$-\ c_{16}$
A_2	c_{21}	$+\ c_{22}$	$-\ c_{23}$	c_{24}	c_{25}	c_{26}
A_3	$+\ c_{31}$	$-\ c_{32}$	$+\ c_{33}$	c_{34}	$-\ c_{35}$	c_{36}
A_4	c_{41}	c_{42}	$-\ c_{43}$	c_{44}	c_{45}	$+\ c_{45}$
A_5	$-\ c_{51}$	$+\ c_{52}$	$+\ c_{53}$	c_{54}	$-\ c_{55}$	c_{56}

Покажем справедливость того, что цена $\gamma_{ij} = c_{ij} - \widetilde{c}_{ij}$.

В соответствии с табл. 1.35 цена цикла для клетки (5, 3) определяется выражением:

$$\gamma_{53} = c_{53} - c_{43} + c_{46} - c_{16} + c_{15} - c_{55} =$$

$$= c_{53} - (\alpha_4 + \beta_3) + (\alpha_4 + \beta_6) - (\alpha_1 + \beta_6) + (\alpha_1 + \beta_5) - (\alpha_5 + \beta_5) =$$

$$= c_{53} - (\beta_3 + \alpha_5) = c_{53} - \widetilde{c}_{53},$$

что соответствует указанному выше утверждению.

Рассмотрим пример, представленный табл. 1.36, элементы которой соответствуют элементам табл. 1.34.

Выберем клетку с максимальной разностью $\widetilde{c}_{ij} - c_{ij}$, т.е. клетку (3, 1), и построим для нее цикл пересчета. Прямая, выходящая из клетки, совершает повороты в базисных клетках и возвращается в начальную.

Теперь нужно перераспределить базисные переменные, находящиеся в вершинах цикла, так, чтобы в свободной клетке появилось положительное значение, а одна из базисных переменных приняла нулевое значение. При этом должно сохраниться требование, что все $x_{ij} \geq 0$. Доказано, что при таком перемещении будет получено

новое решение транспортной задачи с уменьшенным значением целевой функции, если цена цикла $\gamma_{ij} < 0$.

Таблица 1.36

A	B			
	B_1	B_2	B_3	B_4
A_1	5 – 60	7 + 20	1 3	3 6
A_2	4 6	8 – 20	4 + 70	2 7
A_3	1 + 8	5 10	6 – 20	9 70

$$f(\bar{x}) = 1630 .$$

В примере наименьшее значение базисной переменной, которую можно переместить по циклу, равно 20. Таким образом, это число нужно прибавить к базисным переменным вершин цикла, имеющих знак «+», и вычесть из переменных в вершинах цикла со знаком «–». В результате, в клетке (3, 1) появится переменная $x_{31} = 20$. Значение переменной более 20 выбрать нельзя, так как это приведет к появлению отрицательных значений x_{ij}.

После перемещения по циклу получим табл. 1.37.

Таблица 1.37

A	B				
	B_1	B_2	B_3	B_4	a_i
A_1	40	40			80
A_2		0 (ε)	90		90
A_3	20		0	70	90
b_j	60	40	90	70	

$$f(\bar{x}) = 1490 .$$

Новый опорный план соответствует меньшему значению целевой функции, причем уменьшение составляет величину $\gamma_{ij} x_{ij}$ $(7 \cdot 20 = 140)$, где x_{ij} – перемещенное по циклу число.

Следует отметить одну особенность полученной таблицы: вместо одной переменной нулю стали равны две переменные x_{22}, x_{33}. Это означает, что задача вырожденная, так как число базисных переменных меньше $(n+m-1)$. Для устранения вырожденности предположим, что в одной из клеток, например в клетке (A_2, B_2), имеется бесконечно малая величина $\varepsilon > 0$, которую в окончательном плане примем равной нулю.

Подводя итоги, сформулируем *алгоритм решения транспортной задачи.*

1. В качестве первого приближения берется допустимый план, полученный одним из методов нахождения начального опорного плана.

2. Для данного плана определяется система платежей исходя из условия, что в любой базисной клетке стоимость равна псевдостоимости.

3. Подсчитывается псевдостоимость для всех свободных клеток.

Если окажется, что для всех клеток плана $\tilde{c}_{ij} \le c_{ij}$, то задача решена, найденный план оптимален.

В противном случае, т.е. если существуют клетки, в которых $\tilde{c}_{ij} > c_{ij}$, переходим к шагу 4.

4. Для свободной клетки с максимальной разностью $\tilde{c}_{ij} - c_{ij} > 0$ строится цикл пересчета, после чего составляется новый опорный план и осуществляется переход на шаг 2.

Отметим, что логические операции, составляющие содержание процедуры поиска цикла, гораздо более экономны в смысле затрат времени ЭВМ по сравнению с арифметическими операциями симплекс-метода.

В заключение, обратим внимание на любопытную *особенность транспортной задачи*: если все числа a_i, b_j $\left(i = \overline{1,m},\ j = \overline{1,n}\right)$ – целые, то любой опорный план транспортной задачи (в том числе оптимальный) – целочисленный.

Действительно, когда все числа a_i, b_j – целые, начальный опорный план, полученный диагональным или другим методом, является целочисленным. Кроме того, на каждой итерации метода потенциалов перемещаемое число x_{ij} является целым, поэтому любой следующий опорный план – целочисленный.

1.6.4. Транспортные задачи с неправильным балансом

Ранее был рассмотрен метод решения *простейшей транспортной задачи* в случае выполнения условия $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$, т.е. когда суммарный объём возможных поставок равнялся суммарному объёму потребностей (*закрытая* транспортная задача). Между тем чаще возникают транспортные задачи, в которых условие баланса нарушено в ту или иную сторону (*открытые* транспортные задачи).

Так в случае, когда $\sum_{i=1}^{m} a_i > \sum_{j=1}^{n} b_j$ задачу можно сформулировать следующим образом:

найти

$$\min\left\{ f(\overline{x}) = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \right\}$$

при ограничениях:

$$\sum_{j=1}^{n} x_{ij} \le a_i, i = \overline{1, m};$$

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

Это означает, что у некоторых поставщиков остаётся неотправленная продукция, суммарный объём которой равен $\left(\sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j \right)$.

- Если безразлично, у каких именно поставщиков и в каких количествах останется продукция, то задача с неправильным балансом может быть сведена к эквивалентной ей задаче с правильным балансом, т.е. к закрытой транспортной задаче. Для этого вводится еще один фиктивный потребитель с заявкой

$$b_{n+1} = \sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j$$

и стоимостью перевозок от каждого поставщика

$$c_{i\,n+1} = 0, \quad i = \overline{1, m}.$$

Тогда, поставленная задача может быть записана в виде:
найти

$$\min f(\overline{x}) = \sum_{i=1}^{m} \sum_{j=1}^{n+1} c_{ij} x_{ij}$$

при ограничениях:

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

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

Полученная задача действительно эквивалентна исходной, так как фиктивные перевозки не влияют на значение целевой функции, поскольку $c_{i\,n+1} = 0$.

Совершенно аналогично можно поступить в случае $\sum_{i=1}^{m} a_i < \sum_{j=1}^{n} b_j$, т.е. когда продукции недостаточно для полного удовлетворении потребностей. В этом случае вводится в рассмотрение фиктивный поставщик с объемом возможных поставок, равным недостатку продукции $\left(\sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i \right)$. Стоимости перевозок от

фиктивного поставщика ко всем потребителям принимаем также равными нулю, т.е. $c_{m+1\,j} = 0, \quad j = \overline{1, n}.$

- Если не безразлично у какого поставщика и сколько останется продукции (или какой потребитель и сколько недополучит), то для сведения задачи к закрытой транспортной задаче надо прибегать к более сложным приёмам.

Пусть, например, суммарный объём продукции недостаточен; т.е. $\sum_{i=1}^{m} a_i < \sum_{j=1}^{n} b_j$. И пусть убыток, к которому приводит недополучение продукции *j*-м потребителем, оценен в r_j единиц тех же, в которых измерены c_{ij}. Тогда естественно, данную задачу математически можно записать так:

найти

$$\min f(\overline{x}) = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} + \sum_{j=1}^{n} r_j y_j \ . \tag{1.8}$$

В соотношение (1.8)

$$y_j = b_j - \sum_{i=1}^{m} x_{ij}$$

представляет собой объём неудовлетворенного спроса *j*-го потребителя при ограничениях:

$$\sum_{j=1}^{n} x_{ij} = a_i, \quad i = \overline{1, m}; \tag{1.9}$$

$$\sum_{i=1}^{m} x_{ij} + y_j = b_j, \quad j = \overline{1, n}. \tag{1.10}$$

Эта задача принимает вид обычной закрытой транспортной задачи, если ввести фиктивного $(m+1)$-го поставщика с объёмом поставок

$$a_{m+1} = \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i$$

и принять

$$c_{m+1}\, j = r_j, \quad \text{а} \quad x_{m+1\,j} = y_j.$$

Недостающее ограничение $\sum_{j=1}^{n} x_{m+1\,j} = a_{m+1}$ получается как сумма по j ограничений (1.10).

$$\sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} + \sum_{j=1}^{n} y_j = \sum_{j=1}^{n} b_j$$

или

$$\sum_{i=1}^{m} a_i + \sum_{j=1}^{n} x_{m+1\,j} = \sum_{j=1}^{n} b_j .$$

Откуда

$$\sum_{j=1}^{n} x_{m+1\,j} = \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i = a_{m+1} .$$

Таким образом, получаем задачу:

найти

$$\min f(\overline{x}) = \sum_{i=1}^{m+1} \sum_{j=1}^{n} c_{ij} x_{ij}$$

при ограничениях:

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

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

В заключение следует отметить, что:

метод потенциалов является не единственным методом решения простейшей транспортной задачи;

к аналогичному виду математической модели может привести задача, по своему содержанию никак не связанная с транспортом и с планированием перевозок.

Замечания. Многие практические задачи, связанные с планированием перевозок, не укладываются в рамки рассмотренной задачи, так как осложнены теми или иными дополнительными ограничениями.

1. Например, транспортные магистрали, связывающие источники продукции с потребителями, имеют *ограниченные* (в ограниченный промежуток времени) *пропускные способности*, что может не позволить реализовать оптимальный план

перевозок, найденный без их учета. Поэтому ограничения, связанные с пропускными способностями, необходимо включать в задачу.

Так вместо условия $x_{ij} \geq 0$ нужно использовать ограничение $\left(0 \leq x_{ij} \leq d_{ij}\right)$, где d_{ij} максимальный объем продукции, который может быть перевезен по этой магистрали (из i в j) за рассматриваемый промежуток времени (т.е. пропускная способность d_{ij} может быть равна бесконечности).

Ясно, что задача принципиально отличается от простейшей транспортной задачи, для её решения имеется метод являющийся видоизменением метода потенциалов.

2. Задача может быть осложнена наличием *промежуточных транспортных узлов*, в которых производится обработка груза (например, перевалка на другой вид транспорта). Для таких задач также имеются более или менее эффективные методы решения.

3. Принципиально меняется характер модели задачи даже в простейшем случае, если за критерий оптимальности выбрано время, необходимое для обеспечения всех потребителей нужным объёмом продукции, и требуется это время минимизировать – *транспортная задача по критерию времени*. Математическая модель этой задачи оказывается нелинейной за счет нелинейной целевой функции. Методы решения такой задачи известны.

4. Часто требуется составить оптимальный план перевозок не одного, а нескольких видов продукции. В постановке задачи появляется объём перевозок x_{ijk} k-го вида продукции от i-го поставщика к j-му потребителю. В остальном формулировка задачи аналогична простейшей. Получается, так называемая, *трёхиндексная* транспортная задача.

Бывают задачи и с большим количеством индексов.

Например, x_{ijkl} – объем перевозок k-го вида продукции l-м видом транспорта от i-го поставщика j-му потребителю, $k = \overline{1, p}$, $l = \overline{1, q}$.

Следовательно, существует класс *многоиндексных транспортных задач*. Можно отметить, что:

простых методов решения таких задач, подобных методу потенциалов, нет и быть не может;

при их решении надо использовать симплекс-метод, как для обычной задачи линейного программирования, но при этом могут возникать проблемы, связанные с большой размерностью задачи при больших значениях m, n, p, q и т.д.

В заключение следует отметить, что для многих задач транспортного типа имеются методы решения, основанные на математическом анализе свойств графов (или сетей). Сетевые методы решения подобных задач часто оказываются более эффективными, чем методы в рассмотренной нами матричной постановке.

1.7. Дискретное программирование

Дискретное программирование – особый класс задач линейного программирования.

1.7.1. Постановка задачи

Математическая модель общей задачи дискретного программирования может быть представлена следующим образом:

найти

$$\max\left\{ f(\overline{x}) = \sum_{j=1}^{n} c_j x_j + c_0 \right\}$$

при ограничениях:

$$\sum_{j=1}^{n} a_{ij} x_j \le (\ge) b_i, \quad i = \overline{1, m};$$

$$x_j \in A_j = \left\{ 0, A_{j1}, A_{j2}, ..., A_{jk_j} \right\}.$$

Итак, в этой задаче для каждой переменной x_j заранее определен набор значений, которые она может принимать.

Частным случаем задач дискретного программирования являются задачи *целочисленного программирования*:

найти

$$\max\left\{ f(\overline{x}) = \sum_{j=1}^{n} c_j x_j + c_0 \right\}$$

при ограничениях:

$$\sum_{j=1}^{n} a_{ij} x_j \le (\ge) b_j, \quad i = \overline{1, m};$$

$$x_j \ge 0, \quad j = \overline{1, n};$$

$$x_j \text{ – целые}, \quad j = \overline{1, n_1}.$$

Если $n_1 < n$, то задача называется *частично целочисленной*, если $n_1 = n$ – *полностью целочисленной*.

Частным случаем задач целочисленного программирования являются *задачи с булевыми переменными.*

Постановка задачи точно такая же, только на x накладывается ограничение

$$x_j = \begin{cases} 0, \\ 1, \end{cases} \qquad j = \overline{1, n}.$$

Как и в линейном программировании, вектор $\overline{x}$, удовлетворяющий всем ограничениям, называется *допустимым*, а допустимый вектор, оптимизирующий целевую функцию, – *оптимальным*.

Рассмотрим ряд классических задач целочисленного программирования.

1. Задача о ранце. Собравшийся в поход путешественник должен упаковать в ранец различные полезные предметы, всего n наименований, причем могут потребоваться несколько одинаковых предметов. Имеется m ограничений, например: на вес, на объем, на линейные размеры и т.д. – т.е. имеется m характеристик, по которым возможности ранца (или путешественника) ограничены.

Пусть:

a_{ij} – i-я характеристика предмета j-го наименования;

b_i – ограничивающая величина для i-й характеристики;

c_j – полезность одного j-го предмета;

x_j – количество предметов j-го наименования, запланированное к загрузке в ранец.

Тогда математическая формулировка задачи о ранце имеет вид:

найти

$$\max\left\{ f(\overline{x}) = \sum_{j=1}^{n} c_j x_j \right\}$$

при ограничениях:

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, \quad i = \overline{1, m};$$

$$x_j \ge 0, \quad x_j \text{ – целые}, \quad j = \overline{1, n}.$$

К задачам подобного вида приводится большое число практических задач, когда в качестве x_j фигурируют отдельные предметы (машины, кирпичи, люди и т.д.). В первоначальной постановке задачи о ранце рассматривалась загрузка бомбардировщика бомбовым запасом.

2. Задача о назначениях. Имеется *n* различных самолетов, которые требуется распределить между *n* авиалиниями. Известно, что на j-й авиалинии *i*-й самолет будет приносить доход c_{ij}. Требуется так распределить самолеты по линиям, чтобы максимизировать суммарный доход.

Положим

$$x_{ij} = \begin{cases} 1, & \text{если } i\text{-й самолет направлен на } j\text{-ю авиалинию,} \\ 0, & \text{в противном случае.} \end{cases}$$

Тогда математическая модель задачи может быть представлена в следующем виде:

найти

$$\max\left\{ f(\overline{x}) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \right\}$$

при ограничениях:

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

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

$$x_{ij} = \begin{cases} 1, \\ 0, \end{cases} \quad i = \overline{1,n}, \ j = \overline{1,n}.$$

Ограничения этой задачи отражают требования того, что каждый самолет назначается только на одну авиалинию и на каждую авиалинию назначается один самолет.

Очевидно, данная задача является задачей с булевыми переменными.

3. Задача о коммивояжере. Пусть имеются города, занумерованные числами 0, 1, 2, ..., *n*. Выехав из города 0, коммивояжер

должен объехать все остальные города, побывав в каждом из них по одному разу, и вернуться в исходный город. Известны расстояния c_{ij} между городами i и j ($i = 0, \ldots, n, j = 0, \ldots, n$).

Требуется найти самый короткий маршрут.

Введем следующие переменные:

$$x_{ij} = \begin{cases} 1, \text{если коммивояжер из города } i \text{ переезжает в город } j, \quad i \neq j; \\ 0, \text{ в противном случае.} \end{cases}$$

Таким образом, получаем следующую задачу:

найти

$$\min \left\{ f(\bar{x}) = \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij} \right\}$$

при ограничениях:

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

(в город j коммивояжер приезжает только один раз);

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

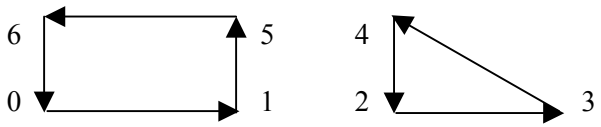
(из города i коммивояжер выезжает только один раз).

Для обеспечения *связности* маршрута введем дополнительные переменные u_i, u_j, которые удовлетворяют следующему неравенству:

$$u_i - u_j + n x_{ij} \leq n - 1; \quad i, j = \overline{1, n}; \quad x_{ij} = (0 \cup 1).$$

Вспомогательные переменные u_i, u_j, участвующие только в ограничениях, совершенно необходимы.

Например, маршрут, изображенный на приведенном ниже рисунке, удовлетворяет первым двум ограничениям, но не подходит под условия задачи о коммивояжере.



1.7.2. О решении задач линейного целочисленного программирования

На первых порах требование целочисленности переменных явилось существенным препятствием для исследования и решения таких задач. Самым естественным путем было попытаться использовать обычные методы линейного программирования, несколько видоизменив их.

Казалось бы, что естественный путь решения целочисленной задачи состоит в решении соответствующей линейной задачи с последующим округлением компонент ее оптимального вектора до ближайших целых чисел. На самом деле такой путь в большинстве случаев не только уводит от оптимума, но даже приводит иногда к недопустимому решению задачи.

Пример. Найти

$$\max\left\{f(\overline{x}) = x_1 + x_2\right\}$$

при ограничениях

$$\begin{cases} 2x_1 + 11x_2 \le 38, \\ 4x_1 - 5x_2 \le 5, \end{cases}$$

$$x_1 \ge 0,\ x_2 \ge 0,\ \ x_1,\ x_2 - \text{целые}.$$

Решим задачу геометрически (рис. 1.7).

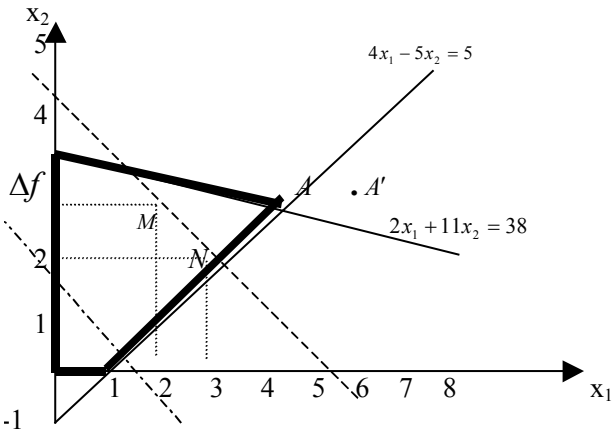


Рис. 1.7. Геометрическое решение задачи

На рис. 1.7 максимум $f(\overline{x}) = 7{,}17$ достигается в вершине $A(4{,}54;\ 2{,}63)$. Поскольку нас интересуют лишь точки с целочисленными координатами, следовательно, точка A не является допустимым решением задачи. Если округлить значения координат точки A, то получим точку $A'(5;\ 3)$, которая вообще не принадлежит области поиска. Можно показать, что целочисленный оптимум достигается в точках $N(3;\ 2)$ и $M(2;\ 3)$ и равен 5. Обе точки находятся внутри области поиска.

Итак, для задач с требованием целочисленности необходимо иметь особые методы решения, и, кроме того, оказывается, что оптимальное решение задач целочисленного программирования не обязательно принадлежит границе допустимого многогранника, что было характерно для обычных задач линейного программирования.

Тем не менее, некоторые из целочисленных задач могут быть непосредственно сведены к задачам линейного программирования. Так, обратимся к задаче о назначениях и заменим условие $x_{ij} = \begin{cases} 1 \\ 0 \end{cases}$, условием $x_{ij} \geq 0$, $i,\ j = \overline{1, n}$. После такого преобразования задача о назначениях принимает вид транспортной задачи. Поскольку числа в правых частях ограничений – целые, то по известному свойству транспортной задачи все опорные планы такой задачи целочисленны. Следовательно, ее можно решить методом потенциалов.

Условия же неотрицательности и остальные ограничения $\left(\sum x_{ij} = 1\right)$ обеспечат, чтобы каждая компонента этого решения была равна 0, или 1.

Однако целочисленную задачу линейного программирования более общего вида, такого, как задача о ранце, уже нельзя свести так просто к обычной задаче линейного программирования. Поэтому теория развития этих задач развивалась в двух направлениях.

Одно из них, предназначенное для решения общей целочисленной задачи, носит название «метод отсечения» и реализует идею сопоставления целочисленной задачи некоторой обычной задаче линейного программирования.

Вторая группа методов носит название «метод ветвей и границ». Здесь также используются иногда свойства соответствующих задач линейного программирования. Кроме того, для специфических задач разрабатываются принципиально новые комбинаторные принципы. Например, для задач о коммивояжере – есть особый метод из класса «ветвей и границ», а для задачи о назначениях – есть очень простой метод, получивший название – «венгерский алгоритм». Последний существенно проще процедуры сведения к транспортной задаче.

1.7.3. Метод отсечения. Первый алгоритм Гомори

Сущность этого метода состоит в том, что если после решения линейной задачи (без учета условия целочисленности) симплекс-методом не получен целочисленный ответ, то строят дополнительное линейное ограничение, уменьшающее допустимую область. Дополнительное ограничение, называемое отсечением, отсекает такую часть области, в которой не содержится допустимых решений целочисленной задачи, но есть найденное оптимальное решение нецелочисленной задачи. Таким образом, отсечение не позволяет потерять оптимальное решение целочисленной задачи. Далее процесс повторяется.

Рассмотрим процесс формирования алгоритма отсечения по Гомори. Постановка задачи в канонической форме имеет вид:

найти

$$\max\left\{ f(\overline{x}) = \sum_{j=1}^{n} c_j x_j + c_0 \right\} \tag{1.11}$$

при ограничениях

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = \overline{1,m}\,; \tag{1.12}$$

$$x_{ij} \geq 0 \ \text{– целые}, \ \ i = \overline{1,n}\,. \tag{1.13}$$

Искомое отсечение представляет собой линейное уравнение и строится по i-й строке симплекс-таблицы. Очевидно, i-я строка соответствует некоторой дробной базисной переменной. Отсечение имеет вид:

$$x_{n+1} = -\{\beta_i\} + \sum_{j\in\omega}\{\alpha_{ij}\}x_j\,, \qquad (1.14)$$

где x_{n+1} – дополнительная переменная, на которую также накладывается условие целочисленности, $\{\alpha_{ij}\}$, $\{\beta_i\}$ – дробные части от соответствующих чисел, ω – множество индексов свободных переменных.

Как известно, дробная часть положительного числа d определяется по формуле:

$$\{d\} = d - [d]\,,$$

где $[d]$ – ближайшее целое число, не превосходящее d (для отрицательных чисел последнее соотношение может не выполняться).

Например, для числа 5,8 будем иметь: $\{d\} = 0{,}8$, а для числа $-5{,}9$, соответственно, получим, $\{d\} = 0{,}1$.

Очевидно, что $0 \le \{d\} < 1$ и $d = \{d\} + [d]$.

Докажем, что построенное отсечение (1.14) исключает текущее оптимальное решение, но сохраняет все допустимые значения переменных в целочисленной задаче. Такое отсечение является *правильным*.

Введем понятие правильного отсечения более строго.

Определение. Пусть $\bar{x}^*$ – оптимальное решение задачи (1.11), (1.12), не являющееся целочисленным. Неравенство

$$\sum_{j=1}^{n}\mu_j x_j \le \mu_0$$

называется *правильным отсечением*, если оно удовлетворяет требованиям:

1) для вектора $\bar{x}^*$ неравенство не выполняется;

2) если $\bar{x}$ – целочисленное допустимое решение задачи (т.е. удовлетворяет условиям (1.12)), то для $\bar{x}$ неравенство выполняется.

Теорема. Пусть $\bar{x}^*$ – оптимальное решение задачи (1.11), (1.12). Если $\bar{x}_i^*$ – нецелое число, то неравенство

$$\sum_{j\in\omega}\{\alpha_{ij}\}x_j \geq \{\beta_i\} \qquad (1.15)$$

является правильным отсечением.

(Уравнение (1.14) получается из неравенства (1.15) при добавлении ослабляющей переменной $\bar{x}_{n+1}$, что необходимо для приведения задачи к каноническому виду при решении симплекс-методом.)

Доказательство.

а) Пусть $\bar{x}^*$ – базисное решение. Тогда

$$\begin{cases} x_i^* = \beta_i \text{ – нецелое, следовательно, } \{\beta_i\} > 0; \\ x_j^* = 0,\ j \in \omega. \end{cases} \qquad (1.16)$$

Таким образом, получим,

$$\sum_{j\in\omega}\{\alpha_{ij}\}x_j = 0,$$

т.е. из неравенства (1.15) следует, что $\{\beta_i\} \leq 0$. Это противоречит условию (1.16).

б) Пусть $\bar{x}$ – целочисленное, допустимое решение задачи. Следовательно, оно удовлетворяет системе ограничений. Рассмотрим i-ю строку системы ограничений, а именно:

$$x_i = \beta_i - \sum_{j\in\omega}\alpha_{ij}x_j,$$

что эквивалентно

$$x_i = [\beta_i] + \{\beta_i\} - \sum_{j\in\omega}([\alpha_{ij}] + \{\alpha_{ij}\})x_j.$$

Запишем это уравнение по-другому:

$$x_i - [\beta_i] + \sum_{j\in\omega}[\alpha_{ij}]x_j = \{\beta_i\} - \sum_{j\in\omega}\{\alpha_{ij}\}x_j.$$

Итак, левая часть уравнения – целое число, следовательно, правая часть уравнения (обозначим ее через z) – тоже целое число:

$$z = \{\beta_i\} - \sum_{j\in\omega}\{\alpha_{ij}\}x_j \text{ – целое.}$$

Причем $z < 1$, так как $\{\beta_i\} < 1$, а $\{\alpha_{ij}\} \geq 0$ и $x_j \geq 0$.

Отсюда следует, что $-z > -1$, но так как $-z$ – целое число, то $-z \geq 0$, или, с учетом предыдущего выражения,

$$-\{\beta_i\} + \sum_{j \in \omega} \{\alpha_{ij}\} x_j \geq 0, \quad \text{т.е.} \quad \sum_{j \in \omega} \{\alpha_{ij}\} x_j \geq \{\beta_i\}.$$

Таким образом, доказано, что любое целочисленное допустимое решение задачи удовлетворяет неравенству (1.15).

Алгоритм метода отсечения.

1. Найти оптимальное решение задачи линейного программирования (1.11), (1.12) без учета условия целочисленности.

Если все базисные переменные полученного решения целые, то целочисленная задача решена. В противном случае перейти к шагу 2.

2. Выбрать (в оптимальном решении) дробную базисную переменную с максимальной дробной частью x_i и построить дополнительное линейное ограничение по i-й строке последней симплекс-таблицы

$$x_{n+1} = -\{\beta_i\} + \sum_{j \in \omega} \{\alpha_{ij}\} x_j.$$

3. Добавить к исходной задаче (1.11) – (1.13) новое ограничение и перейти к шагу 1. При этом исходной симплекс-таблицей для новой задачи служит последняя симплекс-таблица предыдущей задачи, в которую добавлены строка и столбец, соответствующие новому ограничению.

Можно показать, что в результате применения такого алгоритма через конечное число шагов либо будет найдено целочисленное решение, либо будет обнаружена неразрешимость задачи.

Замечания.

1. После введения отсечения в число ограничений задача становится задачей с недопустимым базисом, так как появляется отрицательный свободный член – $\{\beta_i\}$, поэтому решение задачи осуществляется двойственным симплекс-методом.

2. По мере ввода дополнительных ограничений размерность решаемой задачи возрастает. Ограничить рост размерности можно следующим образом: если в базисе осталась переменная, введенная как дополнительная на предыдущем шаге, то надо вычеркнуть соответствующие ей строку и столбец.

1.7.4. Метод ветвей и границ

Метод ветвей и границ заключается в следующем: множество допустимых решений делят на два непересекающихся подмножества, в каждом из которых вычисляют оптимальные значения целевой функции. Из двух подмножеств выбирают то, для которого значение целевой функции лучше. Выбранное подмножество снова делят на две части. В результате процесс поиска экстремума разветвляется. При этом возможен возврат к отбрасываемым ветвям.

На рис. 1.8 приведен пример поиска оптимального решения методом ветвей и границ.

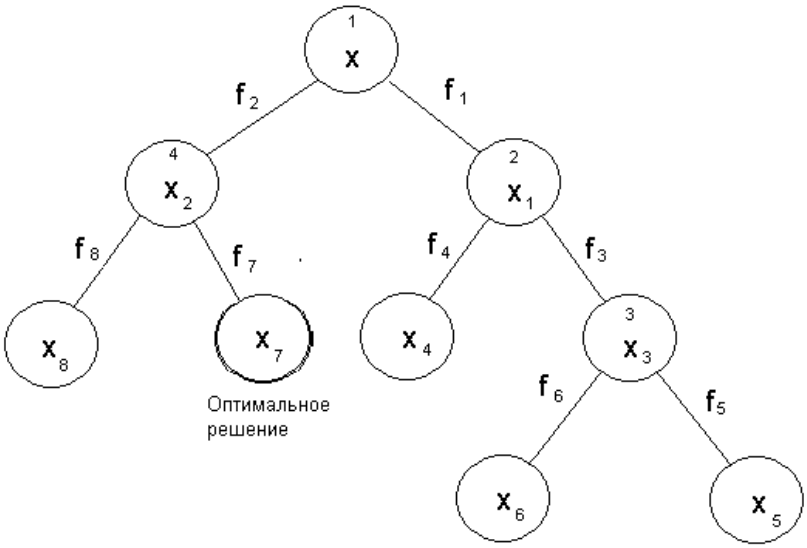


Рис. 1.8. Принцип формирования дерева метода ветвей и границ

Для данного примера, допустимое решение f_1 лучше f_2, однако решения f_6 и f_5, которые получены из решения f_1, оказались хуже чем f_2. В результате, оптимальным решением является f_7.

Как правило, метод ветвей и границ применяется к задачам следующего вида:

найти

$$\max\left\{f(\overline{x}) = c_0 + \sum_{j=1}^{n} c_j x_j\right\}$$

при ограничениях:

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, \quad j = \overline{1,m};$$

$$M_j \le x_j \le N_j, \quad j = \overline{1,n}; \quad x_j \ge 0 - \text{целые}.$$

Числа M_j, N_j – целые числа. Возможна ситуация, когда $M_j = 0$, а $N_j = \infty$. В этом случае последнее условие просто эквивалентно условию неотрицательности $\overline{x}$.

Идея метода такова: сначала решается обычная задача линейного программирования. Если решение не целое, т.е. имеется дробная базисная переменная, например x_i^*, тогда из ограничения $M_i \le x_i \le N_i$ для x_i формируют два ограничения:

$$\left[x_i^*\right] + 1 \le x_i \le N_i \quad \text{и} \quad M_i \le x_i \le \left[x_i^*\right].$$

Здесь $\left[x_i^*\right]$ – целая часть переменной x_i^*.

Это можно проиллюстрировать рис. 1.9.

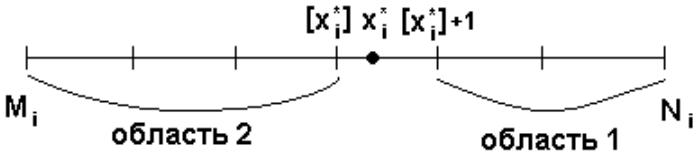


Рис. 1.9. Формирование областей поиска решения

Итак, в результате получаются две задачи линейного программирования. Процесс деления последовательно продолжается.

Алгоритм метода ветвей и границ.

0-я итерация. Первоначальный список задач состоит из исходной задачи без условия целочисленности. За начальную оценку це-

левой функции $\hat{f}^1(\bar{x})$ берут либо очевидное минимальное значение целевой функции при допустимых значениях переменных, либо $\hat{f}^1(x) = -\infty$ в трудных случая.

k-я итерация. 1. Выбрать из основного списка и решить очередную задачу с помощью симплекс алгоритма. Если список пуст, то последняя оценка целевой функции и соответствующие ей значения переменных представляют собой искомое целочисленное решение задачи.

2. Если выбранная задача не имеет решения или если полученное оптимальное значение целевой функции меньше текущей оценки $\hat{f}^k(\bar{x})$, то оставить в качестве оценки целевой функции $\hat{f}^k(\bar{x})$, т.е. положить $\hat{f}^{k+1}(\bar{x}_{k+1}) = \hat{f}^k(\bar{x})$. Перейти к шагу 1.

В противном случае, т.е. если выбранная задача имеет решение и оптимальное значение целевой функции больше оценки $\hat{f}^k(\bar{x})$, перейти к шагу 3.

3. Если полученное оптимальное решение задачи целочисленное, то зафиксировать его, принять в качестве новой оценки $\hat{f}^{k+1}(\bar{x})$ полученное оптимальное значение целевой функции и вернуться к шагу 1.

В противном случае, т.е. если решение задачи не является целочисленным, перейти к шагу 4.

4. Выбрать любую дробную переменную x_i^* из полученного оптимального решения и дополнить основной список двумя новыми задачами, отличающимися от решенной только *i*-м ограничением.

В первой задаче *i*-е ограничение имеет вид:

$$M_i \le x_i \le \left[x_i^*\right] ,$$

а во второй

$$\left[x_i^*\right] + 1 \le x_i \le N_i .$$

5. Сохранить имеющуюся оценку целевой функции для следующей итерации, т.е. положить $\hat{f}^{k+1}(\bar{x}) = \hat{f}^k(\bar{x})$, и перейти к шагу 1.

1.7.5. Метод зондирования решений

Данным методом решают задачу целочисленного программирования с булевыми переменными:

найти

$$\max\left\{ f(\overline{x}) = \sum_{j=1}^{n} c_j x_j \right\}$$

при ограничениях:

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, \quad i = \overline{1, m};$$

$$x_j = \begin{cases} 0, \\ 1, \end{cases} \quad j = \overline{1, n}.$$

Решением задачи является набор переменных $\{x_1, x_2, ..., x_n\}$, каждая из которых принимает значение 0 или 1. Всего существует 2^n таких наборов. Для получения решения поставленной задачи линейного программирования, в сущности, необходимо перебрать все эти наборы. Непосредственный перебор приводит к длительным вычислениям. Метод зондирования позволяет эти вычисления сократить. Метод относится к классу «ветвей и границ».

Совокупность ℓ переменных, где $1 \le \ell \le n$, принимающих значения 0 или 1 называется *пробным зондированием*. Переменные, не входящие в пробное решение, называются *свободными*. Любой конкретный набор свободных переменных образует *дополнение пробного решения*. Если имеется n переменных, ℓ из которых входят в пробное решение, то можно построить $2^{n-\ell}$ дополнений пробного решения.

Пусть $n = 8$ и пробное решение имеет вид $\{x_3 = 1, x_6 = 0\}$, тогда число свободных переменных равно $n - \ell = 8 - 2 = 6$. Произвольным дополнением пробного решения будет, например, такой набор свободных переменных

$$\{x_1 = 1, x_2 = 0, x_4 = 0, x_5 = 1, x_7 = 1\}.$$

Всего же можно составить $2^6 = 64$ дополнений пробного решения.

Пусть каким-то образом выбрано пробное решение. *Зондированием* называется проверка всех его дополнений на допустимость и оптимальность. Для сокращения объема вычислений при построении алгоритма зондирования используют две теоремы, позволяющие сделать заключение о невозможности найти дополнение, которое улучшает текущую оценку целевой функции, или позволяет включить в пробное решение какую-нибудь свободную переменную.

Пример. Пусть требуется найти

$$\max\left\{f(\overline{x}) = x_1 + 8x_2 + 4x_3 + 9x_4 - x_5 + 3x_6\right\}.$$

Предположим, что оценка целевой функции на k-й итерации $\hat{f}^k(\overline{x}) = 15$.

Рассмотрим пробное решение $\left\{x_2 = 0,\ x_3 = 1,\ x_4 = 0\right\}$, оценка целевой функции, следовательно, будет

$$\hat{f}^{k+1}(\overline{x}) = x_1 + 4 - x_5 + 3x_6 .$$

Максимальное значение целевой функции, которое при этом возможно (при максимальных значениях свободных переменных $x_1 = 1,\ x_5 = 0,\ x_6 = 1$) – это $f(\overline{x}) = 8$. А текущая оценка $f^k(\overline{x}) = 15$. Следовательно, нельзя подобрать дополнения пробного решения, увеличивающего значение целевой функции.

Теорема дополнения. Дополнения, улучшающего пробное решение, не существует, если выполняется хотя бы одно из неравенств:

$$\sum_{\substack{\text{по свободным}\\ \text{переменным}}} \min(a_{ij},0) > b_i - \sum_{\substack{\text{по пробным}\\ \text{переменным}}} a_{ij}x_j, \qquad i = \overline{1, m+1};$$

где

$$a_{m+1,j} = -c_j, \quad b_{m+1} = -\hat{f}^k(\overline{x}) - 1 .$$

Теорема расширения. Свободную переменную x_k можно включить в пробное решение, если хотя бы при одном i выполняется неравенство

$$\sum_{\substack{\text{по свободным}\\ \text{переменным}}} \min(a_{ij},0) + \left|a_{ik}\right| > b_i - \sum_{\substack{\text{по пробным}\\ \text{переменным}}} a_{ij}x_j, \quad i = \overline{1, m+1}.$$

Причем

$$x_k = 0, \quad \text{если} \quad a_{ik} \geq 0;$$

$$x_k = 1, \quad \text{если} \quad a_{ik} < 0.$$

Алгоритм метода зондирования решений.

0-я итерация. 1. Взять в качестве оценки целевой функции $\hat{f}^1(\overline{x})$ нуль или сумму всех отрицательных коэффициентов целевой функции (в трудных случаях $\hat{f}^1(\overline{x}) = -\infty$).

2. Сформировать первое пробное решение, состоящее из одной переменной. Переменную x_ℓ, входящую в первое пробное решение выбрать произвольно или, чтобы улучшить сходимость, выбрать переменную, имеющую максимальный коэффициент в целевой функции. Занести в основной список две задачи. В одной из них $x_\ell = 0$, в другой – $x_\ell = 1$.

k-я итерация. 1. Прекратить вычисления, если основной список пуст. В противном случае выбрать задачу из основного списка.

2. С помощью теоремы дополнения установить, существует ли допустимое дополнение пробного решения, у которого значение целевой функции превосходит текущую оценку $\hat{f}^k(\overline{x})$. Если такого дополнения не существует оставить оценку $\hat{f}^{k+1}(\overline{x}) = \hat{f}^k(\overline{x})$ и вернуться к шагу 1.

В противном случае перейти к шагу 3.

3. С помощью теоремы расширения найти свободные переменные, которые следует включить в пробное решение.

4. Если расширенное пробное решение является полным, т.е. содержит все n переменных, то (если значение целевой функции превышает оценку) зафиксировать его, принять оценку $\hat{f}^{k+1}(\overline{x})$ равной соответствующему данному решению значению целевой функции и вернуться к шагу 1.

В противном случае перейти к шагу 5.

5. Выбрать любую свободную переменную x_k, не входящую в пробное решение. Внести в основной список две задачи. В одной из них $x_k = 0$, в другой $x_k = 1$.

6. Если в полученных задачах пробные решения не являются полными, то взять значение $\hat{f}^{k+1}(\overline{x}) = \hat{f}^{k}(\overline{x})$ и вернуться к шагу 3.

Если пробные решения обеих задач полные, то проверить каждое из них на допустимость. Для допустимых решений посчитать значения целевой функции. Сравнить их с оценкой $\hat{f}^{k}(\overline{x})$. Выбрать и зафиксировать решение у которого $f(\overline{x}) > \hat{f}^{k}(\overline{x})$, положить $\hat{f}^{k+1}(\overline{x}) = f(\overline{x})$. Если таких решений нет, то положить $\hat{f}^{k+1}(\overline{x}) = \hat{f}^{k}(\overline{x})$, а полученные задачи отбросить.

Перейти к шагу 1.

1.7.6. Примеры решения задач целочисленного программирования

Пример 1. Метод отсечения.

Найти

$$\max\{f(\overline{x}) = 21x_1 + 11x_2\}$$

при ограничениях:

$$7x_1 + 4x_2 \le 13,$$

$$x_1,\ x_2 \ge 0 - \text{целые}.$$

В канонической форме задача выглядит так:

найти

$$\max\{f(\overline{x}) = 21x_1 + 11x_2\}$$

при ограничениях:

$$7x_1 + 4x_2 + x_3 = 13;$$

$$x_1,\ x_2,\ x_3 \ge 0 - \text{целые}.$$

1. Решение задачи симплекс-методом.

Составим симплекс-таблицу (табл. 1.38).

Таблица 1.38

Базис	Свободный член	x_1	x_2	x_3
x_3	13	7	4	1
$f(\overline{x})$	0	–21	–11	0

Преобразуя табл. 1.38 в соответствии с симплекс-методом, получим табл. 1.39.

Таблица 1.39

Базис	Свободный член	x_1	x_2	x_3
x_1	13/7	1	4/7	1/7
$f(\bar{x})$	39	0	1	3

Исходя из последней симплекс-таблицы, оптимальным решением будет: $x_1 = \frac{13}{7}$, $x_2 = x_3 = 0$, $f(\bar{x}) = 39$. Очевидно, x_1 – дробное число.

2. Введем отсечение по строке для x_1

$$x_4 = -\frac{6}{7} + \frac{4}{7}x_2 + \frac{1}{7}x_3 .$$

Построим следующую симплекс-таблицу (табл. 1.40).

Таблица 1.40

Базис	Свободный член	x_1	x_2	x_3	x_4
x_1	13/7	1	4/7	1/7	0
x_4	–6/7	0	– 4/7	–1/7	1
$f(\bar{x})$	39	0	1	3	0

Преобразовав табл. 1.40 в соответствии с симплекс-методом, получим табл. 1.41.

Таблица 1.41

Базис	Свободный член	x_1	x_2	x_3	x_4
x_1	1	1	0	0	1
x_2	3/2	0	1	¼	–7/4
$f(\bar{x})$	75/2	0	0	11/4	7/4

Из табл. 1.41 следует, что оптимальное решение:

$$x_1 = 1, \quad x_2 = 3/2, \quad x_4 = x_3 = 0, \quad f(x) = 75/2$$

3. Введем отсечение по строке для x_2, в результате получим:

$$x_5 = -1/2 + 1/4x_3 + 1/4x_4;$$

$$\{-7/4\} = -7/4 - (-8/4) = 1/4.$$

Используя симплекс-метод, проведем ряд последовательных преобразований табл. 1.41. Получим табл. 1.42 – 1.44.

Таблица 1.42

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5
x_1	1	1	0	0	1	0
x_2	3/2	0	1	1/4	–7/4	0
x_5	–1/2	0	0	–1/4	$-1/4$	1
$f(\bar{x})$	75/2	0	0	11/4	7/4	0

Таблица 1.43

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5
x_1	–1	1	0	–1	0	4
x_2	5	0	1	2	0	–7
x_4	2	0	0	1	1	–4
$f(\bar{x})$	34	0	0	1	0	7

Таблица 1.44

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5
x_3	1	1	0	1	0	–4
x_2	3	2	1	0	0	1
x_4	1	1	0	0	1	0
$f(\bar{x})$	33	1	0	0	0	11

Таким образом, получаем следующее *оптимальное решение исходной задачи*:

$$x_1^* = 0, \quad x_2^* = 3, \quad f(\bar{x}^*) = 33.$$

Пример 2. Метод ветвей и границ.

Найти

$$\max\left\{f(\bar{x}) = 5x_1 - x_2\right\} \qquad (1)$$

при ограничениях:

$$\begin{cases} 3x_1 - 8x_2 \le 3; & (2) \\ -4x_1 - 8x_2 \le -12; & (3) \\ 7x_1 + 6x_2 \le 42; & (4) \\ 0 \le x_1 \le 9; \\ 0 \le x_2 \le 9; \\ x_1, x_2 - \text{целые}. \end{cases}$$

В качестве начальной оценки целевой функции возьмем $\hat{f}^1(\bar{x}) = -\infty$, так как найти явно минимальное значение затруднительно.

1-я итерация. Имеем **задачу 1** – исходная задача без условия целочисленности.

Решение задачи 1. Исходя из исходной системы ограничений, имеем:

$$\begin{cases} x_3 = 3 - (3x_1 - 8x_2); \\ x_4 = -12 - (-4x_1 - 8x_2); \\ x_5 = 42 - (7x_1 + 6x_2); \\ x_6 = 9 - x_1; \\ x_7 = 9 - x_2. \end{cases}$$

На основе симплекс-метода построим табл. 1.45.

Таблица 1.45

Базис	Свободный член	x_1	x_2	x_3	x_4	x_5	x_6	x_7
x_3	3	3	8	1	0	0	0	0
x_4	–12	–4	–8	0	1	0	0	0
x_5	42	7	6	0	0	1	0	0
x_6	9	1	0	0	0	0	1	0
x_7	9	0	1	0	0	0	0	1
$f(\bar{x})$	0	–5	1	0	0	0	0	0

Так как коэффициенты в строке целевой функции не все положительны, то двойственный алгоритм симплекс-метода не пригоден. Для решения задачи воспользуемся методом искусственного базиса. При этом система ограничений приводится к виду:

$$\begin{cases} y_1 = 3 - (3x_1 - 8x_2 + x_3); \\ y_2 = 12 - (4x_1 + 8x_2 - x_4); \\ y_3 = 42 - (7x_1 + 6x_2 + x_5); \\ \quad y_4 = 9 - (x_1 + x_6); \\ \quad y_5 = 9 - (x_2 + x_7). \end{cases}$$

Целевая функция имеет вид:

$$\Phi(\overline{y}) = y_1 + y_2 + y_3 + y_4 + y_5 =$$

$$= 75 - (16x_1 + 6x_2 + x_3 - x_4 + x_5 + x_6 + x_7).$$

Оптимальным решением задачи 1 будет:

$$x_1 = 4{,}78; \quad x_2 = 1{,}42; \quad f(\overline{x}) = 22{,}5.$$

Полученное решение не является целочисленным, значение $f(\overline{x}) > f^1(\overline{x}) = -\infty$, следовательно, необходимо принять новую оценку целевой функции равной прежней, т.е. $\hat{f}^2(\overline{x}) = -\infty$ и образовать две новые задачи, взяв дробную переменную $x_1 = 4{,}78$.

Задача 2. Соотношения (1), (2), (3), (4) представляют собой систему ограничений, при этом

$$\begin{cases} 0 \le x_1 \le 4; \\ 0 \le x_2 \le 9. \end{cases}$$

Задача 3. Соотношения (1), (2), (3), (4) представляют собой систему ограничений, при этом

$$\begin{cases} 5 \le x_1 \le 9; \\ 0 \le x_2 \le 9. \end{cases}$$

На рис. 1.10 графически изображены допустимые множества, соответствующие исходной задаче и задачам 2 и 3.

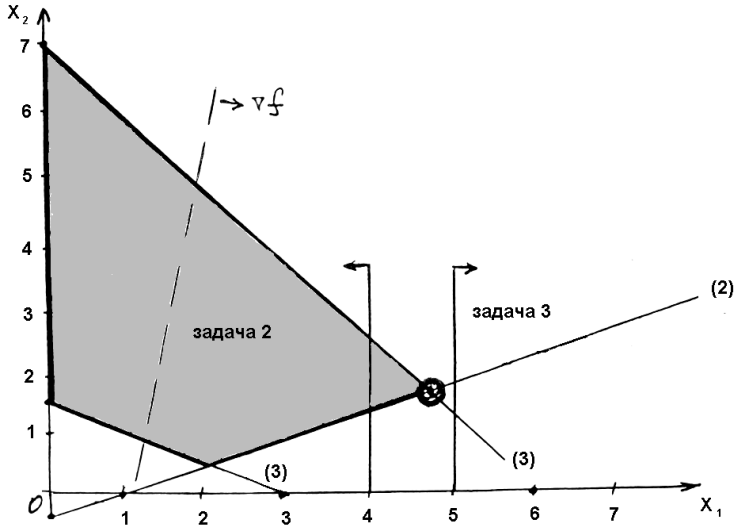


Рис. 1.10. Графическое изображение допустимых множеств задач 1, 2, 3

2-я итерация. *Задача 3 – решения нет* ($\hat{f}^3(\overline{x}) = -\infty$).

3-я итерация. *Оптимальное решение задачи 2:*

$$x_1 = 4; \quad x_2 = 1{,}125; \quad f(\overline{x}) = 18{,}875.$$

Как видим, решение не является целочисленным. Причем значение $f(\overline{x}) > \hat{f}^3(\overline{x})$, следовательно, $\hat{f}^4(\overline{x}) = -\infty$ и образуются две задачи, при этом используется дробная переменная $x_1 = 1{,}125$.

Задача 4. Соотношения (1), (2), (3), (4) – система ограничений, при этом

$$\begin{cases} 0 \le x_1 \le 4; \\ 0 \le x_2 \le 1. \end{cases}$$

Задача 5. Соотношения (1), (2), (3), (4) – система ограничений, при этом

$$\begin{cases} 0 \le x_1 \le 4; \\ 2 \le x_2 \le 9. \end{cases}$$

Изобразим графически допустимые множества задач 4 и 5 (рис. 1.11).

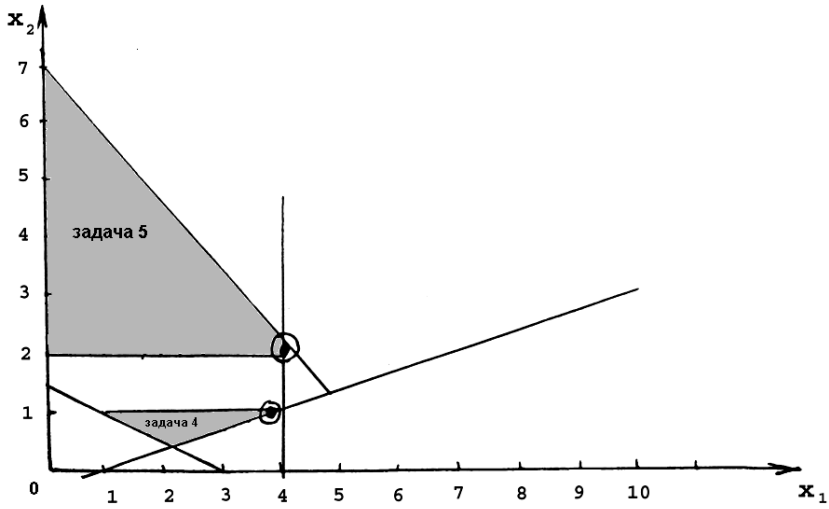


Рис. 1.11. Графическое изображение допустимых множеств задач 4 и 5

4-я итерация. *Оптимальное решение задачи 4:*

$$x_1 = 3{,}66, \quad x_2 = 1, \quad f(\overline{x}) = 17{,}33.$$

Решение не целочисленное, значение $f(\overline{x}) > \hat{f}^4(\overline{x})$, следовательно, $\hat{f}^5(\overline{x}) = -\infty$, при этом образуются две задачи.

Задача 6. Соотношения (1), (2), (3), (4) – система ограничений, при этом

$$\begin{cases} 0 \le x_1 \le 3; \\ 0 \le x_2 \le 1. \end{cases}$$

Задача 7. Соотношения (1), (2), (3), (4) – система ограничений, при этом

$$\begin{cases} x_1 = 4; \\ 0 \le x_2 \le 1. \end{cases}$$

Очевидно, оптимальным будет решение, соответствующее решению задач 5, 6 или 7.

5-я итерация. *Оптимальное решение задачи 5:*

$$x_1 = 4, \quad x_2 = 2, \quad f(\bar{x}) = 18.$$

Это целочисленное решение, следовательно, в качестве оценки целевой функции возьмем $\hat{f}^6(x) = 18$.

6-я итерация. *Оптимальное решение задачи 6:*

$$x_1 = 3, \quad x_2 = 0{,}75, \quad f(\bar{x}) = 14{,}75.$$

Очевидно, $f(\bar{x}) < \hat{f}^6(\bar{x})$, поэтому переходим в основной список задач, где осталась одна нерешенная задача 7.

7-я итерация. *Задача 7 – не имеет решения.*

8-я итерация. Итак, основой список пуст, следовательно, *оптимальное целочисленное решение*:

$$x_1 = 4, \quad x_2 = 2, \quad f(\bar{x}) = 18.$$

Решение задачи методом ветвей и границ хорошо иллюстрировать с помощью дерева решения (рис. 1.12).

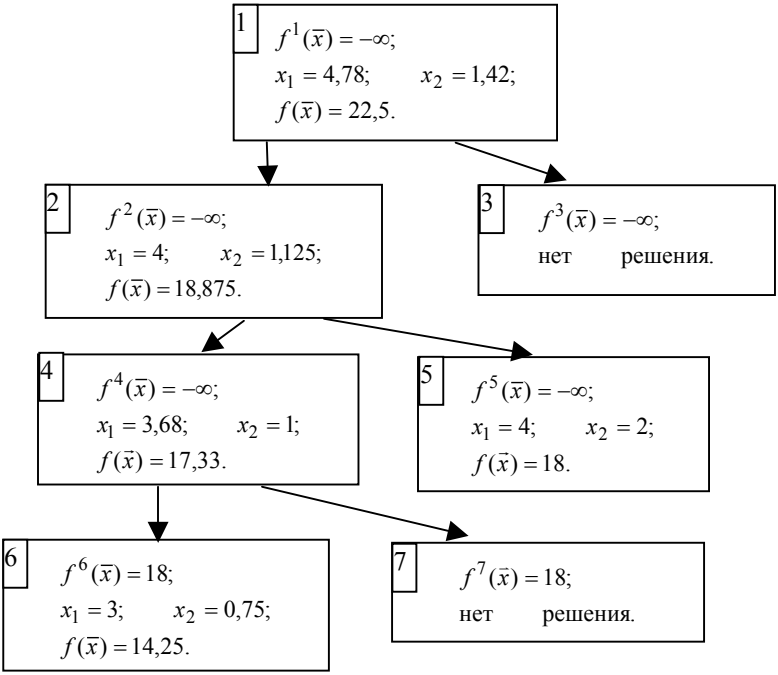


Рис. 1.12. Дерево решения задачи методом ветвей и границ

Глава 2

ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ В ПРИКЛАДНЫХ ЗАДАЧАХ ОПТИМИЗАЦИИ

2.1. Применение линейного программирования в теоретико-игровых методах исследования сложных систем

2.1.1. Теоретические основы матричных игр

Матричной игрой будем называть антагонистическую игру, в которой каждый игрок имеет конечное множество стратегий.

Как известно [28], игра двух лиц задается прямоугольной матрицей:

$$A = \left\| a_{ij} \right\|, \quad i = \overline{1,m}, \quad j = \overline{1,n},$$

где a_{ij} есть значение выигрыша игрока 1, если он выбрал свою i-ю стратегию, а игрок 2 выбрал свою j-ю стратегию. Данная игра называется игрой размерности ($m \times n$), а матрица A – *платежной матрицей*.

Если игрок 1 выбирает номер строки i, а второй – j, то в результате выбора игроками независимых стратегий игрок 2 платит игроку 1 выигрыш a_{ij}. Следовательно, игрок 1 может гарантировать себе выигрыш не менее значения: $v_1 = \max\limits_i \min\limits_j a_{ij}$ – *нижняя цена игры*, а второй гарантировать себе проигрыш не более величины $v_2 = \min\limits_j \max\limits_i a_{ij}$ – *верхняя цена игры*. В общем случае $v_1 \le v \le v_2$.

Если $\min\limits_j \max\limits_i a_{ij} = \max\limits_i \min\limits_j a_{ij} = a_{i^* j^*} = v$, то в такой игре минимаксные стратегии i^* и j^* являются оптимальными, так как реше-

ния о принятых стратегиях игроков получены независимо. В этом случае решение игры является ситуацией равновесия, а стратегии i^* и j^* называются чистыми стратегиями.

Если $\min\limits_j \max\limits_i a_{ij} > \max\limits_i \min\limits_j a_{ij}$, то в игре нет ситуации равновесия в чистых стратегиях. Для разрешения данной ситуации вводятся смешанные стратегии игроков.

Смешанной стратегией игрока 1 называется вектор $X = (x_1, ..., x_m)^{\mathrm{T}}$, удовлетворяющий условиям:

$$\sum_{i=1}^{m} x_i = 1, x_i \geq 0, \quad i = \overline{1,n},$$

где число x_i – вероятность, с которой игрок 1 выбирает свою i-ю чистую стратегию.

Величина

$$H(X,Y) = \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij} x_i y_j = X^{\mathrm{T}} AY \tag{2.1}$$

представляет собой математическое ожидание выигрыша 1-го игрока в ситуации (X,Y).

Ситуация (X^*, Y^*) называется *ситуацией равновесия* в смешанном расширении матричной игры, если для любых X и Y выполняются неравенства

$$H(X,Y^*) \leq H(X^*,Y^*) \leq H(X^*,Y). \tag{2.2}$$

В матричной игре может быть несколько ситуаций равновесия.

Можно доказать следующее свойство:

пусть $X^* \in S_m$, $Y^* \in S_n$, v – действительное число, тогда для того чтобы X^*, Y^* были оптимальными стратегиями, а v – ценой игры, необходимо и достаточно, чтобы выполнялось соотношения:

$$\begin{cases} H(i, Y^*) = \sum_{j=1}^{n} a_{ij} y^*{}_j \le v, \quad i = \overline{1, m}; \\ H(X^*, j) = \sum_{i=1}^{m} a_{ij} x^*{}_i \ge v \quad j = \overline{1, n}. \end{cases} \tag{2.3}$$

Под решением матричной игры будем понимать нахождение векторов X и Y, а также значения цены игры v.

И так, пусть дана платежная матрица

$$A = \begin{pmatrix} a_{11} & a_{12} & . & a_{1n} \\ . & . & . & . \\ a_{m1} & a_{m1} & . & a_{mn} \end{pmatrix}.$$

Положим, что решения в чистых стратегиях нет.

Будем искать решение игры ($m \times n$) в смешанных стратегиях в виде:

$$\begin{aligned} X &= (x_1, ..., x_m)^{\mathrm{T}}, \quad x_i \ge 0, \quad i = \overline{1, m}; \\ Y &= (y_1, ..., y_m)^{\mathrm{T}}, \quad y_j \ge 0, \quad j = \overline{1, n} \end{aligned} \tag{2.4}$$

при ограничениях:

$$\sum_{i=1}^{m} x_i = 1, \quad \sum_{j=1}^{n} y_j = 1.$$

Тогда, если X^*, Y^* – оптимальные смешанные стратегии первой и второй стороны, то, в соответствии с соотношениями (2.3), выполняются неравенства:

$$\begin{cases} \sum_{j=1}^{n} a_{ij} y_j^* \le v, \quad i = \overline{1, m}; \\ \sum_{i=1}^{m} a_{ij} x_i^* \ge v, \quad j = \overline{1, n}, \end{cases} \tag{2.5}$$

где v – цена игры.

2.1.2. Сведение матричной игры к задаче линейного программирования

Как правило, любая матричная игра может быть сведена к паре двойственных задач линейного программирования. Чтобы преобразование было корректным, необходимо выполнение условия $v > 0$. Это условие обязательно выполняется, если все элементы платежной матрицы неотрицательны. Добиться неотрицательности можно путем прибавления достаточно большой константы ко всем элементам матрицы, что не меняет множества оптимальных стратегий игроков, так как приводит к игре стратегически эквивалентной исходной [28], [29].

В теории игр доказывается следующая теорема.

Теорема 2.1. Пусть множество V' состоит из тех и только тех чисел v', для которых существует такая стратегия Y игрока 2, что справедливы неравенства

$$H(i, Y) \le v', \quad i = \overline{1, m}, \tag{2.6}$$

тогда значение цены матричной игры v равно наименьшему из чисел множества V', а вектор Y^*, для которого эти неравенства справедливы при $v' = v$, является оптимальной стратегией игрока 2.

Таким образом, чтобы найти v и Y^*, надо определить минимальное значение v', удовлетворяющее неравенствам (2.6), которые удобнее записать так:

$$\sum_{j=1}^{n} a_{ij} y_j \le v', \quad i = \overline{1, m}. \tag{2.7}$$

Разделим обе части неравенства (2.7) на v' и положим $u_j = y_j / v'$, тогда оно примет вид

$$\sum_{j=1}^{n} a_{ij} u_j \le 1, \quad i = \overline{1, m}. \tag{2.8}$$

С учетом того, что

$$\sum_{j=1}^{n} u_j = \frac{1}{v'} \sum_{j=1}^{n} y_j = \frac{1}{v'}, \tag{2.9}$$

задача минимизации v' сводится к задаче максимизации $\frac{1}{v'}$ или, что то же самое, максимизации функции $f(\overline{u})$, т.е. получили задачу линейного программирования, которую можно сформулировать в виде:

найти

$$\max\left\{f(\overline{u})=\sum_{j=1}^{n}u_j\right\} \tag{2.10}$$

при ограничениях:

$$\begin{cases}\sum_{j=1}^{n}a_{ij}u_j\le 1, & i=\overline{1,m};\\ u_j\ge 0,\vdots\, j=\overline{1,n}.\end{cases}$$

Пусть $\overline{u}^*=\left(u_1^*,...,u_n^*\right)$ – оптимальное решение полученной задачи линейного программирования, тогда искомая цена игры определяется соотношением:

$$v=\frac{1}{f(\overline{u}^*)}. \tag{2.11}$$

Исходя из этого, компоненты оптимальной стратегии второго игрока можно представить в виде

$$y_j^*=vu_j^*,\quad j=\overline{1,n}. \tag{2.12}$$

Необходимость упомянутого ранее условия неотрицательности ($\nu>0$) для корректности проведенных преобразований очевидна.

Приведенную выше теорему 2.1 можно аналогичным образом применить и к стратегиям первого игрока. Из неё следует, что для определения цены игры ν и оптимальной стратегии X^* необходимо найти максимальное значение числа ν'', удовлетворяющее неравенствам:

$$\sum_{i=1}^{m}a_{ij}x_i\ge\nu'',\quad j=\overline{1,n}.$$

В результате, полагая $t_i = x_i / \nu''$, приходим к следующей задаче линейного программирования:

найти

$$\min \left\{ \varphi(\bar{t}) = \sum_{i=1}^{m} t_i \right\} \tag{2.13}$$

при ограничениях:

$$\begin{cases} \sum_{i=1}^{m} a_{ij} t_i \geq 1, \quad j = \overline{1,n}; \\ t_i \geq 0, \, i = \overline{1,m}. \end{cases}$$

Тогда, если $\bar{t}^* = \left(t_1^*, ..., t_m^*\right)$ – оптимальное решение этой задачи, то цена игры определяется соотношением аналогичным выражению (2.11):

$$v = \frac{1}{\varphi\left(\bar{t}^*\right)}, \tag{2.14}$$

и, соответственно, компоненты оптимальной стратегии первого игрока задаются в виде

$$x_i^* = v t_i^*, \quad i = \overline{1,m}. \tag{2.15}$$

Как видно, вторая задача (2.13) является двойственной к первой (2.10), поэтому справедливо равенство:

$$f\left(\bar{u}^*\right) = \varphi\left(\bar{t}^*\right). \tag{2.16}$$

Как отмечено в разд. 1.5 данной книги, решение двойственных задач линейного программирования осуществляется одновременно, путем однократного применения симплекс-метода.

Таким образом, сведение любой матричной игры к задаче линейного программирования приводит к необходимости, с одной стороны, максимизации функции $f(\bar{u})$:

$$\max \left\{ f(\bar{u}) = \sum_{j=1}^{n} u_j \right\}$$

при ограничениях:

$$\begin{cases} \sum_{j=1}^{n} a_{ij} u_j \le 1 \quad i = \overline{1,m}; \\ u_j \ge 0, \quad j = \overline{1,n}, \end{cases}$$

где $u_j = y_j / \nu'$.

С другой стороны, минимизации функции $\varphi(\bar{t})$:

$$\min \left\{ \varphi(\bar{t}) = \sum_{i=1}^{m} t_i \right\}$$

при ограничениях:

$$\begin{cases} \sum_{i=1}^{m} a_{ij} t_i \ge 1, \quad j = \overline{1,n}; \\ t_i \ge 0, \quad i = \overline{1,m}, \end{cases}$$

где $t_i = x_i / v''$.

Легко видеть, что вторая задача является двойственной к первой, более того, и первая и вторая задача прекрасно сводятся к канонической форме записи задачи линейного программирования, которая успешно решается методами, изложенными в гл. 1 настоящего издания.

2.2. Оптимальное распределение запасов реактивности при работе системы ядерных реакторов в переменном суточном графике нагрузки

2.2.1. Физическая постановка задачи

В условиях широкого ввода ядерно-энергетических мощностей и растущего разуплотнения графиков нагрузок энергосистем, использование атомных электростанций (АЭС) в базовой части графика нагрузок будет представлять все большие трудности. Поэтому часть электростанций вынуждена будет работать в соответствии с суточными и сезонными колебаниями потребности в электроэнергии.

При этом, решение проблемы маневренности энергоблоков АЭС возможно только после проведения комплекса научно-исследова-

тельских и опытно-конструкторских работ. Основные из них следующие:

1) снятие ограничений на число циклов пуск-остановка для всего оборудования АЭС;

2) улучшение схем пусков и остановок АЭС и снижение потерь тепла при расхолаживании блоков за счет рационального использования остаточного энерговыделения ядерного топлива и теплоаккумулирующей способности графитовой кладки;

3) решение проблемы выбора оптимального типа топлива, защитных оболочек для него и конструкций тепловыделяющих элементов с учетом работы реакторов при переменных нагрузках;

4) приведение в соответствие с требованиями эксплуатации систем управления и защиты реакторов и обеспечение реакторов необходимым запасом реактивности.

Первые три проблемы носят, в основном, теплотехнический и прочностной характер, в то время как последняя связана с физикой ядерного реактора и обусловлена нестационарным отравлением реактора ксеноном.

Обеспечение реактора оперативным запасом реактивности, позволяющим компенсировать нестационарное отравление ксеноном, приводит к снижению энерговыработки реактора. Потерю энерговыработки реактора, работающего в переменном графике нагрузки, по сравнению с энерговыработкой при работе на номинальной мощности, можно связать с резервируемым запасом реактивности соотношением

$$\Delta Q = \frac{\Delta\rho}{a}, \qquad (2.17)$$

где a – темп выгорания, 1/(кВт · сут); $\Delta\rho$ – запас реактивности, отн. ед.; $\Delta Q = Q_m - Q$ – разность между энерговыработкой при работе в базисном режиме (Q_m) и режиме переменных нагрузок (Q), кВт · сут.

Величина резервируемого запаса реактивности, а следовательно, и потеря энерговыработки реактора, зависит от требуемой длительности работы реактора на пониженной мощности W, от степени снижения мощности реактора $\varepsilon = \frac{W}{W_н}$, где $W_н$ – номинальная мощ-

ность реактора, от величины плотности потока нейтронов при работе на номинальной мощности и других характеристик реактора.

В рамках точечной модели запас реактивности, обеспечивающий работу реактора на пониженной мощности $\varepsilon W_{\text{н}}$ в течение произвольного времени после снижения мощности равен

$$\Delta\rho = x_{\text{м}} - x_{\text{р}}, \tag{2.18}$$

где $x_{\text{м}}$ и $x_{\text{р}}$ – максимальная и равновесная концентрации ксенона, нормированные на $\dfrac{\nu_f \Sigma_f}{\sigma_X}$; Σ_f – макроскопическое сечение деления активной зоны реактора; ν_f – среднее число вторичных нейтронов на акт деления; σ_X – сечение поглощения ксенона, см^2.

На АЭС, как правило, устанавливается несколько энергоблоков. В общем случае энергоблоки могут отличаться электрической мощностью, темпом выгорания, другими характеристиками и работать независимо друг от друга. Условием, связывающим реакторы, входящие в состав АЭС, является выработка заданного количества электроэнергии атомной электростанцией.

При ограниченном оперативном запасе реактивности, например, для реакторов с непрерывной перегрузкой или в конце кампании для реакторов с дискретной перегрузкой возникает задача оптимизации распределения запасов реактивности в системе реакторов с целью минимизации суммарной потери энерговыработки при удовлетворении заданному графику суточного снижения мощности.

Математически задача формулируется следующим образом:

найти

$$\min_{\Delta\rho_1 \ldots \Delta\rho_N} \left\{ \sum_{i=1}^{N} \frac{\Delta\rho_i}{a_i} \right\}$$

при ограничениях:

$$\begin{aligned} &\sum_{i=1}^{N} \delta_i \varepsilon_i(\Delta\rho_i) = \alpha; \\ &0 \le \Delta\rho_i \le \Delta\rho_{im}, \; i = 1, \ldots, N, \end{aligned} \tag{2.19}$$

где N – число реакторов на станции; $\Delta\rho_i$ – оперативный запас реактивности i-го реактора; ε_i – степень снижения мощности i-го реактора; α – заданная степень снижения мощности АЭС, $\alpha = \dfrac{\sum_{i=1}^{N} W_i}{\sum_{i=1}^{N} W_{i\text{н}}}$; δ_i – доля электрической мощности i-го реактора, $\delta_i = \dfrac{W_{i\text{н}}}{\sum_{i=1}^{N} W_{i\text{н}}}$.

Заметим, что оптимизация возможна только при $< 0 < \alpha < 1$, так как при $\alpha = 0$ и $\alpha = 1$ значения оперативных запасов реактивности определены:

$$\Delta\rho_i = \Delta\rho_{i\text{м}},$$

$$\Delta\rho_i = 0.$$

Поэтому качественно понятно, что эффект от оптимизации будет сильнее всего сказываться в средней части диапазона изменения α. Вид зависимости $\varepsilon(\Delta\rho)$ возможной степени снижения мощности реактора от запаса реактивности, вообще говоря, определяется режимом изменения мощности реактора.

Наиболее простым режимом для анализа является снижение мощности реактора с максимально возможной скоростью до определенного уровня и поддержание реактора на данном уровне до момента выхода АЭС на номинальную мощность. Характер зависимости $\varepsilon(\Delta\rho)$ при различных плотностях потоков нейтронов этом случае показан на рис. 2.1.

Как видно из рисунка, при малых плотностях потока нейтронов зависимость $\varepsilon(\Delta\rho)$ представляет собой практически линейную функцию. С увеличением плотности потока нейтронов нелинейность в характере зависимости $\varepsilon(\Delta\rho)$ увеличивается.

В данном разделе рассмотрено решение задачи для системы реакторов с линейной зависимостью возможной степени снижения мощности от запаса реактивности. (Решение задачи для системы

реакторов с нелинейной зависимостью ε(Δρ) рассмотрено в разд. 4.2.)

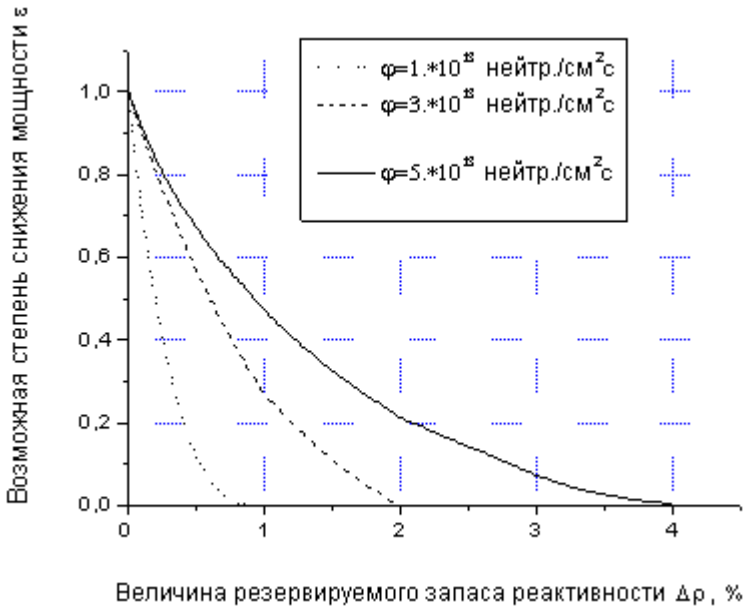


Рис. 2.1. Зависимость возможной степени снижения мощности от величины резервируемого запаса реактивности

2.2.2. Оптимальное распределение запасов реактивности в системе реакторов с линейной зависимостью возможной степени снижения мощности от запаса реактивности

В этом случае математическая постановка задачи такова: найти

$$\min_{\Delta\rho_i}\left\{\sum_{i=1}^{N}\frac{\Delta\rho_i}{a_i}\right\}$$

при ограничениях:

$$\sum_{i=1}^{N} \delta_i \varepsilon_i (\Delta\rho_i) = \alpha;$$
$$0 \le \Delta\rho_i \le \Delta\rho_{im}, \quad i = 1, ..., N, \tag{2.20}$$

где $\varepsilon_i = 1 - K_i \Delta\rho_i$.

Как видно из соотношения (2.20) оптимизационная задача относится к классу задач линейного программирования и может быть решена симплекс-методам (см. разд. 1.2). Однако вследствие того, что задача содержит всего одно уравнение связи между переменными, она может быть решена для произвольного числа реакторов, входящих в систему, достаточно просто без применения ЭВМ [66].

Решение задачи. Графическая интерпретация. Введем для удобства новые переменные

$$z_i = \frac{\Delta\rho_i}{\Delta\rho_{i\text{м}}}, \quad i = 1, ..., N,$$

имеющие смысл относительных запасов реактивности. Примем во внимание, что в широком диапазоне изменения плотностей потоков нейтронов $\left(\varphi > 6 \div 7 \cdot 10^{12} \ \frac{\text{нейтр.}}{\text{см}^2 \cdot \text{с}}\right)$ зависимость $\Delta\rho_{\text{м}}(\varphi)$ линейна, т.е. $\Delta\rho_{\text{м}} = b \cdot \varphi$, где $b = \text{const}$.

Запишем оптимизационную задачу (2.20) для новых переменных:

найти

$$\min_{z_1 ... z_N} \sum_{i=1}^{N} \frac{\varphi_i}{a_i} z_i$$

при ограничениях:

$$\sum_{i=1}^{N} \delta_i \ (1 - z_i) = \alpha;$$
$$0 \le z_i \le 1 \quad i = 1, ..., N. \tag{2.21}$$

Учитывая, что $\sum_{i=1}^{N} \delta_i = 1$, и обозначив

$$y_i = \delta_i z_i,$$

$$\mu_i = \frac{\varphi_i}{\delta_i a_i},$$

приведем оптимизационную задачу (2.21) к стандартному виду задач линейного программирования:

найти

$$\min_{z_1 \ldots z_N} \sum_{i=1}^{N} \mu_i y_i$$

при ограничениях:

$$\sum_{i=1}^{N} \delta_i \ (1 - z_i) = \alpha; \qquad (2.22)$$

$$0 \le z_i \le 1 \quad i = 1, \ldots, N.$$

Перенумеруем реакторы, входящие в систему, в порядке возрастания коэффициентов μ_i.

Пусть $1 - \alpha \le \delta_1$, тогда ясно, что сумма $S = \sum_{i=1}^{N} \mu_i y_i$ минимальна, если

$$y_1 = 1 - \alpha, \quad y_2 = \ldots = y_i = \ldots = y_N = 0.$$

Действительно, если распределение $\{y_i\}$ имеет любой другой вид, например

$$y_1 = \gamma_1, \quad y_2 = \gamma_2, \quad \ldots, \quad y_N = 1 - \alpha - \sum_{i=1}^{N-1} \gamma_i, \quad \text{причем} \quad \gamma_1 = 1 - \alpha,$$

то минимизируемая сумма $\sum_{i=1}^{N} \mu_i y_i$ имеет величину

$$S' = \mu_1 \gamma_1 + \mu_2 \gamma_2 + \ldots + \mu_N \left(1 - \alpha - \sum_{i=1}^{N-1} \gamma_i \right),$$

при оптимальном же распределении

$$S' = \mu_1 \gamma_1 + \mu_1 \gamma_2 + \ldots + \mu_1 \left(1 - \alpha - \sum_{i=1}^{N-1} \gamma_i \right).$$

Так как $\mu_1 \le \mu_2 \le ... \le \mu_N$, то очевидно, что $S < S'$.

Рассуждая аналогичным образом для случаев

$$\delta_1 < 1-\alpha \le \delta_1 + \delta_2, \quad \delta_1 + \delta_2 < 1-\alpha \le \delta_1 + \delta_2 + \delta_3, \quad ...,$$

$$\sum_{i=1}^{N-1} \delta_i < 1-\alpha \le \sum_{i=1}^{N} \delta_i = 1,$$

можно получить следующее решение задачи (2.22):

$$y_1 = 1-\alpha, \quad y_2 = ... = y_i = ... = y_N = 0 \quad \text{при} \quad 0 < 1-\alpha \le \delta_1;$$

$$y_1 = \delta_1, \quad y_2 = 1-\alpha-\delta_1, \quad y_3 = ... = y_N = 0 \quad \text{при} \quad \delta_1 < 1-\alpha \le \delta_1 + \delta_2;$$

$$y_1 = \delta_1, \quad y_2 = \delta_2, \quad y_{N-1} = \delta_{N-1}, \quad y_N = 1-\alpha - \sum_{i=1}^{N-1} y_i$$

$$\text{при} \quad \sum_{i=1}^{N-1} \delta_i < 1-\alpha \le \sum_{i=1}^{N} \delta_i = 1.$$

Возвращаясь к прежним переменным $z_i = \dfrac{y_i}{\delta_i}$, получим решение исходной оптимизационной задачи (2.21):

$$\left.\begin{array}{l} z_1 = \dfrac{1-\alpha}{\delta_1}, \quad z_2 = ... = z_i = ... = z_N = 0 \quad \text{при} \quad 1-\delta_1 \le \alpha < 1; \\ z_1 = 1 \quad z_2 = \dfrac{1-\alpha-\delta_1}{\delta_2}, \quad z_3 = ... = z_N = 0 \quad \text{при} \quad 1-\delta_1-\delta_2 \le \alpha < 1; \\ z_1 = z_2 = ... = z_{N-1} = 1, \quad z_N = \dfrac{1-\alpha - \sum\limits_{i=1}^{N-1} \delta_i}{\delta_N} \quad \text{при} \quad 0 < \alpha < 1 \sum\limits_{i=1}^{N-1} \delta_i. \end{array}\right\} \tag{2.23}$$

Оптимальное распределение относительных запасов реактивности при заданной степени снижения мощности системы α является координатами точки в N-мерном пространстве изменения перемен-

ных $z_1, ..., z_N$. Совокупность точек для $0 < \alpha < 1$ представляет собой оптимальную траекторию распределения запасов реактивности. Оптимальные траектории находятся на границе области изменения переменных.

В качестве примера, рассмотрим систему, состоящую из двух реакторов. Область изменения переменных представляет собой квадрат *ODEL* (рис. 2.2).

Рис. 2.2. Траектории оптимальных распределений относительных запасов реактивности в системе двух реакторов с низким потоком (линейная зависимость $\varepsilon(\Delta\rho)$)

Степень отличия реактора будем характеризовать величиной

$$F = \frac{\mu_1}{\mu_2} = \frac{\delta_1 a_1}{\varphi_1} \Big/ \frac{\delta_2 a_2}{\varphi_2},$$

в дальнейшем называемой параметром системы.

Если параметр системы больше единицы, что точка, координаты которой являются решением оптимизационной задачи, лежит на стороне *OD* при $1-\delta_1 \le \alpha < 1$ и на стороне *DE* при $0 < \alpha \le 1-\delta_1$.

Оптимальной траекторией запасов реактивности является ломаная *ODE*. Если параметр системы меньше единицы, то точка с оптимальными координатами лежит на стороне *OL* при $1-\delta_2 \le \alpha < 1$ и на стороне *LE* при $0 < \alpha \le 1-\delta_2$. Оптимальной траекторией является ломаная *OLE*.

При параметре системы, равном единице, целевая функция от распределения относительных запасов реактивности не зависит и координаты любой точки, принадлежащей квадрату *ODEL*, являются решением задачи.

Физическая интерпретация оптимального распределения запаса реактивности. Запас реактивности в первую очередь необходимо резервировать в реакторе с большой величиной комплекса $\frac{\delta a}{\varphi}$. Если заданное снижение мощности станции можно обеспечить только за счет реактора с максимально величиной $\frac{\delta a}{\varphi}$, то в остальных реакторах запас реактивности на работу в переменном режиме не резервируется. В противном случае, в реакторе с максимальным значением комплекса $\frac{\delta a}{\varphi}$ резервируется максимальный запас реактивности, а в следующем по величине комплекса $\frac{\delta a}{\varphi}$ реакторе – запас реактивности, необходимый для обеспечения заданного снижения мощности АЭС и т.д. На рис. 2.3 показаны траектории оптимальных степеней снижения мощностей для системы двух низкопоточных реакторов, соответствующие оптимальному распределению запасов реактивности.

Как видно из рис. 2.3, при параметре системы $F > 1$ первый реактор работает в полупиковом режиме, вплоть до полной остановки (если $\alpha < 1-\delta_1$), второй реактор работает в базисном режиме. При $F < 1$ характер изменения степеней снижения мощности реакторов противоположен.

Рис. 2.3. Траектории оптимальных степеней снижения мощности в системе двух реакторов с низким потоком (линейная зависимость $\varepsilon(\Delta\rho)$)

Максимально возможный эффект оптимизации. Об эффективности оптимизации будем судить по величине:

$$\Delta S(\alpha) = \frac{S_{\max} - S_{\min}}{\sum_{i=1}^{N} \frac{\varphi_i}{a_i}}$$

где $S_{\max}$ и $S_{\min}$ – максимальная и минимальная потеря энерговыработки АЭС при степени снижения мощности α; $\sum_{i=1}^{N} \frac{\varphi_i}{a_i}$ – потеря энерговыработки АЭС при резервировании запаса реактивности на полную остановку реакторов.

По своему физическому смыслу величина ΔS характеризует максимально возможный проигрыш в энерговыработке системы реакторов от пренебрежения оптимизацией и дает возможность судить о целесообразности оптимизации данной системы. На рис. 2.4, 2.5 показана зависимость максимально возможного эффекта оптимизации от степени снижения мощности АЭС для сис-

тем из двух низкопоточных реакторов. Различные системы реакторов характеризуются величиной параметра $F = \frac{\delta_1 a_1}{\varphi_1} \Big/ \frac{\delta_2 a_2}{\varphi_2}$.

На рисунках отражено, соответственно, два случая: доли мощности реакторов одинаковы $\delta_1 = \delta_2 = 0{,}5$, доли мощности реакторов отличаются в два раза $\frac{\delta_1}{\delta_2} = 2$.

Таким образом, в системе реакторов с линейной зависимостью степени снижения мощности реактора от запаса реактивности оптимальное распределение запасов реактивности находится на границе области изменения переменных, т.е. оптимальный режим работы реакторов формируется из полупиковых и базисных режимов работы отдельных реакторов. Эффект оптимизации тем больше, чем большее значение имеет параметр системы.

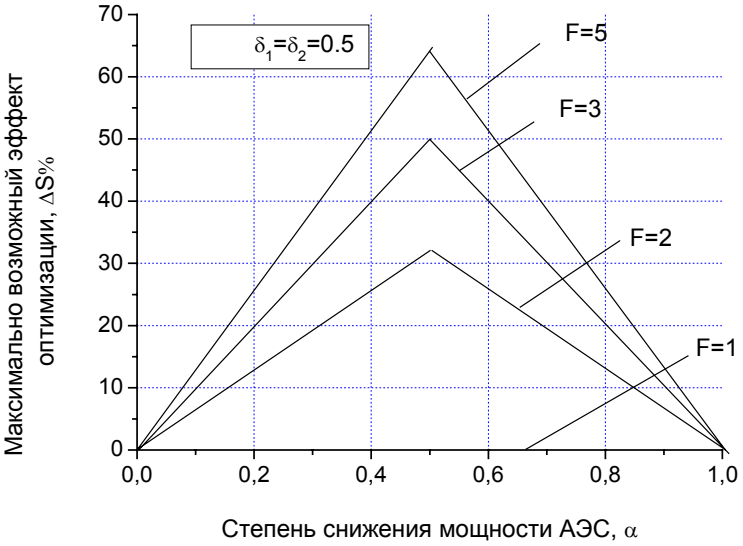


Рис. 2.4. Зависимость величины максимально возможного эффекта оптимизации от степени снижения мощности системы при различных величинах параметра *F*

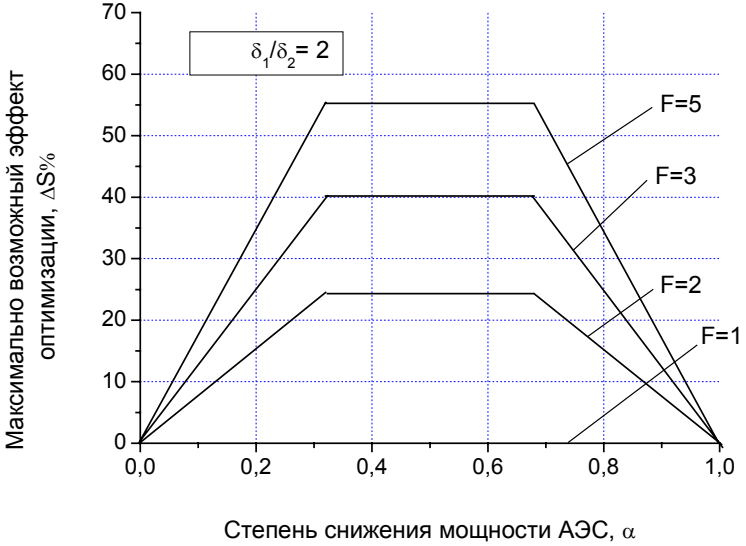


Рис. 2.5. Зависимость величины максимально возможного эффекта оптимизации от степени снижения мощности системы при различных величинах параметра *F*

2.3. Оптимальная кластеризация как задача линейного программирования

2.3.1. Математическая постановка задачи кластерного анализа

Термин «кластерный анализ» предложен Р. Трайоном в 1939 г. (cluster – гроздь). Часто в литературе используется как синонимы следующие термины: автоматическая классификация, таксономия, распознавание образов без обучения, самообучение и т.д. Кластер и класс – суть синонимы. Цепью кластерного анализа является автоматическая (или автоматизированная) классификация (кластеризация) объектов по признаку схожести.

Схожесть объектов определяется близостью их векторов дескрипторов (параметров описания) в некотором пространстве параметров.

Примером успешного неавтоматизированного кластерного анализа является классификация растений и животных в биологии и

зоологии, таблица Д.И. Менделеева в химии, разделение стран на развитые и развивающиеся в экономике и т.д.

Объекты, подлежащие классификации, называются операционными таксономическими единицами (ОТЕ).

Признаковое пространство (векторов-дескрипторов) называется П-пространством. Обычно $\dim(\Pi) < K$, где K – количество всех возможных признаков ОТЕ, так как между отдельными признаками возможна корреляция (если П – ЛВП) и в более общем случае – зависимость (часто некоррелированность).

Если для принадлежности некоторых ОТЕ к одному классу необходимо совпадение всех (или подгруппы) признаков, то такой кластер называется монотетическим или таксоном.

Если в этом нет необходимости, и лишь требуется принадлежность всех, агрегируемых в кластер, ОТЕ к заданной окрестности точки П-пространства, то такие кластеры называют политетическими.

Если вместо всего множества независимых признаков (количество таких признаков = dim(П)) применяется их часть, тем самым неявно вводится взвешивание признаков (отброшенным приписывается нулевой вес, а оставленным вес). При математической постановке задачи кластерного анализа взвешивание признаков (не только с весами 0, 1) является достаточно распространенным приемом.

Пусть имеется P независимых признаков (дескрипторов) для n ОТЕ. Тогда значение i признака для j объекта будем обозначать x_{ij} ($i = 1, ..., p$; $j = 1, ..., n$). Вектор $x_j = (x_{1j}, ..., x_{pj})$ и будет вектором признаков для j объекта. Матрица, составленная из векторов-столбцов $x_j^{\text{т}}$, иногда называется в кластерном анализе матрицей данных.

Если значения признаков не меняются в течение времени кластеризации, то p-мерное П-пространство есть конечное множество, состоящее из n элементов. В этом случае его как ЛВП над полем **P** рассматривать нельзя.

Если П-пространство считать метрическим, то метрики $\rho_l = \rho_s(x_l, x_s)$, являющиеся расстояниями между ОТЕ с номерами l, s ($l, s = 1, ..., n$), могут составить так называемую матрицу расстояний p размерности n x n,симметричную, с нулевой диагональю.

В табл. 2.1 приведены некоторые примеры метрических пространств.

Существуют расстояния, не являющиеся метриками в строгом смысле, например расстояние Джеффриса – Мачуситы:

$$M = \left(\sum_{i=1}^{p} \left(\sqrt{x_{il}} - \sqrt{x_{is}} \right)^2 \right)^{1/2},$$

или «коэффициент дивертенции»

$$CD = \left[\frac{1}{p} \sum_{i=1}^{p} \left(\frac{x_{il} - x_{is}}{x_{il} + x_{is}} \right)^2 \right]^{1/2}.$$

Таблица 2.1

Название пространства	Формула для $\rho(x_l, x_s)$
R_2^p	$\rho = \left[\sum_{i=1}^{p} (x_{il} - x_{is})^d \right]^{1/d}, \quad d \in \mathbf{N}$
R_1^p	$\rho = \sum_{i=1}^{p} \lvert x_{il} - x_{is} \rvert$
R_∞^p	$\rho = \sup_{i=1,\ldots,p} \{ \lvert x_{il} - x_{is} \rvert \}$
Пространство Махаланобиса	$(\vec{x}_l - \vec{x}_s) W^{-1} (\vec{x}_l - \vec{x}_s)^{\mathrm{T}}$, W – заданная матрица

Математическая постановка задачи кластерного анализа выглядит следующим образом: пусть $m \le n$. Надо на основании данных о П-пространстве разбить множество из n ОТЕ на m подмножеств (кластеров) $K_1, \ldots, K_m$ так, чтобы ни одна ОТЕ не принадлежала двум или более кластерам. Обычно $m < n$.

Для того, чтобы подчеркнуть, что объекты в кластеры объединяются по признаку сходства, иногда используется понятие функции сходства.

Функцией сходства называется такая неотрицательная вещественная функция $s(x_l, x_s)$, такая, что:

1) $0 \le s(\vec{x}_l, \vec{x}_s) < 1$, $\vec{x}_l \ne \vec{x}_s$;

2) $s(\vec{x}_l, \vec{x}_l) = 1$;

3) $s(\vec{x}_l, \vec{x}_s) = s(\vec{x}_s, \vec{x}_l)$; $S(\vec{x}_l, \vec{x}_s) = S_{l,s}$.

Примером кластеризации может служить разбиение n стран по уровню развития, опираясь на три признака ($p = 3$): x_1 – ВНП на душу населения, x_2 – личное потребление на душу населения, x_3 – электроэнергия на душу населения.

Решением задачи кластерного анализа является разбиение, соответствующее множеству из m кластеров, которое формируется на основании некоторого критерия, который может быть минимизируемой целевой функцией, где термин «функция» понимается в самом общем смысле.

Рассмотрим в качестве примера целевой функции (критерия) максимальное внутрикластерное расстояние до центра кластера. Если пространство евклидово, то, считая центром кластера его центр «масс» (при том, что «массы» всех ОТЕ одинаковы) при $p = 1$, получим, что ОТЕ принадлежат одному кластеру, если

$$(x_{1l} - x_{\text{цм}})^2 \le r \text{ – фиксированное число}, \quad l = 1, ..., n, \qquad (2.24)$$

где $x_{\text{цм}} = \dfrac{\sum_{l=1}^{\tilde{n}} x_{1l}}{\tilde{n}}$.

Если $m < n$, то необходимо допустить, что есть кластер K_ν ($\nu \in \{1, ..., m\}$), которому принадлежит более чем одна ОТЕ. Поэтому рассматривается множество всех различных пар ОТЕ, вычисляются их $x_{\text{цм}}$, а затем проверяется условие (2.24) по парам.

Если оно ни для одной пары не выполняется, то задача кластеризации для $m < n$ не имеет решения как для пар, так и для троек и т.д. В случае нахождения такой пары (пар) переходим к рассмотрению троек и т.д., пока не определятся самые крупные кластеры, т.е. такие, которые содержат сразу n ОТЕ или $n - 1$, или $n - 2$ и т.д. В данном примере рассмотрено так называемое «естественное» задание кластеров (так как «центры» выбирались, исходя из свойств самих ОТЕ). Если бы «центры» выбирались априорно, то распределение по кластерам было бы другим.

Значения функции сходства составляют симметричную *матрицу сходства*

$$S = \begin{pmatrix} 1 & s_{12} & \dots & s_{1n} \\ s_{21} & 1 & \dots & s_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ s_{n1} & s_{n2} & \dots & 1 \end{pmatrix},$$

величину $s_{l,s}$ иногда называют коэффициентом сходства.

Величина

$$S_\rho = \frac{1}{2}\sum_{l=1}^{n}\sum_{s=1}^{n}\rho(x_l, x_s)$$

называется общим рассеянием для данной функции расстояния.

Величина $\overline{S}_\rho = S_\rho / N_\rho$, где $N_\rho = \dfrac{n^2 - n}{2}$ называется средним рассеянием.

Величина S_ρ представляет собой сумму n^2 расстояний (элементов матрицы расстояний D), из которых n равны нулю, а $\dfrac{n^2 - n}{2}$, вообще говоря, различны и не равны. Поэтому $\overline{S}_\rho$ есть среднее арифметическое ненулевых (вообще говоря) расстояний между неповторяющимися парами ОТЕ.

Матрица S размерности $p \times p$ вида

$$S_x = \sum_{l=1}^{n}(x_l - \overline{x})(x_l - \overline{x})^{\mathrm{T}},$$

где $\overline{x} = \dfrac{1}{n}\sum_{l=1}^{n}\overline{x}_l$ называется матрицей рассеяния множества ОТЕ. След матрицы S_x:

$$\mathrm{tr}\, S_x = \sum_{l=1}^{n}\sum_{i=1}^{p}(x_{il} - \overline{x}_i)^2$$

называется статическим рассеянием множества ОТЕ. Легко показать, что

$$\frac{1}{n}\sum_{\substack{l=1\\ l<s}}^{n}\sum_{s=1}^{n}\rho^2(\bar{x}_l, \bar{x}_s).$$

Для введения критерия кластеризации (целевой функции) необходимо установить признак сходства внутри кластеров и признак различия между кластерами.

Пусть $\{I_1, ..., I_{n_1}\}$ и $\{I_{n_1+1}, ..., I_n\}$ принадлежат различным кластерам K_1 и K_2 соответственно. Тогда

$$\rho_1(K_1, K_2) = \min_{\substack{l=1,\,...,\,n_1\\ s=n_1+1,\,...,\,n}} \rho(\vec{x}_l, \vec{x}_s)$$

называется минимальным локальным расстоянием между кластерами K_1 и K_2 в данном метрическом П-пространстве, а функция

$$\rho_2 = \min_{\substack{l=1,\,...,\,n_1\\ s=n_1+1,\,...,\,n}} \rho(x_l, x_s)$$

называется максимальным локальным расстоянием;

$$\rho_3 = \sum_{l=1}^{n_1}\sum_{s=n_1+1}^{n}\rho(x_l, x_s)/n_1(n-n_1)$$

называется средним расстоянием;

$$\rho_4 = \frac{n_1(n-n_1)}{n}(\bar{\vec{x}}' - \bar{\vec{x}}'')^{\mathrm{T}}(\bar{\vec{x}}' - \bar{\vec{x}}''), \qquad (2.25)$$

где

$$\bar{x}' = \sum_{l=1}^{n_1} x_l / n_1; \quad \bar{x}'' = \sum_{l=n_1+1}^{n} x_l /(n-n_1)$$

называется статистическим расстоянием между кластерами K_1 и K_2.

Формулу (2.25) можно пояснить так: пусть кластер $K = K_1 \cup K_2$. Тогда для K

$$S = \sum_{l=1}^{n_1} (x_p - \bar{x})(x_p - \bar{x})^{\mathrm{T}} + \sum_{s=n_1+1}^{n} (x_l - \bar{x})(x_l - \bar{x})^{\mathrm{T}} , \qquad (2.26)$$

где

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} \vec{x}_i = \frac{n_1 \bar{x}' + (n - n_1)\bar{x}''}{n}, \qquad (2.27)$$

тогда из (2.26) можем записать

$$n_1 \cdot (\bar{x}' - \bar{x})(\bar{x}' - \bar{x})^{\mathrm{T}} = \frac{n_1 (n - n_1)^2}{n^2} (\bar{x}' - \bar{x}'')(\bar{x}' - \bar{x}'')^{\mathrm{T}} ,$$

$$n_2 \cdot (\vec{\bar{x}}'' - \vec{\bar{x}})(\bar{x}'' - \bar{x})^{\mathrm{T}} = \frac{n_1^2 (n - n_1)}{n^2} (\bar{x}' - \bar{x}'')(\bar{x}' - \bar{x}'')^{\mathrm{T}} . \qquad (2.28)$$

Складывая (2.27) и (2.28), получим

$$\frac{n_1 (n - n_1)}{n} (\bar{x}' - \bar{x}'')(\bar{x}' - \bar{x}'')^{\mathrm{T}} = M , \qquad (2.29)$$

где M – матрица межгруппового рассеяния, тогда

$$S_x = S_{x'} + S_{x''} + M , \qquad (2.30)$$

$$\rho_4 = \mathrm{tr}\, M . \qquad (2.31)$$

Пусть дано n объектов, разделенных по m кластерам. Тогда матрица A размерности $m \times n$ называется матрицей толерантности, если $a_{ij} = 1$ при том, что объект j принадлежит кластеру i, $a_{ij} = 0$ – в противном случае.

2.3.2. Математические критерии оптимальной кластеризации

Проблема определения процедуры разбиения анализируемых данных на кластеры остается открытой и после выбора меры сходства объектов. Критерий кластеризации может либо воспроизводить некие эвристические соображения, либо основываться на минимизации (или максимизации) какого-нибудь показателя качества.

При эвристическом подходе решающую роль играют интуиция и опыт. Он предусматривает задание набора правил, которые обес-

печивают использование выбранной меры сходства для отнесения объектов к одному из кластеров. Евклидово расстояние хорошо приспособлено для подобного подхода, что связано с естественностью его интерпретации как меры близости. Поскольку, однако, близость двух объектов является относительной мерой их подобия, обычно приходится вводить порог, чтобы установить приемлемые степени сходства для процесса отыскания кластеров.

Подход к кластеризации, предусматривающий использование показателя качества, связан с разработкой процедур, которые обеспечат минимизацию или максимизацию выбранного показателя качества. Одним из наиболее популярных показателей является сумма квадратов ошибки

$$J = \sum_{j=1}^{N_c} \sum_{x \in S_j} \left\| x - m_j \right\|^2 , \qquad (2.32)$$

где N_c – число кластеров; S_j – множество объектов, относящихся к кластеру j;

$$m_j = \frac{1}{N_j} \sum_{x \in S_j} x - \qquad (2.33)$$

вектор выборочных средних значений для множества S_j; N_j характеризует количество объектов, входящих во множество S_j. Показатель качества определяет общую сумму квадратов отклонений характеристик всех объектов, входящих в некоторый кластер, от соответствующих средних значений по кластеру.

Другой широко применяемый критерий – критерий полной суммы внутрикластерных расстояний

$$J = \sum_{i=1}^{N} \sum_{k=1}^{N} \sum_{j=1}^{N} \rho_{kj} a_{ik} a_{ij} . \qquad (2.34)$$

Здесь ρ_{ij} – евклидово расстояние между i-м и j-м объектом.

На матрицу толерантности накладывается ограничение «каждый объект принадлежит одному и только одному кластеру»

$$\sum_{i=1}^{N} a_{ij} = 1, \quad \text{где} \quad j = 1, ..., N. \tag{2.35}$$

Существует много показателей качества помимо рассмотренных: среднее квадратов расстояний между объектами в кластере; среднее квадратов расстояний между объектами, входящими в разные кластеры; показатели, основанные на понятии матрицы рассеяния; минимум и максимум дисперсии и т.д.

Недостатком таких критериев качества является их нелинейность и, как следствие, возрастание сложности оптимизации, как алгоритмической, так и вычислительной. Поэтому надо стремиться использовать линейные критерии в сочетании с выбором линейных же ограничений. В качестве примера можно рассмотреть критерий вида

$$J = \sum_{i=1}^{N} \sum_{j=1}^{N} \rho_{ij} a_{ij} + \sum_{i=1}^{N} h \cdot a_{ii} \to \min \tag{2.36}$$

при ограничениях

$$\begin{aligned} &\sum_{i=1}^{N} a_{ij} = 1, \text{ где } j = 1, ..., N; \\ &\sum_{j=1, j \neq i}^{N} a_{ij} - N \cdot a_{ii} \leq 0, \text{ где } i = 1, ..., N. \end{aligned} \tag{2.37}$$

Здесь N – число объектов, подлежащих кластеризации; a_{ij} – элементы матрицы толерантности; ρ_{ij} – евклидово расстояние между i-м и j-м объектом; h – неотрицательный весовой коэффициент. Первое ограничение гарантирует, что каждый объект должен входить в один и только один кластер. Второе ограничение предотвращает возникновение кластеров без головного объекта. Для кластеров, содержащих головной объект, первое слагаемое критерия определяет сумму внутрикластерных расстояний.

Нередко применяются алгоритмы отыскания кластеров, основанные на совместном использовании эвристического подхода и показателя качества. Подобной комбинацией является алгоритм ИСОМАД (ISODATA).

В настоящее время разработано много методов решения задачи кластеризации. К точным методам относятся методы, явно оптимизирующие некоторый критерий качества (целевую функцию), например, это методы целочисленного программирования. Приближенные методы имеют в большей или меньшей степени эвристическое происхождение, однако могут неявно также оптимизировать некоторый критерий качества. В качестве примера приближенных методов можно назвать семейство алгоритмов иерархической кластеризации и нейросетевые методы.

2.3.3. Методы оптимальной кластеризации

Как известно, решение задачи кластеризации с математической точки зрении принято представлять в виде матрицы толерантности $A = (a_{ij})$, которая по определению унимодулярна. Критерий качества кластеризации зависит от элементов этой матрицы a_{ij}, т.е. a_{ij} – переменные, по которым проводится оптимизация. Отсюда следует, что задача оптимизации такого критерия качества является задачей целочисленного программирования с булевыми переменными.

Целевая функция в общем случае является нелинейной, что сильно затрудняет (прежде всего, с точки зрения скорости) ее оптимизацию. Как правило, в задаче кластеризации стараются выбрать линейный критерий, который может быть оптимизирован методами линейного программирования. Одним из наиболее известных является целочисленный симплекс-метод (ЦСМ).

Обычный симплекс-метод не позволяет явно учесть ограничение целочисленности переменных и в общем случае дает нецелочисленное решение, поэтому Гомори (разд. 1.7.3) предложил метод отсечений, позволяющий за конечное число шагов привести полученное обычным симплекс-методом нецелочисленное решение к целочисленному. Если после решения задачи симплекс-методом без учёта целочисленности не получен целочисленный результат, то строят дополнительные линейные ограничения, уменьшающие допустимую область. Дополнительное ограничение называется правильным отсечением, если отсекает такую часть допустимой области, в которой содержится оптимальное решение нецелочисленной задачи, но нет допустимых решений целочисленной.

На практике реализация этого метода сопряжена со значительным трудностями. ЦСМ основан на методе отсечений Гомори, при использовании которого на основе дробных частей элементов нецелочисленной симплекс-таблицы строятся правильные отсечения. В то же время представление дробных чисел в современных компьютерах осуществляется в основном типами данных с плавающей запятой, которые имеют ограниченную по числу значащих цифр точность. Это приводит к тому, что последние цифры таких чисел являются незначащими и могут иметь произвольные значения, что, в свою очередь, приводит к невозможности достоверно сравнивать такие числа между собой. Одним из выходов является сравнение с некоторой допустимой погрешностью $\varepsilon << 1$, однако со временем при длительных преобразованиях эти погрешности накапливаются, и в некоторый момент превышают некоторые элементы симплекс-таблицы. Поэтому данный подход можно признать тупиковым.

Заметим, что исходная симплекс-таблица при использовании ограничений вида (2.37) является целочисленной, но не унимодулярной, что является причиной появления дробных чисел. В этом случае, если представлять элементы симплекс-таблицы не числами с плавающей запятой, а рациональными числами в виде пары двух целых чисел – числителя и знаменателя, то можно добиться абсолютной точности. Однако практика показала, что даже этот весьма ёмкий по меркам современных персональных компьютеров тип данных быстро (менее чем за 1 с расчетов) переполняется. Выходом является реализация с помощью ассемблера целочисленного типа большей разрядности.

Аддитивный алгоритм Балаша, как и целочисленный симплекс-метод (ЦСМ), позволяет максимизировать линейную целевую функцию (ЦФ) при линейных же ограничениях, но, в отличие от ЦСМ, работает только с булевыми переменными. Для решения задач такого вида Э. Балаш предложил алгоритм частичного (неявного) перебора, который получил название аддитивного, поскольку в силу условия (2.28) для вычисления по алгоритму используются лишь операции сложения (вычитания). Данный алгоритм позволяет либо найти оптимальное решение, либо установить отсутствие такового, не проводя перебора всех допустимых значений переменных.

Приведем краткое описание этого метода [7]. В обобщенном виде задача оптимальной кластеризации как задача линейного программирования имеет вид (1.1). Очевидно, что задача (2.36) является ее частным случаем – достаточно переобозначить и перенумеровать коэффициенты и переменные для приведения двойной суммы к одинарной.

Пусть z – ранее достигнутое значение ЦФ. Тогда зондируемое решение не имеет допустимого дополнения, улучшающего значение ЦФ, если

$$\sum_{j\in S}\min(a_{ij},0) > b_i - \sum_{j\in\Omega} a_{ij}x_j \text{ при } \forall\, i = 1,\ldots,m, \tag{2.38}$$

где левая часть неравенства – сумма отрицательных коэффициентов при свободных переменных в i строке ограничений; S – совокупность номеров свободных переменных; Ω – совокупность номеров частичных решений.

Если для некоторой свободной переменной x_k выполняется

$$\sum_{j\in S}\min(a_{ij},0) + |a_{ik}| > b_i - \sum_{j\in\Omega} a_{ij}x_j \text{ для } \forall i, \text{ то} \tag{2.39}$$

$$x_k = \begin{cases} 0, \text{ если } a_{ik} > 0, \\ 1, \text{ если } a_{ik} < 0. \end{cases} \tag{2.40}$$

Этапы аддитивного алгоритма таковы.

1. Если основной список задач пуст, то переход к п. 5. Если нет, то переход к п. 2.

2. Проверка условия (2.38). Если оно выполняется, то переход к новой задаче из списка основных задач (п. 1). Если нет, то вычисляются (2.39), (2.40) и переход к п. 3.

3. Если допустимое дополнение частичного решения и зондируемое решение в совокупности состоят из n переменных, то это решение запоминается, вычисляется новое значение ЦФ и переход к п. 1. Если нет, то переход к п. 4.

4. Выбирается любая оставшаяся свободная переменная и в список основных задач вводятся две задачи: в одной эта переменная нулевая, в другой – единичная. Переход к п. 1.

5. Вывод результата: либо решения нет, либо максимум ЦФ и список из оптимальных значений n переменных.

В отличие от ЦСМ, данный метод не порождает сложностей в программной реализации. Однако весомым недостатком этого метода, как и метода зондирования решений, является длительное время их работы. Это вызвано тем, что используемая форма матрицы толерантности допускает до $N!$ (N – число ОТЕ, подлежащих кластеризации) возможных представлений одного и того же разбиения по кластерам за счет перестановок строк этой матрицы. Решаемая этими методами задача имеет почти в $N!$ раз большую размерность, чем на самом деле. Это и является причиной быстрого роста времени кластеризации при незначительном увеличении мощности множества ОТЕ.

Для решения этой проблемы можно предложить использовать альтернативную матрицу толерантности вида

$$\widetilde{A} = (\widetilde{a}_{ij}),$$

где

$$\widetilde{a}_{ij} = \begin{cases} 1, & x_i \text{ и } x_j \in \text{одному кластеру}, \\ 0, & \text{иначе}, \end{cases} \qquad (2.41)$$

$i, j = 1, 2, \ldots, N$.

Заметим, что матрица $\widetilde{A}$ – симметричная, что позволяет рассматривать только ее верхний правый (или нижний левый) треугольник. В этом случае каждое разбиение по кластерам представляется только одним возможным видом матрицы толерантности, и решаемая методами упорядоченного перебора задача оптимизации приобретает адекватную размерность.

Метод зондирования решений является разновидностью аддитивного алгоритма Балаша и часто не выделяется отдельно. Его отличие заключается в добавлении к системе ограничений дополнительного линейного ограничения, зависящего от достигнутой на данном шаге оценки ЦФ. Это ограничение позволяет значительно (в несколько раз) сократить множество перебираемых значений переменных. Однако это не решает проблему с факториальным ростом размерности задачи, характерную для аддитивного алгоритма.

Метод градиентного спуска (МГС) – итерационный метод нахождения минимума функции, в общем случае, многих переменных.

Пусть $\vec{x}$ – точка в N-мерном пространстве, $f(\vec{x})$ – ЦФ, значение которой нужно минимизировать, тогда общая формула градиентного метода имеет вид:

$$x_{k+1} = x_k - \lambda \nabla f(x_k), \tag{2.42}$$

где x_k – значение точки минимума $f(x)$, достигнутое на k-м шаге метода; $\nabla f(x_k)$ – значение градиента функции $f(x)$ в точке x_k; λ – положительный шаг. Для запуска метода необходимо задать начальное приближение x_0. В зависимости от способа выбора шага λ существуют различные градиентные методы.

Применение данного метода к задаче кластеризации имеет следующие особенности:

1) оптимизируемая целевая функция имеет в качестве аргумента булеву матрицу – матрицу толерантности;

2) шаг λ может принимать одно из трех возможных значений: $\{-1; 0; 1\}$;

3) градиентный метод не позволяет явно учесть ограничения (например, вида (2.35));

4) даже при учете ограничения (2.35) оптимизация традиционных оценочных критериев кластеризации, таких как полная сумма внутрикластерных расстояний (2.34) или сумма квадратов отклонений от центров кластеров (2.32), приводит к вырожденному решению, где каждый объект входит в монокластер.

Градиентный метод был использован для кластеризации при минимизации значения критерия:

1) полной суммы внутрикластерных расстояний (2.34) при ограничении (2.35);

2) псевдополной суммы внутрикластерных расстояний (2.36) при ограничениях (2.37).

В обоих случаях алгоритм градиентного спуска был модифицирован таким образом, чтобы на каждом шаге метода, включая начальный, матрица толерантности удовлетворяла условию (2.35). Практическое применение метода показало, что, как правило, для

улучшения значения критерия необходимо сделать шаг только по одному направлению (по которому производная от критерия максимальна), шаг по еще одному направлению без пересчета значений производных критерия приводил к ухудшению значения критерия.

Оптимизация критерия полной суммы внутрикластерных расстояний (2.34) с ограничением (2.35) приводит к полностью или частично вырожденному решению, в зависимости от начального приближений.

При оптимизации критерия псевдополной суммы внутрикластерных расстояний (2.36) на каждой итерации производилась проверка на выполнение второго ограничения системы (2.37), и запоминались лучшие решения, удовлетворяющие этой системе. Однако значения, которые принимает матрица толерантности в процессе минимизации, в подавляющем большинстве случаев не удовлетворяют второму ограничению системы (2.37), и кластеризация не производится.

Наиболее перспективным выходом из описанных трудностей является использование других оптимизируемых критериев, не требующих введения в процесс градиентной оптимизации искусственного контроля за ограничениями.

2.3.4. Приближенные методы кластеризации

В приближенных методах кластеризации явно оптимизируемая ЦФ отсутствует. Как и в точных методах, результат кластеризации представляется в виде матрицы толерантности, однако ее элементы могут не принимать непосредственного участия в процессе кластеризации, как это имеет место, например, при использовании нейросетей (НС).

Метод иерархической кластеризации заключается в объединении объектов в достаточно большие кластеры, используя заданную меру сходства или расстояние между объектами. Ход такой кластеризации можно представить в виде иерархического дерева (дендограммы) [2] (рис. 2.6).

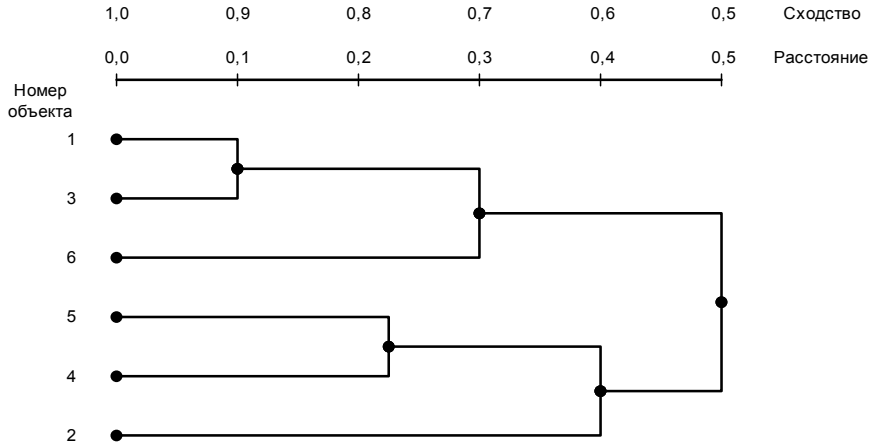


Рис. 2.6. Пример дендограммы

Выполнение метода начинается с того, что каждый объект в выборке считается монокластером – кластером, состоящим из одного объекта. Далее последовательно «ослабляется» критерий о том, какие объекты являются уникальными, а какие нет. Другими словами, понижается порог, относящийся к решению об объединении двух или более объектов в один кластер. В результате, связывается вместе всё большее и большее число объектов и агрегируется (объединяется) все больше и больше кластеров, состоящих из все сильнее различающихся элементов. Окончательно, на последнем шаге все объекты объединяются вместе.

На первом шаге, когда каждый объект представляет собой отдельный кластер, расстояния между этими объектами определяются выбранной мерой. Однако когда связываются вместе несколько объектов, возникает вопрос об определении расстояния между кластерами. Необходимо правило объединения или связи для двух кластеров. Можно связать два кластера вместе, когда любые два объекта в двух кластерах ближе друг к другу, чем соответствующее расстояние связи: используется «правило ближайшего соседа» для определения расстояния между кластерами; этот метод называется методом одиночной связи. Это правило строит «волокнистые» кластеры, т.е. кластеры, «сцепленные вместе» только отдельными элементами, случайно оказавшимися ближе остальных друг к другу.

Как альтернативу можно использовать соседей в кластерах, которые находятся дальше всех остальных пар объектов друг от друга. Этот метод называется методом полной связи. Существует также множество других методов объединения кластеров, основанных на различных подходах к объединению кластеров. Наиболее часто встречающиеся варианты иерархической кластеризации рассмотрены ниже.

Одиночная связь (метод ближайшего соседа). Как было описано выше, в этом методе расстояние между двумя кластерами определяется расстоянием между двумя наиболее близкими объектами (ближайшими соседями) в различных кластерах. Это правило должно, в известном смысле, «нанизывать» объекты вместе для формирования кластеров, и результирующие кластеры имеют тенденцию быть представленными длинными «цепочками».

Полная связь (метод наиболее удаленных соседей). В этом методе расстояния между кластерами определяются наибольшим расстоянием между любыми двумя объектами в различных кластерах («наиболее удаленными соседями»). Этот метод обычно работает очень хорошо, когда объекты происходят на самом деле из реально различных «рощ». Если же кластеры имеют в некотором роде удлиненную форму или их естественный тип является «цепочечным», то этот метод непригоден.

Невзвешенное попарное среднее. В этом методе расстояние между двумя различными кластерами вычисляется как среднее расстояние между всеми парами объектов в них. Метод эффективен, когда объекты в действительности формируют различные «рощи», однако он работает одинаково хорошо и в случаях протяженных («цепочного» типа) кластеров. Снит и Сокэл вводят аббревиатуру UPGMA для ссылки на этот метод, как на метод невзвешенного попарного арифметического среднего – unweighted pair-group method using arithmetic averages.

Взвешенное попарное среднее. Метод идентичен методу невзвешенного попарного среднего, за исключением того, что при вычислениях размер соответствующих кластеров (т.е. число объектов, содержащихся в них) используется в качестве весового коэффициента. Поэтому предлагаемый метод должен быть использован (скорее даже, чем предыдущий), когда предполагаются неравные размеры кластеров. Снит и Сокэл вводят аббревиатуру WPGMA

для ссылки на этот метод, как на метод взвешенного попарного арифметического среднего – weighted pair-group method using arithmetic averages.

Невзвешенный центроидный метод. В этом методе расстояние между двумя кластерами определяется как расстояние между их центрами тяжести. Снит и Сокэл используют аббревиатуру UPGMC для ссылки на этот метод, как на метод невзвешенного попарного центроидного усреднения – unweighted pair-group method using the centroid average.

Взвешенный центроидный метод (медиана). Этот метод идентичен предыдущему, за исключением того, что при вычислениях используются веса для учёта разницы между размерами кластеров (т.е. числами объектов в них). Поэтому, если имеются (или подозреваются) значительные отличия в размерах кластеров, этот метод оказывается предпочтительнее предыдущего. Снит и Сокэл использовали аббревиатуру WPGMC для ссылок на него, как на метод взвешенного попарного центроидного усреднения – weighted pair-group method using the centroid average.

Метод Варда. Этот метод отличается от всех других методов, поскольку он использует методы дисперсионного анализа для оценки расстояний между кластерами. Метод минимизирует сумму квадратов (SS) для любых двух (гипотетических) кластеров, которые могут быть сформированы на каждом шаге. Метод является очень эффективным, однако он стремится создавать кластеры малого размера.

К новым приближенным методам кластериации можно отнести и нейросетевые подходы. Рассмотрим их возможности на примере достаточно часто применяемой нейросети Кохонена. Нейросеть (НС) Кохонена относится к нейросетям без учителя. Это означает, что подстройка весов нейронов (обучение) такой сети производится исключительно на основе поступающих на ее вход данных. В традиционной нейросети без учителя используется т.н. соревновательное обучение, при этом выходы сети максимально скоррелированы: при любом значении входа активность всех нейронов, кроме нейрона-победителя, одинакова и равна нулю. Такой режим функционирования сети называется победитель забирает все.

Нейрон-победитель (с индексом i^*), свой для каждого входного вектора, будет служить прототипом этого вектора. Поэтому побе-

дитель выбирается так, что его вектор весов $\mathbf{w}_{i^*}$, определенный в том же d -мерном пространстве, находится ближе к данному входному вектору $\mathbf{x}$, чем у всех остальных нейронов: $\left|\mathbf{w}_{i^*} - \mathbf{x}\right| \le \left|\mathbf{w}_i - \mathbf{x}\right|$ для всех i. Если, как это обычно и делается, применять правила обучения нейронов, обеспечивающие одинаковую нормировку всех весов, например, $\left|\mathbf{w}_i\right| = 1$, то победителем окажется нейрон, дающий наибольший отклик на данный входной стимул: $\mathbf{w}_{i^*} \cdot \mathbf{x} \ge \mathbf{w}_i \cdot \mathbf{x} \;\; \forall i$. Выход такого нейрона усиливается до единичного, а остальных – подавляется до нуля.

Количество нейронов в соревновательном слое определяет максимальное разнообразие выходов и выбирается в соответствии с требуемой степенью детализации входной информации. Обученная сеть может затем классифицировать входы: нейрон-победитель определяет, к какому классу относится данный входной вектор.

По предъявлении очередного примера корректируются лишь веса нейрона-победителя

$$\Delta\mathbf{w}_{i^*}^{\tau} = \eta(\mathbf{x}^{\tau} - \mathbf{w}_{i^*}) . \qquad (2.43)$$

Здесь η – скорость обучения; $\mathbf{w}_{i^*}$ – вектор весов нейрона-победителя; $\mathbf{x}^{\tau}$ – предъявленный пример.

Описанный выше базовый алгоритм обучения на практике обычно несколько модифицируют, так как он, например, допускает существование так называемых мертвых нейронов, которые никогда не выигрывают, и, следовательно, бесполезны. Самый простой способ избежать их появления – выбирать в качестве начальных значений весов случайно выбранные в обучающей выборке входные вектора. Такой способ хорош еще и тем, что при достаточно большом числе прототипов он способствует равной «нагрузке» всех нейронов-прототипов. Это соответствует максимизации энтропии выходов в случае соревновательного слоя. В идеале каждый из нейронов соревновательного слоя должен одинаково часто становились победителем, чтобы априори невозможно было бы предсказать, какой из них победит при случайном выборе входного вектора из обучающей выборки.

Наиболее быструю сходимость обеспечивает пакетный (batch) режим обучения, когда веса изменяются лишь после предъявления всех примеров. В этом случае можно сделать приращения не малыми, помещая вес нейрона на следующем шаге сразу в центр тяжести всех входных векторов, относящихся к его ячейке. Такой алгоритм сходится за $O(1)$ итераций.

Записав правило соревновательного обучения в градиентном виде: $<\Delta\mathbf{w}>=-\eta\dfrac{\partial E}{\partial\mathbf{w}}$, легко убедиться, что оно минимизирует квадратичное отклонение входных векторов от их прототипов – весов нейронов-победителей:

$$E=\frac{1}{2}\sum\nolimits_{\alpha}\left|\mathbf{x}^{\alpha}-\mathbf{w}_{*}^{\alpha}\right|. \qquad (2.44)$$

Иными словами, сеть осуществляет кластеризацию данных: находит такие усредненные прототипы, которые минимизируют ошибку огрубления данных. Недостаток такого варианта кластеризации очевиден – «навязывание» количества кластеров, равного числу нейронов.

Один из вариантов модификации базового правила обучения соревновательного слоя состоит в том, чтобы обучать не только нейрон-победитель, но и его «соседи», хотя и с меньшей скоростью. Такой подход – «подтягивание» ближайших к победителю нейронов – применяется в топографических картах Кохонена (Kohonen, 1982). Небольшой модификацией соревновательного обучения можно добиться того, что положение нейрона в выходном слое будет коррелировать с положением прототипов в многомерном пространстве входов сети: близким нейронам будут соответствовать близкие значения входов. Тем самым, появляется возможность строить топографические карты, чрезвычайно полезные для визуализации многомерной информации. Обычно для этого используют соревновательные слои в виде двумерных сеток. Такой подход сочетает квантование данных с отображением, понижающим размерность. Причем это достигается с помощью всего лишь одного слоя нейронов, что существенно облегчает обучение.

Нейроны выходного слоя упорядочиваются, образуя одно- или двумерные решетки, т.е. теперь положение нейронов в такой решетке маркируется векторным индексом $\mathbf{i}$. Такое упорядочение

естественным образом вводит расстояние между нейронами $|\mathbf{i}-\mathbf{j}|$ в слое. Модифицированное Кохоненом правило соревновательного обучения учитывает расстояние нейронов от нейрона-победителя:

$$\Delta \mathbf{w}_{\mathbf{i}}^{\tau} = \eta \Lambda(\left|\mathbf{i}-\mathbf{i}^{*}\right|)(\mathbf{x}^{\tau} - \mathbf{w}_{\mathbf{i}}). \qquad (2.45)$$

Функция соседства $\Lambda(\left|\mathbf{i}-\mathbf{i}^{*}\right|)$ равна единице для нейрона-победителя с индексом $\mathbf{i}^{*}$ и постепенно спадает с расстоянием, например по закону $\Lambda(a) = \exp(-a^2/\sigma^2)$. Как темп обучения η, так и радиус взаимодействия нейронов σ постепенно уменьшаются в процессе обучения, так что на конечной стадии обучения мы возвращаемся к базовому правилу адаптации весов только нейронов-победителей.

После обучения нейросети Кохонена вектора весов нейронов единственного слоя представляют собой ни что иное, как центры кластеров. Принадлежность объекта кластеризуемого множества к определенному кластеру определяется близостью вектора объекта к вектору весов нейрона.

Тестовые множества могут различаться одним или несколькими из следующих параметров: размерность пространства дескрипторов (П-пространства) *d*; общее количество кластеризуемых объектов *N*.

Основные характеристики компьютера, на котором проводилось тестирование, приведены в табл. 2.2.

Таблица 2.2

Название параметра	Значение параметра
Процессор	AMD Athlon 1200 МГц
Объем и тип оперативной памяти	512 МБ DDR-266
Операционная система	Windows 2000 Professional SP3

Параметры тестовых множеств, представлены в табл. 2.3.

Таблица 2.3

Название параметра	Значение параметра
Диапазон значений параметра	[0; 10]
Распределение значений в диапазоне	равномерное

Настройки для методов кластеризации UPGMC и НС Кохонена приведены в табл. 2.4.

Отметим, что для каждой серии экспериментов число кластеров необходимо задавать, явно (НС Кохонена) или неявно (UPGMC). Для НС Кохонена число нейронов в слое является ограничением сверху числа кластеров и одновременно наиболее вероятным (по данным экспериментов) числом выделяемых кластеров. В некоторых случаях нейросеть Кохонена с N нейронами в единственном слое выделяет $N - 1$ кластер, что можно отметить как положительное свойство НС учитывать реальную структуру данных.

Таблица 2.4

Название параметра	Значение параметра
Активационная функция нейронов НС Кохонена	Линейная: $f(x) = x$
Число нейронов в слое НС Кохонена	равно заданному числу кластеров
Начальные веса слоя НС Кохонена	Случайно выбранные объекты кластеризуемого множества
Точность обучения НС Кохонена	0,01
Начальный радиус взаимодействия нейронов	1,0
Коэффициент изменения радиуса взаимодействия	0,9
Начальная скорость обучения	1,0
Коэффициент изменения скорости обучения	1,0
Пакетный режим обучения	Да
Критическое расстояние UPGMC	Для получения заданного числа кластеров
Частота обновления информации, 1/итер	100

На основе известного тестового множества подбирается критическое расстояние UPGMC для получения заданного числа класте-

ров. Затем найденное расстояние используется для проведения эксперимента.

Для НС Кохонена проводилось по три эксперимента с различными начальными приближениями, и в качестве результата бралось среднее арифметическое. Используются следующие обозначения для критериев оценки качества кластеризации: полная сумма внутрикластерных расстояний – ПСВКР; сумма квадратов отклонений от центра кластера – СКО.

2.3.5. Зависимость времени и качества кластеризации от количества объектов, кластеров и размерности признакового пространства

Положим число кластеров $m = 5$, размерность признакового пространства $d = 3$. Результаты испытаний приведены в табл. 2.6 и на рис. 2.7, 2.8.

Таблица 2.6

Параметр	Количество кластеризуемых объектов			
	10	50	75	100
Время UPGMC, мин:с, мс	00:00.019	00:04.325	00:41.000	03:42.000
Время НС Кохонена, мин:с, мс	00:00.060	00:00.477	00:00.811	00:01.191
Полная сумма внутрикластерных расстояний UPGMC	24	1134	2867	9673
Полная сумма внутрикластерных расстояний НС Кохонена	28	903	1989	4412
Сумма квадратов отклонений UPGMC	30	440	784	1427
Сумма квадратов отклонений НС Кохонена	45	335	508	900

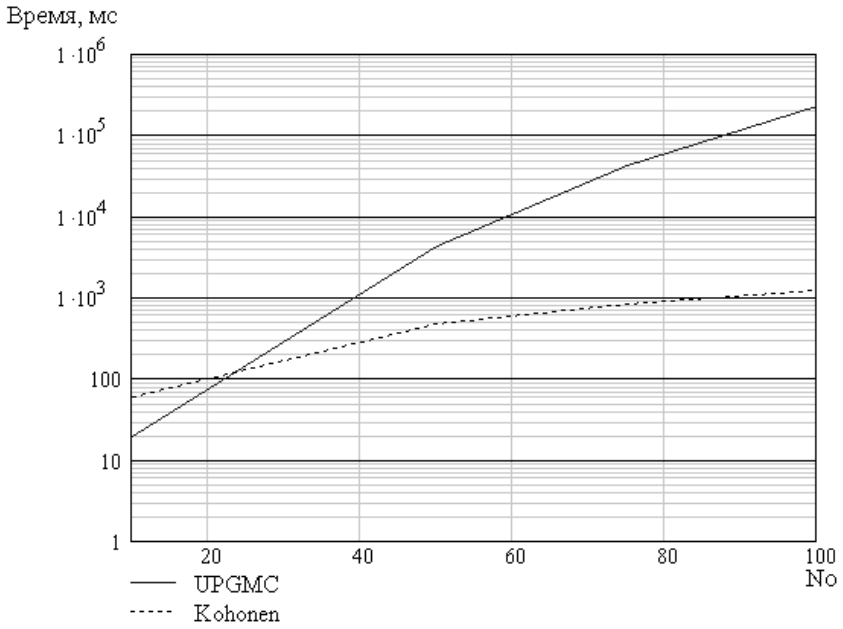


Рис. 2.7. Зависимость времени кластеризации от количества объектов

Как видно из рис. 2.11, время кластеризации для обоих алгоритмов растет экспоненциально, однако для алгоритма НС Кохонена коэффициент роста значительно ниже, в результате уже примерно при 25 объектах в кластеризуемом множестве метод UPGMC начинает проигрывать по времени. Из рисунков видно, что во всех случаях НС Кохонена дает лучшее значение обоих критериев.

В данном эксперименте положим число объектов в кластеризуемом множестве $N = 50$, размерность признакового пространства $d = 3$. Будем кластеризовать одно и то же множество для получения разного количества кластеров (напомним, что оно задается числом нейронов в слое НС Кохонена и неявно – критическим расстоянием в UPGMC). Результаты испытаний приведены в табл. 2.7.

Как следует из рис. 2.9, время кластеризации для метода UPGMC практически не зависит от количества кластеров (это подтверждается и теоретически, так как число итераций алгоритма невзвешенного попарного усреднения равно разности числа объектов кластеризуемого множества и количества выделяемых методом

кластеров). Для НС Кохонена наблюдается примерно экспоненциальный рост, связанный с тем, что количество выделяемых НС кластеров определяется числом нейронов в единственном слое; следовательно, при одном и том же числе объектов в кластеризуемом множестве увеличение числа нейронов увеличивает также и число операций на каждой итерации алгоритма нейросетевой кластеризации.

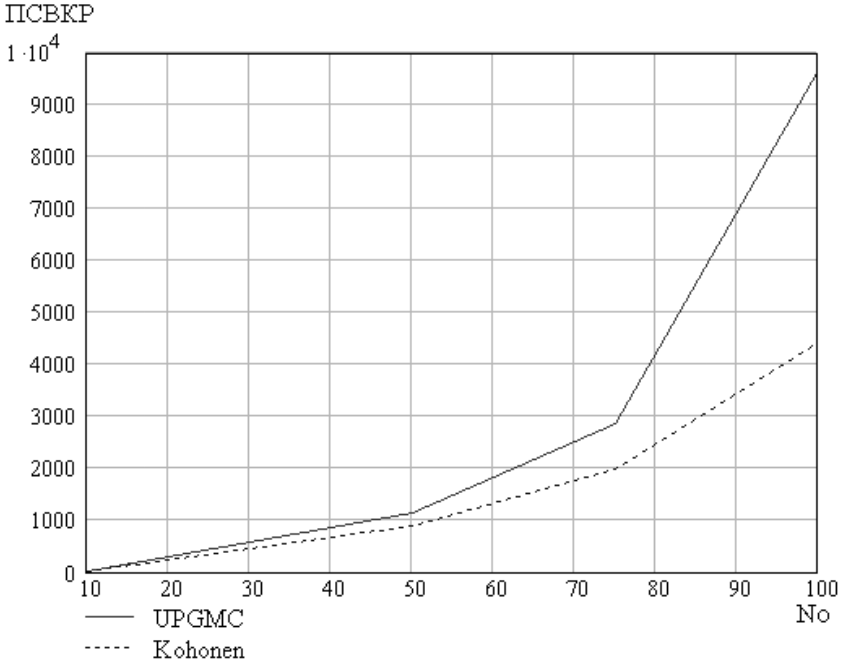


Рис. 2.8. Зависимость значения критерия качества от количества объектов (ПСВКР)

Рис. 2.10 и 2.11 показывают превосходство алгоритма НС Кохонена по качеству кластеризации над методом UPGMC.

Зависимость времени и качества кластеризации от размерности пространства исследовалась при следующих условиях: количество кластеров $m = 5$, число объектов в кластеризуемом множестве $N = 50$. Результаты приведены в табл. 2.8.

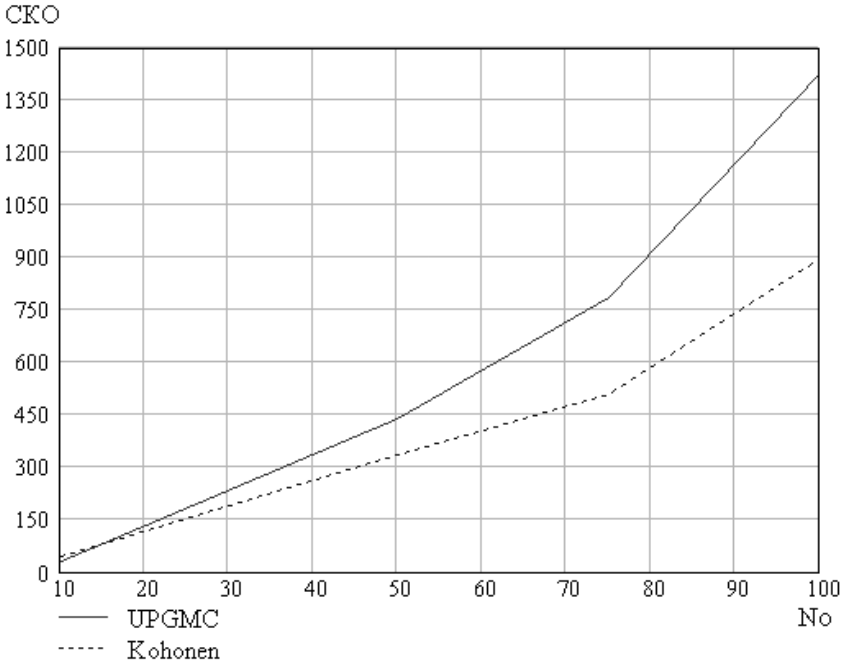


Рис. 2.9. Зависимость значения критерия качества от количества объектов (СКО)

Таблица 2.7 (начало)

Параметр	Количество кластеров			
	2	3	4	5
Время UPGMC, мин:с, мс	00:05.366	00:05.927	00:05.588	00:04.325
Время НС Кохонена, мин:с, мс	00:00.143	00:00.317	00:00.400	00:00.477
Полная сумма внутрикластерных расстояний UPGMC	4574	2353	1784	1134
Полная сумма внутрикластерных расстояний НС Кохонена	3167	1832	1266	903
Сумма квадратов отклонений UPGMC	921	665	579	440
Сумма квадратов отклонений НС Кохонена	759	549	435	335

Таблица 2.7 (окончание)

Параметр	Количество кластеров			
	6	7	8	9
Время UPGMC, мин:с, мс	00:05.336	00:04.847	00:04.920	00:05.257
Время НС Кохонена, мин:с, мс	00:00.537	00:01.001	00:01.171	00:00.987
Полная сумма внутрикластерных расстояний UPGMC	949	905	825	681
Полная сумма внутрикластерных расстояний НС Кохонена	645	534	459	409
Сумма квадратов отклонений UPGMC	365	333	314	272
Сумма квадратов отклонений НС Кохонена	281	237	210	192

Из рис. 2.12 виден слабый рост времени для нейросети и отсутствие роста для иерархического алгоритма. Если рассмотреть суть последнего, то станет ясно, что подсчет центров кластеров на каждой итерации также приводит к увеличению времени, однако оно весьма незначительно. Для алгоритма Кохонена этот рост вызван введением с увеличением размерности признакового пространства дополнительных входов нейронов. При числе признаков в несколько десятков, возможно, нейросетевой алгоритм уступит UPGMC, однако на практике задачи такой размерности редки.

Как любой итерационный метод, алгоритм кластеризации при помощи НС Кохонена чувствителен к выбору начального приближения. Графики минимальных, средних и максимальных значений времени и качества кластеризации можно видеть на рис. 2.13, 2.14.

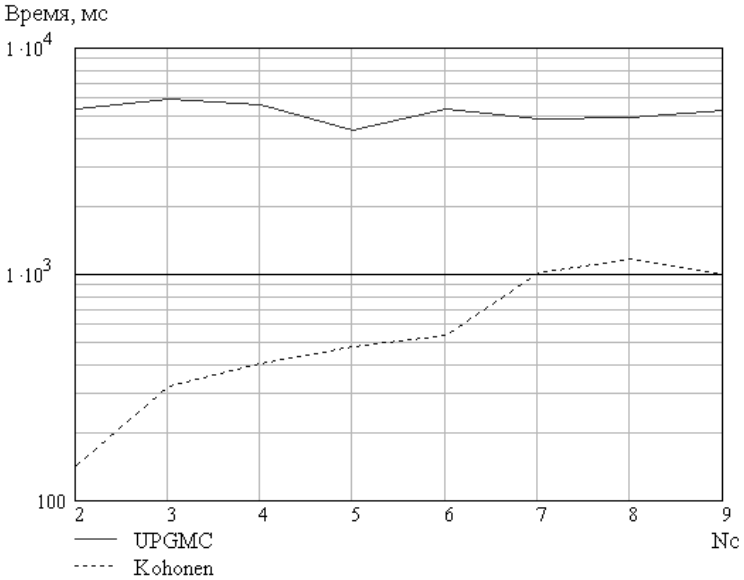


Рис. 2.10. Зависимость времени кластеризации от количества кластеров

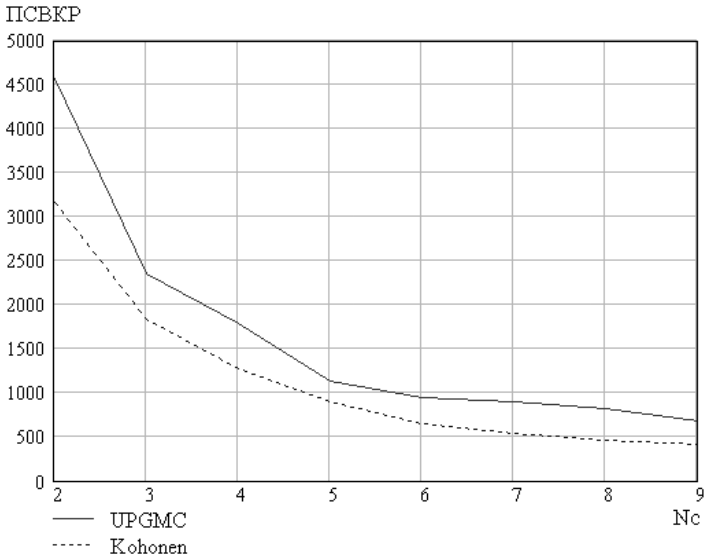


Рис. 2.11. Зависимость значения критерия качества
от количества кластеров (ПСВКР)

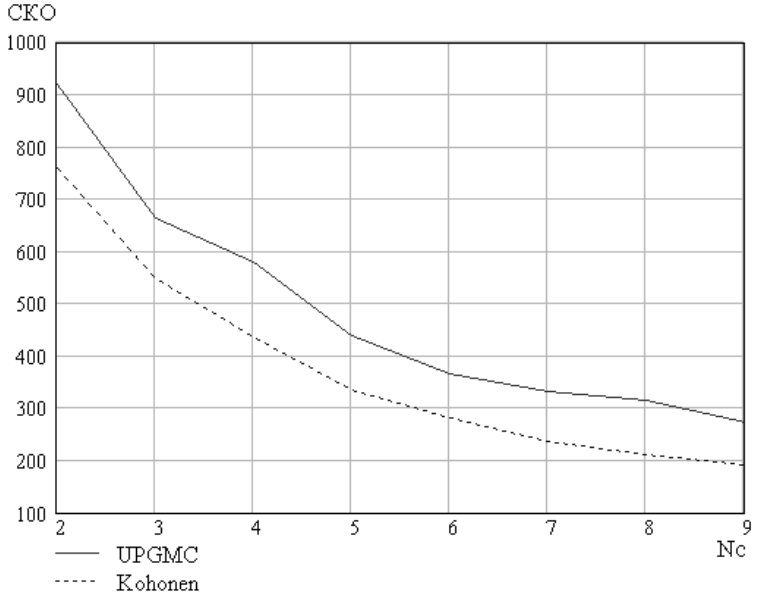


Рис. 2.12. Зависимость значения критерия качества от количества кластеров (СКО)

Таблица 2.8 (начало)

Параметр	Размерность пространства			
	2	3	4	5
Время UPGMC, мин:с, мс	00:04.966	00:04.325	00:05.268	00:05.106
Время НС Кохонена, мин:с, мс	00:00.637	00:00.477	00:01.155	00:00.777
Полная сумма внутрикластерных расстояний UPGMC	1033	1134	1647	4727
Полная сумма внутрикластерных расстояний НС Кохонена	585	903	1402	1692
Сумма квадратов отклонений UPGMC	255	440	821	1478
Сумма квадратов отклонений НС Кохонена	155	335	720	1058

Таблица 2.8 (окончание)

Параметр	Размерность пространства			
	6	7	8	9
Время UPGMC, мин:с, мс	00:05.086	00:05.098	00:04.916	00:04.386
Время НС Кохонена, мин:с, мс	00:00.607	00:00.683	00:00.854	00:01.233
Полная сумма внутрикластерных расстояний UPGMC	5834	2851	2775	6392
Полная сумма внутрикластерных расстояний НС Кохонена	1645	2089	2247	2658
Сумма квадратов отклонений UPGMC	1851	1937	2366	2811
Сумма квадратов отклонений НС Кохонена	1224	1627	2041	2504

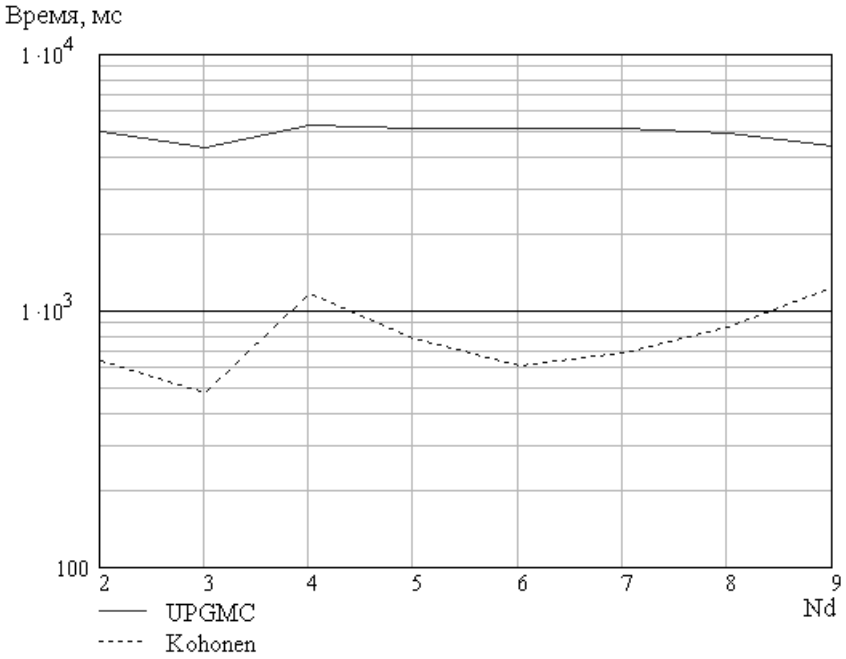


Рис. 2.13. Зависимость времени кластеризации от размерности пространства

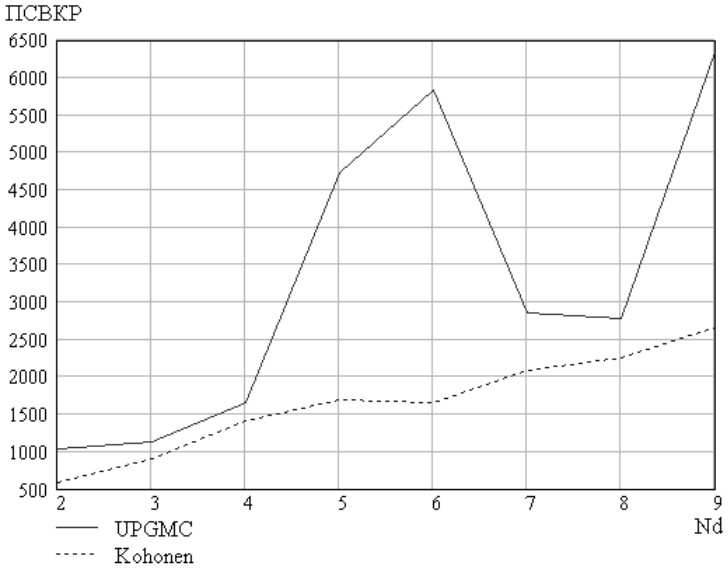


Рис. 2.14. Зависимость значения критерия качества от размерности пространства (ПСВКР)

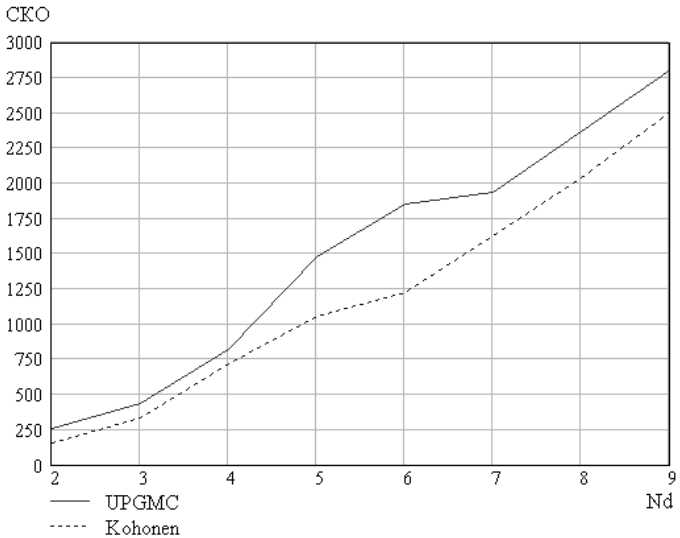


Рис. 2.15. Зависимость значения критерия качества от размерности пространства (СКО)

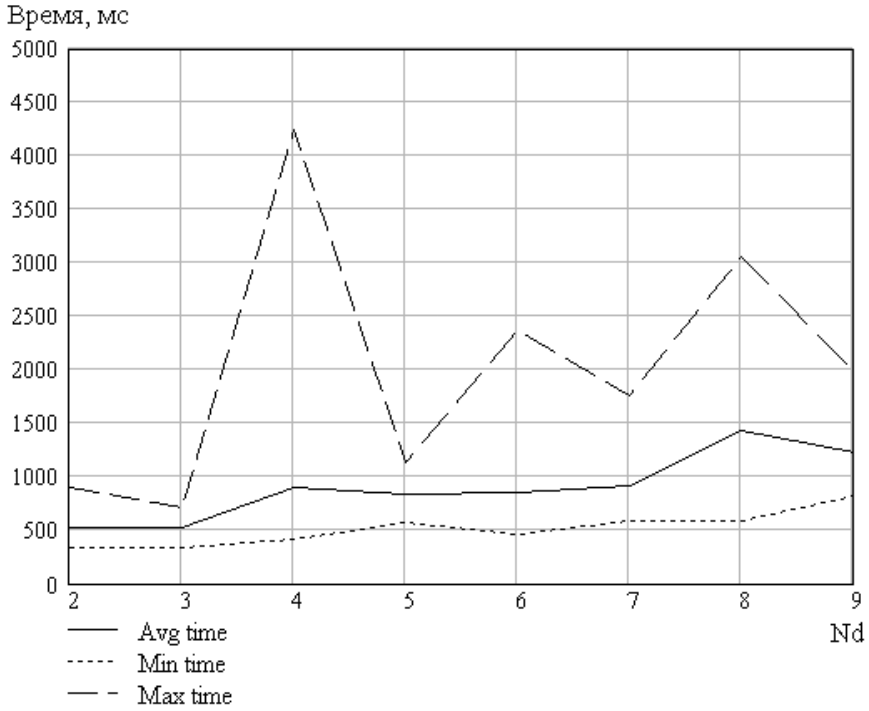


Рис. 2.16. Зависимость диапазона значений времени кластеризации при помощи НС Кохонена от размерности пространства

Для исследования диапазона значений времени и критериев качества кластеризации была проведена серия экспериментов со следующими параметрами: число кластеров $m = 5$, число объектов в кластеризуемом множестве $N = 50$, размерность П-пространства – переменная от 2 до 9. Для каждой совокупности из 50 ОТЕ в пространстве соответствующей размерности было проведено 10 экспериментов. В качестве начального приближения берется случайная выборка из множества кластеризуемых объектов.

Как следует из графиков, разброс времени кластеризации в зависимости от начального приближения может достигать сотен процентов. Разброс значений критериев и не превышает 40 % для критерия ПСВКР, а для критерия СКО – 15 %.

Можно сделать вывод о превосходстве метода кластеризации с помощью НС Кохонена над алгоритмом невзвешенного попарного центроидного усреднения как по времени работы, так и по качеству. Для 100 объектов в кластеризуемом множестве и 5 выделяемых кластеров НС Кохонена находит на 219 % (по критерию полной суммы внутрикластерных расстояний) более качественное решение в 186 раз быстрее.

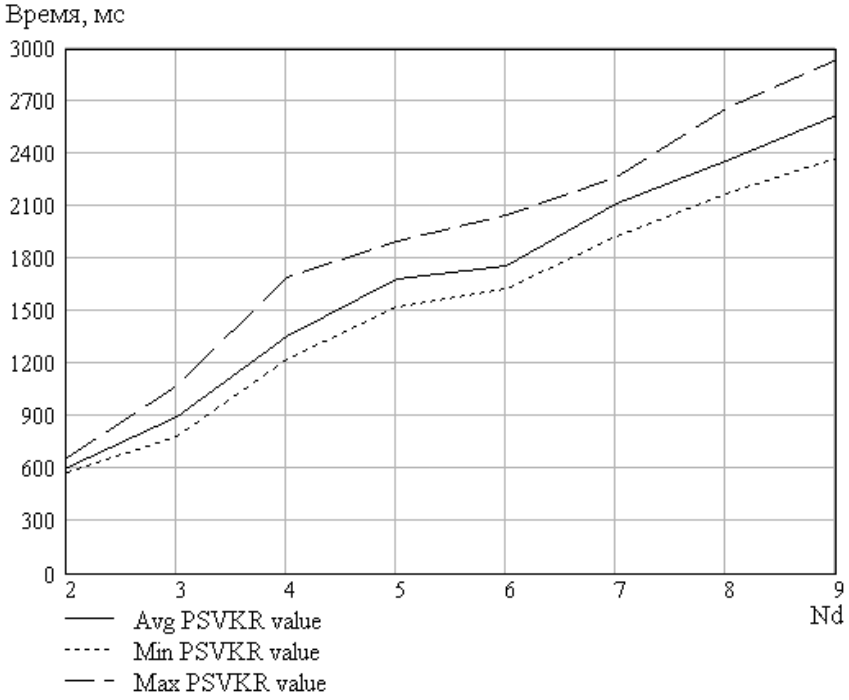


Рис. 2.17. Зависимость диапазона значений критерия ПСВКР при кластеризации с помощью НС Кохонена от размерности пространства

В то же время метод кластеризации при помощи НС Кохонена обладает значительным разбросом времени и качества в зависимости от начального приближения, что является существенным недостатком.

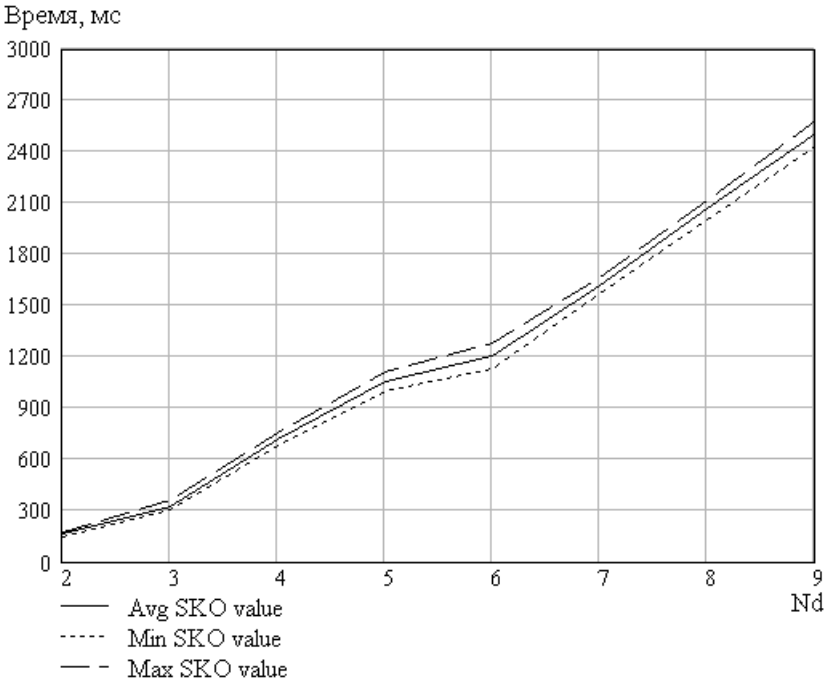


Рис. 2.18. Зависимость диапазона значений критерия СКО при кластеризации с помощью НС Кохонена от размерности пространства

Необходимо подчеркнуть, что сделанные выводы касаются лишь сравнения приближенных методов кластеризации. Недавние сравнительные исследования нейросетевых подходов с модифицированным аддитивным алгоритмом «рекурсивного ветвления» показали в некоторых классах задач кластеризации превосходство алгоритма оптимальной кластеризации над нейросетью Хопфилда по качеству при сравнимых скоростных характеристиках.

Глава 3

НЕЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ

3.1. Постановка задачи

Общая задача нелинейного программирования формулируется так.

Найти $\min f(x_1, \ldots, x_n)$ при ограничениях

$$g_i(x_1, \ldots, x_n) \le b_i, \quad i = \overline{1, m}.$$

Пример. *Задача о размещении.* Эта задача напоминает транспортную задачу.

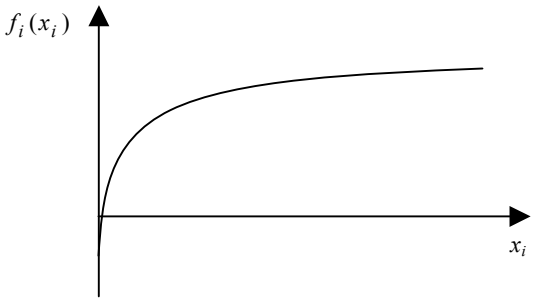


Рис. 3.1. Типичный вид зависимости стоимости производства от объема производства

Пусть имеются n пунктов потребления некоторой продукции, причем b_j $(j = \overline{1, n})$ – объем потребления в j-м пункте; имеются также m пунктов производства. Будем считать, что для каждого i-го пункта производства известна зависимость стоимости производства f_i от объема производства x_i, т.е. функции $f_i(x_i)$, $i = \overline{1, m}$ – это, как правило, нелинейные функции (рис. 3.1).

Из рис. 3.1 следует, что чем больше объем, тем меньше себестоимость единицы продукции.

Наконец, задана матрица транспортных расходов $\|c_{ij}\|$, элементами которой являются стоимости перевозок единицы продукции из i-го пункта производства в j-й пункт потребления.

Требуется найти такие объемы перевозок x_{ij} из i-го в j-й пункт и такие объемы производства $x_i = \sum_{j=1}^{n} x_{ij}$, которые обеспечивают потребности по всем продуктам в j-м пункте назначения ($b_j = \sum_{i=1}^{m} x_{ij}$) и минимизируют суммарные расходы.

В результате возникает следующая задача нелинейного программирования с ограничениями:

найти

$$\min\left\{ f(x_{ij}) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}\, x_{ij} + \sum_{i=1}^{m} f_i(x_i) \right\}$$

при ограничениях

$$\begin{cases} \sum_{i=1}^{m} x_{ij} = b_j, & j = \overline{1,m}; \\ x_i = \sum_{j=1}^{n} x_{ij}, & i = \overline{1,n}; \\ x_{ij} \geq 0. \end{cases}$$

Как видно, последние соотношения, описывающие поставленную задачу, представляют собой совокупность нелинейной целевой функции и линейных ограничений.

3.1.1. Минимизация функции одной переменной

Минимизация функции одной переменной является, как правило, необходимым элементом большинства методов минимизации многомерных функций.

На первый взгляд кажется, что задача минимизации функции одной переменной является, довольно, элементарной. В самом деле, если функция $f(x)$, которую нужно минимизировать на отрезке $[a, b]$, дифференцируема, то достаточно найти нули производной, присоединить к ним концы отрезка, выделить из этих точек локальные минимумы и, наконец, среди последних найти ту точку, в которой достигается абсолютный (глобальный) минимум. Этот метод является классическим методом. Он основан на дифференциальном исчислении и довольно подробно описан в литературе.

Однако для широкого класса функций эта задача не так уж проста, и классический метод имеет весьма ограниченное применение, поскольку задача решения уравнения $f'(x) = 0$ может оказаться весьма сложной. К тому же в практических задачах часто неизвестно, является ли $f(x)$ дифференцируемой функцией. Кроме того, во многих практических задачах часто невозможно найти явную зависимость $f(x)$. Поэтому существенное значение приобретают методы минимизации, не требующие вычисления производной.

Существование локальных минимумов функции $f(x)$, почти всегда затрудняет поиск глобального минимума. Поэтому многие приближенные методы минимизации применимы только тогда, когда любой локальный минимум функции $f(x)$ является одновременно и глобальным. Это дает *гарантию* сходимости методов. Если же такие сведения о функции не известны, то методы применять можно, но без гарантии сходимости.

Одним из классов функций, удовлетворяющих указанному условию, является класс *унимодальных* функций.

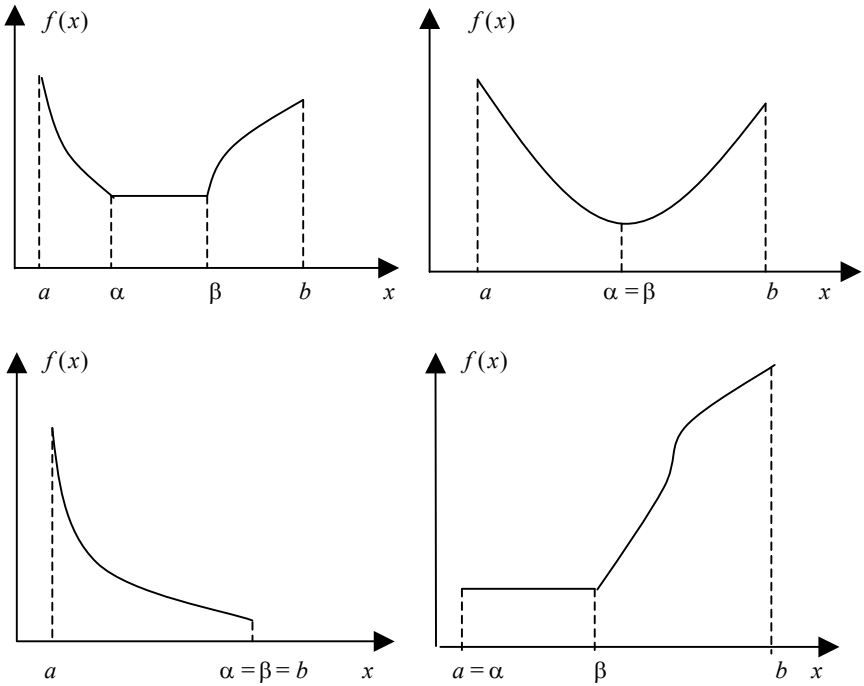
Определение. Функция $f(x)$ называется унимодальной на отрезке $[a, b]$, если она непрерывна на $[a, b]$ и существуют такие числа α и β $(a \le \alpha \le \beta \le b)$, что:

1) на отрезке $[a, \alpha]$ при $a < \alpha$, функция $f(x)$ монотонно убывает;

2) на отрезке $[\beta, b]$ при $\beta < b$, функция $f(x)$ монотонно возрастает;

3) $f(x) = f^* = \min\limits_{[a,b]} f(x)$ при $x \in [\alpha, \beta]$, т.е. данная функция имеет минимум.

Далее приведены некоторые варианты унимодальных функций.



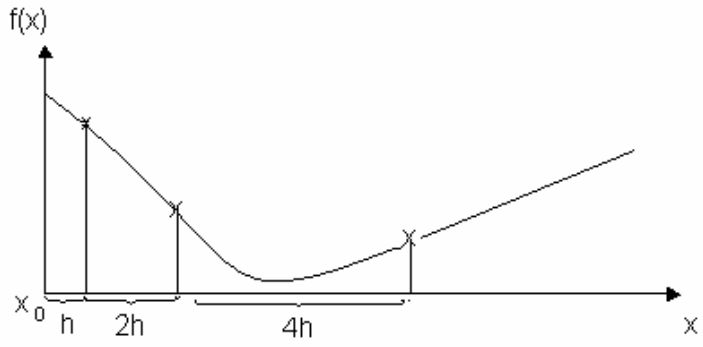
Прежде чем приступить к процедуре минимизации, следует по возможности установить принадлежность целевой функции классу, для которого гарантирована сходимость процесса. Во многих случая существует дополнительная информация, позволяющая установить необходимую принадлежность. Часто такая информация связана с физическим содержанием той реальной задачи, для которой построена исследуемая математическая модель.

Заметим, что предположение об унимодальности функции в окрестности точки экстремума весьма естественно. Получение информации о таком промежутке является важным предварительным этапом процедуры минимизации.

3.1.2. Поиск отрезка, содержащего точку минимума

Сущность поиска отражена на рисунке, изображенном ниже, и заключается в том, что, начиная с некоторой точки, осуществляют-

ся возрастающие по величине шаги до тех пор, пока не будет пройдена точка минимума функции $f(x)$.



Алгоритм.

1. Положить $k = 1$.

2. Выбрать точку x_0 и определить направление убывания функции $f(x)$.

Для этого положить шаг $h > 0$ и вычислить значение функции $f(x_0 + h)$.

Если $f(x_0 + h) < f(x_0)$, то перейти к п. 3, положив $x_1 = x_0 + h$.

Если $f(x_0 + h) \geq f(x_0)$, то положить $h = -h$ и вычислить $f(x_0 + h)$.

Если $f(x_0 + h) < f(x_0)$, то перейти к п. 3, положив $x_1 = x_0 + h$.

Если $f(x_0 + h) \geq f(x_0)$, то положить $h = h/2$ и повторить п. 2.

3. Удвоить шаг, т.е. положить $h = 2h$ и вычислить $x_{k+1} = x_k + h$.

4. Вычислить $f(x_{k+1})$.

Если $f(x_{k+1}) < f(x_k)$, то положить $k = k + 1$ и перейти к п. 3.

Если $f(x_{k+1}) \geq f(x_k)$, то поиск прекратить и в качестве отрезка, содержащего точку минимума, выбрать отрезок $[x_{k-1}, x_{k+1}]$.

3.2. Методы одномерной минимизации

Рассмотрим с общих позиций ряд методов, позволяющих найти минимум функции $f(x)$ при ограничениях $x \in [a, b]$.

3.2.1. Методы нахождения глобального минимума унимодальных функций

3.2.1.1. Прямые методы минимизации

Данные методы основаны на вычислении значений функции $f(x)$ в некоторых точках; они не используют значений производных оптимизируемой функции.

Метод перебора – простейший, но редко используемый в серьезных задачах.

Согласно этому методу отрезок $[a, b]$ делится на n равных частей точками $x_i = a + i\left(\dfrac{b-a}{n}\right)$, $i = \overline{1, n}$. Вычисляются значения функции в этих точках, и путем сравнения определяется точка минимума x_m:

$$f(x_m) = \min_{0 \le i \le n} f(x_i).$$

В качестве точки экстремума полагается: $x^* \approx x_m$, $f^* \approx f(x_m)$. При этом погрешность ε в определении точки минимума x^*, очевидно, равна отрезку деления, т.е. $\varepsilon = \left(\dfrac{b-a}{n}\right)$.

Метод перебора, предполагающий предварительный выбор точек x_i, называется также *пассивной стратегией* поиска точки минимума x^*. На практике точки x_i выбираются заранее, когда удобно провести $(n+1)$ независимый эксперимент по измерению значений функции $f(x)$, а последовательное измерение этих значений трудоемко или невозможно по каким-либо причинам.

Использование информации о функции $f(x)$ для выбора очередной точки x_i измерения (вычисления) функции $f(x)$ уже полученной в предыдущих экспериментах, однако, приводит к более эффективному поиску точки x^*.

Методы минимизации, в которых точки x_i определяются в процессе поиска с помощью найденных ранее значений $f(x)$, называ-

ются *последовательными* методами. Практически все рассматриваемые ниже методы относятся к этому классу.

Метод золотого сечения. Метод состоит в том, что исходный отрезок $[a, b]$ уменьшается по определенному закону, постепенно стягиваясь к точке минимума (рис. 3.2). Сокращение отрезка происходит за счет его деления и отбрасывания частей, не содержащих экстремальной точки. Отрезок делится в отношении «золотого сечения» (отсюда название).

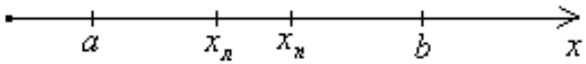


Рис. 3.2. Схема разбиения отрезка поиска экстремума

Прежде всего, при реализации этого метода необходимо задать желаемую точность ε вычисления точки x^*. Количество вычислений функции заранее не задается и полностью определяется точностью ε.

Метод Фибоначчи. Этот метод почти полностью совпадает с методом золотого сечения, но есть два отличия:

1) отрезок делится с помощью чисел Фибоначчи –

$$\gamma_0 = \gamma_1 = 1; \quad \gamma_n = \gamma_{n-1} + \gamma_{n-2}, \, n \geq 2,$$

в результате получаем следующую последовательность чисел:

$$1, 1, 2, 3, 5, 8, 13, 21, \ldots;$$

2) требуется до начала работы метода задать число шагов n (так как на первой итерации отрезок делится пропорционально $\frac{\gamma_{n-2}}{\gamma_n}$ и $\frac{\gamma_{n-1}}{\gamma_n}$, а величина n изменяется в обратную сторону к нулю).

На практике нередко встречаются функции, нахождение значений которых в каждой точке связано с большим объемом вычислений или дорогостоящими экспериментами. В таких случаях число n, определяющее количество вычислений, может быть заранее жестко задано и превышение его недопустимо.

Следует отметить, что как в методе «золотого сечения» так и в методе с использованием чисел Фибоначчи, легко аналитически рассчитать, количество вычислений функции на отрезке $[a, b]$, если желаемая точность вычисления $x^* - \varepsilon$, и наоборот, какая будет точность при n вычислениях функции.

Метод золотого сечения имеет несколько меньшую скорость сходимости, чем метод Фибоначчи.

К недостаткам последних из рассмотренных методов можно отнести то, что в результате округлений в процессе решения задачи (на ЭВМ) накапливаются ошибки в вычислении точек деления отрезка. Методы очень чувствительны к этому и могут даже расходиться.

В этом отношении более предпочтительными оказываются методы, основанные на *полиномиальной аппроксимации.*

Идея метода такова: если на отрезке $[a, b]$ с внутренней точкой минимума есть основание полагать, что функция $f(x)$ достаточно хорошо аппроксимируется многочленом (2-й, 3-й степени), то за приближенное значение x^* целесообразно взять точку минимума этого многочлена.

3.2.1.2. Методы минимизации, основанные на использовании производных функции

Во многих случаях эти методы обладают более высокой скоростью сходимости, по сравнению с прямыми методами поиска экстремума.

Метод дихотомии. При использовании этого метода вычисляется середина отрезка, в которой находится производная функции $f(x)$; в зависимости от знака производной отбрасывается одна из половин отрезка. За счет чего отрезок стягивается.

Скорость сходимости данного метода выше, чем у методов «золотого сечения» и с использованием чисел Фибоначчи.

Метод касательных. Данный метод используется только для выпуклых функций и имеет простой геометрический смысл: находят абсциссу c точки пересечения касательных к графику функции

$f(x)$, проведенных в граничных точках отрезка (рис. 3.3), для этого нужны производные в этих точках.

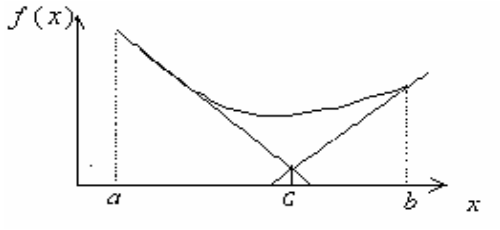


Рис. 3.3. Определение точки *c*

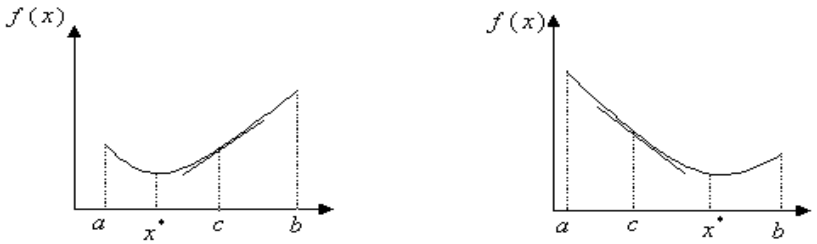


Рис. 3.4. Возможные наклоны касательной в точке *c*

Затем анализируется знак производной в этой точке *c*. При этом возможны следующие ситуации (рис. 3.4).

Если $f'(c) > 0$, то в качестве нового отрезка берется отрезок $[c, b]$, если $f'(c) < 0$, то в качестве нового отрезка берется отрезок $[a, c]$.

Таким образом, отрезок уменьшается от итерации к итерации.

Метод Ньютона. Данный метод использует не только первую, но и вторую производные функции. При определенных условиях он обеспечивает значительно более высокую скорость сходимости к точке минимума, чем рассмотренные выше методы минимизации.

Для реализации этого метода функция $f(x)$ должна быть выпуклой, дважды дифференцируемой функцией.

Прежде всего, выбирается начальное приближение x_0 и строится последовательность:

$$x_n = x_{n-1} - \frac{f'(x_{n-1})}{f''(x_{n-1})}, \quad n = 1, 2, \ldots$$

Вычисления заканчиваются, например, если $|f'(x)| \le \varepsilon$.

При неудачном выборе x_0 метод может расходиться.

3.2.2. Методы поиска глобального минимума многоэкстремальных функций

Во многих практических случаях достаточно сложно, а иногда невозможно, установить является ли функция $f(x)$ унимодальной. В этом случае наиболее известным методом поиска глобального минимума на фоне множества локальных минимумов является метод ломаных.

Метод ломаных. Этот метод может быть использован для поиска глобального минимума функции, удовлетворяющей условию Липшица на отрезке $[a, b]$. Функция $f(x)$ удовлетворяет условию Липшица на отрезке $[a, b]$, если существует число $L > 0$ (константа Липшица), такое, что

$$|f(x') - f(x'')| \le L\,|x' - x''| \quad \text{для всех} \quad x', x'' \in [a, b].$$

Геометрический смысл метода ломаных состоит в построении последовательности ломаных, приближающихся к графику функции $f(x)$ снизу и имеющих угловые коэффициенты всех звеньев $\pm L$.

Метод ломаных невозможно реализовать без знания константы Липшица L. Ее оценку получить можно, но иногда это представляет значительные трудности.

3.2.3. Методы минимизации унимодальных функций

Рассмотрим ряд методов, широко используемых в практике поиска экстремума нелинейных функций, подробнее.

3.2.3.1. Метод золотого сечения

Как известно, «золотым сечением» отрезка называется деление отрезка на две неравные части такие, что выполняется соотношение:

$$\frac{\text{весь отрезок}}{\text{большая часть}} = \frac{\text{большая часть}}{\text{меньшая часть}}.$$

Нетрудно проверить, что золотое сечение отрезка $[a, b]$ производится двумя точками:

$$x_1 = a + r_1(b - a),$$

$$x_2 = a + r_2(b - a),$$

где

$$r_1 = \frac{3 - \sqrt{5}}{2} = 0{,}381966...,$$

$$r_2 = \frac{\sqrt{5} - 1}{2} = 0{,}618033...$$

На рис. 3.5 изображен пример деления отрезка $[a, b]$ в пропорции «золотого сечения».

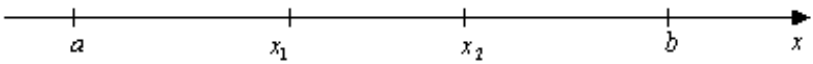


Рис. 3.5. Пример деления отрезка [*a*, *b*] в пропорциях «золотого сечения»

Точки x_1 и x_2 расположены симметрично относительно середины отрезка, при этом выполняются соотношения:

$$r_1 + r_2 = 1, \quad r_1 = r_2^2.$$

Замечательно здесь то, что точка x_1, в свою очередь, производит «золотое сечение» отрезка $[a, x_2]$:

$$\frac{x_2 - a}{x_1 - a} = \frac{x_1 - a}{x_2 - x_1}.$$

Аналогично, точка x_2 производит золотое сечение отрезка $[x_1, b]$.

Опираясь на это свойство «золотого сечения» можно предложить следующий метод минимизации унимодальной функции $f(x)$ на отрезке $[a, b]$.

Идея метода такова. На отрезке $[a, b]$ возьмем точки x_1 и x_2, производящие золотое сечение отрезка, и вычислим значения функций $f(x_1)$ и $f(x_2)$ в этих точках (рис. 3.6).

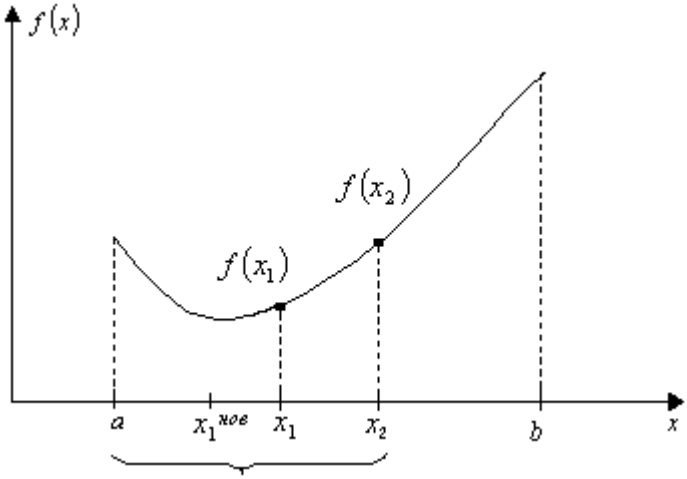


Рис. 3.6. Выбор отрезка, содержащего точку минимума

Если $f(x_1) \le f(x_2)$, то при $x > x_2$ функция не может убывать, т.е. точка минимума лежит на отрезке $[a, x_2]$. Если же $f(x_1) > f(x_2)$, то наоборот минимум функции лежит на отрезке $[x_1, b]$.

Итак, отрезок, на котором ищется минимум, уменьшился. Процесс повторяется сначала. При этом весьма важно то, что внутри отрезка $[a, x_2]$ (это справедливо для рассматриваемого примера) содержится точка x_1 с вычисленным значением $f(x_1)$, которая производит «золотое сечение» отрезка $[a, x_2]$. Следовательно, на

данном шаге достаточно вычислить лишь одно значение $f(x)$ во второй, новой точке, производящей золотое сечение.

Аналогично в случае, если выбран отрезок $[x_1, b]$.

Алгоритм метода золотого сечения.

1. Вычислить длину отрезка $[a, b] - \Delta = b - a$.

Взять внутри отрезка две точки:

$$x_л = a + r_1\Delta, \quad x_п = a + (1 - r_1)\Delta.$$

2. Вычислить $f(x_л)$ и $f(x_п)$.

3. Если $f(x_л) \le f(x_п)$ перейти к шагу 4, если $f(x_л) > f(x_п)$ – к шагу 5.

4. Положить $a = a$, $b = x_п$. Вычислить $\Delta = b - a$. Если $\Delta \le \varepsilon$, то процесс прекратить. В противном случае положить:

$x_п = x_л$, $\quad f(x_п)$ – известно;

$x_л = a + r_1\Delta$, $\quad f(x_л)$ – вычислить.

Перейти к шагу 3.

5. Положить $b = b$, $a = x_л$. Вычислить $\Delta = b - a$. Если $\Delta \le \varepsilon$, то процесс прекратить. В противном случае положить

$x_л = x_п$, $\quad f(x_л)$ – известно;

$x_п = a + (1 - r_1)\Delta$, $\quad f(x_п)$ – вычислить.

Перейти к шагу 3.

Блок-схема алгоритма изображена на рис. 3.7.

Замечания. 1. После каждого шага длина отрезка уменьшается в $1/r_2$ раз. После n итераций длина отрезка составляет $\Delta = r_2^n(b - a)$.

2. Численная реализация метода обладает существенным недостатком. Так, приближенные значения чисел r_1 и r_2 порождают накопление ошибок, которое довольно быстро может привести к нарушению пропорций отрезков в смысле отношения «золотого сечения» и, как следствие, к нарушению условий вложенности отрезков и тем самым к расходимости процесса. Таким образом, метод *не является устойчивым* по отношению к ошибкам в определении параметров r_1 и r_2.

Рекомендации:

с максимально возможной для используемой ЭВМ точностью определять эти параметры и точки деления отрезка;

при определенных условиях можно воспользоваться модификацией метода, делающей процесс устойчивым.

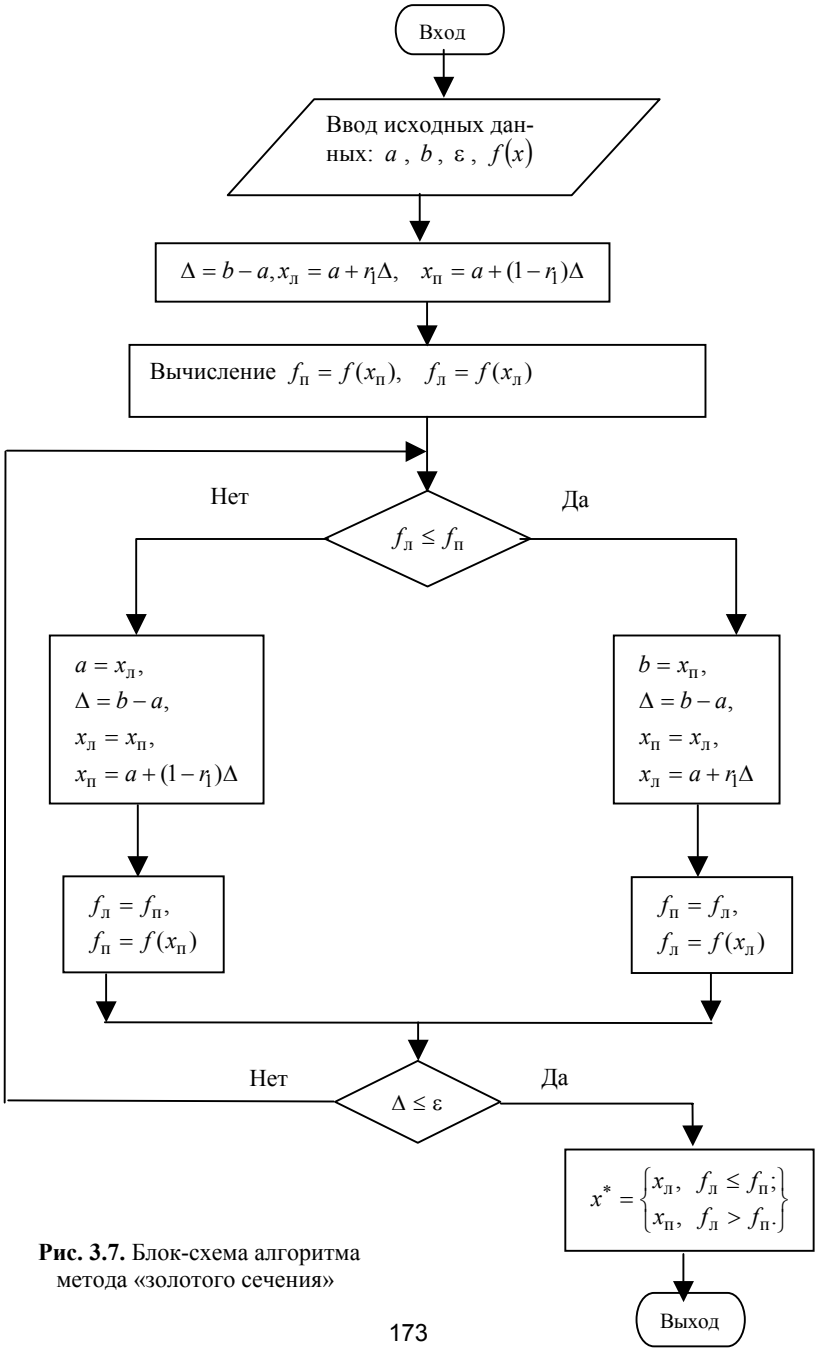


Рис. 3.7. Блок-схема алгоритма метода «золотого сечения»

В качестве примеров использования метода «золотого сечения» рассмотрим следующие задачи.

Задача 1. Сколько шагов n метода «золотого сечения» обеспечивают заданную точность ε.

Решение. Как отмечалось выше, условием достижения точности является выполнение неравенства

$$\Delta_n \le \varepsilon .$$

Поскольку

$$\Delta_n = r_2^n (b-a) ,$$

то отсюда

$$r_2^n (b-a) \le \varepsilon ,$$

и, как следствие:

$$r_2^n \le \frac{\varepsilon}{b-a} ,$$

$$\ln\left(r_2^n\right) \le \ln\left(\frac{\varepsilon}{b-a}\right),$$

$$n \cdot \ln r_2 \le \ln\left(\frac{\varepsilon}{b-a}\right),$$

так как $r_2 < 1$, то $\ln r_2 < 0$. Следовательно,

$$n \ge \ln\left(\frac{\varepsilon}{b-a}\right) \Big/ \ln r_2 \approx -2{,}1 \ln\left(\frac{\varepsilon}{b-a}\right).$$

Так как

$$\ln\left(\frac{\varepsilon}{b-a}\right) < 0, \quad \text{то} \quad n \approx 2 .$$

Задача 2. Найти методом золотого сечения точку минимума функции $f(x) = x^2 - x$ на отрезке [0, 2] с точностью $\varepsilon = 0{,}1$.

Решение. Очевидно, данная задача может быть легко решена классическим методом. Взяв производную и разрешив относительно x^* полученное уравнение, т.е.:

$$\frac{df}{dx} = 2x - 1 = 0 \to x^* = 0{,}5, \quad f^* = -0{,}25.$$

Решим задачу с помощью метода «золотого сечения». В соответствии с рассматриваемым методом, параметры $r_1 = 0{,}382$, $r_2 = 0{,}618$. Учитывая это, результаты пошаговых расчетов можно свести в табл. 3.1.

Таблица 3.1

n	Δ_n	b_n	a_n	$x^{л}$	$x^{п}$	$f(x^{л})$	$f(x^{п})$
1	2	0	2	0,764	1,236	–0,180	0,26
2	1,236	0	1,236	0,472	0,764	–0,241	–0,18
3	0,764	0	0,764	0,292	0,472	–0,207	–0,24
4	0,472	0,292	0,764	0,472	0,584	–0,249	–0,2
5	0,292	0,292	0,584	0,404	0,472	–0,241	–0,2
6	0,18	0,404	0,584	0,472	0,515	–0,249216	–0,2
7	0,112	0,472	0,584	0,515	0,541	–0,249775	–0,2

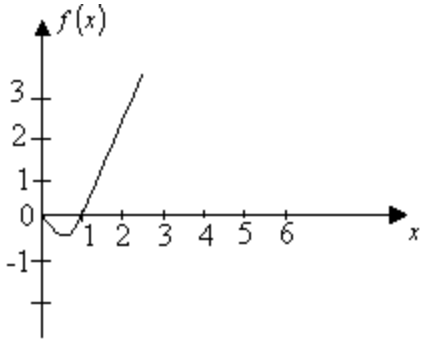


Рис. 3.8. Графическое изображение функции (задача 2)

Как следует из таблицы, $x^* \approx 0{,}515$, $f^* \approx -0{,}249775$, что совпадает с результатами, полученными с помощью производной. График, соответствующий исследуемой функции, приведен на рис. 3.8.

3.2.3.2. Метод дихотомии

В этом методе используются производные целевой функции, поэтому он применим только в том случае, когда производная функции вычисляется достаточно просто.

В основу метода положен тот очевидный факт, что производная унимодальной функции меняет знак на отрезке поиска только один раз – в точке минимума (рис. 3.9).

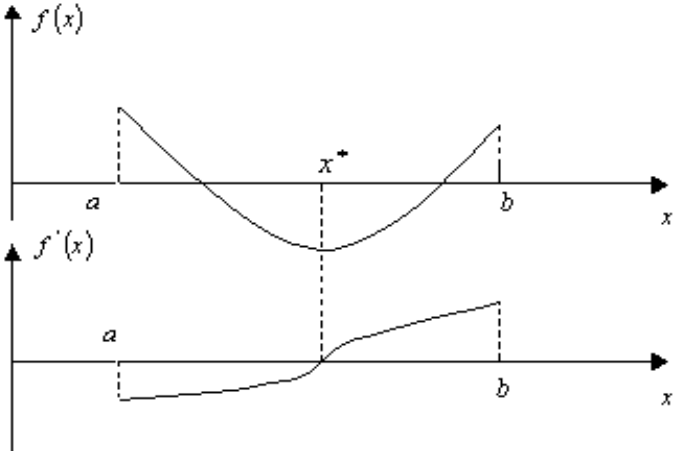


Рис. 3.9. Характер производной унимодальной функции

Идея метода такова: на концах рассматриваемого отрезка вычисляют производные целевой функции, отрезок делят пополам и вычисляют производную в средней точке. Для следующей итерации выбирают тот отрезок из двух получившихся, на концах которого знаки производной различны, так как именно этот отрезок содержит точку экстремума.

Это общая идея метода, которая фактически сводит рассматриваемый метод к поиску корня уравнения $f'(x)=0$ на отрезке $[a,b]$ известным *методом дихотомии*. Эта идея работает, когда минимум функции $f(x)$ является внутренней точкой отрезка $[a,b]$. Если x^* лежит на границе, то ситуация несколько иная, однако в алгоритме, который будет изложен ниже, эта проблема не затрагивается.

Метод дихотомии дает более быстрое уменьшение отрезка, чем метод «золотого сечения». После n итераций длина отрезка составляет $0{,}5^n(b-a)$, тогда как при использовании метода «золотого сечения» длина отрезка будет – $r_2^n(b-a)$, $r_2 \approx 0{,}61$. Приведем

таблицу коэффициентов уменьшения исходного отрезка $[a, b]$ после n итераций для методов дихотомии и «золотого сечения» (табл. 3.2).

Таблица 3.2

n	1	2	3	…	10
r_2^n	0,61	0,38	0,23	...	0,0080
$0{,}5^n$	0,5	0,25	0,125	…	0,001

Основной недостаток метода заключается в вычислительных трудностях, связанных с определением производных функции $f(x)$.

Алгоритм метода дихотомии (поиск минимума).

1. Задан отрезок $[a, b]$. Вычислить длину отрезка $\Delta = b - a$.

2. Вычислить среднюю точку отрезка $x_{\text{c}} = a + 0{,}5\Delta$ и значение производной $\dfrac{df}{dx}(x_{\text{c}})$.

Если производная равна нулю (с точностью ε_1), то вычисления прекращаются – $x^* \approx x_{\text{c}}$.

Если производная меньше нуля, перейти к шагу 3.

Если производная больше нуля – к шагу 4.

3. Положить $a = x_{\text{c}}$. Вычислить $\Delta = b - a$.

Если $\Delta \le \varepsilon_2$, то закончить вычисления, положить $x^* \approx x_{\text{c}}$.

В противном случае – перейти к шагу 2.

4. Положить $b = x_{\text{c}}$. Вычислить $\Delta = b - a$.

Если $\Delta \le \varepsilon_2$, то закончить вычисления, положить $x^* \approx x_{\text{c}}$.

В противном случае – перейти к шагу 2.

Блок-схема алгоритма приведена на рис. 3.10.

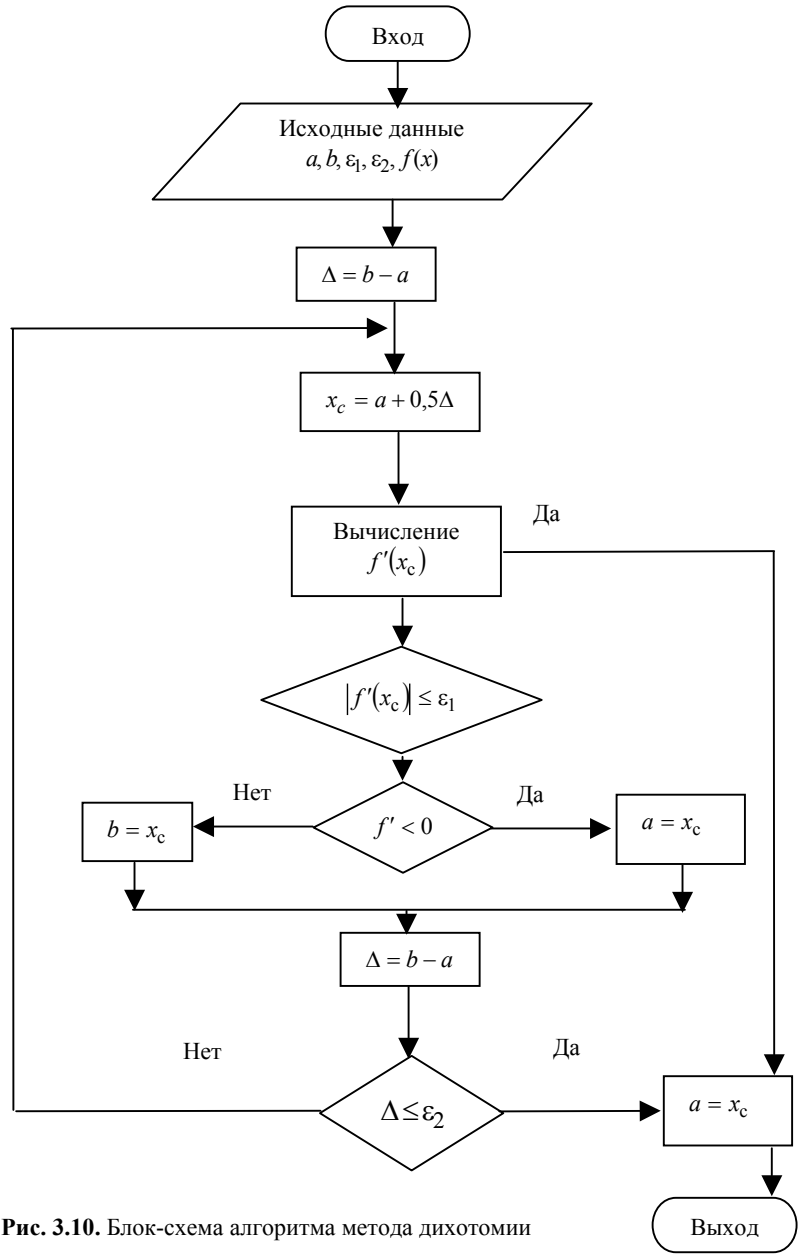


Рис. 3.10. Блок-схема алгоритма метода дихотомии

3.2.3.3. Метод парабол (метод полиномиальной аппроксимации)

Идея метода весьма проста: если на отрезке $[a, b]$, внутри которого лежит точка минимума x^*, функция $f(x)$ достаточно хорошо аппроксимируется многочленом второй степени, то за приближенное значение x^* целесообразно взять точку минимума этого многочлена. В случае когда функция гладкая и унимодальная, можно предполагать, что в окрестности точки x^* она весьма близка к параболе. Ввиду этого метод парабол целесообразно применять после того как найден отрезок достаточно малой длины, содержащий x^* (например, после нескольких шагов по методу золотого сечения).

Пусть известны три значения $x_1 < x_2 < x_3$, таких, что $f(x_1) > f(x_2) < f(x_3)$. В этом случае $x^* \in [x_1, x_3]$. Построим многочлен второго порядка, который проходит через три данные точки (рис. 3.11).

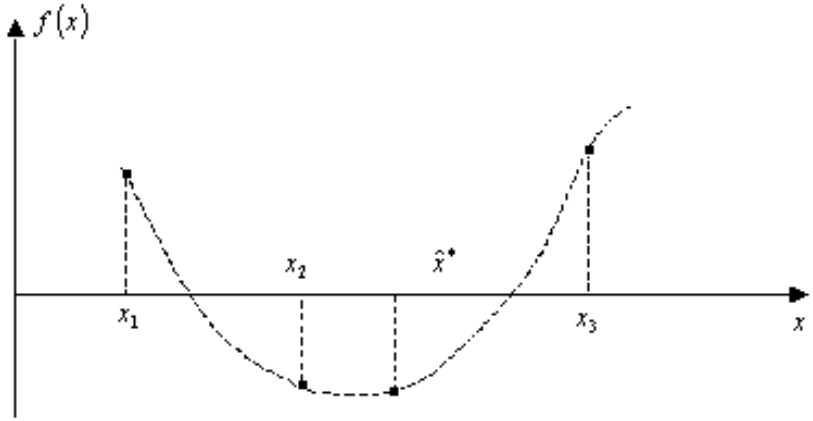


Рис. 3.11. Графическое представление функции $f(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2$

Коэффициенты $\alpha_0, \alpha_1, \alpha_2$ определяются из системы уравнений:

$$\begin{cases} f(x_1) = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_1^2, \\ f(x_2) = \alpha_0 + \alpha_1 x_2 + \alpha_2 x_2^2, \\ f(x_3) = \alpha_0 + \alpha_1 x_3 + \alpha_2 x_3^2. \end{cases}$$

Минимум полинома определяется из условия

$$\left.\frac{df(x)}{dx}\right|_{x=\hat{x}^*} = \alpha_1 + 2\alpha_2 \hat{x}^* = 0 .$$

Откуда получаем,

$$\hat{x}^* = -\frac{\alpha_1}{2\alpha_2} .$$

Найдя α_1 и α_2 из линейной системы уравнений, получим

$$\hat{x}^* = -\frac{1}{2}\begin{vmatrix} 1 & f(x_1) & x_1^2 \\ 1 & f(x_2) & x_2^2 \\ 1 & f(x_3) & x_3^2 \end{vmatrix} \Bigg/ \begin{vmatrix} 1 & x_1 & f(x_1) \\ 1 & x_2 & f(x_2) \\ 1 & x_3 & f(x_3) \end{vmatrix} = \qquad (*)$$

$$= \frac{1}{2}\frac{(x_2^2 - x_3^2)f(x_1) + (x_3^2 - x_1^2)f(x_2) + (x_1^2 - x_2^2)f(x_3)}{(x_2 - x_3)f(x_1) + (x_3 - x_1)f(x_2) + (x_1 - x_2)f(x_3)} .$$

Алгоритм метода парабол.

1. Найти тройку чисел $x_1 < x_2 < x_3$, таких, что $f(x_1) > f(x_2) < f(x_3)$ (это можно сделать, например, используя алгоритм поиска отрезка, содержащего точку минимума).

2. Вычислить оценку $\hat{x}^*$ по формуле (*).

3. Если $\left|\hat{x}^* - x_2\right| \le \varepsilon$, закончить процесс, положив $x^* = \hat{x}^*$.

В противном случае – вычислить $f(\hat{x}^*)$.

4. Из чисел $x_1, x_2, x_3, \hat{x}^*$ выбрать необходимую тройку чисел и перейти к шагу 2.

3.3. Минимизация функций без ограничений (безусловная минимизация)

В настоящем разделе будут изучены методы нахождения минимума функции многих переменных $f(x_1, \ldots, x_n)$ по всему пространству R^n, т.е. при условии, что при поиске экстремума каждая из переменных x_i может принимать значения от $-\infty$ до $+\infty$. Допустим, что $f(\bar{x})$ обладает единственным минимумом и что все частные производные, по крайней мере первого порядка, существуют при любых значениях $\bar{x}$.

Прежде всего, рассмотрим методы отыскания экстремума функций без ограничений, а затем перейдем к более трудной задаче отыскания экстремума с ограничениями. Это необходимо потому, что задача оптимизации с ограничениями часто решается путем преобразования ее к задаче оптимизации без ограничений.

Все методы безусловной минимизации сводятся к нахождению последовательности точек $\bar{x}_0, \bar{x}_1, \ldots, \bar{x}_n$, значения функции в которых убывают:

$$f(\bar{x}_0) > f(\bar{x}_1) > \cdots > f(\bar{x}_n). \qquad (3.1)$$

Эти методы называются *методами спуска* (или *релаксационными методами*).

При получении таких последовательностей точек в задачах с ограничениями, прежде всего, необходимо выбрать начальную точку $\bar{x}_0$, удовлетворяющую всем ограничениям.

В задаче без ограничений можно в качестве $\bar{x}_0$ взять любую точку. При этом, конечно, нужно использовать любую имеющуюся информацию о поведении функции $f(\bar{x})$ с тем, чтобы выбрать $\bar{x}_0$ не слишком далеко от точки минимума.

После того, как какая-то точка выбрана, необходимо:

1) выбрать направление, вдоль которого предполагается расположить следующую точку;

2) решить, какой величины шаг должен быть сделан в выбранном направлении.

При любом методе спуска последовательность точек выглядит так:

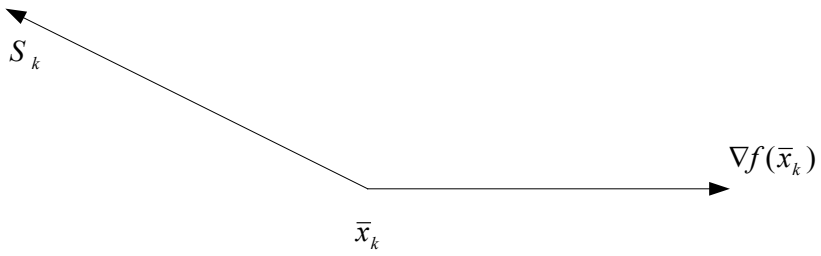
$$\bar{x}_{k+1} = \bar{x}_k + \lambda_k S_k\,, \tag{3.2}$$

где единичный вектор S_k задает направление, а λ_k – величину шага. Изменяя процедуру выбора S_k и λ_k, можно получать различные методы спуска.

Для задач без ограничений любое направление является возможным, однако не все направления приемлемы. *Приемлемым* называется такое направление S_k, для которого выполняется условие

$$\left\langle \nabla f(\bar{x}_k), S_k \right\rangle < 0\,, \tag{3.3}$$

геометрически представленное на следующем рисунке.



Действительно, для достаточно малых λ_k можно написать разложение функции в ряд Тейлора

$$f(\bar{x}_k + \lambda S_k) \approx f(\bar{x}_k) + \underbrace{\lambda\left\langle \nabla f(\bar{x}_k), S_k \right\rangle}_{\substack{\text{если этот член} < 0\text{, то} \\ f(\bar{x}_k + \lambda S_k) < f(\bar{x}_k)}}\,. \tag{3.4}$$

Все методы спуска можно разбить на две группы:

1) *прямые* методы или методы *нулевого порядка* – методы, использующие только значения функций;

2) методы *первого порядка* – методы, использующие, кроме того, первые производные;

3) методы *второго порядка* – методы, использующие кроме первых производных еще и вторые производные.

Производные могут вычисляться аналитически или численно.

Вообще говоря, методы третьей группы при наименьшем числе шагов приводят к точкам, достаточно близким к точкам минимума. Это, однако, не означает, что они являются наиболее эффективны-

ми методами в отношении расхода машинного времени, необходимого для решения задачи. Иногда функция $f(\bar{x})$ представляет собой настолько сложную функцию, что ее первая или вторая производные не могут быть получены аналитически, а их численные приближения оказываются очень грубыми. Кроме того, вычисление этих производных может потребовать больше машинного времени, чем вычисление значений функции в необходимом для метода нулевого порядка числе точек.

Таким образом, невозможно выделить какой-либо метод, пригодный в любом случае. Для отыскания метода, наиболее пригодного для оптимизации заданной функции, необходимы опыт, интуиция и, может быть, предварительные исследования.

3.3.1. Методы нулевого порядка

3.3.1.1. Метод покоординатного спуска

Наиболее простым способом определения направления спуска является выбор в качестве $\hat{S}_k$ одного из координатных векторов $\bar{e}_1, \bar{e}_2, \ldots, \bar{e}_n$.

Через $\bar{e}_i = (0, \ldots, 0, \underset{i}{1}, 0, \ldots, \underset{n}{0})$ $(i = \overline{1, n})$ обозначим единичный вектор, у которого *i*-я компонента равна 1, а остальные – нулю. Иначе говоря, в методе покоординатного спуска на каждой итерации поиск точки с меньшим значением функции осуществляется изменением одной компоненты вектора $\bar{x}$ при неизменных остальных.

Существует несколько вариантов покоординатного спуска. Мы рассмотрим некоторые из них.

Покоординатный спуск с удвоением шага
(первый вариант)

Опишем первый цикл метода, состоящий из n итераций.

Пусть заданы точка $\bar{x}_0$ и шаг λ_0.

В точке $\bar{x}_0$ выбирают начальное направление $S_1 = \bar{e}_1$ и выбирают величину шага λ_1 способом удвоения. Этот способ состоит в следующем:

1) выбирают произвольное начальное значение шага λ_1;

2) если $f(\bar{x}_0 + \lambda_1 S_1) < f(\bar{x}_0)$, то полагают, $\lambda_1 = 2\lambda_1$ и процесс удвоения шага продолжают до тех пор, пока убывание функции не прекратится;

3) если $f(\bar{x}_0 + \lambda_1 S_1) \geq f(\bar{x}_0)$, то выбирают $\lambda_1 = \lambda_1 / 2$ и переходят к п. 2.

Шаг дробят до тех пор, пока он не уменьшится до некоторой малой величины ε. Это означает, что в данном направлении функция не убывает.

Если в направлении $\bar{e}_1$ функция $f(\bar{x})$ убывает, то фиксируют λ_1 и переходят к следующей итерации. В противном случае выбирают направление $S_1 = -\bar{e}_1$ и снова определяют величину шага λ_1 способом удвоения. Если и в данном направлении функция не убывает, то фиксируют неудачный поиск в данном направлении; в качестве значения λ_1 берут начальное значение шага λ_0 и переходят к следующей итерации.

Если в направлении $-\bar{e}_1$ функция убывает, то фиксируют найденную величину шага λ_1 и переходят к следующей итерации.

На следующей итерации выбирают направление $S_2 = \bar{e}_2$ и полагают начальное значение шага $\lambda_2 = \lambda_1$, а начальную точку $\bar{x}_1 = \bar{x}_0 + \lambda_1 S_1$ (если поиск – неудачный, то $\bar{x}_1 = \bar{x}_0$) и повторяют процесс, как на первой итерации.

Цикл заканчивается при $k = n$, т.е. после того, как пройдет поиск по всем n направлениям $S_1 = \pm\bar{e}_1, S_2 = \pm\bar{e}_2, \ldots, S_n = \pm\bar{e}_n$.

Если поиск по всем n направлениям оказался неудачным, то процесс прекращается: $\bar{x}^* = \bar{x}_n$.

В противном случае начинается новый цикл: $S_{n+1} = \bar{e}_1$ и т.д.

Покоординатный спуск с удвоением шага (второй вариант)

Каждый цикл этого метода характеризуется тем, что величина шага λ в течение всех n итераций цикла остается постоянной. Предполагается, что в результате завершения предыдущего цикла получена некоторая величина шага λ.

Рассмотрим i-ю итерацию цикла ($1 \le i \le n$), которая состоит в следующем:

1) если

$$f(\overline{x} + \lambda \overline{e}_i) < f(\overline{x}), \quad \overline{x} \text{ – текущая точка,} \tag{3.5}$$

то полагают $\overline{x} = \overline{x} + \lambda \overline{e}_i$ и переходят к следующей итерации;

2) если

$$f(\overline{x} + \lambda \overline{e}_i) \ge f(\overline{x}), \tag{3.6}$$

то вычисляют

$$f(\overline{x} - \lambda \overline{e}_i); \tag{3.7}$$

3) если

$$f(\overline{x} - \lambda \overline{e}_i) < f(\overline{x}), \tag{3.8}$$

то полагают $\overline{x} = \overline{x} - \lambda \overline{e}_i$ и переходят к следующей итерации;

4) если

$$f(\overline{x} - \lambda \overline{e}_i) \ge f(\overline{x}), \tag{3.9}$$

то $\overline{x}$ не меняют (полагают $\overline{x} = \overline{x}$) и переходят к следующей итерации.

Таким образом, осуществляется n итераций цикла.

Если неравенства (3.6) и (3.9) имеют место для всех $1 \le i \le n$, то уменьшают величину λ, полагая, как правило, $\lambda = \lambda / 2$ и переходят к следующему циклу, то есть повторяют все процедуры предыдущего цикла, но уже с шагом, уменьшенным в два раза.

Если неравенство (3.6) или (3.9) имеют место не для всех i, то шаг λ не дробится.

Процесс заканчивается, когда величина шага становится меньше заданной величины ε, определяющей требуемую точность вычисления точки минимума.

Покоординатный спуск с одномерной минимизацией

Цикл данного метода опять содержит n итераций. Каждая итерация состоит в поиске минимума функции вдоль одной из координат при неизменных остальных. Поиск минимума осуществляется одним из методов одномерной минимизации. Изобразим алгоритм в виде блок-схемы, представленной на рис. 3.12.

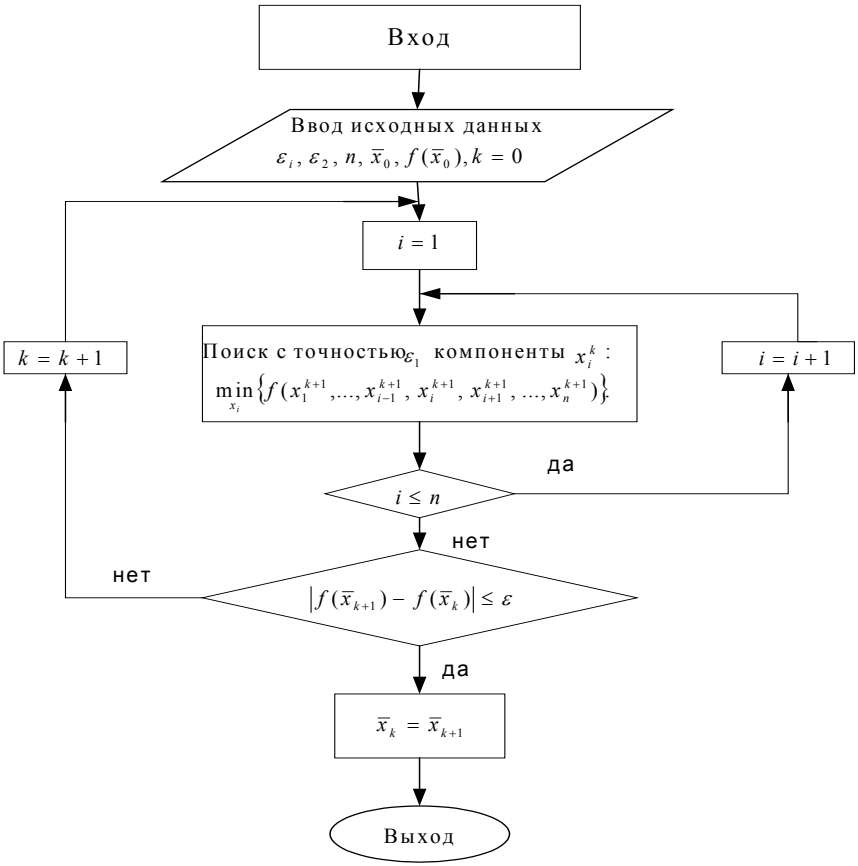


Рис. 3.12. Блок-схема алгоритма метода покоординатного спуска

Недостаток метода покоординатного спуска (всех его вариантов) заключается в том, что он может «застревать», т.е. он может

остановиться вдали от точки минимума и не обеспечить дальнейшего улучшения. Такая ситуация может возникнуть, если поверхности уровня целевой функции обладают острыми углами или очень изогнуты. На рис. 3.13, приведенном ниже и отображающем расположение линий уровня $f(\bar{x}) = C$, представлен пример такой ситуации.

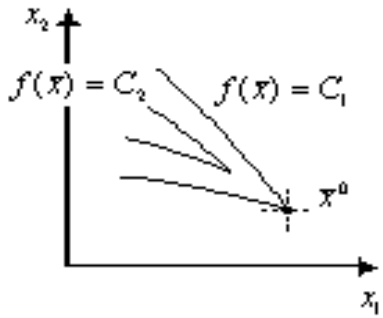


Рис. 3.13. Линии уровня в методе покоординатного спуска

Если процесс решения привел в точку $\bar{x}^0$, то каким бы малым ни брать шаг в направлении x_1 или x_2, нельзя получить уменьшение значения функции. Метод «застревает» в точке $\bar{x}^0$.

Рассмотрим несколько примеров применения метода покоординатного спуска.

Задача 1. Построить блок-схему метода покоординатного спуска с удвоением шага (второй вариант).

Блок-схема метода покоординатного спуска с удвоением шага при поиске минимуму по каждой координате приведена на рис. 3.14. При этом использованы следующие обозначения:

k – номер цикла;

i – номер (внутри цикла) итерации движения по i-му направлению ($i = \overline{1, n}$);

j – счетчик неудачных шагов, когда $\bar{x}$ не меняется;

$\bar{x}$ – текущее значение аргумента (без индекса).

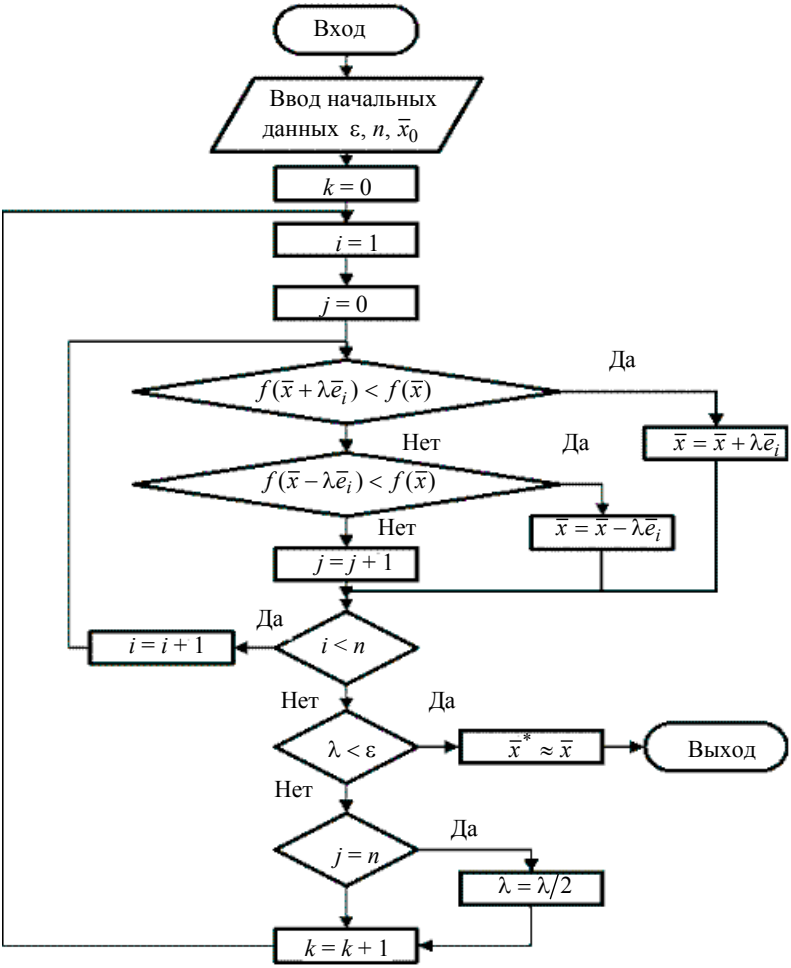


Рис. 3.14. Блок-схема метода покоординатного спуска с удвоением шага

Задача 2. Найти методом покоординатного спуска с одномерной минимизацией по каждой координате минимум функции

$$f(\bar{x}) = x_1^2 + x_2^2 - 4x_1 + 5$$

с точностью $\varepsilon = 0{,}01$.

Схему решения задачи можно представить в виде следующей последовательности действий.

Выберем начальную точку $x_1^0 = 1$, $x_2^0 = 1$, $f^0 = 4$.

<u>Цикл</u> $k = 0$.

<u>Итерация</u> $i = 1$:

$$x_1 - \text{меняется};$$
$$x_2 = 1,$$

необходимо найти $\min\limits_{x_1}\left\{f(x_1) = x_1^2 - 4x_1 + 7\right\}$.

Из условия экстремума:

$$\frac{\partial f}{\partial x_1} = 2x_1 - 4 = 0 \to x_1 = 2.$$

Итак:

$$\begin{cases} x_1 = 2; \\ x_2 = 1, \end{cases} \quad f^1 = 3, \quad \left|f^1 - f^0\right| = 1 > \varepsilon.$$

<u>Итерация</u> $i = 2$:

$$\begin{cases} x_1 = 2; \\ x_2 - \text{меняется}, \end{cases}$$

необходимо найти $\min\limits_{x_2}\left\{f(x_2) = x_2^2 + 1\right\}$.

Из условия экстремума:

$$\frac{\partial f}{\partial x_2} = 2x_2 = 0 \to x_2 = 0.$$

Итак:

$$\begin{cases} x_1 = 2; \\ x_2 = 0, \end{cases} \quad f^2 = 1, \quad \left|f^2 - f^1\right| = 2 > \varepsilon.$$

<u>Цикл</u> $k = 1$.

Соответственно:

<u>итерация</u> $i = 1$:

$$\begin{cases} x_1 - \text{меняется}; \\ x_2 = 0. \end{cases}$$

Исходя из

$$\min_{x_1} \left\{ f(x_1) = x_1^2 - 4x_1 + 5 \right\} \to x_1 = 2 .$$

Итак:

$$\begin{cases} x_1 = 2; \\ x_2 = 0, \end{cases} \quad f^3 = 1, \quad \left| f^3 - f^2 \right| = 0 < \varepsilon .$$

Ответ:

$$\begin{cases} x_1^* = 2; \\ x_2^* = 0, \end{cases} \quad f^* = 1 .$$

Линии уровня, соответствующие рассматриваемой функции приведены ниже:

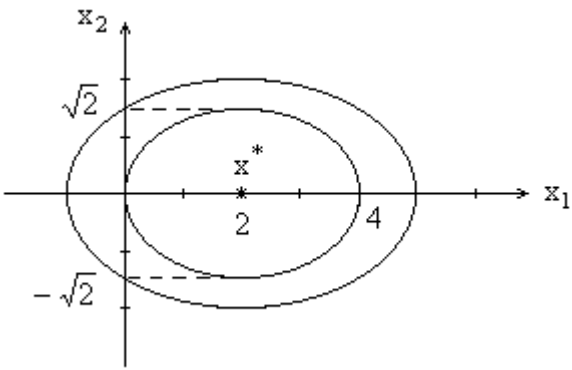


Рис. 3.15. Линии уровня функции $f(\overline{x}) = x_1^2 + x_2^2 - 4x_1 + 5$

3.3.1.2. Метод ортонормальных направлений (метод Розенброка)

В этом методе в каждом цикле производится поиск вдоль n взаимно ортогональных направлений. После завершения цикла с помощью процедуры Грама-Шмидта строится новая система ортогональных направлений. Преимущество этого метода состоит в следующем. Если целевая функция имеет узкий искривленный гребень (точнее, овраг), то поиск по n взаимно ортогональным направлениям эффективен тем, что результирующее направление стремится расположиться вдоль оси оврага. Это существенно ускоряет процесс по сравнению с методом покоординатного спуска.

Рассмотрим процедуры, выполняемые на k -м цикле метода.

Пусть в начале цикла имеется точка $\bar{x}_0^k$, числа $\lambda_1, \lambda_2, \ldots, \lambda_n$ – длины шагов, $\hat{S}_1^k, \hat{S}_2^k, \ldots, \hat{S}_n^k$ – единичные векторы, задающие направления поиска экстремума. (Для первого цикла $\bar{x}_0^1$, $\lambda_1, \lambda_2, \ldots, \lambda_n$ выбираются произвольно, направления, как правило, совпадают с координатными осями.)

В направлении $\hat{S}_1^k$ делается шаг $\lambda_1 \lambda_1$.

Если $f(\bar{x}_0^k + \lambda_1 \hat{S}_1^k) \le f(\bar{x}_0^k)$, то шаг считается успешным и полученная точка занимает место $\bar{x}_0^k$, т.е. $\bar{x}_0^k = \bar{x}_0^k + \lambda_1 \hat{S}_1^k$. Величина λ_1 умножается на число α ($\alpha > 1$, обычно $\alpha = 3$), т.е. теперь $\lambda_1 = \alpha \lambda_1$.

Если $f(\bar{x}_0^k + \lambda_1 \hat{S}_1^k) > f(\bar{x}_0^k)$, то шаг считается неуспешным, точка $\bar{x}_0^k$ остается без изменений, а величина λ_1 умножается на число β ($\beta < 0$, обычно $\beta = -0{,}5$), т.е. $\lambda_1 = \beta \lambda_1$.

Далее та же самая процедура повторяется для остальных направлений $\hat{S}_2^k, \ldots, \hat{S}_n^k$. В результате будет получена новая точка $\bar{x}_0^k$ и совокупность шагов $\lambda_1, \lambda_2, \ldots, \lambda_n$ тоже новая. Затем снова возвращаемся к направлению $\hat{S}_1^k$ и задаем шаг либо $\alpha \lambda_1$, либо $\beta \lambda_1$ в зависимости от того успешным или неуспешным был шаг до этого.

Процесс поиска вдоль направления повторяется сначала. Это происходит до тех пор, пока за успешным движением на каждом направлении не последует неудача. На этом заканчивается первый этап цикла.

Второй этап цикла состоит в выборе исходных данных для следующего цикла.

В качестве начальной точки следующего цикла $\overline{x}_0^{k+1}$ берется последняя успешная точка.

В качестве шагов $\lambda_i, i = \overline{1,n},$ – последние получившиеся значения. После этого выбираются новые направления.

Строится система векторов:

$$\begin{aligned}
\overline{q}_1 &= \Lambda_1 \hat{S}_1^k + \Lambda_2 \hat{S}_2^k + \ldots + \Lambda_n \hat{S}_n^k; \\
\overline{q}_2 &= \qquad\quad \Lambda_2 \hat{S}_2^k + \ldots + \Lambda_n \hat{S}_n^k; \\
&\ldots \\
\overline{q}_n &= \qquad\qquad\qquad\quad \Lambda_n \hat{S}_n^k,
\end{aligned}$$

здесь Λ_i – алгебраическая сумма длин успешных шагов по *i*-му направлению.

Если изобразить условно направления на плоскости, то получим следующий рисунок (рис. 3.16).

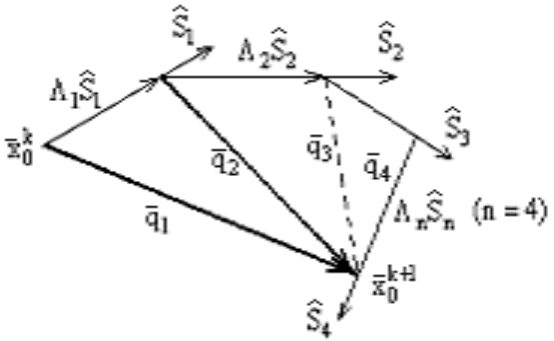


Рис. 3.16. Графическое представление метода Розенброка на одной из итераций

Как видно из рисунка, вектор $\overline{q}_1$ соединяет точку, в которой процесс находился в начале цикла, с точкой, в которую он попал в конце цикла.

Данная система векторов $\overline{q}_1,\ldots,\overline{q}_n$ ортогонализируется с помощью процедуры Грама-Шмидта.

В качестве начального (первого) направления принимается $\overline{q}_1$ (так как это направление располагается вдоль предполагаемого гребня функции).

Процедура ортогонализации Грама-Шмидта. Пусть имеем n векторов $\overline{q}_1,\ldots,\overline{q}_n$. Необходимо получить n единичных ортогональных векторов $\hat{S}_1^k,\ldots,\hat{S}_n^k$.

Будем находить эти вектора по формуле: $\hat{S}_i=\dfrac{\overline{y}_i}{\|\overline{y}_i\|}$, где $\overline{y}_i$ – вспомогательный вектор, $i=\overline{1,n}$, $\|\overline{y}_i\|=\langle\overline{y}_i,\overline{y}_i\rangle^{\frac{1}{2}}$ – норма вектора.

Вектора $\overline{y}_i$ вычисляются по рекуррентным формулам:

$$\overline{y}_1=\overline{q}_1,$$

$$\ldots$$

$$\overline{y}_i=\overline{q}_i-\sum_{j=1}^{i-1}\frac{\langle\overline{q}_i,\overline{y}_j\rangle}{\langle\overline{y}_j,\overline{y}_j\rangle}\overline{y}_j,\quad i=\overline{1,n};$$

$$\hat{S}_1=\overline{q}_1,$$

$$\ldots$$

$$\hat{S}_i=\overline{q}_i-\sum_{j=1}^{i-1}\frac{\langle\overline{q}_i,\hat{S}_j\rangle}{\langle\hat{S}_j,\hat{S}_j\rangle}\hat{S}_j,\quad i=\overline{1,n}.$$

Блок-схема алгоритма может быть представлена в виде, изображенном на рис. 3.17. На рисунке использованы следующие обозначения:

k – номер цикла;

$i=\overline{1,n}$ – номер итерации при движениях по $\hat{S}_i$;

j – счетчик неудачных шагов;

$\overline{x}_0$ – это $\overline{x}$ в начале цикла;

$\overline{x}$ – текущий аргумент.

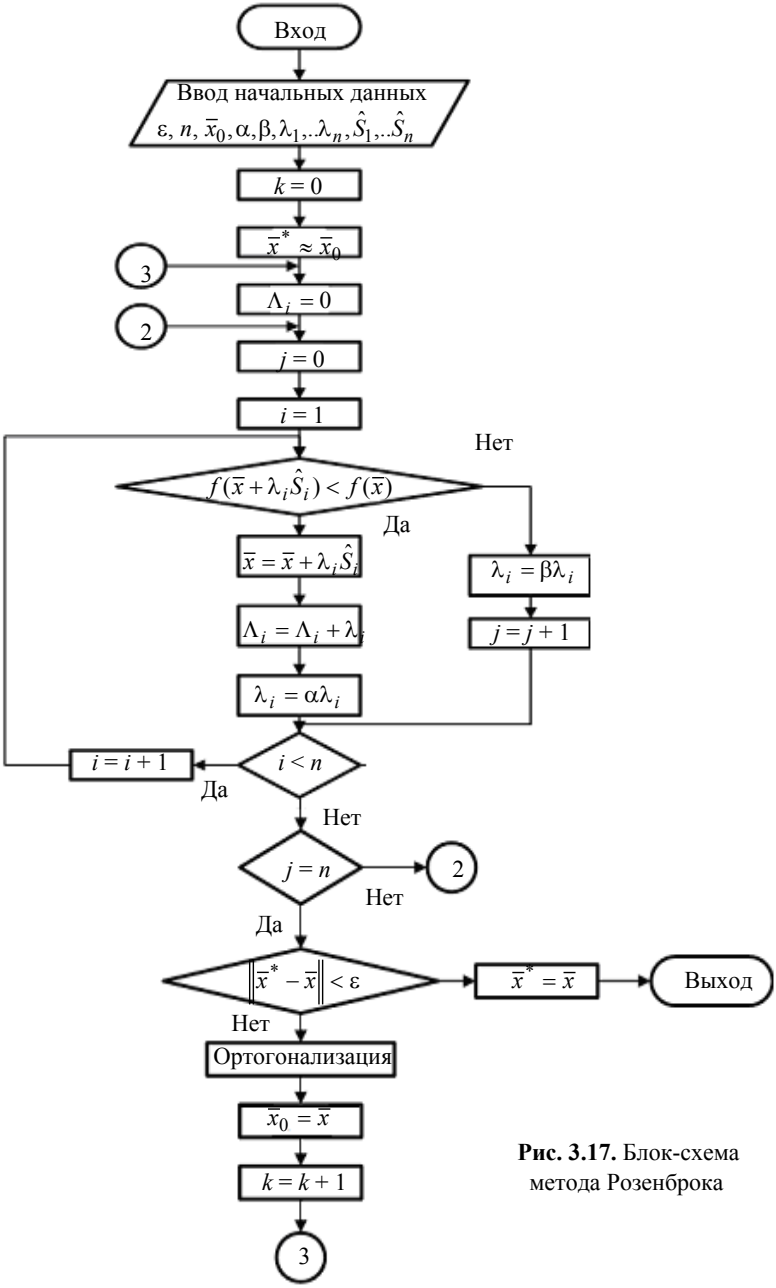


Рис. 3.17. Блок-схема метода Розенброка

3.3.1.3. Метод сопряженных направлений (метод Пауэлла)

Два вектора $\overline{x}$ и $\overline{y}$ называются Q-сопряженными (или сопряженными по отношению к матрице Q), если $\overline{x}^{\mathrm{T}} Q \overline{y} = 0$, где Q – положительно определенная матрица ($\overline{x}$ и $\overline{y}$ называют взаимно ортогональными, если $\overline{x}^{\mathrm{T}} \overline{y} = 0$. Таким образом, понятие сопряженности является обобщением понятия ортогональности).

Направления $S_1, S_2, \ldots, S_n$ являются *сопряженными* относительно матрицы Q, если

$$S_i^{\mathrm{T}} Q S_j = 0 \quad \text{при} \quad i \neq j, \quad i, j = \overline{1, n}.$$

Сущность метода сопряженных направлений состоит в том, чтобы построить систему направлений, сопряженных относительно матрицы Гессе целевой функции, и осуществить последовательно одномерную минимизацию вдоль каждого из этих направлений.

Это самая общая идея метода. Существуют несколько способов построения системы сопряженных направлений и, соответственно, несколько различных вариантов метода сопряженных направлений. Метод Пауэлла – один из таких вариантов, относящийся к методам нулевого порядка, т.е. методам, не требующим вычисления производных.

На чем основана идея сопряженных направлений и в чем преимущество поиска вдоль таких направлений? Дело в том, что если имеем квадратичную целевую функцию

$$f(\overline{x}) = a + \overline{x}^{\mathrm{T}} \overline{b} + \frac{1}{2} \overline{x}^{\mathrm{T}} Q \overline{x},$$

где $\overline{x}$ – n-мерный вектор, Q – матрица Гессе (Q – положительно определенная матрица), то минимум такой функции может быть найден ровно за «n» шагов от любой начальной точки, если поиск вести вдоль системы из n Q-сопряженных направлений.

На рис. 3.18 приведен пример формирования Q-сопряженных направлений для случая квадратичной целевой функции $f(\overline{x})$.

Рис. 3.18. Определение сопряженных направлений (n = 2)

Может возникнуть вопрос, зачем понадобилось тратить усилия на минимизацию квадратичной функции, если она минимизируется аналитически и дает результат $\bar{x}^* = -Q^{-1}\bar{b}$?

Квадратичные функции рассматриваются по двум причинам.

Во-первых, если метод не пригоден для квадратичной функции, то очень мало шансов, что он пригоден для функций с более сложной структурой. Поэтому любой метод, как правило, вначале испытывают на квадратичной функции.

Во-вторых, не квадратичную функцию можно аппроксимировать квадратичной функцией, если ограничиться разложением в ряд Тейлора членами не выше второго порядка. При этом минимум функции может быть получен посредством минимизации последовательности квадратичных функций, аппроксимирующих исходную функцию относительно точек, последовательно приближающих к точке истинного минимума.

Метод Пауэлла основывается на следующей особенности: любая прямая, которая проходит через точку минимума квадратичной формы, пересекает под равными углами поверхности уровня. Очевидно, это эквивалентно тому, что все касательные, проведенные к линиям уровня в точках их пересечения прямой, проходящей через экстремум, будут параллельны. На рис. 3.19 приведена иллюстрация этого свойства для квадратичной функции $f(\bar{x})$.

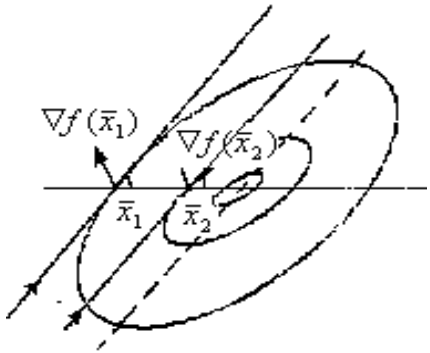


Рис. 3.19. Расположение касательных к линиям уровня в точках $\bar{x}_1, \bar{x}_2$

Из этого следует, что если $\bar{x}_1$ и $\bar{x}_2$ – два минимума, расположенные на двух параллельных направлениях, то минимум квадратичной функции нужно искать на направлении $\bar{x}_1 - \bar{x}_2$. Причем легко убедится, что это направление является Q-сопряженным с S. Градиент квадратичной функции:

$$\nabla f(\bar{x}) = \bar{b} + Q\bar{x}\,.$$

Поскольку $\bar{x}^i$ – это точки минимума на направлении S, градиент функции в этих точках ортогонален к S, т.е.

$$S^{\mathrm{T}}\nabla f(\bar{x}^1) = 0 = S^{\mathrm{T}}(\bar{b} + Q\bar{x}^1),$$
$$S^{\mathrm{T}}\nabla f(\bar{x}^2) = 0 = S^{\mathrm{T}}(\bar{b} + Q\bar{x}^2).$$

Вычитая второе уравнение из первого, получим

$$0 = S^{\mathrm{T}}Q(\bar{x}_1 - \bar{x}^2)\,,$$

т.е. S и $(\bar{x}^1 - \bar{x}^2)$ – Q-сопряженные направления.

Алгоритм метода Пауэлла.

1. Выбрать в качестве начальных направлений $S_1^0, S_2^0, \ldots, S_n^0$, совпадающие с координатными осями ($k = 0$). Выбрать начальную точку поиска $\bar{x}_0^k$.

2. Определить с помощью одномерного поиска

$$\min_{\lambda_1} f(\bar{x}_0^k + \lambda_1 S_1^k).$$

Положить $\bar{x}_1^k = \bar{x}_0^k + \lambda_1^* S_1^k$. Затем последовательно осуществить одномерный поиск вдоль направлений $S_2^k, \ldots, S_n^k$, определив последовательность точек

$$\bar{x}_i^k = \bar{x}_{i+1}^k + \lambda_i^* S_i^k, \quad i = \overline{2,n}.$$

3. Вычислить направление $S = \bar{x}_n^k - \bar{x}_0^k$ и сформировать новые направления для следующей итерации:

$$\left\{S_1^{k+1}, S_2^{k+1}, \ldots, S_n^{k+1}\right\} = \left\{S_2^k, S_3^k, \ldots, S_n^k, S\right\}.$$

4. Найти λ^* из условия $\min_{\lambda} f(\bar{x}_n^k + \lambda S_1^k)$ (т.е. найти минимум вдоль нового полученного направления) и взять в качестве начальной точки для следующей итерации $\bar{x}_0^{k+1} = \bar{x}_n^k + \lambda^* S^k$, положив $k = k + 1$, и перейти к п. 2.

Блок-схема алгоритма метода Пауэлла приведена на рис. 3.20.

В результате такой процедуры определения направлений поиска поочередно заменяются принятые в начале координатные направления. При квадратичной форме после n шагов все направления поиска окажутся взаимно сопряженными (это можно доказать) и будет найдена точка минимума.

В случае неквадратичной функции метод сопряженных направлений становится итеративным и обычно не заканчивается за n шагов. Неквадратичные функции локально аппроксимируют последовательность квадратичных функций. Направления определяются в соответствии с квадратичными аппроксимациями этих неквадратичных функций. Поэтому при функциях, локальные квадратичные аппроксимации которых быстро меняются от итерации к итерации или для которых матрица Гессе перестает быть положительно определенной, метод сопряженных направлений может не сходиться.

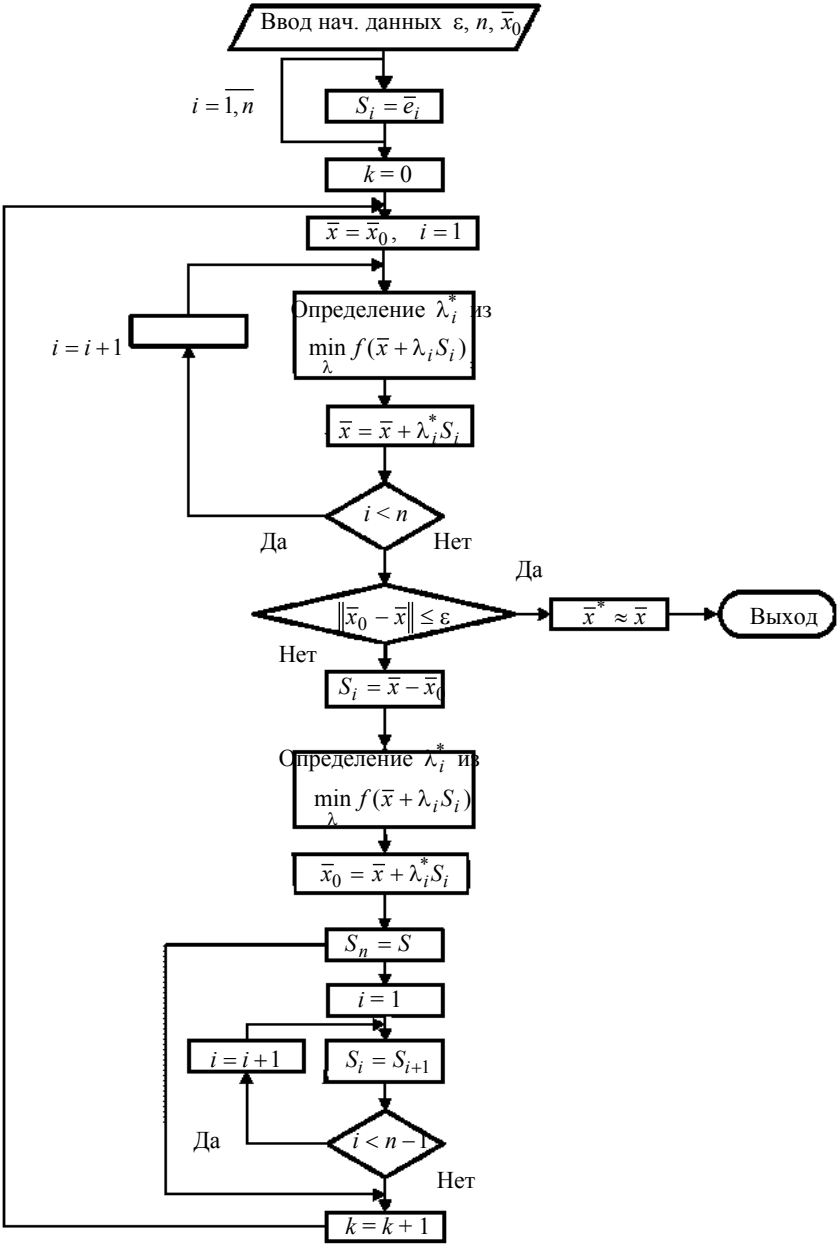


Рис. 3.20. Блок-схема алгоритма метода Пауэлла

3.3.2. Методы первого порядка

3.3.2.1. Градиентные методы

Приемлемым направлением, т.е. направлением, в котором функция убывает, как было показано раньше, является направление S, для которого

$$\langle \nabla f(\bar{x}), S \rangle < 0 .$$

Поэтому направление $S = -\nabla f(\bar{x})$ является приемлемым. Более того, можно показать, что с направлением антиградиента совпадает направление наибыстрейшего убывания функции в точке $\bar{x}$. Это свойство градиента лежит в основе ряда итерационных методов минимизации функций, в том числе градиентных методов.

Градиентным называется метод, согласно которому точка $\bar{x}_{k+1}$ выбирается в соответствии с соотношением

$$\bar{x}_{k+1} = \bar{x}_k - \lambda_k \nabla f(\bar{x}_k) .$$

В зависимости от способа выбора шага λ_k можно получить различные варианты градиентного метода.

1 вариант (метод наискорейшего спуска). Согласно этому методу λ_k определяется путем решения одномерной задачи минимизации

$$\min_{\lambda_k} f(\bar{x}_k - \lambda_k \nabla f(\bar{x}_k)) .$$

В отдельных случаях эта задача может быть решена точно, аналитически.

Интересная особенность метода наискорейшего спуска состоит в том, что предыдущее и последующее направления поиска ортогональны.

Двумерная иллюстрация метода представлена рис. 3.21.

Действительно, точка $\bar{x}_{k+1}$ лежит на данном направлении в точке его касания с линией уровня $f(\bar{x}) = f(\bar{x}_{k+1})$ поскольку она найдена из условия минимума на направлении $-\nabla f(\bar{x}_k)$. Следующим направлением поиска будет $-\nabla f(\bar{x}_{k+1})$. Известно, что градиент

направлен по нормали к линии уровня, т.е. перпендикулярно касательной к ней.

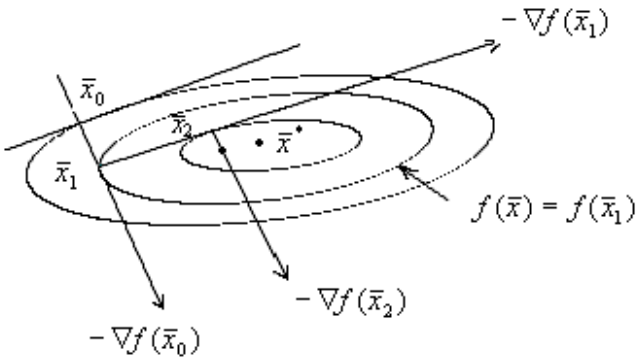


Рис. 3.21. Метод наискорейшего спуска

Аналитически это можно показать следующим образом.

Обозначим

$$\varphi(\lambda) = f(\bar{x}_k - \lambda \nabla f(\bar{x}_k)) .$$

Поскольку шаг λ выбирается из условия минимума

$$\left.\frac{d\varphi}{d\lambda}\right|_{\lambda_k} = 0 .$$

Найдем $\dfrac{d\varphi}{d\lambda}$ по правилу дифференцирования сложной функции; т.е. исходя из зависимости $\varphi[\bar{x}(\lambda)]$, значение производной определяется соотношением

$$\frac{d\varphi}{d\lambda} = \frac{d\varphi}{dx}\frac{dx}{d\lambda} .$$

Откуда

$$\left.\frac{d\varphi}{d\lambda}\right|_{\lambda_k} = < \underbrace{\nabla f(\bar{x}_k - \lambda_k \nabla f(\bar{x}_k))}_{\text{производная } \frac{d\varphi}{dx}}, \underbrace{-\nabla f(\bar{x}_k)}_{\text{производная } \frac{dx}{d\lambda}} > =$$

$$= - < \nabla f(\bar{x}_{k+1}), \nabla f(\bar{x}_k) > = 0 .$$

Следовательно, направления ортогональны.

Важно отметить, что эпитет наискорейший по отношению к спуску является исторически сложившимся, он не означает, что этот метод обладает способностью осуществлять отыскание минимальной точки за минимальное число шагов или при минимальном объеме вычислений. Существуют более эффективные в этом смысле методы.

2 вариант градиентного метода. На практике нередко довольствуются выбором шага, обеспечивающего убывание функции в направлении антиградиента $(f(\bar{x}_{k+1}) < f(\bar{x}_k))$, вместо того чтобы отыскивать минимум функции в данном направлении. Трудоемкость итерации при таком способе выбора шага меньше. При этом скорость сходимости обоих вариантов в конечном счете примерно одинакова.

Опишем способ выбора шага λ_k на k-й итерации.

1. Выбирается некоторое произвольное значение λ (одно и то же на всех итерациях) и определяется точка $\bar{x} = \bar{x}_k - \lambda \nabla f(\bar{x}_k)$.

2. Вычисляется $f(\bar{x}) = f(\bar{x}_k - \lambda \nabla f(\bar{x}_k))$.

3. Производится проверка неравенства

$$f(\bar{x}_k) - f(\bar{x}) \geq \varepsilon\lambda(<\underbrace{\nabla f(\bar{x}_k), \nabla f(\bar{x}_k)}_{\|\nabla f(\bar{x})\|^2}>),$$

где $0 < \varepsilon < 1$ – произвольно выбранная константа (одна и та же на всех итерациях).

Возможно использование более простого неравенства:

$$f(\bar{x}) < f(\bar{x}_k).$$

4. Если неравенство выполняется, то значение λ и берется в качестве искомого: $\lambda_k = \lambda$. В противном случае производится дробление λ (путем умножения λ на произвольное число $\alpha < 1$) до тех пор, пока неравенство не окажется справедливым.

Следует отметить, что если выбрать в качестве λ слишком маленький шаг, то процесс будет сходиться очень долго, поэтому время от времени можно пробовать увеличить λ, но так чтобы не нарушить выбранное неравенство.

Направления поиска в этом варианте метода уже не будут ортогональны (рис. 3.22).

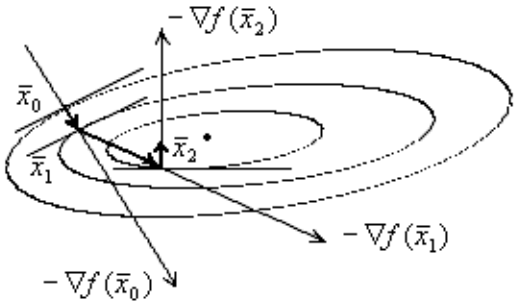


Рис. 3.22. Градиентный метод (второй вариант)

Можно предложить несколько критериев окончания итерационного процесса:

1) по разности аргументов на последовательных шагах

$$\|\bar{x}_{k+1} - \bar{x}_k\| \le \varepsilon_1,$$

2) по разности значений функции

$$\| f(\vec{x}_{k+1}) - f(\vec{x}_k) \| \le \varepsilon_2,$$

3) по значению градиента

$$\| \nabla f(\vec{x}_k) \| \le \varepsilon_3 .$$

В большинстве практических случаев невозможно получить явное выражение градиента. В этом случае используются численные методы вычисления производных:

Например. Для функции $f(x_1, x_2)$

$$\frac{\partial f(x^k)}{\partial x_1} = \frac{f(x_1^k + \delta_1, x_2^k) - f(x_1^k, x_2^k)}{\delta_1},$$

$$\frac{\partial f(x^k)}{\partial x_2} = \frac{f(x_1^k, x_2^k + \delta_2) - f(x_1^k, x_2^k)}{\delta_2}.$$

Сходимость градиентных методов. Тот факт, что градиентные методы при определенных условиях сходятся, отражается в теореме, которую приведем без доказательства.

Теорема. Если функция $f(\bar{x})$ ограничена снизу, ее градиент $\nabla f(\bar{x})$ удовлетворяет условию Липшица:

$$\| \nabla f(\bar{x}) - \nabla f(\bar{y}) \| \le R \| \bar{x} - \bar{y} \|$$

при любых $\bar{x}, \bar{y} \in E^n$, а выбор шага λ_k производится указанными ранее способами, то для любой начальной точки $\bar{x}_0$ градиентный метод сходится, т.е. $\| \nabla f(\bar{x}_k) \| \to 0$ при $k \to \infty$.

Оценка скорости сходимости проведена при еще более жестких требованиях к функции (выпуклость, гладкость и т.д.). Для достаточно хороших, с точки зрения решения задачи минимизации функций (гладких и выпуклых), градиентные методы сходятся к минимуму со скоростью геометрической прогрессии

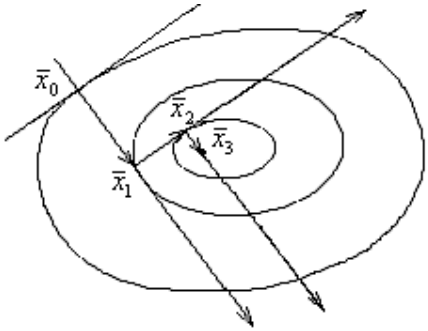
$$\| \bar{x}_{k+1} - \bar{x}_* \| \le q \| \bar{x}_k - \bar{x}_* \|, \quad 0 < q < 1$$

(такая скорость сходимости называется *линейной*).

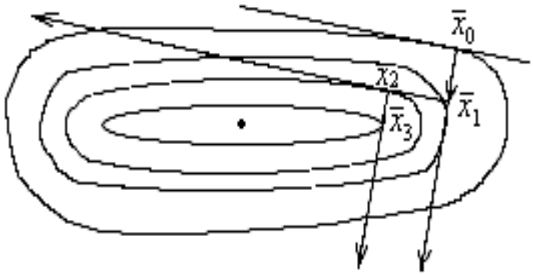
Величина знаменателя прогрессии q зависит от соотношения наибольшего M и наименьшего m собственных значений матрицы вторых производных $f(\bar{x})$. Достаточно малым знаменатель q будет лишь в том случае, когда эти собственные числа мало отличаются друг от друга. В этом случае градиентные методы будут сходиться с высокой скоростью. Чем меньше отношение $\frac{m}{M}$, тем ближе к единице знаменатель q и тем медленнее сходятся градиентные методы.

Можно дать геометрическую интерпретацию этого факта (рис. 3.23).

Когда отношение $\frac{m}{M}$ близко к единице линии уровня функции $f(\bar{x}) = \text{const}$ близки к окружности. Для таких линий уровня градиентный метод сходится быстро (рис. 3.23, *а*).



а)



б)

Рис. 3.23. Типы линий уровня: *а* – линии уровня близкие к окружности, *б* – линии уровня «овражного типа»

С уменьшением отношения $\frac{m}{M}$ линии уровня становятся все более вытянутыми, и направление вектора градиента в большинстве точек все более существенно отклоняется от направления в точку минимума. Это приводит к замедлению скорости сходимости (рис. 3.23, *б*).

Особенно медленно градиентные методы сходятся, когда функция имеет «овражный» характер. Это означает, что небольшое изменение некоторых переменных приводит к резкому изменению значений функции – эта группа характеризует «склон оврага», а по остальным переменным, задающим направление «дна оврага», функция меняется незначительно. Если линии уровня у такой функции сильно вытянуты, то градиентные методы сходятся медленно.

Это происходит из-за того, что кривая поиска для таких функций обычно быстро спускается на «дно оврага», а затем начинает медленное перемещение к экстремуму, так как градиент оказывается почти перпендикулярным к оси оврага (рис. 3.24).

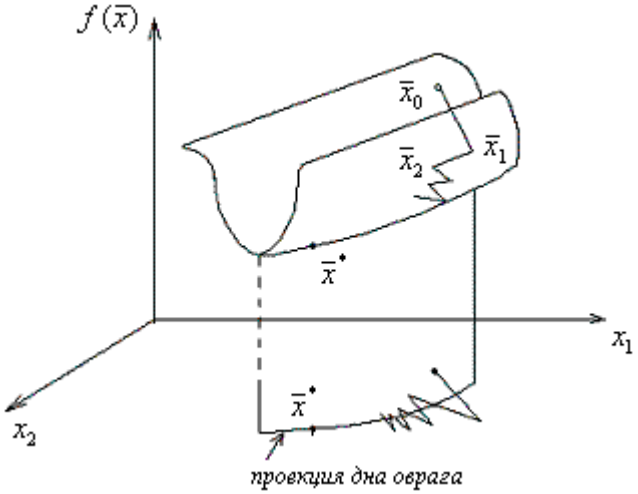


Рис. 3.24. Использование градиентного метода в случае функции «овражного» типа

Для ускорения сходимости при поиске минимума «овражной» функции существуют специальные методы – они называются «овражными».

Так для ускорения сходимости градиентных методов используют преобразование системы координат, таким образом чтобы линии уровня функции стали близки к окружностям.

Пример. Рассмотрим функцию $f(\overline{x}) = x_1^2 + 25x_2^2$.

Очевидно, линии уровня данной функции вытянуты вдоль оси x_1.

Если ввести замену переменных: $y_1 = x_1, y_2 = 5x_2$, то функция $f(\overline{y}) = y_1^2 + y_2^2$ будет иметь линии уровня в виде окружностей и, в результате, градиентный метод сходится за один шаг.

В реальной задаче подобрать нужные преобразования, как правило, оказывается достаточно сложно.

Блок-схема градиентного метода (2-й вариант) изображена на рис. 3.25.

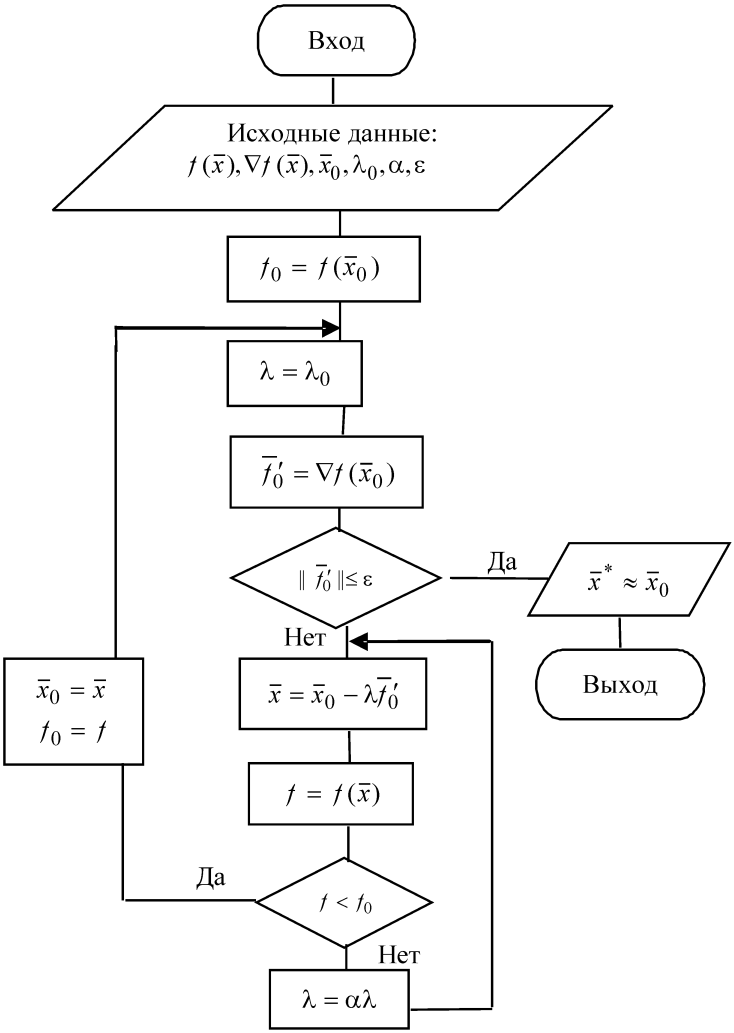


Рис. 3.25. Блок-схема градиентного метода, где $\bar{x}_0$ – аргумент в начале итерации; $\bar{x}$ – текущее значение аргумента; f_0 – функция в точке $\bar{x}_0$; f – функция в точке $\bar{x}$; f_0' – производная в точке $\bar{x}$; λ_0 – начальное λ для каждой итерации

В данном методе окончание итерационного процесса осуществляется по условию – $\| f'(\bar{x}_k) \| \le \varepsilon$.

Задача. Найти методом наискорейшего спуска минимум функции $f(x) = 2x_1^2 + x_2^2 + x_1x_2 + x_1 + x_2$ с точностью $\varepsilon = 0{,}05$ (по норме градиента). В качестве начальной точки принять точку $\bar{x}^0 = (0{,}0)$. Значения функции и градиента в этой точке, соответственно, будут:

$$f(\bar{x}^0) = 0\,;$$

$$\nabla f(\bar{x}_0) = \left.\begin{pmatrix} \dfrac{\partial f}{\partial x_1} = 4x_1 + x_2 + 1 \\ \dfrac{\partial f}{\partial x_2} = 2x_2 + x_1 + 1 \end{pmatrix}\right|_{\bar{x}=\bar{x}_0} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Очевидно, нетрудно найти точное решение данной задачи:

$$x_1^* = -\frac{1}{7}; \quad x_2^* = -\frac{3}{7}; \quad f^* = -\frac{2}{7} \approx -0{,}2857\,.$$

Решим задачу с использованием градиентного метода. В результате получим следующую последовательность действий.

1-я итерация:

$$\nabla f(\bar{x}^0) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}; \quad \bar{x}^1 = \bar{x}^0 - \lambda \nabla f(\bar{x}^0) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} - \begin{pmatrix} \lambda \\ \lambda \end{pmatrix} = \begin{pmatrix} -\lambda \\ -\lambda \end{pmatrix},$$

$$\left\| \nabla f(\bar{x}_0) \right\| \approx 1{,}4 > \varepsilon\,.$$

Так как $x_1^1 = -\lambda$ и $x_2^1 = -\lambda$, то $f(\bar{x}^1(\lambda)) = f(\lambda) = 4\lambda^2 - 2\lambda$.

Таким образом, значение λ, обеспечивающее минимум функции по направлению $\nabla f(\bar{x}^0) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ из точки $\bar{x}^0$, будет $\lambda = \dfrac{1}{4}$.

Окончательно получаем для первой итерации:

$$\bar{x}^1 = \begin{pmatrix} -1/4 \\ -1/4 \end{pmatrix}; \quad f(\bar{x}^1) = -1/4 = -0{,}25\,.$$

2-я итерация:

$$\nabla f(\bar{x}^1) = \begin{pmatrix} -1/4 \\ 1/4 \end{pmatrix}, \quad \| \nabla f(\bar{x}_1) \| \approx 0{,}35 ;$$

$$\bar{x}^{(2)} = \begin{pmatrix} -1/4 \\ -1/4 \end{pmatrix} - \lambda \begin{pmatrix} -1/4 \\ 1/4 \end{pmatrix} = \begin{pmatrix} \dfrac{\lambda - 1}{4} \\ -\dfrac{(\lambda + 1)}{4} \end{pmatrix};$$

$$f(\lambda) = \frac{2\lambda^2 - 2\lambda - 4}{16}, \quad f'(\lambda) = \frac{1}{16}(4\lambda - 2) = 0 \to \lambda = \frac{1}{2};$$

$$\bar{x}^2 = \begin{pmatrix} -1/8 \\ -3/8 \end{pmatrix}, \quad f(\bar{x}^2) = -0{,}281 .$$

3-я итерация:

$$\nabla f(\bar{x}^2) = \begin{pmatrix} 1/8 \\ 1/8 \end{pmatrix}, \quad \| \nabla f(\bar{x}^2) \| \approx 0{,}18 ;$$

$$\bar{x}^{(3)} = \begin{pmatrix} -1/8 \\ -3/8 \end{pmatrix} - \lambda \begin{pmatrix} 1/8 \\ 1/8 \end{pmatrix} = \begin{pmatrix} -\dfrac{(\lambda + 1)}{8} \\ -\dfrac{(\lambda + 3)}{8} \end{pmatrix};$$

$$f(\lambda) = \frac{4\lambda^2 - 2\lambda - 18}{64}, \quad f'(\lambda) = \frac{1}{64}(8\lambda - 2) = 0 \to \lambda = \frac{1}{4};$$

$$\bar{x}^3 = \begin{pmatrix} -5/32 \\ -13/32 \end{pmatrix}, \quad f(\bar{x}^3) = -0{,}2852 .$$

4-я итерация:

$$\nabla f(\bar{x}^3) = \begin{pmatrix} -1/32 \\ 1/32 \end{pmatrix}, \quad \| \nabla f(\bar{x}^3) \| \approx 0{,}044 < \varepsilon .$$

Как видим, условие точности решения задачи выполнено; окончательно имеем:

$$\bar{x}^* = \bar{x}^3 \approx \begin{pmatrix} -5/32 \\ -13/32 \end{pmatrix}, \quad f(\bar{x}^*) = -0{,}2852 ,$$

что, с заданной степенью точности совпадает истинным решением задачи.

3.3.2.2. Метод сопряженных градиентов

Метод сопряженных градиентов является частным случаем метода сопряженных направлений.

Вновь предположим, что $f(\bar{x})$ – квадратичная функция вида:

$$f(\bar{x}) = a + \bar{x}^{\mathrm{T}}\bar{b} + \frac{1}{2}\bar{x}^{\mathrm{T}}Q\bar{x},$$

где Q – положительно определенная матрица, которая в данном случае совпадает с матрицей Гессе.

В том случае, если $f(\bar{x})$ не является квадратичной, то Q также будет обозначать матрицу вторых производных – матрицу Гессе. При этом матрица Гессе будет изменяться при получении каждой новой точки. Для того, чтобы метод был применимым, необходимо, чтобы матицы Гессе неквадратичных функций менялись не очень быстро.

Существуют различные варианты метода сопряженных градиентов. Одним из них является метод Флетчера – Ривса.

Вместо того чтобы использовать антиградиент в качестве направления спуска, метод сопряженных градиентов преобразует антиградиент в направление S_k, которое является Q-сопряженным с ранее найденными направлениями.

Эти направления вырабатываются следующим образом:

$$S_0 = -\nabla f(\bar{x}_0),$$

$$S_k = -\nabla f(\bar{x}_k) + \beta_{k-1}S_{k-1}, \qquad (3.10)$$

где коэффициент β_{k-1} выбирается так, чтобы сделать S_k Q-сопряженным с S_{k-1}. Иначе говоря, из условия

$$0 = < S_k, QS_{k-1} > = - < \nabla f(\bar{x}_k), QS_{k-1} > + \beta_{k-1} < S_{k-1}, QS_{k-1} >.$$

Отсюда

$$\beta_{k-1} = \frac{< \nabla f(\bar{x}_k), QS_{k-1} >}{< S_{k-1}, QS_{k-1} >}. \qquad (3.11)$$

Точка $\bar{x}_k$ находится с помощью одномерного поиска в направлении S_k из точки $\bar{x}_{k-1}$, т.е.

$$\bar{x}_k = \bar{x}_{k-1} + \lambda^*_{k-1} S_{k-1}, \qquad (3.12)$$

где

$$\lambda^*_{k-1} = \arg(\min_{\lambda_{k-1}} f(\bar{x}_{k-1} + \lambda_{k-1} S_{k-1})).$$

Можно показать, что вырабатываемые с помощью данной процедуры направления $S_0, S_1, ..., S_k$ являются Q-сопряженными.

Таким образом, *метод сопряженных градиентов сходится для квадратичных функций за число шагов, не превышающих n*.

Для того чтобы распространить метод сопряженных градиентов на случай поиска экстремума неквадратичных функций (в том числе заданных неявно), рассмотрим способ выбора направления спуска, не использующий в явном виде матрицу Гессе Q.

Для этого воспользуемся выражением градиента квадратичной функции:

$$\nabla f(\vec{x}) = Q\bar{x} + \bar{b}.$$

Тогда, принимая во внимание итерационное соотношение (3.12), можно записать,

$$\nabla f(\bar{x}_k) - \nabla f(\bar{x}_{k-1}) = Q(\bar{x}_k - \bar{x}_{k-1}) = \lambda^*_{k-1} Q S_{k-1}. \qquad (3.13)$$

Откуда получаем

$$QS_{k-1} = \frac{\nabla f(\bar{x}_k) - \nabla f(\bar{x}_{k-1})}{\lambda^*_{k-1}}.$$

Подставим полученное выражение в формулу (3.11) для β_{k-1}:

$$\beta_{k-1} = \frac{< \nabla f(\bar{x}_k), \nabla f(\bar{x}_k) - \nabla f(\bar{x}_{k-1}) >}{< S_{k-1}, \nabla f(\bar{x}_k) - \nabla f(\bar{x}_{k-1}) >},$$

или, раскрывая скалярные произведения и производя несложные операции, получим

$$\beta_{k-1} = \frac{< \nabla f(\vec{x}_k), \nabla f(\vec{x}_k) > - \overbrace{< \nabla f(\bar{x}_k), \nabla f(\bar{x}_{k-1}) >}^{1}}{\underbrace{< S_{k-1}, \nabla f(\bar{x}_k) >}_{2} - \underbrace{< S_{k-1}, \nabla f(\bar{x}_{k-1}) >}_{3}}.$$

Рассмотрим каждый из занумерованных членов.

2-й член. Так как $\bar{x}_k$ – точка минимума на направлении S_{k-1}, то градиент в этой точке ортогонален к направлению поиска (рис. 3.26). Откуда получаем:

$$< S_{k-1}, \nabla f(\bar{x}_k) >= 0 .$$

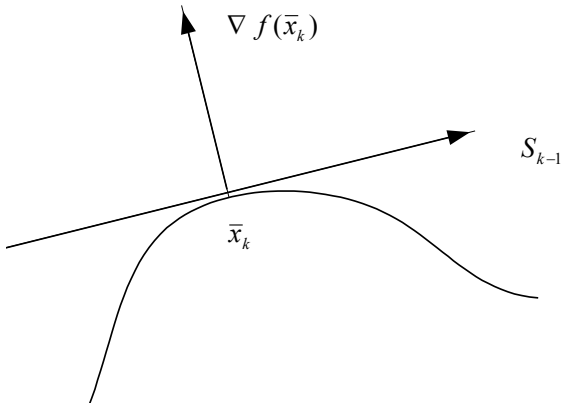


Рис. 3.26. Ортогональность градиента $\nabla f(\bar{x}_k)$ и направления S_{k-1}

3-й член. Подставим вместо S_{k-1} его выражение, определяемое формулой (3.10). Тогда получим:

$$< S_{k-1}, \nabla f(\bar{x}_{k-1}) >=< -\nabla f(\bar{x}_{k-1}), \nabla f(\bar{x}_{k-1}) > +\beta_{k-2} < S_{k-2}, \nabla f(\bar{x}_{k-1}) > .$$

Очевидно, второе слагаемое равно нулю, как скалярное произведение двух ортогональных векторов. Таким образом,

$$< S_{k-1}, \nabla f(\bar{x}_{k-1}) >= - < \nabla f(\bar{x}_{k-1}), \nabla f(\bar{x}_{k-1}) > .$$

1-й член. Выразим $\nabla f(\bar{x}_{k-1})$ в соответствии с формулой (3.10), тогда получим

$$< \nabla f(\bar{x}_k), \nabla f(\bar{x}_{k-1}) >=< \nabla f(\bar{x}_k), -S_{k-1} + \beta_{k-2} S_{k-2} >=$$

$$= - < \nabla f(\bar{x}_k), S_{k-1} > +\beta_{k-2} < \nabla f(\bar{x}_k), S_{k-2} > .$$

Разрешим (3.13) относительно $\nabla f(\bar{x}_k)$ и подставим в последнее выражение. Тогда получим:

$$< \nabla f(\bar{x}_k), \nabla f(\bar{x}_{k-1}) >= \beta_{k-2} < \nabla f(\bar{x}_{k-1}) + \lambda^*_{k-1} Q S_{k-1}, S_{k-2} >=$$

$$= \beta_{k-2} < \nabla f(\bar{x}_{k-1}), S_{k-2} > + \beta_{k-2} \lambda^*_{k-1} < Q S_{k-1}, S_{k-2} > .$$

Легко заметить, что получили сумму двух скалярных произведений ортогональных векторов. Таким образом,

$$< \nabla f(\bar{x}_k), \nabla f(\bar{x}_{k-1}) >= 0 .$$

Это означает, что вектора $\nabla f(\bar{x}_k), \nabla f(\bar{x}_{k-1})$ ортогональны. Вообще можно показать, что все градиенты $\nabla f(\bar{x}_0), \nabla f(\bar{x}_1), ..., \nabla f(\bar{x}_k)$ взаимно ортогональны.

Итак, в окончательном виде имеем:

$$\beta_{k-1} = \frac{< \nabla f(\bar{x}_k), \nabla f(\bar{x}_k) >}{< \nabla f(\bar{x}_{k-1}), \nabla f(\bar{x}_{k-1}) >} = \frac{\left\| \nabla f(\bar{x}_k) \right\|^2}{\left\| \nabla f(\bar{x}_{k-1}) \right\|^2} .$$

Последнюю формулу можно применять для неквадратичных функций, так как в нее не входит матрица Q.

Метод сопряженных градиентов является одним из наиболее эффективных методов минимизации достаточно гладких функций. При определенных условиях метод обладает квадратичной скоростью сходимости:

$$\left\| \bar{x}_{k-1} - \bar{x}^* \right\| \le c \left\| \bar{x}_k - \bar{x}^* \right\|^2, \quad c > 0 .$$

3.3.3. Методы второго порядка

3.3.3.1. Метод Ньютона (метод Ньютона – Рафсона)

В методах первого порядка для определения направления убывания функции используется лишь линейная часть разложения функции в ряд Тейлора. Если минимизируемая функция дважды непрерывно дифференцируема, а первая и вторая производные вычисляются достаточно просто, то возможно применение методов минимизации второго порядка, которые используют квадратичную часть разложения этой функции в ряд Тейлора. Поскольку квадра-

тичная часть разложения аппроксимирует функцию гораздо точнее, чем линейная, то естественно ожидать, что методы второго порядка сходятся быстрее, чем методы первого порядка.

Метод Ньютона является прямым обобщением метода отыскания корня уравнения $\varphi(x) = 0$, где $\varphi(x)$ – функция скалярной переменной. Разложение в ряд Тейлора до первого члена позволяет переписать уравнение в следующем виде:

$$0 = \varphi(x_{k+1}) \approx \varphi(x_k) + \varphi'(x_k)(x_{k+1} - x_k).$$

Тогда при определенных условиях можно улучшить приближение к значению корня следующим образом:

$$x_{k+1} = x_k - \frac{\varphi(x_k)}{\varphi'(x_k)}.$$

Нас интересует n-мерная задача оптимизации, сводящаяся фактически к определению корня уравнения $\nabla f(\bar{x}) = \bar{0}$. Разложение в ряд Тейлора в этом случае дает:

$$\bar{0} = \nabla f(\bar{x}_{k+1}) \approx \nabla f(\bar{x}_k) + Q(\bar{x}_k)(\bar{x}_{k+1} - \bar{x}_k),$$

где $Q(\bar{x}_k)$ – матрица Гессе.

Отсюда

$$\bar{x}_{k+1} = \bar{x}_k - Q^{-1}(\bar{x}_k)\nabla f(\bar{x}_k)$$

при условии, что существует обратная матрица $Q^{-1}(\bar{x}_k)$. Эта формула и определяет метод Ньютона минимизации функции.

Если функция квадратичная

$$f(\bar{x}) = a + \bar{x}^{\mathrm{T}}\bar{b} + \frac{1}{2}\bar{x}^T Q\bar{x},$$

где Q – положительно определенная матрица, то, исходя из произвольной начальной точки $\bar{x}_0$, с помощью полученной выше формулы можно получить следующую точку:

$$\bar{x}_1 = \bar{x}_0 - Q^{-1}(\bar{b} + Q\bar{x}_0) = -Q^{-1}\bar{b}.$$

Эта точка является точкой минимума квадратичной функции.

Таким образом, для квадратичной функции метод Ньютона определяет точку минимума за один шаг.

3.3.3.2. Сходимость метода Ньютона

На сходимость метода Ньютона большое влияние оказывает матрица Гессе. Одной из причин расходимости метода является то, что матрица Гессе не является положительно определенной. В этом случае функция будет увеличиваться в направлении $Q^{-1}(\bar{x}_k)\nabla f(\bar{x}_k)$, а не уменьшаться. Кроме того, матрица Гессе на протяжении всего процесса должна быть невырожденной, так как необходимо существование обратной матрицы.

Если матрица удовлетворяет этим условиям и, кроме того, является ограниченной и удовлетворяет условию Липшица, то существует некоторая окрестность точки минимума $\bar{x}^*$, такая, что для любой начальной точки $\bar{x}_0$ из этой окрестности метод Ньютона сходится с квадратичной скоростью:

$$\left\|\bar{x}_{k-1} - \bar{x}^*\right\| \le c\left\|\bar{x}_k - \bar{x}^*\right\|^2, \quad c \ge 0 .$$

Таким образом, для сходимости метода начальная точка $\bar{x}_0$ должна выбираться достаточно близко к искомой точке минимума $\bar{x}^*$.

Подводя итог, можно следующим образом сформулировать достоинства и недостатки метода.

Недостатки:

необходимость вычислять и обращать матрицу вторых производных. В ряде задач трудоемкость итерации метода Ньютона может за счет этого оказаться непомерно большой;

сходимость метода зависит от выбора начальной точки $\bar{x}_0$. В связи с этим возникает проблема выбора начальной точки, которая должна находиться в достаточно малой окрестности минимума.

Достоинства:

вблизи точки минимума метод обеспечивает более быструю сходимость, чем градиентные методы;

общий объем вычислений может оказаться меньше, хотя трудоемкость каждой итерации в методе Ньютона больше, чем в методах 1-го порядка.

Ниже приводятся некоторые модификации метода Ньютона, направленные на устранение указанных выше недостатков.

3.3.3.3. Метод Ньютона с регулировкой шага

В некоторых случаях можно управлять сходимостью метода Ньютона, изменяя размеры шагов, т.е. определяя последовательность точек по формуле

$$\bar{x}_{k+1} = \bar{x}_k - \rho_k Q^{-1}(\bar{x}_k)\nabla f(\bar{x}_k),$$

где $0 < \rho_k \le 1$ (обычному методу Ньютона соответствует $\rho_k = 1$).

В этом случае метод называется методом Ньютона с регулировкой шага, или обобщенным методом Ньютона.

Существуют два способа выбора параметра ρ_k.

1. Первый способ заключается в следующем:

1) выбирается $\rho_k = 1$ и вычисляется точка $\bar{x} = \bar{x}_k + \bar{p}_k$, где

$$\bar{p}_k = -Q^{-1}(\bar{x}_k)\nabla f(\bar{x}_k);$$

2) вычисляется $f(\bar{x}) = f(\bar{x}_k + \bar{p}_k)$;

3) производится проверка неравенства

$$f(\bar{x}) - f(\bar{x}_k) \le (\varepsilon\rho_k < \nabla f(\bar{x}_k), \bar{p}_k >),$$

$$0 < \varepsilon < \frac{1}{2},$$

4) если это неравенство выполняется, то в качестве искомого значения ρ_k берем $\rho_k = 1$. В противном случае производится дробление ρ_k до тех пор, пока неравенство не выполнится.

2. В другом варианте метода, значение ρ_k выбирается из условия минимума функции в направлении движения:

$$f(\bar{x}_k - \rho_k Q^{-1}(\bar{x}_k)\nabla f(\bar{x}_k)) = \min_{\rho \ge 0} f\{\bar{x}_k - \rho Q^{-1}(\bar{x}_k)\nabla f(\bar{x}_k)\}.$$

Преимущество данной модификации метода Ньютона состоит в том, что метод сходится при любом выборе начального приближения $\bar{x}_0$. Скорость сходимости при этом остается квадратичной, трудоемкость каждой итерации увеличивается. Однако при первом способе выбора параметра ρ_k трудоемкость каждой итерации несколько увеличивается.

Сравнение двух способов регулировки длины шага говорит в пользу первого, потому что он оказывается менее трудоемким по количеству вычислений и обеспечивает такой же порядок скорости сходимости.

Замечания. Как было отмечено в п. 3.3.3.2, на сходимость метода Ньютона большое влияние оказывает матрица Гессе. Если она не является положительно определенной, то метод расходится. Если она положительно определенная (т.е. выпуклая $f(\bar{x})$), то метод сходится в окрестности точки минимума.

Определить, является ли матрица положительно определенной можно либо используя критерий Сильвестра, либо собственные числе матрицы Гессе. Матрица будет положительно определенная, если все угловые миноры матрицы положительны (критерий Сильвестра), или все собственные числа матрицы положительны.

В качестве примера рассмотрим следующие задачи.

Задача 1. Рассмотрим функцию

$$f(\bar{x}) = 2x_1^2 + x_2^2 + \sin(x_1 + x_2).$$

Проверить является ли для данной функции матрица Гессе положительно определенной?

Решение. Градиент функции имеет вид:

$$\nabla f(\bar{x}) = \begin{vmatrix} 4x_1 + \cos(x_1 + x_2) \\ 2x_2 + \cos(x_1 + x_2) \end{vmatrix}.$$

Матрица Гессе:

$$Q(\bar{x}) = \begin{vmatrix} 4 - \sin(x_1 + x_2) & -\sin(x_1 + x_2) \\ -\sin(x_1 + x_2) & 2 - \sin(x_1 + x_2) \end{vmatrix}.$$

Угловыми минорами будут:

$$\Delta_1 = 4 - \sin(x_1 + x_2) \quad \text{и} \quad \Delta_2 = 8 - 6\sin(x_1 + x_2) > 0 .$$

Так как $\left|\sin(x_1 + x_2)\right| \le 1$, то $\Delta_1 > 0$ и $\Delta_2 > 0$.

Следовательно, согласно критерию Сильвестра $Q(x)$ – положительно определенная матрица.

Задача 2. Проверить положительную определенность матрицы Гессе функции:

$$f(\overline{x}) = 4x_1^2 - x_2^2 - 2x_1x_2 + 6x_1 - x_2 - 2 .$$

Решение. Легко установить, что матрица Гессе данной функции будет:

$$Q = \begin{vmatrix} 8 & -2 \\ -2 & -2 \end{vmatrix}.$$

По критерию Сильвестра матрица Гессе не является положительно определенной, так как её главный определитель $\Delta = -16 - 4 = -20 < 0$.

Проверим полученный результат, используя собственные числа матрицы Гессе:

$$\begin{vmatrix} 8-\lambda & -2 \\ -2 & -2-\lambda \end{vmatrix} = \lambda^2 - 6\lambda - 20 = 0 ,$$

$$\lambda_1 = \frac{6+\sqrt{116}}{2} > 0 , \ \lambda_2 = \frac{6-\sqrt{116}}{2} < 0 .$$

Как видим, не все собственные числа положительны. Следовательно, матрица Гессе – не положительно определенная. Метод Ньютона расходится.

3.3.4. Метод переменной метрики (метод Девидона)

Как было отмечено выше, использование метода Ньютона связано с необходимостью вычисления матрицы Гессе исследуемой функции и последующего обращения этой матрицы. Во многих

случаях вычисление матрицы Гессе может быть связано с большими трудностями (например, она может быть получена только численными методами). Кроме того, известно, что число умножений при обращении матрицы размера $[n \times n]$, приблизительно, пропорционально n^3, что представляет собой большую величину уже при относительно небольших значениях n.

Рассмотрим метод, позволяющий обойти указанные выше трудности. В тех случаях, когда есть возможность вычисления градиентов, метод переменной метрики оказывается наиболее эффективным (т.е. он относится к группе методов первого порядка, но сохраняет высокую скорость сходимости метода Ньютона и, по этой причине, относится к квазиньютоновским методам).

Основная итерационная формула метода имеет вид:

$$\bar{x}_{k+1} = \bar{x}_k - \lambda_k \eta_k \nabla f(\bar{x}_k), \tag{3.14}$$

где η_k – матрица, аппроксимирующая обратную матрицу Гессе, а λ_k – шаг, который выбирается путем минимизации функции в направлении $\left(-\eta_k \nabla f(x_k)\right)$.

Кстати, формула (3.14) является общей для градиентных методов и метода Ньютона. В методе наискорейшего спуска роль η_k играет единичная матрица, а в методе Ньютона – обратная матрица Гессе.

В рассматриваемом методе матрица η_k вычисляется по рекуррентной формуле:

$$\eta_{k+1} = \eta_k + A_k - B_k, \tag{3.15}$$

где

$$A_k = \frac{\Delta\bar{x}_k (\Delta\bar{x}_k)^{\mathrm{T}}}{(\Delta\bar{x}_k)^{\mathrm{T}} \Delta\bar{g}_k}, \quad B_k = \frac{\eta_k \Delta\bar{g}_k (\Delta\bar{g}_k)^{\mathrm{T}} \eta_k{}^{\mathrm{T}}}{(\Delta\bar{g}_k)^{\mathrm{T}} \eta_k \Delta\bar{g}_k},$$

$$\Delta\bar{x}_k = \bar{x}_{k+1} - \bar{x}_k,$$

$$\Delta\bar{g}_k = \nabla f(\bar{x}_{k+1}) - \nabla f(\bar{x}_k).$$

В качестве начального значения η_0 для организации рекуррентного процесса (3.15) принимается $\eta_0 = E$, где E – единичная матрица. Таким образом, начальное направление поиска экстремума совпадает с направлением поиска в методе наискорейшего спус-

ка, при этом в ходе работы алгоритма осуществляется постепенный переход к ньютоновскому направлению. Доказательство этого может быть дано лишь для квадратичной функции

$$f(\bar{x}) = a + \vec{x}^{\mathrm{T}}\bar{b} + \frac{1}{2}\bar{x}^{\mathrm{T}}Q\bar{x}$$

с положительно определенной матрицей Гессе Q. Оказывается, что роль матрицы A_k состоит в том, чтобы обеспечить сходимость матрицы η_{k+1} к обратной матрице Гессе Q^{-1}, а роль матрицы B_k в том, чтобы обеспечить положительную определенность матрицы η_{k+1} и, в пределе, исключить влияние произвольного задания матрицы η_0.

Действительно,

$$\begin{aligned}
&\eta_0 = E,\\
&\eta_1 = E + A_0 - B_0,\\
&\eta_2 = \eta_1 + A_1 - B_1 = E + (A_0 + A_1) - (B_0 + B_1),\\
&\vdots\\
&\eta_{k+1} = E + \sum_{i=0}^{k} A_i - \sum_{i=0}^{k} B_i.
\end{aligned}$$

В случае квадратичной функции, сумма матриц A_i ($i = \overline{1,k}$), при $k = n-1$, равна обратной матрице Q^{-1}, а сумма матриц B_i равна матрице E. Это легко доказать, если в начале показать с помощью математической индукции, что вырабатываемые методом Девидона векторы направлений $\Delta\bar{x}_0$, $\Delta\bar{x}_1$, …, $\Delta\bar{x}_{n-1}$ образуют множество Q-сопряженных векторов.

Данный метод можно рассматривать как один из вариантов метода сопряженных направлений. Он решает задачу минимизации квадратичной функции за конечное число шагов, не превосходящее n.

Покажем, что $\sum_{i=0}^{n-1} A_i = Q^{-1}$.

Из соотношений (3.15), учитывая, что $\nabla f(\bar{x}) = Q\bar{x} + \bar{b}$, получим:

$$\Delta\bar{g}_k = \nabla f(\bar{x}_{k+1}) - \nabla f(\bar{x}_k) = Q(\bar{x}_{k+1} - \bar{x}_k) = Q\Delta\bar{x}_k \cdot$$

Принимая во внимание последнее выражение, можно записать:

$$\left(\sum_{i=0}^{n-1} A_i\right) = \sum_{i=0}^{n-1} \frac{\Delta\bar{x}_i (\Delta\bar{x}_i)^{\mathrm{T}}}{(\Delta\bar{x}_i)^{\mathrm{T}} Q \Delta\bar{x}_i}.$$

Рассмотрим выражение

$$\left(\sum_{i=0}^{n-1} A_i\right) Q\Delta\bar{x}_j = \sum_{i=0}^{n-1} \frac{\Delta\bar{x}_i (\Delta\bar{x}_i)^{\mathrm{T}} Q \Delta\bar{x}_j}{(\Delta\bar{x}_i)^{\mathrm{T}} Q \Delta\bar{x}_i}. \qquad (3.16)$$

Так как вектора $\Delta\bar{x}_i, \Delta\bar{x}_j$ $(i, j = \overline{1,n}; i \neq j)$ – взаимно сопряженные, то все слагаемые в формуле (3.16), кроме j-го слагаемого, равны нулю.

Таким образом, получаем:

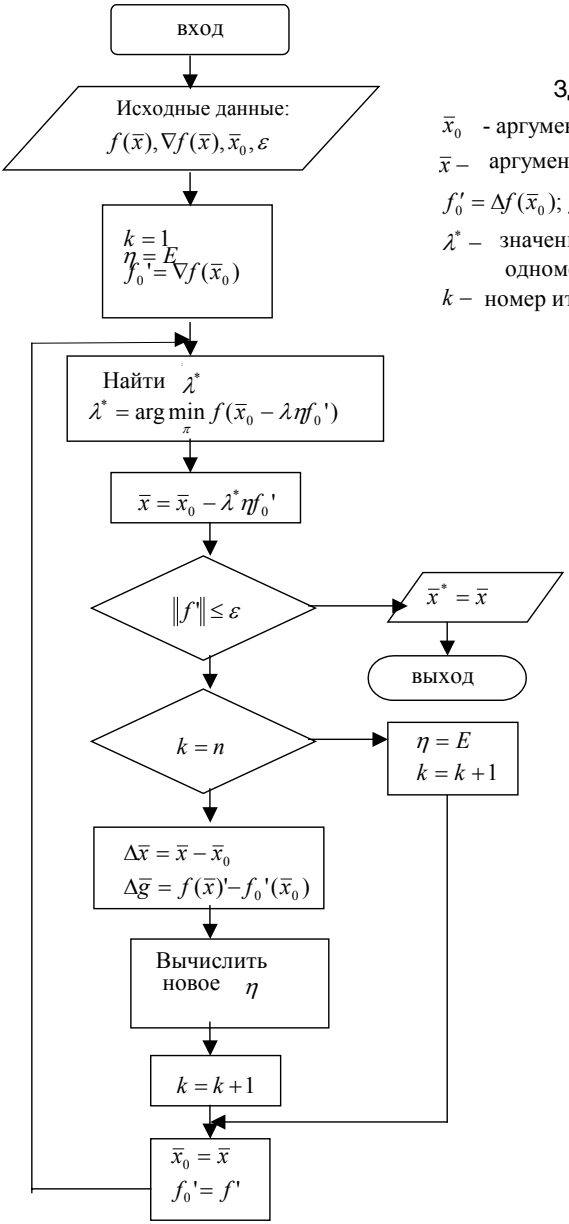
$$\left(\sum_{i=0}^{n-1} A_i\right) Q\Delta\bar{x}_j = \frac{\Delta\bar{x}_j (\Delta\bar{x}_j)^{\mathrm{T}} Q \Delta\bar{x}_j}{(\Delta\bar{x}_j)^{\mathrm{T}} Q \Delta\bar{x}_j} = \Delta\bar{x}_j.$$

Очевидно, такое возможно только, если $\left(\sum_{i=0}^{n-1} A_i\right) = Q^{-1}$.

Также нетрудно доказать, что $\sum_{i=0}^{n-1} B_i = E$.

Метод переменной метрики, так же как и метод сопряженных направлений, требует, чтобы нахождение минимума функции на данном направлении осуществлялось очень точно. В противном случае направления поиска не будут Q-сопряженными.

Если сравнить метод переменной метрики с методом сопряженных градиентов, то оказывается, что первый обеспечивает существенно более быструю сходимость, чем второй, но в большей мере подвержен влиянию ошибок вычислений. Поэтому часто используют метод переменной метрики со следующей модификацией: через конечное число шагов (обычно n) осуществляют обновление матрицы η_k, т.е. полагают $\eta_k = E$ и процесс начинается, как бы, сначала. На рис. 3.26 приведена блок-схема метода переменной метрики с периодическим обновлением матрицы η_k.



Здесь:

$\bar{x}_0$ - аргумент в начале итерации;

$\bar{x}$ – аргумент в конце итерации;

$f_0' = \Delta f(\bar{x}_0)$; $f' = \Delta f(\bar{x})$.

λ^* – значение λ, доставляющее одномерный минимум;

k – номер итерации.

Рис. 3.26. Блок-схема метода переменной метрики

3.4. Минимизация функций с ограничениями

При формулировке реальных задач оптимизации обычно приходится на переменные накладывать некоторые ограничения. В результате точки $\bar{x}$ выбираются не произвольно, а лишь из некоторого подмножества X пространства R^n. Подмножество X обычно задается неявно системой уравнений ограничений, которая состоит из ограничивающих равенств, неравенств или тех и других вместе.

Множество X называют допустимой областью, а точку $\bar{x} \in X$ – допустимым решением.

Итак, задача состоит в следующем:

найти $\min(f(\bar{x}))$ при ограничениях

$$g_i(\bar{x}) \le 0, \quad i = \overline{1,m}. \tag{3.17}$$

3.4.1. Метод штрафных функций

Так как минимизация функции без ограничений представляет собой более легкую задачу, чем минимизация с ограничениями, то естественными являются попытки преобразования задач с ограничениями в задачи без ограничений. Существует несколько способов такого преобразования. В некоторых случаях для исключения ограничений, накладываемых на $\bar{x}$, может быть использована простая замена переменных.

Например, если скалярная переменная x ограничена условием $|x| \le 1$, то можно использовать следующую замену переменной:

$$x = \sin\theta \quad \text{или} \quad x = \cos\theta.$$

Если переменная x ограничена условием $x \ge 0$, то снять ограничения можно заменой $x = y^2$. При этом переменная y не ограничена.

Если $a \le x \le b$, то возможна замена: $x = a + (b-a)\sin^2\theta$. Эти приемы имеют ограниченное применение, тем не менее ими пренебрегать нельзя.

Более широкое применение получил, так называемый, *метод штрафных функций*. Это один из наиболее простых и широко из-

вестных методов решения задач нелинейного программирования. Основная идея метода состоит в приближенном сведении задачи минимизации при ограничениях к задаче минимизации некоторой функции без ограничений. При этом вспомогательная функция подбирается так, чтобы она совпадала с заданной минимизируемой функцией внутри допустимой области и быстро возрастала вне ее. Вспомогательная функция представляет собой сумму минимизируемой функции и функции штрафа:

$$F(\bar{x},l) = f(\bar{x}) + \sum_i \psi_i(g_i(\bar{x}),l_i),$$

где $\bar{l}$ – некоторый векторный параметр $\bar{l} = \{l_i\}$, $i = \overline{1,m}$; $\psi_i(g_i(\bar{x}),l_i)$, $i = \overline{1,m}$ – функция штрафа, которая обладает следующим свойством:

$$\lim_{l_i \to \infty} (\psi_i(g_i(\bar{x}),l_i)) = \begin{cases} 0 & \text{при} \quad g_i(\bar{x}) \le 0, \quad i = \overline{1,m}; \\ +\infty & \text{в противном случае.} \end{cases}$$

При таком задании штрафных функций $F(\bar{x},\bar{l})$ в области X близко к $f(\bar{x})$, а вне области X эта функция принимает большие значения.

Идея метода штрафных функций состоит в том, чтобы вместо задачи (3.17) рассматривать задачу минимизации функции $F(\bar{x},\bar{l})$ при больших l_i.

В общем случае $F(\bar{x},\bar{l})$ строится так, чтобы она была гладкой и была возможность применить один из быстро сходящихся методов минимизации функций без ограничений.

Например, штрафная функция может быть построена как взвешенная сумма квадратов невязок

$$F(\bar{x},\bar{l}) = f(\bar{x}) + \sum_{i=1}^{m} l_i \varphi^2(g_i(\bar{x})),$$

где

$$\varphi(g_i(\bar{x})) = \begin{cases} g_i(\bar{x}), \text{если } g_i(\bar{x}) \ge 0, \quad i = \overline{1,m}; \\ 0, \text{если } g_i(\bar{x}) < 0, \quad i = \overline{1,m}. \end{cases}$$

Замена решения задачи (3.17) минимизацией функции $F(\bar{x},\bar{l})$ при больших $\bar{l}$ позволяет приблизиться к решению исходной задачи.

Однако здесь возникают следующие вычислительные трудности.

1. Если функции $g_i(\bar{x})$ – невыпуклые, то $F(\bar{x},\bar{l})$ также не будет выпуклой по $\bar{x}$. Поэтому она может обладать локальными минимумами. Так как все изученные нами методы предназначены для нахождения локального минимума, то при плохом начальном приближении $\bar{x}_0$ будет найден локальный минимум функции $F(\bar{x},\bar{l})$, не совпадающий с минимумом исходной задачи.

Если функции $g_i(\bar{x})$ – выпуклые, то $F(\bar{x},\bar{l})$ также будет выпуклой и данная проблема устраняется.

2. Для получения хорошего приближения следует брать большие значения $\bar{l}$. При этом все производные по $\bar{x}$ также будут большими, ибо они пропорциональны $\bar{l}$. Однако это приводит к ухудшению сходимости методов безусловной минимизации (градиентный метод, метод сопряженных градиентов и другие методы, использующие первую производную). Окрестность, в которой методы обладают высокой скоростью сходимости, становится очень маленькой.

3. Функция $F(\bar{x},\bar{l})$ в точках $\bar{x}$, для которых $g_i(\bar{x})=0$ при некоторых $\bar{l}$ не имеет вторых производных, т.е. градиент в этих точках имеет разрыв. Но если решение $\bar{x}^*$ лежит на границе допустимой области (а это бывает часто), то возникает трудность со сходимостью, так как все быстросходящееся методы требуют наличия у минимизируемой функции вторых производных, по крайней мере, в некоторой окрестности точки.

Все указанные трудности, как правило, проявляют себя в практических расчетах, что снижает эффективность метода.

3.4.2. Метод Фиакко и Мак-Кормика (метод барьерных функций, метод внутренней точки)

Этот метод основан на идее, близкой к методу штрафных функций, при этом в нем используется иная форма записи вспомогательной функции.

Итак, надо найти $\min f(\bar{x})$ при ограничениях $g_i(\bar{x}) \le 0$, где функции $g_i(\bar{x})$ – выпуклые и допустимая область не пуста.

Составим вспомогательную функцию

$$F(\vec{x},k) = f(\bar{x}) - k\sum_{i=1}^{m}\frac{1}{g_i(\bar{x})}, \quad k > 0.$$

Когда $\bar{x}$ приближается к границам области X (изнутри), значения, по меньшей мере, одной из ограничивающих функций приближаются из области отрицательных значений к нулю. В этом случае к функции $f(\bar{x})$ добавляется большая положительная величина. При $k \to 0$ минимум функции $F(\bar{x},k)$ стремится к минимуму функции $f(\bar{x})$ с ограничениями $g_i(\bar{x}) \le 0$.

Для повышения точности важно выбирать малые значения k. Однако при малых k небольшие изменения $\bar{x}$ приводят к резким изменениям функции $F(\bar{x},k)$. Другими словами, изменения градиента $F(\bar{x},k)$ вблизи границ допустимой области становятся более резкими. Это затрудняет поиск минимума и значительно снижает ценность метода.

3.4.3. Методы возможных направлений

Методы возможных направлений – наиболее исследованный класс методов выпуклого программирования (задачи выпуклого программирования – это такие задачи нелинейного программирования, в которых целевые функции и функции ограничений выпуклы). Подробное описание данных методов было произведено в работах Зойтендейка.

Хотя сходимость к глобальному минимуму может быть доказана только в случаях задач выпуклого программирования, тем не менее

методы возможных направлений сходятся к локальному минимуму и в применении к задачам нелинейного программирования вообще.

Рассмотрим сущность этих методов.

Имеем задачу нелинейного программирования:

найти $\min f(\bar{x})$ при ограничениях

$$g_i(\bar{x}) < 0, \quad i = \overline{1, m},$$

где $g_i(\bar{x})$, $f(\bar{x})$ – непрерывно дифференцируемые функции.

Последовательность точек будем, как всегда, строить по итерационной формуле

$$\bar{x}_{k+1} = \bar{x}_k + \lambda_k \cdot S_k .$$

Как известно, имея допустимую точку $\bar{x}$, удовлетворяющую всем ограничениям, необходимо принять два решения:

1) выбрать направление S, которое должно быть *возможным и приемлемым*, т.е. на этом направлении должны лежать точки, принадлежащие допустимой области X (возможность), и функция $f(\bar{x})$ должна в этом направлении убывать (приемлемость);

2) решить, какой величины шаг должен быть сделан в выбранном направлении S.

Вообще говоря, существует много возможных и приемлемых направлений.

При выборе «лучшего» направления S, т.е. такого, в котором функция убывает в наибольшей степени, минимизируют $< \nabla f(\bar{x}), S >$.

Функция убывает в направлении S, если это скалярное произведение меньше нуля, т.е. направление S образует острый угол с антиградиентом (рис. 3.27).

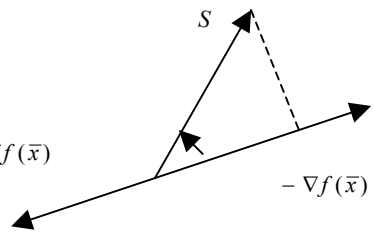


Рис. 3.27. К выбору допустимых направлений

Будем полагать, что в точке $\bar{x}$ имеется хотя бы одно активное ограничивающее уравнение $g(\bar{x}) = 0$, при этом точка $\bar{x}$ лежит на границе допустимой области. В противном случае нет проблем с выбором направления – это антиградиент.

Предположим, что антиградиент не является возможным направлением, то есть при движении по нему нарушается, по крайней мере, одно ограничивающее уравнение.

Если ограничивающее уравнение линейно, то «лучшее» направление S получается проекцией вектора $-\nabla f(\bar{x})$ на многообразие, характеризуемое линейным ограничивающим уравнением (рис. 3.28).

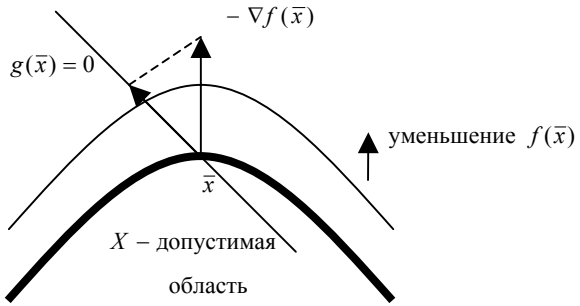


Рис. 3.28. Проекция антиградиента на активное ограничение

Если активное ограничивающее уравнение является нелинейным, проблема выбора направления становится сложнее. Помимо условия $<\nabla f(\bar{x}), S>< 0$ (движение в сторону уменьшения $f(\bar{x})$), должно удовлетворяться условие $<\nabla g(\bar{x}), S>< 0$ (движение в сторону уменьшения $g(\bar{x})$), где $g(\bar{x})$ – активное ограничивающее неравенство (рис. 3.29).

Возможными являются направления S, для которых $<\nabla g(x), S>< 0$ (при условии выпуклости $g(\bar{x})$).

Оказывается, что при выборе возможного направления нужно принимать во внимание не только ограничения, которые в данной точке выполняются точно, но и те, которые выполняются «почти точно», так как в противном случае может произойти сколь угодно сильное измельчение шага вдали от точки минимума. При этом

процесс не обязательно будет приводить в точку минимума. Эти соображения включены в алгоритм выбора направления.

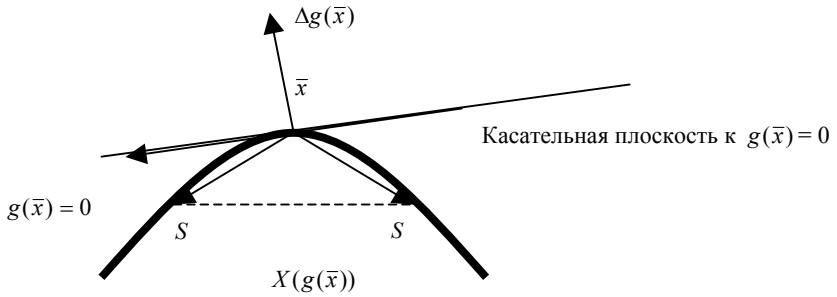


Рис. 3.29. Выбор возможных направлений
при нелинейном ограничивающем уравнении

Возможное и приемлемое направление S *получается путем решения следующей вспомогательной задачи условной оптимизации*:

найти $\max\{\varphi(\sigma, S) = \sigma\}$ при ограничениях:

$$< \nabla g_i(\bar{x}), S > + \sigma \le 0, \quad i \in I_\varepsilon;$$

$$< \nabla f(\bar{x}), S > + \sigma \le 0 .$$

$$< S, S > = 1 .$$

В данной задаче $I_\varepsilon = \{i : -\varepsilon \le g_i(\bar{x}) \le 0; \quad 0 < \varepsilon << 1\}$, т.е. множество I_ε включает в себя индексы «почти активных» ограничений, значения которых находятся в ε-близости от границы ($-\varepsilon_i \le g_i(\bar{x}) \le 0$).

Очевидно, если полученное в результате решения вспомогательной задачи максимальное значение $\sigma > 0$, то смещение в направлении S приводит к уменьшению значения функции $f(\bar{x})$, что следует из второго условия в ограничениях вспомогательной задачи, и не нарушает никаких ограничений основной задачи, что следует из первого условия в ограничениях вспомогательной задачи. Последнее ограничение – это условие нормировки вектора S. Чаще всего это ограничение заменяют следующим: $-1 \le s_j \le 1$,

$j=\overline{1,n}$, s_j – элемент вектора направления S. Таким образом, вспомогательная задача становиться задачей линейного программирования, для решения которой существуют конечно-шаговые методы (симплекс-методы). Неизвестными параметрами в этой задаче линейного программирования являются σ, и элементы вектора S $(s_1, s_2, \ldots, s_n)$.

Рассмотрим один из алгоритмов метода возможных направлений.

В качестве начального приближения $\bar{x}_0$ может быть выбрана любая точка множества X, а ε_0 выбирают из интервала $(0,1]$.

Пусть в результате k-й итерации вычислены $\bar{x}_k$ и ε_k. Опишем $k+1$ итерацию.

1. Решить вспомогательную задачу и вычислить $\sigma_k \geq 0$ и S_k (если I_ε пусто, т.е. $\bar{x}_k$ – внутренняя точка, то S_k совпадает с антиградиентом).

2. Если $\sigma_k \geq \varepsilon_k$, то перейти к вычислению величины шага λ_k. Это можно сделать одним из двух способов:

а) решая задачу одномерной минимизации функции $\varphi(\lambda) = f(\bar{x}_k) + \lambda S_k$, причем λ должна лежать в интервале $0 \leq \lambda \leq \lambda_{\text{доп}}$. Здесь $\lambda_{\text{доп}}$ – расстояние от точки $\bar{x}_k$ до ближайшей по направлению S_k точки границы множества X, т.е. точка $\bar{y} = \bar{x}_k + \lambda_{\text{доп}} S_k$ принадлежит границе множества X (если луч $\bar{x}_k + \lambda_{\text{доп}} S_k$ не пересекается с границей, а полностью принадлежит множеству X, то $\lambda_{\text{доп}} = +\infty$);

б) число λ_k можно выбрать так, чтобы функция $f(\bar{x})$ была в точке $\bar{x}_{k+1}$ меньше, чем в точке $\bar{x}_k$.

Например, в качестве λ_k можно взять наибольшее из чисел, удовлетворяющих соотношениям:

$$f(\bar{x}_k) - f(\bar{x}_k + \lambda_k S_k) \geq \frac{1}{2}\lambda_k \sigma_k ,$$

$$0 \leq \lambda_k \leq \lambda_{\text{доп}} .$$

После определения λ_k вычислить $\bar{x}_{k+1} = \bar{x}_k + \lambda_k S_k$, положить $\varepsilon_{k+1} = \varepsilon_k$ и прейти к шагу 1.

3. Если $0 < \sigma_k < \varepsilon_k$, то положить $\bar{x}_{k+1} = \bar{x}_k$, $\varepsilon_{k+1} = \gamma\varepsilon_k$, где γ удовлетворяет условию $0 < \gamma < 1$, и перейти к шагу 1.

4. Если $\sigma_k = 0$, то вычислить σ_k^*, решив вспомогательную задачу с $\varepsilon = 0$. Если $\sigma_k^* = 0$, то процесс поиска точки минимума закончен ($\bar{x}^* = \bar{x}_k$). В противном случае положить $\bar{x}_{k+1} = \bar{x}_k$, $\varepsilon_{k+1} = \gamma\varepsilon_k$ и перейти к шагу 1.

Замечание. Условие $\sigma_k^* = 0$ является необходимым и достаточным условием оптимальности точки $\bar{x}$, это доказывается с помощью специальной теоремы.

3.4.4. Метод проекции градиента

В случае задач безусловной минимизации весьма распространенным является метод градиентного спуска. Однако для задач с ограничениями направление вдоль антиградиента необязательно является возможным. В случае, когда множество X выпукло, для отыскания направления в точке $\bar{x}_k$ напрашивается мысль спроектировать точку $\bar{y}_k = \bar{x}_k - \nu_k \nabla f(\bar{x}_k)$ (ν_k – некоторое фиксированное положительное число) на множество X и в качестве направления спуска взять $S_k = \bar{p}_k - \bar{x}_k$, где $\bar{p}_k$ – проекция точки $\bar{y}_k$ на допустимое множество X (рис. 3.30). После этого надо осуществлять спуск вдоль полученного направления. Выпуклость множества X гарантирует, что такое направление S_k является возможным. Проекция точки $\bar{y}_k$ на множество – ближайшая к $\bar{y}_k$ точка этого множества.

Итак, метод проекции градиента состоит в вычислении проекции $\bar{p}_k$ точки $\bar{y}_k = \bar{x}_k - \nu_k \nabla f(\bar{x}_k)$ на множество X и в выборе шага λ_k таким образом, чтобы в точке $\bar{x}_{k+1} = \bar{x}_k - \lambda_k S_k$, где $S_k = \bar{p}_k - \bar{x}_k$, выполнялось условие движения к минимуму ($f(\bar{x}_{k+1}) < f(\bar{x}_k)$).

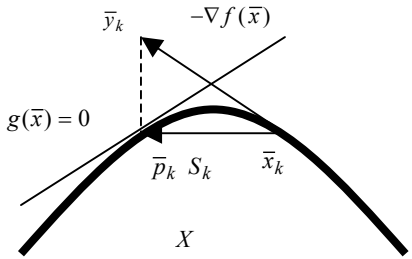


Рис. 3.30. Определение направления в методе проекции градиента

В зависимости от способа выбора шага λ_k можно получить различные варианты метода проекции градиента (можно использовать одномерную минимизацию и т.д.).

Условием прекращения процесса является условие $\bar{p}_k = \bar{x}_k$, которое является необходимым и достаточным условием того, что точка $\bar{x}_k$ – точка минимума (доказывается с помощью специальной теоремы).

Например, $\bar{p}_k = \bar{x}_k$, если $\nabla f(x_k) = 0$, так как тогда $\bar{y}_k = \bar{x}_k$, и нет перемещения из точки $\bar{x}_k$. Аналогичная ситуация возникает, если $\bar{x}_k$ лежит на границе и градиент ортогонален границе допустимой области (рис. 3.31).

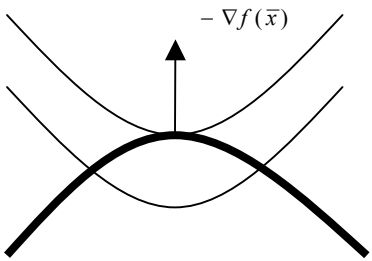


Рис. 3.31. Направление градиента совпадает с направлением градиента к границе допустимой области

Для того, чтобы найти проекцию точки $\bar{y}_k$ на множество X, необходимо решить задачу минимизации квадратичной функции

$\|\bar{y}_k - \bar{x}\|^2$ на множестве X (т.е. $\bar{x} \in X$) – тем самым будет найдена ближайшая к $\bar{y}_k$ точка множества X. В общем случае эта задача того же порядка сложности, что и исходная. Поэтому методом проекции градиента обычно пользуются лишь в тех случаях, когда проекция точки на множество легко определяется.

Например, когда множество X представляет собой шар или параллелепипед в E^n, гиперплоскость или полупространство. В этом случае задача проектирования точки решается просто и в явном виде. В частности, это имеет место, когда ограничивающие уравнения линейны.

Рассмотрим пример явного определения проекции точки на множестве X. В качестве области X будем рассматривать замкнутый шар радиуса r с центром в точке O в пространстве E^n.

Как известно, уравнение шара имеет вид:

$$\sum_{i=1}^{n} x_i^2 = r^2 ;$$

тогда допустимая область X описывается неравенством:

$$\sum_{i=1}^{n} x_i^2 \le r^2 .$$

Пусть $\bar{y}$ – некоторая точка, для которой надо найти проекцию $\bar{p}$ на поверхность шара. Очевидно, данная задаче эквивалентна задаче минимизации функции

$$\phi(\bar{x}) = \|\bar{y} - \bar{x}\|^2; \quad x \in X, \text{ при этом } \bar{p} = \arg\left(\min_{x \in X} \phi(\bar{x})\right).$$

Данную задачу можно записать в виде:

найти

$$\min \sum_{i=1}^{n} (y_i - x_i)^2$$

при ограничение

$$\sum_{i=1}^{n} x_i^2 \le r^2 .$$

Решение этой вспомогательной задачи находиться в явном виде:

$$\overline{p} = \begin{cases} \overline{y}, \text{ если } \sum\limits_{i=1}^{n} y_i^2 \le r^2; \\ \dfrac{r\overline{y}}{\sqrt{\sum\limits_{i=1}^{n} y_i^2}}, \text{ если } \sum\limits_{i=1}^{n} y_i^2 > r^2. \end{cases}$$

3.4.5. Метод проекции градиента при линейных ограничениях

Будем полагать, что ограничивающие уравнения линейны, т.е. имеют вид:

$$A\overline{x} - \overline{b} \le 0,$$

где A – матрица ($m \times n$); $\overline{b}$ – m-мерный вектор.

Назовем гранью многогранника X (допустимой области) подпространство, определяемое любой совокупностью активных ограничений.

Предположим, что при заданном $\overline{x} \in X$ грань, содержащая $\overline{x}$, определяется уравнением

$$g_i(\overline{x}) = 0, \quad i \in I,$$

где $I = \{i_1, i_2, \ldots, i_p\}$, $p < m$.

Предположим, что любая подматрица размерности $(p \times n)$ матрицы A имеет ранг p, и пусть A_I – подматрица, составленная из p строк матрицы A, соответствующих активным ограничениям. Тогда можно предложить следующий алгоритм поиска минимума функции $f(\overline{x})$ при линейных ограничениях.

Алгоритм (один шаг, после того как найдена точка $\overline{x}_k \in X$).

1. Вычислить проекцию градиента на грань, содержащую $\overline{x}_k$ по формуле

$$\Delta\overline{x} = \left[I_n - A_I^{\text{т}} \left(A_I A_I^{\text{т}} \right)^{-1} A_I \right] \nabla f(\overline{x}_k),$$

где I_n – единичная матрица $(n \times n)$. Матрица $\left(A_I A_I^{\text{т}}\right)^{-1}$ существует, так как по условию задачи ранг матрицы A_I равен числу строк в ней.

Если $\bar{x}_k$ лежит внутри X, то множество I пусто, в квадратных скобках остается одна единичная матрица I_n и $\Delta\bar{x} = \nabla f(x_k)$. В этом случае данный метод оказывается обычным градиентным методом.

2. Если $\Delta\bar{x} \neq 0$, то найти $\lambda_{\text{доп}}$, такое, что $\lambda_{\text{доп}}\Delta\bar{x}$ определяет максимальное перемещение в направлении $-\Delta\bar{x}$, которое может быть сделано, не выходя за пределы X; $\lambda_{\text{доп}}$ может быть найдено путем решения задачи:

$$\lambda_{\text{доп}} = \max\{\lambda: \quad (\bar{x}_k - \lambda\Delta\bar{x}_k \in X\}.$$

После этого найти λ^* как решение задачи одномерной оптимизации:

$$\lambda^* = \arg\left(\min_{0 \le \lambda \le \lambda_{\text{доп}}} f(\bar{x}_k - \lambda\Delta\bar{x}_k\right)$$

и вычислить $\bar{x}_{k+1} = \bar{x}_k - \lambda^*\Delta\bar{x}_k$.

Если $\lambda^* < \lambda_{\text{доп}}$, то совокупность активных ограничивающих уравнений остается без изменений.

Если $\lambda^* = \lambda_{\text{доп}}$, то граница области X будет достигнута и новое ограничивающее уравнение станет активным. Тогда, если $g_j(\bar{x}_{k+1}) = 0$, $j \notin I$, то надо включить в I это j, определить новую матрицу A_I и вернуться к шагу 1.

Если $\Delta\bar{x} = 0$, то необходимо вычислить вектор $\bar{\mu}$ (размерности p), исходя из соотношения

$$\bar{\mu} = (A_I A_I^{\text{т}})^{-1} A_I \nabla f(\bar{x}_k).$$

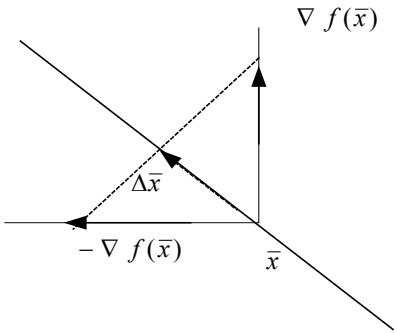
Если все компоненты $\mu_i \le 0$, $i = \overline{1, p}$, то $\bar{x}_k$ оптимально ($\bar{x}_k = \bar{x}^*$), и процесс вычисления прекращается.

В противном случае исключить из A_I строку, соответствующую наибольшей положительной компоненте вектора $\overline{\mu}$, и вернуться к шагу 1.

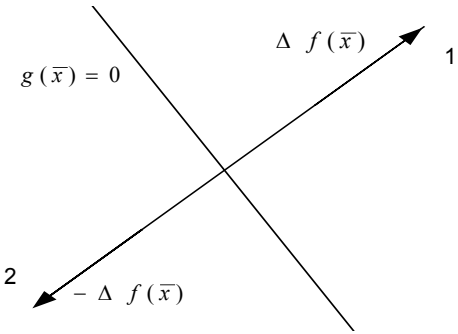
Дело в том, что компоненты вектора $\overline{\mu}$ эквивалентны множителям Лагранжа и точка $\overline{x}_k$ является оптимальной точкой в том случае, когда все $\mu_i \le 0$.

Однако, если некоторые компоненты вектора $\overline{\mu}$ положительны, то функция $f(\overline{x})$ может уменьшаться, если перемещаться от $\overline{x}_k$ в направлении уменьшения $g_i(\overline{x})$, т.е. в направлении, где $g_i(\overline{x}) < 0$. При этом ограничение $g_i(\overline{x})$ становится неактивным, i исключается из I.

Это можно проиллюстрировать следующим образом:



Алгоритм работает так, что проекции градиентов, расположенных симметрично относительно грани, одинаковы:



Следовательно, если $\nabla f(\bar{x}) \neq 0$, то ситуация, когда $\Delta\bar{x} = 0$, может возникнуть в двух случаях:

в случае 1 точка $\bar{x}$ является точкой оптимума, следовательно, вычисления необходимо прекратить;

в случае 2, как видно из рисунка, антиградиент направлен внутрь области X, следовательно, его проектировать на грань не нужно.

Когда, согласно алгоритму, из A_I будет убрана строка, соответствующая ограничению $g_i(\bar{x}) = 0$, в формуле для проекции останется I_n и получится, что $\Delta\bar{x} = \nabla f(\bar{x})$, т.е. направление поиска будет направлено внутрь области X в сторону убывания функции.

3.4.6. Метод условного градиента

Идея метода условного градиента состоит в выборе направления спуска на основе линеаризации функции $f(\bar{x})$ относительно текущей точки $\bar{x}_k$. В результате линеаризации получаем линейную функцию $f_L(\bar{x})$ вида:

$$f_L(\bar{x}) = f(\bar{x}_k) + \Delta f(\bar{x}_k)(\bar{x} - \bar{x}_k).$$

Пусть $\bar{y}_k$ – точка, обеспечивающая минимум функции $f_L(\bar{x})$ на множестве X. Тогда направление S_k поиска экстремума исходной функции $f(\bar{x})$ на множестве X можно определить следующим образом:

$$S_k = \bar{y}_k - \bar{x}_k,$$

при этом основная итерационная формула имеет стандартный вид:

$$\bar{x}_{k+1} = \bar{x}_k + \lambda_k S_k.$$

Таким образом, для определения направления S_k необходимо решить задачу минимизации линейной функции $f_L(\bar{x})$ на множестве X. В общем случае эта задача того же порядка сложности, что и исходная. Поэтому метод условного градиента применяют лишь тогда, когда вспомогательная задача решается просто. Например,

если множество X представляет собой шар, или параллелепипед. В этом случае решение задачи легко найти в явном виде.

Если ограничения, образующие множество X линейны, то рассматриваемая задача превращается в задачу линейного программирования, которая легко может быть решена симплекс-методом.

Причем если в задаче линейного программирования вспомогательная функция $f_L(\bar{x})$ неограничена для текущего $\bar{x}_k$ (такое возможно, несмотря на то, что исходная задача имеет решение), то можно для получения точки $\bar{y}_k$ ограничиться несколькими шагами симплекс-метода в сторону убывания функции $f_L(\bar{x})$.

Существуют различные способы выбора шага λ_k в методе условного градиента. Рассмотрим некоторые из них.

1. Величина шага λ_k может выбираться из условий:

$$0 \le \lambda_k \le 1;$$
$$\varphi(\lambda_k) = \min_{\lambda} f(\bar{x}_k + \lambda S_k).$$

Очевидно, такая задача может быть решена каким-либо методом одномерной минимизации.

2. Можно задавать $\lambda_k = 1$ и проверять условие монотонного убывания функции $f(\bar{x})$ при переходе от точки $\bar{x}_k$ к точке $\bar{x}_{k+1}$, а именно, $f(\bar{x}_{k+1}) < f(\bar{x}_k)$. Если это условие не выполняется, то необходимо дробить шаг λ_k до тех пор, пока не выполнится условие монотонности.

3. Величина шага определяется из условия $\lambda_k = \alpha^{i_0}$, где i_0 – минимальный номер среди номеров $i \ge 0$, удовлетворяющих условию:

$$f(\bar{x}_k) - f(\bar{x}_k + \alpha^i S_k) \ge \alpha^i \varepsilon \left|\eta(\bar{x}_k)\right|.$$

Здесь α, ε – параметры метода, $0 < \alpha < 1; 0 < \varepsilon < 1$. Величина $\eta(\bar{x}_k)$ является минимумом линейной функции $f_L(\bar{x})$, т.е.

$$\eta(\bar{x}_k) = \min_{\bar{x} \in X} f_L(\bar{x}) = \min_{\bar{x} \in X} < \nabla f(\bar{x}_k), (\bar{x} - \bar{x}_k) > .$$

Необходимо отметить, что величина $\eta(\bar{x}_k)$ в условии выбора параметра шага берется по модулю, так как $\eta(\bar{x}_k) \le 0$. Это следует из того, что при $\bar{x} = \bar{x}_k$ вспомогательная линейная функция $f_L(\bar{x})\big|_{\bar{x}=\bar{x}_k} = 0$, следовательно, значение $\eta(\bar{x}_k)$, обеспечивающее минимальное значение $f_L(\bar{x})$ меньше, или равно нулю.

Существуют теоремы, доказывающие сходимость метода условного градиента при любом из перечисленных способов выбора шага.

Процесс вычислений заканчивается в точке $\bar{x}_k$, если $\eta(\bar{x}_k) = 0$. Это может случиться, например, если $\nabla f(\bar{x}_k) = 0$. Из этого условия следует, что если

$$\min_{\bar{x}\in X} < \nabla f(\bar{x}_k), \bar{x} - \bar{x}_k >= 0,$$

то для всех $\bar{x} \in X$ будет выполняться:

$$< \nabla f(\bar{x}_k), \bar{x} - \bar{x}_k > \ge 0. \qquad (*)$$

Последнее, очевидно, является необходимым и достаточным (для выпуклой функции достаточным) условием того, чтобы точка $\bar{x}_k$ являлась минимумом функции $f(\bar{x})$ на множестве X (тоже выпуклом).

Это условие является обобщением условия стационарности $\nabla f(\bar{x}_k) = 0$ для задачи минимизации функций на множествах.

Проиллюстрируем условие $< \nabla f(\bar{x}_k), \bar{x} - \bar{x}_k > \ge 0$ графически (рис. 3.32).

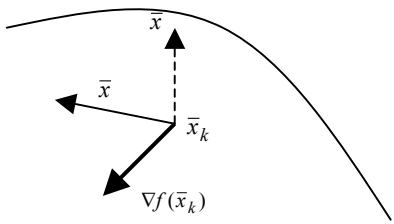


Рис. 3.32. Графическая интерпретация скалярного произведения $< \nabla f(\bar{x}_k),\ \bar{x} - \bar{x}_k > \ge 0$

Оно означает, что угол между градиентом в точке $\bar{x}_k$ и любым вектором $\bar{x} - \bar{x}_k$ для всех точек $\bar{x} \in X$ должен быть острым. Однако, если точка $\bar{x}_k$ лежит внутри области X, то этого не может быть. Следовательно, условие (*) справедливо, только в случае:

$$\nabla f(\bar{x}_k) = 0 .$$

Если точка $\bar{x}_k$ лежит на границе области, то минимум в этой точке достигается лишь тогда, когда антиградиент перпендикулярен границе и направлен из области X (рис. 3.33).

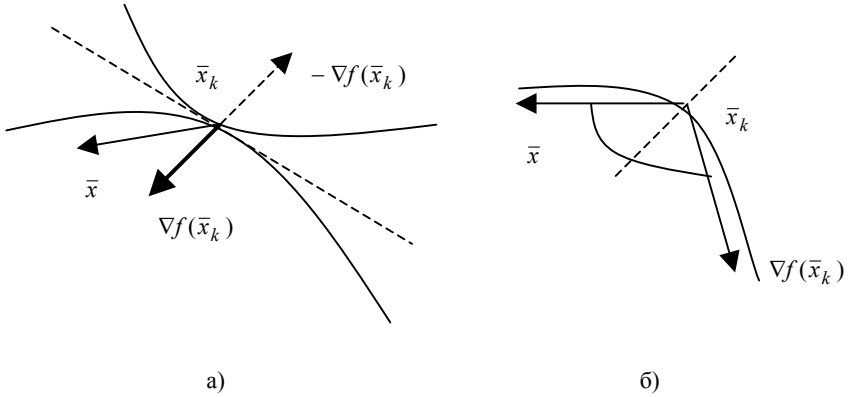


Рис. 3.33. Определение оптимального направления в случае, когда $\bar{x}_k$ лежит на границе области

В этом случае углы между градиентом и любым вектором $(\bar{x} - \bar{x}_k)$ будут острыми (рис. 3.33, *а*).

Если антиградиент отклоняется от данного направления, то найдутся точки $\bar{x}$, для которых эти углы будут тупыми (рис. 3.33, *б*).

Геометрический смысл метода условного градиента поясняет рис. 3.34.

Рассмотрим пример использования метода условного градиента, который решается в явном виде.

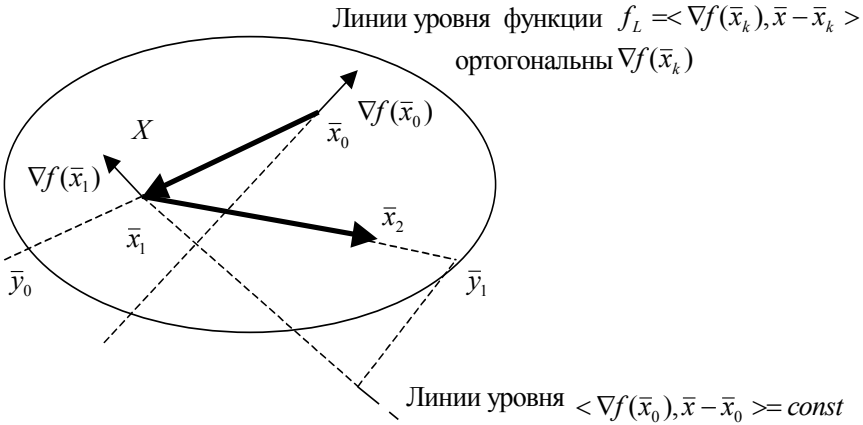


Рис. 3.34. Геометрический смысл метода условного градиента

Пример. Найти $\min f(\bar{x})$ при ограничении

$$\sum_{i=1}^{n} (x_i - \tilde{x}_i)^2 \le r^2 \quad .$$

Как видим, допустимая область представляет собой шар радиуса r в n-мерном пространстве с центром в точке $\tilde{\bar{x}}$.

Вспомогательная задача для метода условного градиента формулируется следующим образам.

Определить точку y_k, являющуюся решением задачи:

найти

$$\min_x \{ f_L(\bar{x}) = < \nabla f(\bar{x}_k), \bar{x} - \bar{x}_k > \}$$

при ограничении

$$\sum_{i=1}^{n} (x_i - \tilde{x}_i)^2 \le r^2 \ .$$

Решение оказывается следующим:

$$y_k = \tilde{\bar{x}} - r \frac{\nabla f(\bar{x}_k)}{\| \nabla f(\bar{x}_k) \|} \ .$$

3.4.7. Метод линеаризации

Пусть требуется:
найти $\min f(\bar{x})$ при ограничениях

$$g_i(\bar{x}) \le 0, \quad i = \overline{1, m}.$$

Заменим в точке $\bar{x}_k$ функцию $f(\bar{x})$ и все ограничения $g_i(\bar{x})$ на линейные путем линеаризации:

$$f_L(\bar{x}) = < \nabla f(\bar{x}_k), \bar{x} - \bar{x}_k >;$$

$$g_i(\bar{x}_k) + < \nabla g(\bar{x}_k), \bar{x} - \bar{x}_k > \le 0, \quad i = \overline{1, m}.$$

В результате получим задачу линейного программирования:
найти

$$\min \left\{ f_L(\bar{x}) = < \nabla f(\bar{x}_k), \bar{x} - \bar{x}_k > \right\}$$

при ограничениях

$$g_i(\bar{x}_k) + < \nabla g(\bar{x}_k), \bar{x} - \bar{x}_k > \le 0, \quad i = \overline{1, m}.$$

Можно было бы решение линеаризованной задачи взять в качестве следующего приближения, как это делается в методе Ньютона для решения систем нелинейных уравнений. К сожалению, как правило, этот путь не приводит к решению исходной задачи, так как, вспомогательная задача линейного программирования часто не имеет решения. Поэтому необходимо наложить некоторые ограничения на приращение вектора $\bar{x}$ в точке $\bar{x}_k$, чтобы решение линеаризованной задачи в точке $\bar{x}_k$ не уходило слишком далеко от $\bar{x}_k$, оставаясь в такой окрестности $\bar{x}_k$, в которой линеаризация еще справедлива. Это осуществляется путем добавления квадратного члена к линеаризованной функции.

Таким образом, после того как получена точка $\bar{x}_k$, в качестве точки $\bar{x}_{k+1}$ берется решение следующей задачи минимизации:
найти

$$\min \left\{ \varphi_k(\bar{x}) = \frac{1}{2} \left\| \bar{x} - \bar{x}_k \right\|^2 + \beta_k < \nabla f(\bar{x}_k), \bar{x} - \bar{x}_k > \right\}$$

при ограничениях:

$$g_i(\bar{x}_k) + < \nabla g(\bar{x}_k),\ \bar{x} - \bar{x}_k > \le 0, \quad i = \overline{1, m},$$

где параметр $\beta_k > 0$.

Данная задача представляет собой задачу квадратичного программирования и может быть решена методами, предназначенными для решения таких задач (методами условного градиента, или методом проекции градиента).

3.4.8. Другие методы минимизации функций с ограничениями

Квадратичное программирование. Задачей квадратичного программирования называется задача, в которой квадратичная функция минимизируется на многогранном множестве:

найти

$$\min\left\{ f(\bar{x}) = a + \bar{b}^{\mathrm{T}}\bar{x} + \frac{1}{2}\bar{x}^{\mathrm{T}}Q\bar{x} \right\},$$

(Q – положительно определенная матрица) при ограничениях:

$$A\bar{x} \le G.$$

Такие задачи возникают в некоторых приложениях и, кроме того, часто возникают как вспомогательные при описании различных методов минимизации, в том числе, метода линеаризации.

Оказывается, для задачи квадратичного программирования, как и для задачи линейного программирования, существуют конечношаговые методы их решения.

Метод Ньютона. Этот метод можно применять для задач с ограничениями; при этом направление поиска находится не явно, а получается в результате решения вспомогательной задачи. При этом формула итерационного процесса имеет стандартный вид:

$$\bar{x}_{k+1} = \bar{x}_k + \lambda_k \bar{S}_k,$$

где вектор $\bar{S}_k = \bar{y}_k - \bar{x}_k$ и определяется из решения задачи минимизации на множестве X квадратичной функции

$$\Psi_k(\bar{x}) = < \nabla f(\bar{x}_k), \bar{x} - \bar{x}_k > + \frac{1}{2}(\bar{x} - \bar{x}_k)^{\mathrm{T}} Q(\bar{x}_k)(\bar{x} - \bar{x}_k).$$

В последней формуле $Q(\bar{x}_k)$ – матрица Гессе.

Шаг λ_k может быть выбран различными способами, в частности можно использовать способ, описанный в п. 3.3.3.3.

Метод Ньютона, как правило, применяют в тех случаях, когда вычисление первых и вторых производных не представляет особых трудностей и вспомогательная задача решается достаточно просто.

Метод покоординатного спуска. Описанный ранее метод покоординатного спуска не трудно модифицировать применительно к задаче минимизации функции на параллелепипеде:

найти $\min\{f(\bar{x})\}$ при ограничениях

$$a_i \le x_i \le b_i, \quad i = \overline{1,n}.$$

3.4.9. Способы определения начальной точки

В рассмотренных ранее методах минимизации требовалось в качестве начальной точки $\bar{x}_0$ выбрать некоторую допустимую точку $\bar{x}_0 \in X$. Для X, таких, как, например, параллелепипед, шар, гиперплоскость, указать такую точку нетрудно. Однако нередко задача определения $\bar{x}_0$ является весьма непростой.

Например, если

$$X = \left\{ \bar{x} : \quad q_i(\bar{x}) = 0, \quad i = \overline{1,m} \right\}, \tag{3.18}$$

то для определения точки $\bar{x}_0 \in X$ нужно решать систему уравнений (в общем случае нелинейных).

Чтобы найти какую-либо точку множества

$$X = \left\{ \bar{x} : \quad q_i(\bar{x}) \le 0, \quad i = \overline{1,m} \right\} \tag{3.19}$$

придется решать систему неравенств, что представляет собой весьма серьезную задачу. Если ограничения $q_i(\bar{x}) \le 0$ линейны, то для определения начальной точки можно использовать тот же прием, который используется в линейном программировании.

Задачу нахождения точки $\overline{x}_0$, принадлежащей множествам (3.18) или (3.19) можно переформулировать в виде задачи минимизации.

В случае множества (3.18) введем функцию

$$F(\overline{x}) = \sum_{i=1}^{m} q_i^2(\overline{x}), \quad \overline{x} \in E^n,$$

а в случае множества (3.19) – функцию

$$F(\overline{x}) = \sum_{i=1}^{m} (\max\{q_i^2(\overline{x}), 0\})^p, \quad \overline{x} \in E^n, \quad p \geq 1,$$

и рассмотрим задачу минимизации:

$$\text{найти } \min F(\overline{x}), \quad \overline{x} \in E^n.$$

Эта задача решается любым из методов безусловной минимизации.

Если множество X не пусто, то условие $\overline{x} \in X$ равносильно условию $F(\overline{x}_0) = \min F(\overline{x}) = 0 = F^*$.

Если $F^* > 0$, то X – пустое множество.

Глава 4

НЕЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ В ПРИКЛАДНЫХ ЗАДАЧАХ ОПТИМИЗАЦИИ

4.1. Применение нелинейного программирования в теоретико-игровых методах исследования сложных систем

4.1.1. Теоретические предпосылки решения матричных игр

Матричной игрой будем называть антагонистическую игру, в которой каждый игрок имеет конечное множество стратегий.

Антагонистической игрой $\Gamma = < J, \{S_i\}_{i \in J}, \{H_i\}_{i \in J} >$ называется игра, в которой число игроков равно двум, а значение функции выигрыша этих игроков в каждой ситуации равны по величине и противоположны по знаку:

$$J = \{1,2\}, \quad H_2(s) = -H_1(s), \quad s \in S .$$

Такая игра задается прямоугольной матрицей (см. разд. 2.1):

$$A = \left\| a_{ij} \right\|, \quad i = \overline{1,m}, \quad j = \overline{1,n},$$

где a_{ij} – значение выигрыша игрока 1, если он выбрал свою i-ю стратегию, а игрок 2 выбрал свою j-ю стратегию. Данная игра называется игрой размерности ($m \times n$), а матрица A – *платежной матрицей*.

Если игрок 1 выбирает номер строки i, а второй – j, то в результате выбора игроками независимых стратегий игрок 2 платит игроку 1 выигрыш a_{ij}. Следовательно, игрок 1 может гарантировать себе выигрыш не менее значения: $v_1 = \max\limits_i \min\limits_j a_{ij}$ – *нижняя цена игры*, а второй гарантировать себе проигрыш не более вели-

чины $v_2 = \min_j \max_i a_{ij}$ – *верхняя цена игры*. В общем случае $v_1 \le v \le v_2$.

Если $\min_j \max_i a_{ij} = \max_i \min_j a_{ij} = a_{i^* j^*} = v$. То в такой игре минимаксные стратегии i^* и j^* являются оптимальными, так как решения о принятых стратегиях игроков получены независимо. В этом случае решение игры является ситуациями равновесия, а стратегии i^* и j^* называются чистыми стратегиями.

Если $\min_j \max_i a_{ij} \ge \max_i \min_j a_{ij}$ то в игре нет ситуации равновесия в чистых стратегиях. Для разрешения данной ситуации вводятся смешанные стратегии игроков.

Смешанной стратегией игрока 1 называется вектор $X = (x_1, ..., x_m)^{\mathrm{T}}$, удовлетворяющий условиям:

$$\sum_{i=1}^{m} x_i = 1, \quad x_i \ge 0, \quad i = \overline{1, n},$$

где число x_i – вероятность, с которой игрок 1 выбирает свою i-ю чистую стратегию.

Величина

$$H(X, Y) = \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij} x_i y_j = X^{\mathrm{T}} A Y \tag{4.1}$$

представляет собой математическое ожидание выигрыша 1-го игрока в ситуации (X, Y).

Ситуация (X^*, Y^*) называется *ситуацией равновесия* в смешанном расширении матричной игры, если для любых X и Y выполняются неравенства

$$H(X, Y^*) \le H(X^*, Y^*) \le H(X^*, Y). \tag{4.2}$$

В матричной игре может быть несколько ситуаций равновесия.

Можно доказать следующее свойство:

пусть $X^* \in S_m, Y^* \in S_n, v$ – действительное число, тогда, для того чтобы X^*, Y^* были оптимальными стратегиями, а v-ценой игры, необходимо и достаточно, чтобы выполнялись соотношения:

$$\begin{cases} H(i,Y) = \sum_{j=1}^{n} a_{ij} y^*{}_j \le v, \quad 1 \le i \le m; \\ H(X,j) = \sum_{i=1}^{m} a_{ij} x^*{}_i \ge v, \quad 1 \le j \le n. \end{cases} \tag{4.3}$$

Под решением матричной игры будем понимать нахождение векторов X и Y, а также значения цены игры v.

4.1.2. Основы метода фон Неймана

Данный метод представляет собой итеративный метод приближенного решения матричных игр размерности ($m \times n$), обладающий достаточно хорошей скоростью сходимости.

Пусть дана платежная матрица

$$A = \begin{pmatrix} a_{11} & a_{12} & . & a_{1n} \\ . & . & . & . \\ a_{m1} & a_{m1} & . & a_{mn} \end{pmatrix}.$$

Положим, что решения в чистых стратегиях нет.

Будем искать решение игры $(m \times n)$ в смешанных стратегиях в виде

$$\begin{aligned} X &= (x_1, ..., x_m)^{\mathrm{T}}, \quad x_i \ge 0, \quad i = \overline{1,m}, \\ Y &= (y_1, ..., y_m)^{\mathrm{T}}, \quad y_j \ge 0, \quad j = \overline{1,n}, \end{aligned} \tag{4.4}$$

при ограничениях

$$\sum_{i=1}^{m} x_i = 1, \quad \sum_{j=1}^{n} y_j = 1.$$

Тогда, если X^*, Y^* – оптимальные смешанные стратегии первой и второй стороны, то, в соответствии с соотношениями (4.3), выполняются неравенства:

$$\begin{cases} \sum_{j=1}^{n} a_{ij} y_j^* \le v, \quad i = \overline{1,m}; \\ \sum_{i=1}^{m} a_{ij} x_i^* \ge v, \quad j = \overline{1,n}, \end{cases} \tag{4.5}$$

где v – цена игры.

Суть метода фон Неймана состоит в численном решении системы линейных неравенств (4.5) путем сведения этой задачи к задаче минимизации функции:

$$f(X,Y,v) = \sum_{j=1}^{n} [z_j(X,v)]^2 + \sum_{i=1}^{m} [u_i(Y,v)]^2 , \tag{4.6}$$

где

$$z_j = \begin{cases} 0 \quad \text{при} \quad v - \sum_{i=1}^{m} a_{ij} x_i \le 0; \\ v - \sum_{i=1}^{m} a_{ij} x_i \quad \text{при} \quad v - \sum_{i=1}^{m} a_{ij} x_i \ge 0; \end{cases} \tag{4.7}$$

$$u_i = \begin{cases} 0 \quad \text{при} \quad \sum_{i=1}^{n} a_{ij} x_j - v \le 0; \\ \sum_{i=1}^{n} a_{ij} y_j - v \quad \text{при} \quad \sum_{i=1}^{n} a_{ij} y_j - v^l \ge 0. \end{cases} \tag{4.8}$$

Функция (4.6) представляет собой сумму квадратов невязок правых частей неравенств (4.5).

Точное решение матричной игры соответствует таким значениям аргументов, при которых достигается минимум функции $f(X,Y,v)$, т.е.

$$f(X^*, Y^*, \nu) = \min f(X,Y,\nu) = 0 .$$

Задача минимизации функции (4.6) относится к области нелинейного программирования (см. гл. 3) и представляет собой задачу минимизации функции многих переменных с ограничениями. Ограничения наложены на компоненты векторов X и Y в соответствии с определением смешанной стратегии.

Дж. фон Нейманом предложена своя процедура решения данной задачи нелинейного программирования, близкая по сути к градиентным методам, но учитывающая особую структуру ограничений.

4.1.3. Алгоритм фон Неймана

Алгоритм заключается в следующем.

1. Начальные оценки векторов X и Y и цены игры ν задаются в виде

$$\begin{aligned} x_i^{(0)} &= \frac{1}{m}, \quad i = \overline{1,m}; \\ y_j^{(0)} &= \frac{1}{n}, \quad j = \overline{1,n}; \\ v^{(0)} &= \frac{\max(a_{ij}) + \min(a_{ij})}{2}. \end{aligned} \tag{4.9}$$

2. На $(l+1)$-й итерации новые значения оценок определяются на основании соотношений:

$$\begin{aligned} x_i^{l+1} &= x_i^l - \theta^{l+1}(x_i^l - \tilde{x}_i^{l+1}), \quad i = \overline{1,m}; \\ y_j^{l+1} &= y_j^l - \theta^{l+1}(y_j^l - \tilde{y}_j^{l+1}), \quad j = \overline{1,n}; \\ v^{l+1} &= v^l - \theta^{l+1}(v^l - \tilde{v}^{l+1}), \end{aligned} \tag{4.10}$$

где

$$\theta^{l+1} = \frac{f(X^l, Y^l, v^l)}{f(X^l, Y^l, v^l) + f(\tilde{X}^{l+1}, \tilde{Y}^{l+1}, \tilde{v}^{l+1})}.$$

Значения $\tilde{x}_i^{l+1}, \tilde{y}_j^{l+1}$ $(i=\overline{1,m};\ j=\overline{1,n})$ представляют собой оценку величин x_i^*, y_j^* на шаге $(l+1)$. Эти значения рассчитываются, с учетом соотношений (4.7) и (4.8), по формулам:

$$\tilde{x}_i^{l+1} = \begin{cases} \dfrac{u_i^l}{s^l} & \text{при } s^l \neq 0, \quad s^l = \sum\limits_{i=1}^{m} u_i^l; \\ x_i^l & \text{при } s^l = 0; \end{cases} \tag{4.11}$$

$$\tilde{y}_j^{l+1} = \begin{cases} \dfrac{z_j^l}{t^l} & \text{при } t^l \neq 0, \quad t^l = \sum\limits_{j=1}^{n} z_j^l; \\ y_j^l & \text{при } t^l = 0. \end{cases} \tag{4.12}$$

Оценки цены игры на $(l+1)$-шаге вычисляются по формуле:

$$\tilde{v}^{l+1} = \tilde{X}^{l+1} A \tilde{Y}^{\text{т}\,l+1},$$

которая в скалярной форме имеет вид

$$\tilde{v}^{l+1} = \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij} \tilde{x}_i^{l+1} \tilde{y}_j^{l+1}. \tag{4.13}$$

3. Итерационный процесс заканчивается, когда значение функции (4.6) становится достаточно малым, т.е. когда выполняется соотношение:

$$f^l \leq \varepsilon, \tag{4.14}$$

где ε – заданная точность вычисления.

Структурная схема алгоритма фон Неймана приведена на рис. 4.1.

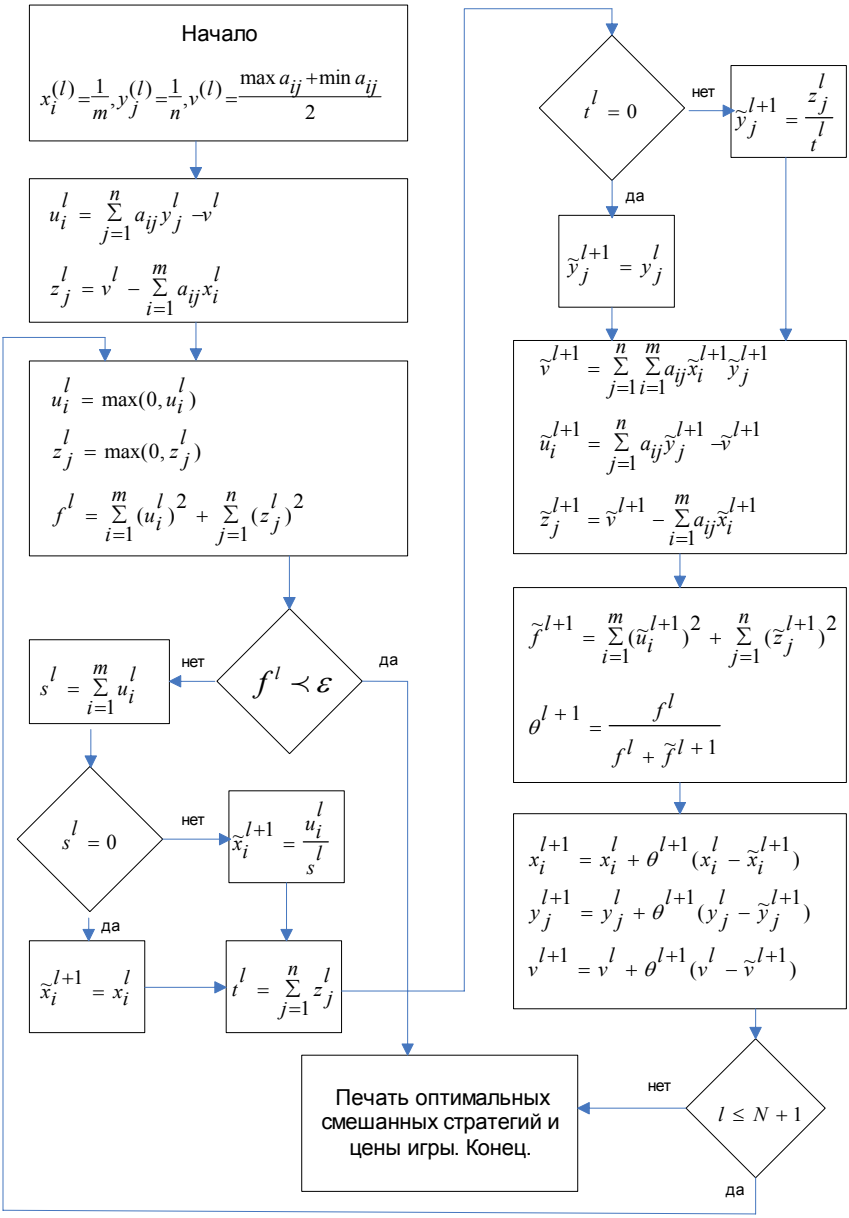


Рис. 4.1. Структурная схема алгоритма фон Неймана

4.1.4. Математическое программирование в теории биматричных игр

4.1.4.1. Биматричные игры. Основные теоретические сведения

Матричная игра является антагонистической игрой [23], т.е. игрой двух лиц с нулевой суммой. Однако на практике часто встречаются такие ситуации, когда интересы сторон не являются прямо противоположными. В этом случае игра имеет произвольную сумму, отличную от нуля, и является неантагонистической.

Конечную игру двух игроков с произвольной суммой можно описать парой матриц – поэтому она называется *биматричной*.

Биматричная игра определяется следующим образом. Два игрока обозначены буквами A и B. Игрок A имеет в своем распоряжении m чистых стратегий, а B – n чистых стратегий. Если A выбирает свою i-ю чистую стратегию, а B – j-ю чистую стратегию, то выигрыш игрока A есть a_{ij}, а выигрыш игрока B есть b_{ij}. Определим матрицы A_1 и B_1 как матрицы размером $m \times n$, (i, j) элементы которых есть a_{ij} и b_{ij}. Игра полностью определена, когда заданы матрицы выигрышей A_1 и B_1.

Смешанная стратегия игрока A есть столбец X из неотрицательных элементов x_i, которые представляют собой относительную частоту, с которой A будет выбирать свою i-ю чистую стратегию. Таким образом,

$$x_1 + x_2 + ... + x_m = 1. \tag{4.15}$$

Аналогично смешанная стратегия игрока B есть столбец Y, неотрицательные элементы y_j которого в сумме равны 1. Если A и B всегда выбирают чистую стратегию случайным образом в соответствии с распределением вероятностей, заданным векторами X и Y, то ожидаемые выигрыши игроков A и B соответственно

$$V_A = \sum_{i=1}^{m}\sum_{j=1}^{n} x_i a_{ij} y_j = \left(X, A_1 Y\right), \quad V_B = \sum_{i=1}^{m}\sum_{j=1}^{n} x_i b_{ij} y_j = \left(X, B_1 Y\right). \tag{2.16}$$

В этих соотношениях выражение в скобках означает скалярное произведение соответствующих векторов.

В отличие от матричных игр, где существует единственный критерий оптимальности поведения игроков, в неантагонистических играх вообще и в биматричных в частности, таких критериев несколько. Различают *ситуации равновесия по Нэшу*, *сильно равновесные ситуации* и *ситуации, оптимальные по Парето*. Каждый из этих подходов к оценке оптимальности поведения игроков имеет свои достоинства и свои недостатки [13].

Далее пойдет речь только о ситуациях равновесия по Нэшу.

Ситуация равновесия в игре есть пара смешанных стратегий (X^0, Y^0) такая, что для любых других смешанных стратегий (X, Y) выполняется соотношение

$$(X^0, A_1 Y^0) \geq (X, A_1 Y^0), \quad (X^0, B_1 Y^0) \geq (X^0, B_1 Y). \qquad (2.17)$$

Иными словами, ситуация равновесия – такая ситуация, отклонение от которой одного из игроков не может увеличить его выигрыш.

Следует отметить, что ситуаций равновесия в биматричной игре может быть несколько. Причем в различных ситуациях равновесия выигрыши неодинаковы, а множество ситуаций равновесия прямоугольно.

Теорема Нэша гарантирует существование ситуаций равновесия в биматричных играх [28], но не дает никаких средств для их нахождения.

Наиболее эффективным из всех известных алгоритмов для практических вычислений ситуации равновесия является алгоритм, предложенный Лемке и Хоусоном в 1963 г. [67], [44]. По своей сути этот алгоритм тесно связан с методами нелинейного программирования, изложенными в разделе 3 настоящего издания. Следует особо подчеркнуть то обстоятельство, что данный алгоритм может быть без особых затруднений реализован на ЭВМ.

Докажем, прежде всего две теоремы, на основании которых можно сформулировать постановку задачи.

Теорема 4.1. Ситуация (X^0, Y^0) является ситуацией равновесия в биматричной игре с матрицами выигрышей A_1 и B_1 в том и только в том случае, когда

$$\begin{aligned} &(X^0, A_1Y^0)L_m \geq A_1Y^0, \\ &(X^0, B_1Y^0)L_n \geq B_1^T X^0, \end{aligned} \tag{4.18}$$

где L_m и L_n – векторы размерности m и n соответственно, составленные из единиц (знаки неравенств используются для покоординатного сравнения векторов).

Доказательство. Пусть (X^0, Y^0) – ситуация равновесия, т.е.

$$(X^0, A_1Y^0) \geq (X, A_1Y^0) \text{ при всех } X \geq 0,\ (X, L_m) = 1,$$

$$(X^0, B_1Y^0) \geq (X^0, B_1Y) \text{ при всех } Y \geq 0,\ (Y, L_n) = 1. \tag{4.19}$$

Возьмем в качестве X и Y векторы, у которых одна из компонент равна единице, а остальные – нулю. Получим $m+n$ соотношений

$$\begin{aligned} &(X^0, A_1Y^0)L_m \geq A_1Y^0, \\ &(X^0, B_1Y^0)L_n \geq B_1^{\text{т}} X^0, \end{aligned} \tag{4.20}$$

Наоборот, пусть выполняются указанные условия.

Вектор

$$X = (x_1, x_2, ..., x_m),\ (X, L_m) = 1 \tag{4.21}$$

является произвольным вектором. Представим его в базисе единичных векторов $f_1, f_2, ..., f_m$ следующим образом:

$$X = x_1 f_1 + x_2 f_2 + ... + x_m f_m. \tag{4.22}$$

Тогда

$$\begin{aligned} (X, A_1Y^0) &= x_1(f_1, A_1Y^0) + ... + x_m(f_m, A_1Y^0) \leq \\ &\leq x_1(X^0, A_1Y^0) + ... + x_m(X^0, A_1Y^0) = \\ &= (x_1 + ... + x_m)(X^0, A_1Y^0) = (X^0, A_1Y^0), \end{aligned} \tag{4.23}$$

что и требовалось доказать.

Применение методов математического программирования возможно, если существует некоторое допустимое множество, соответствующее ограничениям на аргументы целевой функции, в качестве которых рассматриваются компоненты смешанных стратегий игроков. При решении биматричных игр такое множество значений аргументов строится с помощью простой операции: перехода к платежным матрицам A и B, элементы которых – положительные числа.

Рассмотрим матрицы

$$A = dE - A_1, \quad B = dE - B_1, \tag{4.24}$$

где d – достаточно большая положительная константа, такая, что $A > 0$ и $B > 0$, E – матрица размерности $(m \times n)$, составленная из единиц. Вследствие процедуры (4.24) изменяются и условия (4.20).

Теорема 4.2. Ситуация

$$\begin{cases} X^0 = \dfrac{X^*}{(X^*, L_m)}, \\ Y^0 = \dfrac{Y^*}{(Y^*, L_n)} \end{cases} \tag{4.25}$$

является ситуацией равновесия в игре с матрицами выигрышей A_1 и B_1 в том и только в том случае, если пара (X^*, Y^*) удовлетворяют соотношениям

$$B^{\mathrm{T}} X^* \geq L_n, \quad X^* \geq 0, \quad (Y^*, B^{\mathrm{T}} X^* - L_n) = 0, \tag{4.26}$$

$$AY^* \geq L_m, \quad Y^* \geq 0, \quad (X^*, AY^* - L_m) = 0. \tag{4.27}$$

Доказательство. Сначала покажем, что

$$V_A = (X^0, AY^0) = d - \frac{1}{(Y^*, L_n)}. \tag{4.28}$$

Имеем

$$\frac{1}{(X^*, L_m)(Y^*, L_n)} (X^*, ((dE - A_1)Y^* - L_m)) = 0, \tag{4.29}$$

т.е.

$$(X^0, dEY^0) - \left(X^0, A_1 Y^0\right) - \frac{1}{(Y^*, L_n)}(X^0, L_m) = 0. \qquad (4.30)$$

Учитывая, что $EY^0 = L_m$ а $(X^0, L_m) = 1$, получим

$$(X^0, A_1 Y^0) = d - \frac{1}{(Y^*, L_n)}. \qquad (4.31)$$

Это соотношение можно использовать при вычислении цены игры для участника с платежной матрицей A_1.

Поскольку $Y^* \geq 0$, т.е. $(Y^*, L_n) \geq 0$, неравенство

$$(dE - A_1)Y^* \geq L_n \qquad (4.32)$$

можно преобразовать к виду

$$(dE - A_1)Y^0 \geq \frac{L_m}{(Y^*, L_n)} \qquad (4.33)$$

или

$$\left(d - \frac{1}{(Y^*, L_n)}\right)L_m \geq A_1 Y^0. \qquad (4.34)$$

Используя полученный в начале доказательства результат, придем к неравенству

$$(X^0, A_1 Y^0)L_m \geq A_1 Y^0. \qquad (4.35)$$

По аналогии доказывается справедливость соотношения

$$(X^0, B_1 Y^0)L_n \geq B_1^{\text{т}} X^0. \qquad (4.36)$$

Согласно теореме 4.1 выполнение этих условий является необходимым и достаточным, чтобы утверждать, что пара (X^0, Y^0) – ситуация равновесия.

Теорема доказана.

Соотношения

$$(f_i, X^*)((a_i, Y^*) - 1) = 0 \quad \text{для} \quad i = 1,...,m, \tag{4.37}$$

$$(e_j, Y^*)((b_j, X^*) - 1) = 0 \quad \text{для} \quad j = 1,...,n. \tag{4.38}$$

где вектор a_i – i-я строка матрицы A, а вектор b_j – j-й столбец матрицы B, есть эквивалентная форма записи условий теоремы.

4.1.4.2. Нахождение ситуации равновесия в биматричных играх

Доказав теоремы 4.1 и 4.2, получили критерий, с помощью которого удобно определять, является ли пара (X^0, Y^0) ситуацией равновесия.

Действительно, если исходить только из определения ситуации равновесия, то для проверки пары (X^0, Y^0) необходимо перебирать все векторы X и Y из множества смешанных стратегий. Сделать это невозможно. Вот почему потребовалось найти необходимое и достаточное условие, в записи которого фигурируют только векторы X^0 и Y^0.

Постановка задачи нахождения ситуации равновесия. Пусть A и $B > 0$.

Задача состоит в том, чтобы найти пару (X^*, Y^*), для которой выполняются следующие условия.

1. $X^* \geq 0$, $Y^* \geq 0$. (4.39)
2. $(f_i, X^*)((a_i, Y^*) - 1) = 0$, $(a_i, Y^*) \geq 1$ для $i = 1,...,m$. (4.40)
3. $(e_j, Y^*)((b_j, X^*) - 1) = 0$, $(b_j, X^*) \geq 1$ для $j = 1,...,n$. (4.41)

Будем называть любую пару (X^*, Y^*) ситуацией равновесия.

Допустимое множество для поиска смешанных стратегий игрока 1 определяется в соответствии с соотношениями (4.40) и (4.41) следующим образом:

$$\mathrm{X} = \left\{ X \mid X \geq 0, B^{\mathrm{T}} X - L_n \geq 0 \right\}, \tag{4.42}$$

т.е. X состоит из векторов X, для которых

$$(f_i, X) \geq 0 \quad \text{для} \quad i = 1,...,m, \tag{4.43}$$

$$(b_j, X) \geq 1 \quad \text{для} \quad j = 1,...,n. \tag{4.44}$$

Конец вектора X лежит на границе множества X, если хотя бы одно из этих соотношений является равенством.

Аналогично можно рассмотреть множество

$$\mathrm{Y} = \{Y \mid Y \geq 0, AY - L_m \geq 0\}. \tag{4.45}$$

Граница множества Y состоит из точек, удовлетворяющих хотя бы одному из $m+n$ уравнений

$$(a_i, Y) = 1 \quad \text{для} \quad i = 1,...,m, \tag{4.46}$$

$$(e_j, Y) = 0 \quad \text{для} \quad j = 1,...,n, \tag{4.47}$$

а для внутренних точек выполняются неравенства

$$(a_i, Y) > 1 \quad \text{для} \quad i = 1,...,m, \tag{4.48}$$

$$(e_j, Y) > 0 \quad \text{для} \quad j = 1,...,n. \tag{4.49}$$

Легко убедиться, что X и Y – многогранники, у которых соответственно имеются m и n бесконечных ребер.

* * *

Пример 4.1. Рассмотрим игру с матрицами выигрышей

$$A_1 = \begin{pmatrix} 1{,}75 & 1{,}5 \\ 1{,}66 & 1{,}8 \end{pmatrix}, \quad B_1 = \begin{pmatrix} 1{,}5 & 1{,}8 \\ 1{,}67 & 1 \end{pmatrix}.$$

Выберем $d = 2$, получим матрицы

$$A = \begin{pmatrix} 0{,}25 & 0{,}5 \\ 0{,}34 & 0{,}2 \end{pmatrix}, \quad B = \begin{pmatrix} 0{,}5 & 0{,}2 \\ 0{,}33 & 1 \end{pmatrix}.$$

Множество Y, показанное на рис. 4.2, определяется следующими соотношениями:

$$\begin{cases} 0{,}25y_1 + 0{,}5y_2 - 1 \geq 0; \\ 0{,}34y_1 + 0{,}2y_2 - 1 \geq 0; \\ y_1 \geq 0, y_2 \geq 0, \end{cases}$$

а множество X, показанное на рис. 4.3, – соотношениями:

$$\begin{cases} 0{,}5x_1 + 0{,}33x_2 - 1 \geq 0; \\ 0{,}2x_1 + y_2 - 1 \geq 0; \\ x_1 \geq 0, x_2 \geq 0. \end{cases}$$

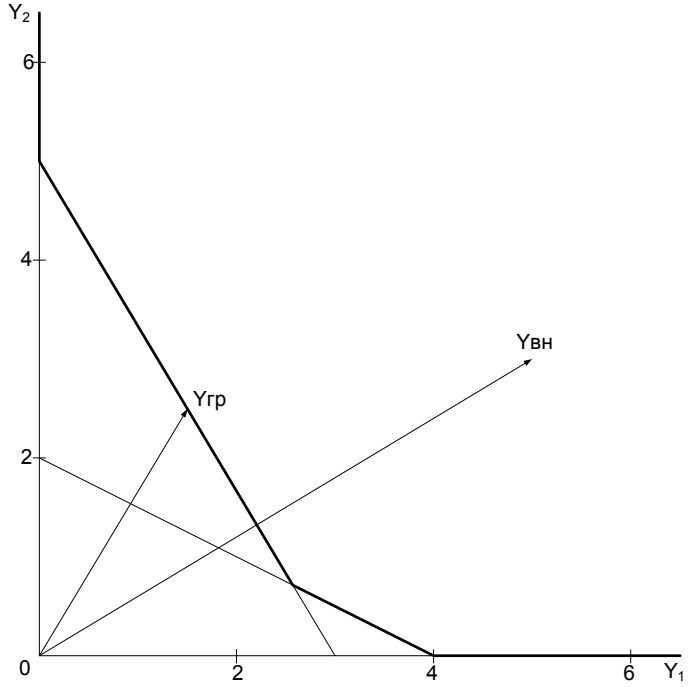


Рис. 4.2. Множество Y : $Y_{вн}$ – внутренняя точка; $Y_{гр}$ – точка на границе

Заметим, что здесь и в дальнейшем используется важное предположение, сделанное при постановке задачи:

$$A > 0 \quad \text{и} \quad B > 0 .$$

Именно это условие гарантирует существование выпуклых открытых многогранников X и Y.

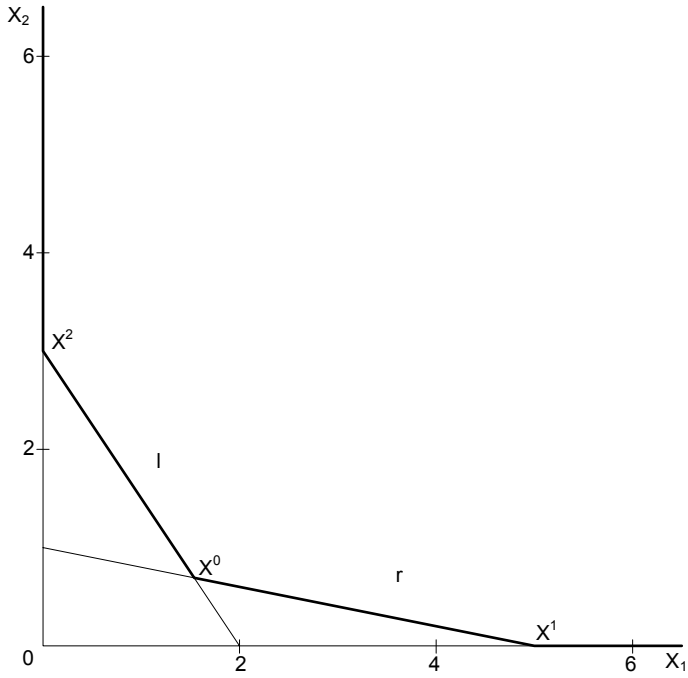


Рис. 4.3. Множество X : X^0, X^1, X^2 – крайние точки; l, r – открытые ребра

* * *

Пусть для любого X через $p(X)$ обозначено множество тех из векторов f_i или b_j, для которых

$$(f_i, X) = 0 \quad \text{или} \quad (b_j, X) = 1. \tag{4.50}$$

Множество $p(X)$ может быть единственным образом записано в виде матрицы

$$p(X) = (p_1, p_2, ..., p_r), \tag{4.51}$$

где p_l $(l = 1,...,r)$ – это соответствующий определению вектор f_i или b_j.

Предположим, что матрица B удовлетворяет условию *невырожденности*: пусть столбцы матрицы C являются столбцами из матрицы (I, B), где I – единичная матрица, тогда если $C = p(X)$ для некоторого $X \in \mathrm{X}$, то ранг матрицы C равен числу ее столбцов.

Рассмотрим следствия, вытекающие из предположения о невырожденности.

Следствие 1. Если для некоторого $X \in \mathrm{X}$ множество $p(X)$ содержит m векторов, то X полностью определяется своими векторами f_i и b_j. В этом случае X называется *крайней точкой* множества X.

Следствие 2. Пусть для $X^0 \in \mathrm{X}$ матрица $p(X^0)$ имеет ранг r, матрица $p(X^0)$ есть матрица размером $(m \times r)$ (r может быть и равным нулю):

$$p(X^0) = (p_1, p_2, ..., p_r)\,. \tag{4.52}$$

Ввиду того, что $p_1, p_2, ..., p_r$ – линейно независимые векторы, можно присоединить к ним векторы $p_{r+1}, p_{r+2}, ..., p_m$ так, чтобы матрица

$$D = (p_1, p_2, ..., p_m) \tag{4.53}$$

была невырожденной.

Существует обратная матрица

$$(D^{-1})^{\mathrm{T}} = (p^1, p^2, ..., p^m)\,. \tag{4.54}$$

Это означает, что

$$(p^i, p_j) = \delta_{ij} = \begin{cases} 1 & \text{при} \quad i = j, \\ 0 & \text{при} \quad i \neq j. \end{cases} \tag{4.55}$$

Теорема 4.3. Пусть $X^0 \in \mathrm{X}$ и $p(X^0) = (p_1, p_2, ..., p_r)$. Существует такая постоянная K, что при

$$\sum_{i=1}^{m} \lambda_i^2 \le K\,, \tag{4.56}$$

где $\lambda_i \geq 0$ для $1 \leq i \leq r$, точка X, определяемая соотношением

$$X = X^0 + \sum_{i=1}^{m} \lambda_i p^i , \tag{4.57}$$

принадлежит множеству X.

Доказательство. Пусть p – любой столбец матрицы (I, B). Тогда

$$(p, X) = (p, X^0) + \sum_{i=1}^{m} \lambda_i (p, p^i) . \tag{4.58}$$

Рассмотрим два случая.

1. Если $p \in p(X^0)$, т.е. $p = p_i$, $1 \leq i \leq r$.

На основании следствия 2 имеем $(p, X) = (p, X^0) + \lambda_i$. При этом

$$(p, X^0) = \begin{cases} 1, \text{ если } p \text{ – столбец из } B, \\ 0, \text{ если } p \text{ – столбец из } I. \end{cases} \tag{4.59}$$

По условию теоремы $\lambda_i \geq 0$, $1 \leq i \leq r$, следовательно, $(p, X) \geq$ $\geq (p, X^0)$ и X является подмножеством X.

2. Если $p \notin p(X^0)$, т.е. p – любой другой столбец из (I, B). В этом случае

$$(p, X^0) > \begin{cases} 1, \text{ если } p \text{ – столбец из } B, \\ 0, \text{ если } p \text{ – столбец из } I. \end{cases} \tag{4.60}$$

Пусть p – столбец из B, тогда $(p, X^0) = 1 + \alpha$, где $\alpha > 0$. Выбрав λ_i, $1 \leq i \leq m$, достаточно малыми, такими, чтобы

$$\left| \sum_{i=1}^{m} \lambda_i (p, p^i) \right| \leq \alpha , \tag{4.61}$$

получим $(p, X) \geq 1$.

По аналогии можно рассмотреть столбец матрицы I. Таким образом, при соответствующем выборе λ_i, $1 \leq i \leq m$, из соотношения

$(p, X^0) \geq 0$ следует, что $(p, X) \geq 0$, а из соотношения $(p, X^0) \geq 1$ следует, что $(p, X) \geq 1$. Следовательно, $X \in \mathrm{X}$.

Теорема доказана.

Таким образом, доказана возможность перемещения вдоль границы многогранника X или Y от одной крайней точки к другой. Именно в таком перемещении и проверке в каждой крайней точке условий (4.37) и (4.38) и заключается суть алгоритма Лемке – Хоусона поиска ситуации равновесия.

Обратимся вновь к рис. 4.3. Точка X^0 является крайней точкой множества X. Поскольку выполняются условия

$$0.5x_1^0 + 0.33x_2^0 = 1 \quad \text{и} \quad 0.2x_1^0 + x_2^0 = 1, \tag{4.62}$$

матрица $p(X^0)$ невырождена:

$$p(X^0) = (p_1, p_2), \tag{4.63}$$

где

$$p_1 = \begin{pmatrix} 0{,}5 \\ 0{,}33 \end{pmatrix}, \quad \text{а} \quad p_2 = \begin{pmatrix} 0{,}2 \\ 1 \end{pmatrix}. \tag{4.64}$$

Согласно соотношению (4.57) точки

$$X = X^0 + \lambda_1 p^1 + \lambda_2 p^2 \tag{4.65}$$

при малых положительных λ_i принадлежат X.

Пусть $\lambda_2 = 0$. Для

$$X = X^0 + \lambda_1 p^1, \quad \lambda_1 > 0 \tag{4.66}$$

матрица $p(X)$ получается из $p(X^0)$ вычеркиванием только одного столбца, а именно p_1. Такие точки образуют *открытое ребро* многогранника X с концом X^0. На рис. 4.3 оно обозначено буквой r.

Для $X^1 \in \mathrm{X}$ матрица $p(X^1) = (p_1)$. При достаточно малом λ_2 точки

$$X = X^1 + \lambda_2 p^2 \qquad (4.67)$$

принадлежат множеству X и для них $p(X) = p(X^1)$. Такие точки образуют открытое ребро 1, содержащее X^1. В общем случае матрица $p(X)$, где X принадлежит открытому ребру X, имеет ранг, равный $(m-1)$.

На рис. 4.3 показаны два *неограниченных ребра*, каждое из которых обладает лишь одной концевой точкой $X = kf_i$ с соответствующим k. Таким образом, у многогранника X имеется ровно m неограниченных ребер, а остальные ребра имеют по две концевые точки, которые называются *смежными крайними точками*. Для смежных крайних точек, например X^2 и X^0, отличаются лишь одним столбцом. В нашем примере

$$p(X^2) = \begin{pmatrix} 0{,}5 & 1 \\ 0{,}33 & 0 \end{pmatrix}, \quad \text{а} \quad p(X^0) = \begin{pmatrix} 0{,}5 & 0{,}2 \\ 0{,}33 & 1 \end{pmatrix}. \qquad (4.68)$$

Аналогичные рассуждения можно провести для множества Y.

Пусть $q(Y)$ – матрица, подобная рассматривавшейся для элементов X множества X матрице $p(X)$. Предположим также, что матрица A удовлетворяет условию невырожденности, аналогично тому, которое было наложено на B.

Рассмотрим $\mathrm{Z} = \mathrm{X} \times \mathrm{Y}$. Точка $Z = (X, Y)$ из Z называется крайней для Z, если X – крайняя точка многогранника X, а Y – крайняя точка Y. Очевидно, такая точка должна удовлетворять $(m+n)$ условиям из системы

$$\begin{cases} (f_i, X) = 0, & i = 1,\dots,m, \\ (b_j, X) - 1 = 0, & j = 1,\dots,n, \\ (e_j, Y) = 0, & j = 1,\dots,n, \\ (a_i, Y) - 1 = 0, & i = 1,\dots,m. \end{cases} \qquad (4.69)$$

Будем говорить, что $Z = (X, Y)$ лежит на открытом ребре многогранника Z, если одна из ее координат является крайней точкой X или Y, а другая лежит на открытом ребре. Для точек, принадле-

жащих открытому ребру, выполняются $(m+n-1)$ из указанных выше соотношений.

Теорема 4.4. Любая ситуация равновесия для невырожденной задачи является крайней точкой множества Z.

Доказательство. Если $Z^* = (X^*, Y^*)$ удовлетворяет условиям постановки задачи, то для $i = 1,...,m$ или $(f_i, X^*) = 0$, или $(a_i, Y^*) - 1 = 0$, и для $j = 1,...,n$ или $(e_j, Y^*) = 0$, или $(b_j, X^*) - 1 = 0$. Из условия невырожденности следует, что $X \in \mathrm{X}$ может удовлетворять не более, чем m соотношениям вида:

$$(f_i, X^*) = 0 \quad \text{или} \quad (b_j, X^*) - 1 = 0, \tag{4.70}$$

а $Y \in \mathrm{Y}$ может удовлетворять не более, чем n соотношениям вида:

$$(e_j, Y^*) = 0 \quad \text{или} \quad (a_i, Y^*) - 1 = 0. \tag{4.71}$$

Но (X^*, Y^*) должны удовлетворять по меньшей мере $(m+n)$ таким соотношениям. Таким образом, удовлетворяются в точности $(m+n)$ соотношений, т.е. для X^* выполнены ровно m из соотношений

$$\begin{aligned} &(f_i, X) = 0, \qquad i = 1,...,m, \\ &(b_j, X) - 1 = 0, \quad j = 1,...,n. \end{aligned} \tag{4.72}$$

Следовательно, X^* – крайняя точка X. Аналогично Y^* – крайняя точка Y, а $Z^* = (X^*, Y^*)$ – крайняя точка Z, что и требовалось доказать.

Одновременно получен следующий критерий: если $Z^* = (X^*, Y^*)$ – ситуация равновесия, то для любого s, $1 \le s \le m+n$, или s-й столбец матрицы (I, B) принадлежит $p(X^*)$, или s-й столбец матрицы (A^{T}, I) принадлежит $q(Y^*)$, но не тот и другой одновременно. Подчеркнем, что это утверждение имеет силу только для ситуаций равновесия.

4.1.4.3. Метод Лемке – Хоусона. Теоретические основы

Механизм алгоритма Лемке – Хоусона заключается в том, чтобы, последовательно двигаясь от одной крайней точки множества Z к другой крайней точке, за конечное число шагов найти одну из ситуаций равновесия. Необходимо доказать несколько теорем, на основании которых строится схема движения. Прежде всего нужно получить само *множество путей* движения. Для этого рассмотрим точки $Z = (X, Y) \in \mathrm{Z}$, удовлетворяющие хотя бы $(m + n - 1)$ из условий

$$(f_i, X^*)\,((a_i, Y^*) - 1) = 0\,, \quad i = 1,...,m\,,$$

$$(e_j, Y^*)\,((b_j, X^*) - 1) = 0\,, \quad j = 1,...,n\,. \qquad (4.73)$$

Пусть через H_s обозначено множество точек $Z \in \mathrm{Z}$, удовлетворяющих всем этим уравнениям, кроме, возможно, уравнения

$$(e_s, Y^*)\,((b_s, X^*) - 1) = 0\,. \qquad (4.74)$$

Пример 4.2. Построить H_1 для биматричной игры, рассмотренной в примере 4.1. Для $Z = (X, Y) \in H_1$ одновременно должны выполняться соотношения:

$y_2 = 0$ или $0{,}2x_1 + x_2 - 1 = 0$ (прямая 4),
$x_1 = 0$ или $0{,}25y_1 + 0{,}5y_2 - 1 = 0$ (прямая 1),
$x_2 = 0$ или $0{,}34y_1 + 0{,}2y_2 - 1 = 0$ (прямая 2),

а соотношение

$y_1 = 0$ или $0{,}5x_1 + 0{,}33x_2 - 1 = 0$ (прямая 3)

может не выполняться.

На рис. 4.4 и 4.5 показаны точки (X, Y) $(X \in \mathrm{X}, Y \in \mathrm{Y})$, принадлежащие множеству H_1. Множество H_1 состоит из:

открытого ребра с концом в крайней точке (X^0, Y^0), образованного точками (X, Y^0), где X удовлетворяет условию $0{,}2x_1 + x_2 - 1 = 0$;

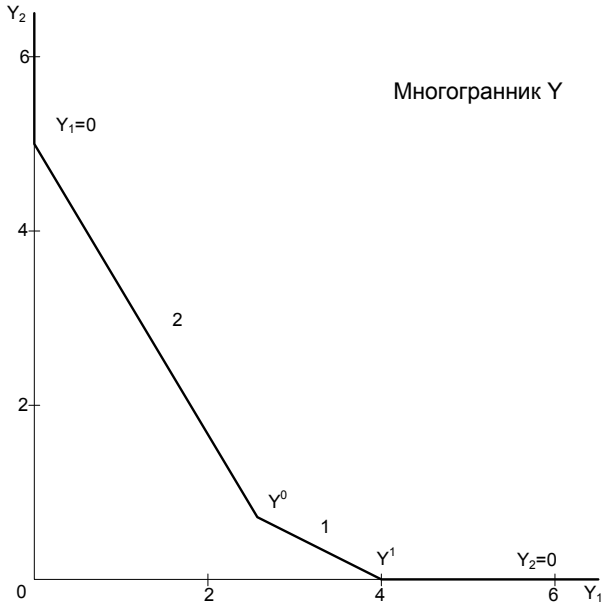


Рис. 4.4. Проекция s -пути на плоскость Y

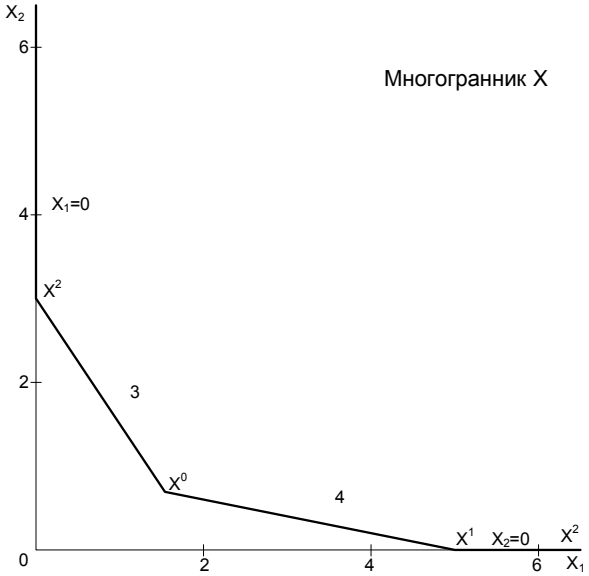


Рис. 4.5. Проекция s -пути на плоскость X

открытого ребра с концом в крайней точке (X^1, Y^1), образованного точками (X^1, Y), где Y удовлетворяет условию $0{,}25y_1 + 0{,}5y_2 - 1 = 0$;

неограниченного ребра (X, Y^1) с концом (X^1, Y^1).

Полученный результат можно обобщить, доказав теоремы 4.5 и 4.6.

Теорема 4.5. Любая точка из H_s или является крайней для Z, или лежит на открытом ребре Z.

Доказательство. Если $Z \in H_s$ удовлетворяет всем $(m+n)$ соотношениям, то она является крайней. Если же она удовлетворяет $(m+n-1)$ из уравнений

$$(f_i, X)\,((a_i, Y) - 1) = 0, \quad i = 1,...,m,$$

$$(e_j, Y)\,((b_j, X) - 1) = 0, \quad j = 1,...,n, \tag{4.75}$$

то в силу условий невырожденности выполнены также $(m+n-1)$ из соотношений

$$\begin{cases} (f_i, X) = 0, & i = 1,...,m, \\ (b_j, X) - 1 = 0, & j = 1,...,n, \\ (e_j, Y) = 0, & j = 1,...,n, \\ (a_i, Y) - 1 = 0, & i = 1,...,m. \end{cases} \tag{4.76}$$

Следовательно, эта точка лежит на ребре.

Теорема 4.6. Точка множества H_s образует единственное неограниченное ребро многогранника Z.

Доказательство. Рассмотрим множество H_1. Пусть $Y^1 = k_0 e_1$. При соответствующем k_0 точка Y^1 будет крайней для Y (см. рис. 4.4). Так как $(e_j, Y^1) = 0$ при $j \neq 1$ и этих соотношений всегда $(n-1)$, то так как Y^1 – крайняя точка Y, следует, что она должна быть определена еще одним соотношением, например $(a_r, Y^1) - 1 = 0$. Итак,

$$q(Y^1) = (e_2, ..., e_n, a_r)\,. \qquad (4.77)$$

Для случая, рассмотренного в примере 4.2, $r = 1$. При достаточно больших k точки $X = kf_r$ принадлежат X. Пусть k_1 таково, что $X^1 = k_1 f_r$ является крайней точкой X. Поскольку соотношения

$$(e_j, Y^1)\,((b_j, X) - 1) = 0\,, \quad j = 2,...,n\,,$$

$$(f_i, X)\,((a_i, Y^1) - 1) = 0\,, \quad i = 1,...,m\,, \qquad (4.78)$$

где $X = kf_r$, $(k \geq k_1)$, выполняются, то можно утверждать, что пары (X, Y^1) составляют открытое ребро многогранника Z, лежащее в H_1. Точка (X^1, Y^1) является концом этого ребра.

Докажем единственность этого ребра. Рассмотрим любое другое неограниченное ребро многогранника Z, например (X, Y^0), где $X = kf_l$, $(l \neq r)$, а Y^0 – крайняя точка множества Y. Такое ребро не принадлежит H_1 в силу того, что либо условие

$$(f_l, X)\,((a_l, Y^0) - 1) = 0\,. \qquad (4.79)$$

либо одно из условий

$$(e_j, Y^0)\,((b_j, X) - 1) = 0\,, \quad j = 2,...,n\,, \qquad (4.80)$$

не выполняется.

Теорема доказана.

Два открытых ребра многогранника Z называются *смежными*, если они имеют общий конец. Последовательность смежных открытых ребер из H_s вместе с их концевыми точками называется s-путем.

При решении задачи, рассмотренной в примере 4.2 (см. рис. 4.4 и 4.5), можно двигаться по следующему s-пути с началом в точке (X^2, Y^1):

двигаемся вдоль неограниченного ребра (X, Y^1) до точки (X^1, Y^1);

из точки (X^1, Y^1) продолжаем движение вдоль ребра (X^1, Y), то есть по прямой 1 в множестве Y, попадаем в крайнюю точку (X^1, Y^0);

(X^1, Y) является концом ребра (X, Y^0), где точки $X \in \mathrm{X}$ принадлежат прямой 4, по этому ребру можно попасть в (X^0, Y^0) – конечную точку рассматриваемого s-пути.

Теорема 4.7. Пусть Z – крайняя точка Z и $Z \in H_s$. Найдутся или одно, или два открытых ребра, лежащих в H_s и имеющих Z своей крайней точкой; Z является ситуацией равновесия в том и только в том случае, когда такое ребро единственное.

Доказательство. Положим, что Z – крайняя точка Z и $Z \in H_s$. Возможны два случая.

1. Пусть $(e_s, Y)((b_s, X) - 1) = 0$. Поскольку справедливы $(m+n)$ соотношений

$$(f_i, X)((a_i, Y) - 1) = 0, \quad i = 1,...,m,$$

$$(e_j, Y)((b_j, X) - 1) = 0, \quad j = 1,...,n, \tag{4.81}$$

точка $Z = (X, Y)$ – ситуация равновесия. Как было показано, при этом или $(e_s, Y) = 0$, или $(b_s, X) - 1 = 0$, но не оба сомножителя равны нулю сразу.

Допустим, что $(b_s, X) - 1 = 0$ ($(e_s, Y) = 0$ рассматривается аналогично). Так как $Z = (X, Y)$ – крайняя точка Z, выполнено m из соотношений

$$\begin{aligned} &(f_i, X) = 0, && i = 1,...,m, \\ &(b_j, X) - 1 = 0, && j = 1,...,n, \end{aligned} \tag{4.82}$$

и n из соотношений

$$\begin{aligned} &(e_j, Y) = 0, && j = 1,...,n, \\ &(a_i, Y) - 1 = 0, && i = 1,...,m. \end{aligned} \tag{4.83}$$

Следовательно, существует $(m+n)$ ребер многогранника Z, имеющих Z своим концом, причем, как было показано, вдоль каждого из них нарушается в точности одно соотношение. Поэтому в H_s будет лежать только то единственное ребро, вдоль которого нарушается условие $(b_s, X)-1=0$.

2. Пусть $(e_s, Y)((b_s, X)-1)>0$. Так как $Z=(X,Y)$ – крайняя точка Z, X определяется m векторами из множества

$$\{f_1,...,f_m, b_1,...b_n\}, \tag{4.84}$$

а Y определяется n векторами множества

$$\{e_1,...,e_n, a_1,...,a_m\}. \tag{4.85}$$

В нашем случае $(e_s, Y)>0$ и $(b_s, X)-1>0$. Но $Z \in H_s$, поэтому найдется такой индекс q, для которого выполнены сразу или оба соотношения

$$(f_q, X)=0 \quad \text{и} \quad (a_q, Y)-1=0, \tag{4.86}$$

или оба соотношения

$$(e_q, Y)=0 \quad \text{и} \quad (b_q, X)-1=0. \tag{4.87}$$

Рассмотрим первый вариант. В H_s лежат два ребра – одно, вдоль которого нарушается условие $(f_q, X)=0$, и другое, вдоль которого нарушается условие $(a_q, Y)-1=0$. При этом Z не является ситуацией равновесия. На любом другом ребре Z, имеющем Z концевой точкой, не выполнено какое-либо из соотношений, определяющих H_s.

Теорема доказана.

Доказанные выше теоремы гарантируют существование крайней точки Z, лежащей в H_s. Начав с такой точки Z_1, можно двигаться по ребру, лежащему в H_s: или это ребро закончится в другой крайней точке Z_2, или это единственное неограниченное ребро, лежащее в H_s. В первом случае или Z_2 – ситуация равновесия и процесс не может быть продолжен, или существует другое ребро с

концом в Z_2, лежащее в H_s, по которому можно продолжать двигаться. Процесс прекращается в следующих случаях, когда:

попадаем на неограниченное ребро;

достигаем ситуации равновесия, отличной от точки Z_1;

приходим в точку, в которой уже были.

Вернуться можно только в начальную точку пути, так как в противном случае имелась бы точка, к которой примыкают три ребра. Таким образом, начав с Z_1, возвращаемся в Z_1, или нет. В первом случае путь называется замкнутым (рис. 4.6, *а*). Если путь не замкнут, то он заканчивается либо в ситуации равновесия (рис. 4.6, *б*), либо на неограниченном ребре (рис. 4.6, *в*).

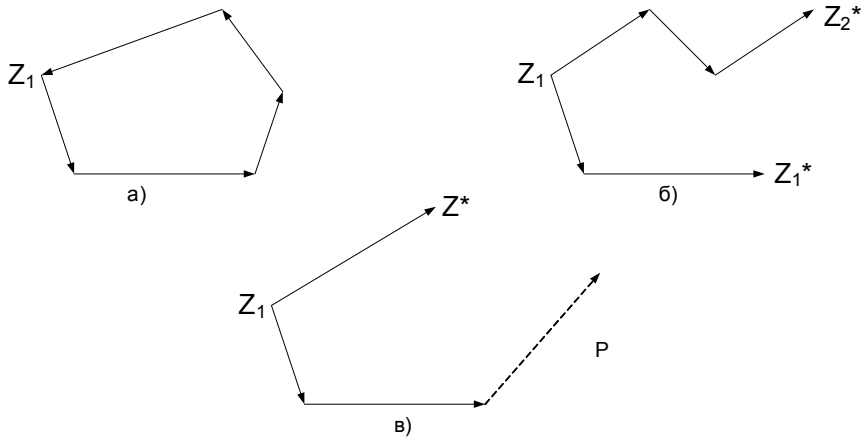


Рис. 4.6. Возможные *s*-пути

Теорема 4.8. Пусть P – s-путь, содержащий неограниченное ребро F. Тогда на P лежит в точности одна ситуация равновесия. Ее можно вычислить, последовательно проходя путь P, начиная с ребра F. Общее число ситуаций равновесия конечно и нечетно.

Доказательство. Единственный s-путь P, начинающийся с F, должен закончиться в некоторой крайней точке, являющейся ситуацией равновесия (см. рис. 4.6, *в*). Отличный от P незамкнутый путь должен иметь две конечные точки (см. рис. 4.6, *б*), каждая из которых является ситуацией равновесия. Отсюда следует утверждение теоремы.

4.1.4.4. Алгоритм Лемке – Хоусона решения биматричных игр

Дана биматричная игра с матрицами A_1 и A_2 размерностью $m \times n$.

1. Положим $N = 0$ – номер итерации. Выбираем $d = \max\limits_{i,j}\left\{\left| a_{ij}^1 \right|, \left| a_{ij}^2 \right|\right\} + 1$. Определяем матрицы $A = dE - A_1$ и $B = dE - A_2$.

2. Выбираем начальные базисы $q^0(Y) = \{e_1, ..., e_n\}$ и $p^0(X) = \{f_1, ..., f_m\}$. Составляем симплекс-таблицы.

Таблица *A*

Базис	a_1	a_2	...	a_m	e_1	e_2	...	e_n
e_1	a_{11}	a_{21}	...	a_{m1}	1	0	...	0
e_2	a_{12}	a_{22}	...	a_{m2}	0	1	...	0
...	...	...	...	...	...	...	...	...
e_n	a_{1n}	a_{2n}	...	a_{mn}	0	0	...	1

Таблица *B*

Базис	b_1	b_2	...	b_n	f_1	f_2	...	f_m
f_1	b_{11}	b_{12}	...	b_{1n}	1	0	...	0
f_2	b_{21}	b_{22}	...	b_{2n}	0	1	...	0
...	...	...	...	...	...	...	...	...
f_m	b_{m1}	b_{m2}	...	b_{mn}	0	0	...	1

3. Выбираем множество H_s, в котором будет производиться поиск ситуации равновесия. Определяем исходную точку (X^0, Y^0):

$$Y^0 = \left(0, 0, ..., 0, \underbrace{1/l}_{s\text{-я позиция}}, 0, ..., 0\right),$$

где $l = a_{i^* s} = \min\limits_i (a_{1s}, a_{2s}, ..., a_{ms})$, т.е. минимум в s-й строке таблицы *A*;

$$X^0 = \left(0, 0, ..., 0, \underbrace{1/k,}_{i^*\text{-я позиция}} 0, ..., 0 \right),$$

где $k = b_{i^* j^*} = \min_j \left(b_{i^* 1}, b_{i^* 2}, ..., b_{i^* n} \right)$, т.е. минимум в строке с номером i^* таблицы B.

Таким образом, (X^0, Y^0) выбрана так, чтобы она была крайней точкой множества Z. Согласно теореме 4.6, $(X^0, Y^0) \in H_s$, где H_s состоит из точек множества $X \times Y$, удовлетворяющих уравнениям

$$(f_i, X)((a_i, Y) - 1) = 0, \quad i = 1,...,m,$$

$$(e_j, Y)((b_j, X) - 1) = 0, \quad j = 1,...,n, \quad j \neq s,$$

и, возможно, уравнению $(e_s, Y^*)((b_s, X^*) - 1) = 0$.

4. Если последнее уравнение выполнено для точки (X^0, Y^0), то это ситуация равновесия. Переходим к шагу 13. В противном случае переходим к следующему шагу.

5. Изменяем базисы p^0 и q^0, для этого найдем множества

$$p(X^0) = \{f_i, b_j \mid (f_i, X^0) = 0, (b_j, X^0) = 1\},$$

$$q(Y^0) = \{e_j, a_i \mid (e_j, Y^0) = 0, (a_i, Y^0) = 1\}.$$

6. Выписываем таблицы A^N и B^N (N – номер итерации) для новых базисов так же, как в симплекс-методе (разд. 1). Чтобы получить таблицу A^N, надо исключить из базиса вектор e_s и ввести вектор a_{i^*}. Для этого вычтем из всех строк таблицы необходимо вычесть s-ю строку, умноженную на множители, подобранные таким образом, чтобы во всех строках, кроме s-й, i^*-й элемент обратился в нуль. Затем разделить s-ю строку на $a_{i^* s}$.

Таблица A^N

Базис	a_1 a_2 ... a_m	e_1 e_2 ... e_n	λ
$q(Y^N)$	α_{11} α_{21} ... α_{m1} α_{12} α_{22} ... α_{m2} ... ... α_{1n} α_{2n} ... α_{mn}	q_{11} q_{12} ... q_{1n} q_{21} q_{22} ... q_{2n} ... ... q_{n1} q_{n2} ... q_{nn}	λ_1 λ_2 ... λ_n
	$\xi_1 - 1$ $\xi_2 - 1$ ... $\xi_m - 1$	y_1^N y_2^N ... y_n^N	

В таблицу A^N добавляется снизу строка и справа один столбец. Значения $\xi_i = (a_i, Y^0)$, причем a_i берется из таблицы, полученной на предыдущем шаге вычислений; y_j^0 – j-я компонента вектора Y^0; заметим, что $\xi_i \geq 1$, $i = 1,...,m$, так как $Y^0 \in \mathrm{Y}$. Значение λ_j для j-й строки получается следующим образом:

вычисляются

$$\lambda^* = \min_{\substack{\alpha_{kj} < 0, q_{jr} < 0 \\ k=\overline{1,m}, r=\overline{1,n}}} \left\{ -\frac{\xi_k - 1}{\alpha_{kj}}, -\frac{y_r^N}{q_{jr}} \right\} \geq 0,$$

$$\lambda^{**} = \min_{\substack{\alpha_{kj} > 0, q_{jr} > 0 \\ k=\overline{1,m}, r=\overline{1,n}}} \left\{ -\frac{\xi_k - 1}{\alpha_{kj}}, -\frac{y_r^N}{q_{jr}} \right\} \leq 0;$$

так как (X^0, Y^0) – крайняя точка, можно показать, что либо λ^*, либо λ^{**} равно нулю. В столбец λ в качестве λ_j добавляется то из чисел λ^* и λ^{**}, которое отлично от нуля. Если они оба равны нулю, добавляется нуль.

Аналогично формируется таблица B^N. Из базиса исключается f_{i^*} и на его место вводится b_{j^*}. В нижней строке таблицы помещаются числа $\eta_j = (b_j, X^N)$ без единицы и компоненты x_i^N. Числа μ_i рассчитываются аналогично λ_j.

Таблица B^N

Базис	b_1 b_2 ... b_n	f_1 f_2 ... f_m	μ
$p(X^N)$	β_{11} β_{12} ... β_{1n} β_{21} β_{22} ... β_{2n} ... ... β_{m1} β_{m2} ... β_{mn}	p_{11} p_{12} ... p_{1m} p_{21} p_{22} ... p_{2m} ... ... p_{m1} p_{m2} ... p_{mm}	μ_1 μ_2 ... μ_m
	$\eta_1 - 1$ $\eta_2 - 1$ ... $\eta_n - 1$	x_1^N x_2^N ... x_m^N	

7. Утверждаем, что $Y = Y^0 + \lambda_j q_j$, где q_j – j-я строка входящей в таблицу матрицы $\|q_{ij}\|$, является крайней точкой множества Y. Покажем, прежде всего, что $Y \in \mathrm{Y}$, т.е. что

$$(a_k, Y^0 + \lambda_j q_j) \geq 1 .$$

Матрица $\|q_{ij}\|$ является обратной матрице, столбцами которой служат векторы базиса, поэтому справедливы соотношения

$$(a_k, q_j) = \alpha_{kj}, \quad k = 1,2,...,m, \quad j = 1,2,...,n .$$

Таким образом,

$$(a_k, Y^0) + \lambda_k (a_k, q_j) = \xi_k + \lambda_j \alpha_{kj} ,$$

причем правую часть равенства преобразуем следующим образом:

$$(\xi_k - 1) + \lambda_j \alpha_{kj} = -\alpha_{kj} \left(-\frac{\xi_k - 1}{\alpha_{kj}} - \lambda_j \right).$$

Если $\lambda_j > 0$, то $-\dfrac{\xi_k - 1}{\alpha_{kj}} - \lambda_j \geq 0$ по определению для λ^*, и потому при $\alpha_{kj} < 0$ имеем

$$(\xi_k - 1) + \lambda_j \alpha_{kj} \geq 0 ,$$

т.е.

$$(a_k, Y^0) + \lambda_k (a_k, q_j) \geq 1.$$

Если же $\lambda_j > 0$ и $\alpha_{kj} > 0$, то $\lambda_j \alpha_{kj} > 0$, поэтому вновь

$$(a_k, Y^0) + \lambda_k (a_k, q_j) \geq 1,$$

т.е. доказали, что $Y \in \mathrm{Y}$.

При $\lambda_j < 0$ – аргументация аналогична. Таким образом, $Y \in \mathrm{Y}$. Точка Y определена так, что она является крайней точкой множества Y. Ввиду того, что Y – конец ребра, другим концом которого является Y^0, числа λ^* и λ^{**} не могут одновременно отличаться от нуля, так как в противном случае

$$(Y^0 + \lambda^* q_j) \in \mathrm{Y}, \quad (Y^0 + \lambda^{**} q_j) \in \mathrm{Y}, \quad \lambda^* > 0, \quad \lambda^{**} < 0,$$

вопреки тому, что Y^0 – крайняя точка.

Таким образом, на данном шаге алгоритма рассчитываем крайнее точки

$$X^i = X^0 + \mu_i p_i, \quad i = 1,2,...,m,$$

$$Y^j = Y^0 + \lambda_j q_j, \quad j = 1,2,...,n.$$

Точки (X^0, Y^j) и (X^i, Y^0) являются смежными с точкой (X^0, Y^0).

8. Находим $q(Y^j)$ и $p(X^i)$ в соответствии с определением базиса (п. 5 настоящего алгоритма).

9. Для каждой пары (X^0, Y^j) и (X^i, Y^0) формируем множество M:

$$M(X^i, Y^j) = \left\{ e_r, f_k \left| \begin{array}{l} e_r \in M, \text{ если } e_r \in q(Y^j), b_r \in p(X^i), \\ f_k \in M, \text{ если } f_k \in p(X^i), a_k \in q(Y^j). \end{array} \right. \right\}$$

10. Если для некоторой пары (X^i, Y^j)

$$M(X^i, Y^j) = \{e_1, e_2, ..., e_n, f_1, ..., f_m\},$$

то процесс вычислений заканчивается. В этом случае (X^i, Y^j) – ситуация равновесия, т.е.

$$(X^*, Y^*) = (X^i, Y^j).$$

Переходим к шагу 13.

Если такой пары не найдется, то переходим к следующему шагу.

11. Выбираем точку $Z = (X, Y)$, для которой в $M(X, Y)$ недостает вектора e_s. Тогда $(X, Y) \in H_s$. По теореме 4.7 найдется пара таких точек Z_1, Z_2, являющихся концами ребер, которые пересекаются в (X^0, Y^0). Если к точке (X^0, Y^0) примыкает неограниченное ребро, такая точка будет одна.

12. Выбираем ту из найденных точек Z_1, Z_2, которая не была включена в s-путь на предыдущих итерациях, изменяем базис $p(X)$ или $q(Y)$ и возвращаемся на шаг 6. (При этом на шаге 6 меняется лишь одна из таблиц A или B.)

Согласно теореме 4.8, через конечное число шагов процесс закончится или в ситуации равновесия, или в исходной точке (X^0, Y^0), если произойдет зацикливание. В последнем случае выбираем новое s и начинаем поиск сначала.

13. Нормируем пару (X^*, Y^*):

$$X^0 = \frac{X^*}{(X^*, L_m)}, \quad Y^0 = \frac{Y^*}{(Y^*, L_n)}.$$

Вычисляем выигрыши игроков:

для игрока 1

$$V_1 = d - \frac{1}{(Y^*, L_n)},$$

для игрока 2

$$V_2 = d - \frac{1}{(X^*, L_m)}.$$

(Последние соотношения выводятся при доказательстве теоремы 4.2.)

4.2. Оптимальное распределение нагрузки в системе ядерных реакторов

4.2.1. Оптимальное распределение запасов реактивности в системе двух реакторов с линейной зависимостью возможной степени снижения мощности от запаса реактивности

В разд. 2.3 рассмотрена оптимизационная задача с линейной зависимостью запаса реактивности от степени снижения мощности. Эта ситуация характерна для низкопоточных реакторов. Однако, в большинстве энергетических реакторов уровень плотности потоков нейтронов больше, чем $1 \cdot 10^{13}\ \dfrac{\text{нейтр.}}{\text{см}^2 \cdot \text{с}}$ при этом зависимость $\varepsilon(\Delta\rho)$, как видно из рис. 2.2, носит нелинейный характер. Для получения качественных результатов рассмотрим систему из двух реакторов. Зависимость $\varepsilon(\Delta\rho)$ будем аппроксимировать полиномом второй степени.

Математически задача в этом случае формулируется следующим образом:

найти

$$\min_{\Delta\rho_i \ldots \Delta\rho_N} \left\{ \sum_{i=1}^{N} \frac{\Delta\rho_i}{a_i} \right\}$$

при ограничениях:

$$\begin{aligned} &\sum_{i=1}^{N} \delta_i \left[1 - 2\frac{\Delta\rho_i}{\Delta\rho_{im}} + \left(\frac{\Delta\rho_i}{\Delta\rho_{im}} \right)^2 \right] = \alpha; \\ &0 \le \Delta\rho_1 \le \Delta\rho_{1m}; \\ &0 \le \Delta\rho_2 \le \Delta\rho_{2m}. \end{aligned} \tag{4.88}$$

Введя новые переменные так же, как в разд. 2.3, т.е.

$$z_1 = \frac{\Delta\rho_1}{\Delta\rho_{1m}}, \quad z_2 = \frac{\Delta\rho_2}{\Delta\rho_{2m}},$$

запишем задачу (4.88) в виде:

найти

$$\min_{z_1, z_2}\left(\frac{z_1}{a_1/\varphi_1}+\frac{z_2}{a_2/\varphi_2}\right)$$

при ограничениях

$$\frac{(z_1-1)^2}{1/\delta_1}+\frac{(z_2-1)^2}{1/\delta_2}=\alpha,$$
$$0\le z_1\le 1, \qquad (4.89)$$
$$0\le z_2\le 1.$$

Решение задачи. Решение задачи можно получить одним из методов, изложенных в разд. 3.4. Однако в данном конкретном случае легко решить задачу графически. Минимизируемая функция $S=\left(\frac{z_1}{a_1/\varphi_1}+\frac{z_2}{a_2/\varphi_2}\right)$ представляет собой прямую, уравнение связи между переменными – либо эллипс при $\delta_1\ne\delta_2$ (см. рис. 4.7), либо окружность при $\delta\ =\delta_2$, область изменения переменных – квадрат ODEL.

Рис. 4.7. Решение оптимизационной задачи для системы двух реакторов с нелинейной зависимостью степени снижения мощности от запаса реактивности

Нетрудно видеть, что оптимальным решением являются координаты точки $M(z_1^*, z_2^*)$ касания прямой и эллипса, если точка касания принадлежит области изменения переменных – квадрату ODEL. Найдем координаты точки касания.

Уравнение касательной к эллипсу в точке (z_1^*, z_2^*) примет вид

$$\frac{z_1(z_1^* - 1)}{1/\delta_1} + \frac{z_2(z_2^* - 1)}{1/\delta_2} = \alpha .$$

Условие совпадения углового коэффициента касательной с угловым коэффициентом целевой функции есть:

$$\frac{z_1^* - 1}{1/\delta_1} \Big/ \frac{z_2^* - 1}{1/\delta_2} = \frac{1}{a_1/\varphi_1} \Big/ \frac{1}{a_2/\varphi_2},$$

откуда

$$\frac{z_1^* - 1}{z_2^* - 1} = \frac{\delta_2 a_2}{\varphi_2} \Big/ \frac{\delta_1 a_1}{\varphi_1} = \frac{1}{F} . \qquad (4.90)$$

Используя полученное соотношение (4.90) и условие связи между переменными

$$\frac{(z_1^* - 1)^2}{1/\delta_1} + \frac{(z_2^* - 1)^2}{1/\delta_2} = \alpha$$

получим оптимальное распределение относительных запасов реактивности

$$\left.\begin{aligned} z_1^* &= 1 - \frac{1}{F}\sqrt{\frac{\alpha}{\delta_2 + \delta_1/F^2}}; \\ z_2^* &= 1 - \sqrt{\frac{\alpha}{\delta_2 + \delta_1/F^2}}. \end{aligned}\right\} \qquad (4.91)$$

Определим интервал изменения α, при котором точка касания $M(z_1^*, z_2^*)$ принадлежит области изменения переменных

$$0 \le z_1 \le 1;$$
$$0 \le z_2 \le 1.$$

Используя (4.91), получим, что

$$z_1 \ge 0 \quad \text{при} \quad \alpha_1 = \alpha \le F^2\delta_2 + \delta_1;$$

$$z_2 \ge 0 \quad \text{при} \quad \alpha_2 = \alpha \le \delta_2 + \delta_1 \big/ F^2 .$$

Понятно, что $z_1 \ge 0$ и $z_2 \ge 0$ одновременно при $\alpha = \min(\alpha_1, \alpha_2)$.

Определим соотношение между α_1 и α_2 в зависимости от значения параметра системы *F*.

$$\alpha_1 - \alpha_2 = \Delta\alpha = F^2\delta_2 + \delta_1 - \frac{F^2\delta_2 + \delta_1}{F^2} = \frac{(F^2 - 1)(F^2\delta_2 + \delta_1)}{F^2}. \quad (4.92)$$

Из выражения (4.92) видно, что при $F > 1$, $\alpha_1 > \alpha_2$. Ограничением в этом случае является условие $\alpha = \alpha_2$, т.е. $\alpha \le \delta_2 + \delta_1 \big/ F^2$:

при $F < 1$, $\alpha_1 > \alpha_2$ и ограничением является условие $\alpha \le \delta_2 F^2 + \delta_1$;

при $F = 1$, $\alpha_1 = \alpha_2$ ограничением является условие $\alpha \le \delta_1 + \delta_2 = 1$, которое выполняется всегда. Следовательно, при $F = 1$ точка касания при любом α принадлежит области изменения переменных и находится внутри нее.

На рис. 4.7 показан случай, когда параметр системы $F > 1$ (для определенности изображена ситуация, когда $\frac{a_1}{\varphi_1} \Big/ \frac{a_2}{\varphi_2} = 1, \frac{\delta_1}{\delta_2} = 2$).

При $\alpha \le \delta_2 + \delta_1 \big/ F^2$ точка касания прямой и эллипса $M(z_1^*, z_2^*)$ принадлежит области изменения переменных – квадрату ODEL. При $\alpha > \delta_2 + \delta_1 \big/ F^2$ точка касания M_1' выходит за область изменения переменных. Ближайшей к точке M_1' является точка M_1 – точка пересечения эллипса с осью z_1. Координаты точки M_1 и будут являться решением задачи.

Таким образом, решением задачи при параметре системы $F > 1$ является:

$$\left.\begin{array}{l}\begin{cases} z_1^* = 1 - \dfrac{1}{F}\sqrt{\dfrac{\alpha}{\delta_2 + \delta_1 / F^2}}; \\ z_2^* = 1 - \sqrt{\dfrac{\alpha}{\delta_2 + \delta_1 / F^2}} \qquad \text{при} \quad 0 < \alpha \le \delta_2 + \delta_1 / F^2; \end{cases} \\ \begin{cases} z_1^* = 1 - \sqrt{\dfrac{\alpha - \delta_2}{\delta_1}}; \\ z_2^* = 0 \qquad \text{при} \quad \delta_2 + \delta_1 / F^2 \le \alpha < 1, \end{cases} \end{array}\right\} \quad (4.93)$$

Решением задачи при параметре системы $F < 1$ будут:

$$\left.\begin{array}{l}\begin{cases} z_1^* = 1 - \dfrac{1}{F}\sqrt{\dfrac{\alpha}{\delta_2 + \delta_1 / F^2}}; \\ z_2^* = 1 - \sqrt{\dfrac{\alpha}{\delta_2 + \delta_1 / F^2}} \qquad \text{при} \quad 0 < \alpha \le \delta_2 F^2 + \delta_1; \end{cases} \\ \begin{cases} z_1^* = 0; \\ z_2^* = 1 - \sqrt{\dfrac{\alpha - \delta_1}{\delta_2}} \qquad \text{при} \quad \delta_2 F^2 + \delta_1 \le \alpha < 1, \end{cases} \end{array}\right\} \quad (4.94)$$

Если параметр системы равен единице, то оптимальным является распределение:

$$z_1^* = z_2^* = 1 - \sqrt{\alpha} \quad \text{при} \quad 0 < \alpha < 1. \qquad (4.95)$$

Из выражений (4.93), (4.94) легко видеть, что

$$z_1^* > z_2^* \quad \text{при} \quad F > 1;$$

$$z_1^* < z_2^* \quad \text{при} \quad F < 1,$$

т.е. больший запас реактивности резервируется в реакторе с большим значением комплекса $\dfrac{\delta a}{\varphi}$.

На рис. 4.8 показаны фазовые диаграммы оптимального распределения запасов реактивности.

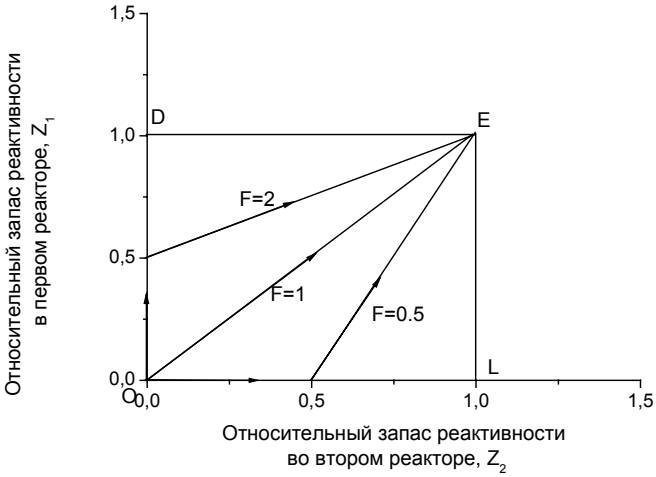


Рис. 4.8. Траектории оптимальных распределений запасов реактивности в системе двух реакторов с нелинейной зависимостью $\varepsilon(\Delta\rho)$

Используя выражения для оптимальных распределений запасов реактивности, можно получить оптимальное распределение степеней снижения мощностей реакторов, с учетом того, что

$$\varepsilon_1^* = (z_1^* - 1)^2 \quad \text{и} \quad \varepsilon_2^* = (z_2^* - 1)^2 .$$

Оптимальное распределение степеней снижения мощностей реакторов есть:

1) параметр системы $F > 1$

$$\left\{\begin{cases} \varepsilon_1^* = \dfrac{1}{F}\dfrac{\alpha}{\delta_2 + \delta_1/F^2}; \\ \varepsilon_2^* = \dfrac{\alpha}{\delta_2 + \delta_1/F^2} \qquad \text{при} \quad 0 < \alpha \le \delta_2 + \delta_1/F^2; \end{cases}\right.$$

$$\left.\begin{cases} \varepsilon_1^* = \dfrac{\alpha - \delta_2}{\delta_1}; \\ \varepsilon_2^* = 1 \qquad \text{при} \quad \delta_2 + \delta_1/F^2 \le \alpha < 1; \end{cases}\right\}$$

2) параметр системы $F < 1$

$$\left\{\begin{array}{ll} \begin{cases} \varepsilon_1^* = \dfrac{1}{F^2}\dfrac{\alpha}{\delta_2 + \delta_1/F^2}; \\ \varepsilon_2^* = \dfrac{\alpha}{\delta_2 + \delta_1/F^2} \end{cases} & \text{при} \quad 0 < \alpha \le \delta_2 F^2 + \delta_1; \\ \begin{cases} \varepsilon_1^* = 1; \\ \varepsilon_2^* = \dfrac{\alpha - \delta_1}{\delta_2} \end{cases} & \text{при} \quad \delta_2 F^2 + \delta_1 \le \alpha < 1; \end{array}\right\}$$

3) параметр системы $F = 1$

$$\varepsilon_1^* = \varepsilon_2^* = \alpha \qquad \text{при} \quad 0 < \alpha < 1 .$$

На рис. 4.9 показаны фазовые диаграммы оптимальных степеней снижения мощностей реакторов.

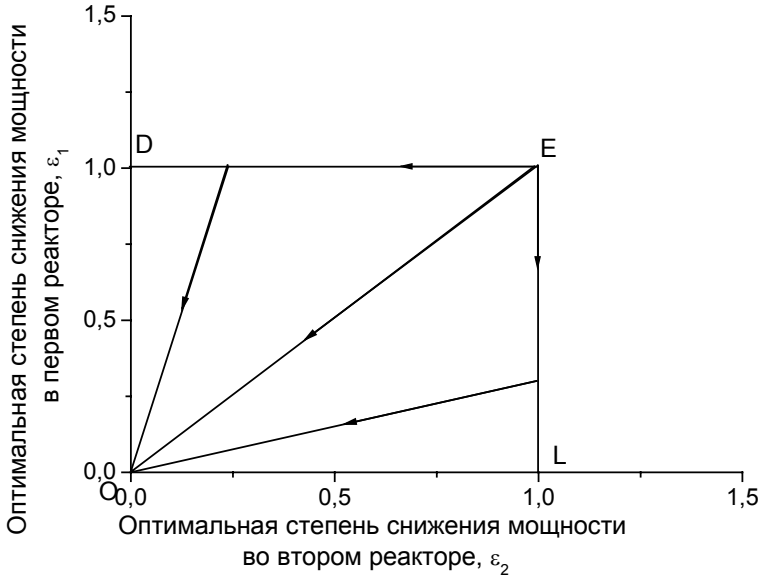


Рис. 4.9. Траектории оптимальных степеней снижения мощности в системе двух реакторов с нелинейной зависимостью $\varepsilon(\Delta\rho)$

Таблица 4.1

Оптимальные режимы эксплуатации системы двух реакторов с нелинейной зависимостью $\varepsilon(\Delta\rho)$ при различных значениях параметра системы

Параметр системы F	Степень снижения мощности АЭС	Оптимальные траектории		Оптимальные режимы	
		запасов реактивности	степеней снижения мощности	первый реактор	второй реактор
$F > 1$	$\delta_2 + \delta_1 / F^2 \le \alpha < 1$	OG	EG	полупиковый	базисный
	$0 < \alpha \le \delta_2 + \delta_1 / F^2$	GE	GO	полупиковый	полупиковый
$F = 1$	$0 < \alpha < 1$	OE	EO	полупиковый	полупиковый
$F < 1$	$\delta_2 F^2 + \delta_1 \le \alpha < 1$	OT	ET	базисный	полупиковый
	$0 < \alpha \le \delta_2 F^2 + \delta_1$	TE	TO	полупиковый	полупиковый

В табл. 4.1 показан характер оптимальных режимов эксплуатации реакторов в зависимости от величины параметра системы F. Для определенности рассмотрены случаи $F = 2$, $F = 1$, $F = 0{,}5$. Как видно из таблицы, оптимальный режим эксплуатации реакторов может быть двух типов при $F \ne 1$ и одного типа при $F = 1$. Если параметр системы не равен единице, то существует интервал снижения мощности АЭС, в пределах которого один из реакторов (а именно тот, у которого величина комплекса $\dfrac{\delta a}{\varphi}$ меньше) работает в базисном режиме, в то время как другой отрабатывает заданную степень снижения мощности АЭС. При параметре системы, равном единице, снижение мощности АЭС отрабатывается одновременно двумя реакторами во всем диапазоне изменения α ($0 < \alpha < 1$), причем степени снижения мощностей реакторов равны. В системе ре-

акторов с нелинейной зависимостью $\varepsilon(\Delta\rho)$ ни в одном из них не резервируется запас реактивности на полную остановку (за исключением случая, когда требуемая степень снижения мощности АЭС равна нулю). Это объясняется тем, что запас реактивности, обеспечивающий заданную степень снижения мощности реактора, резко увеличивается при $\varepsilon \to 0$, что, в свою очередь, приводит к увеличению потери энерговыработки.

4.2.2. Максимально возможный эффект оптимизации

Величина максимально возможного эффекта оптимизации оценивалась так же, как в разд. 2.3. Результаты расчетов для конкретного случая, когда доли мощности реакторов одинаковы, приведены на рис. 4.10.

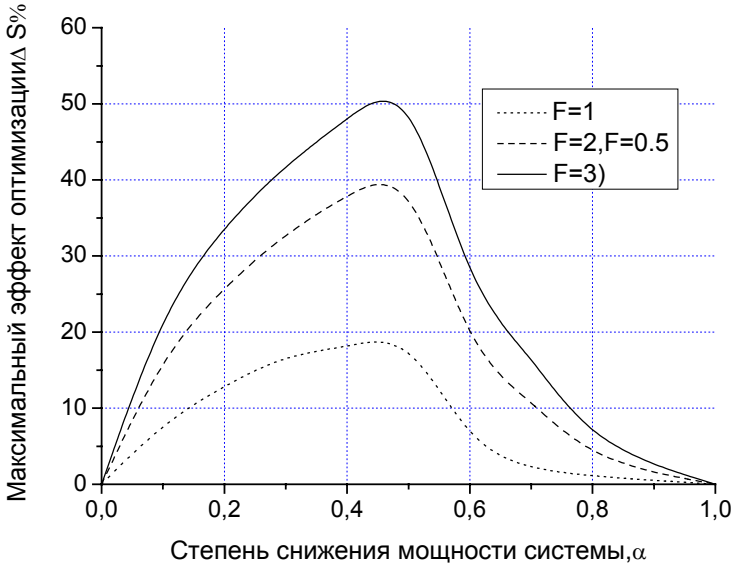


Рис. 4.10. Зависимость величины максимально возможного эффекта оптимизации от степени снижения мощности системы реакторов с нелинейной зависимостью $\varepsilon(\rho)$ при различных величинах параметра F

Как видно из рисунков, оптимизация дает тем больший эффект, чем больше параметр системы F отличается от единицы. Оптими-

зация системы наиболее существенна в области снижения мощности АЭС со 100 до 20 % номинальной.

Полученные результаты для реакторов с нелинейной зависимостью сводятся к следующим:

Характер оптимальных распределений запасов реактивности и оптимальных степеней снижения мощности определяется величиной параметра системы $F = \frac{\delta_1 a_1}{\varphi_1} \Big/ \frac{\delta_2 a_2}{\varphi_2}$.

Оптимальный режим эксплуатации реакторов может быть двух типов при $F \neq 1$ и одного типа при $F = 1$.

Больший относительный запас реактивности следует создавать в реакторе с большей величиной комплекса $\frac{\delta a}{\varphi}$. Этот реактор в первую очередь вовлекается в переменный режим работы.

Эффект оптимизации наиболее существенен в диапазоне снижения мощности АЭС от 100 до 30 % и тем больше, чем больше параметр системы F отличается от единицы. Но даже в случае одинаковых реакторов ($F = 1$) эффект может быть существенным.

Например, из решения оптимизационной задачи для двух реакторов типа РБМК, следует что оптимальным является равномерное снижение мощности, а «антиоптимальным» – отработка переменного графика одним блоком.

При 50 % уровне снижения мощности АЭС максимально возможный эффект за компанию топлива составляет около $\approx 3\,\%$ от энерговыработки системы, что соответствует $\approx 10^5$ МВт · сут

В целом, решение задачи по оптимизации распределения запасов реактивности в системе реакторов позволяет сделать следующие выводы.

1. Характер оптимального распределения запаса реактивности определяется величиной параметра системы F (для системы двух реакторов $F = \frac{\delta_1 a_1}{\varphi_1} \Big/ \frac{\delta_2 a_2}{\varphi_2}$) и зависит от степени снижения мощности системы. В переменный режим работы, как правило, в пер-

вую очередь вовлекается реактор с большей величиной комплекса $\frac{\delta a}{\varphi}$, иначе говоря, с худшим использованием топлива.

2. Возможный проигрыш от пренебрежения оптимизацией наиболее существенен в предполагаемом регулировочном диапазоне работы АЭС. Величина эффекта оптимизации тем больше, чем больше параметр системы отличается от единицы.

3. Решение рассмотренной задачи является также решением обратной задачи – об оптимальном распределении запасов реактивности с целью увеличения маневренных свойств системы (т.е. уменьшения α) при заданной суммарной потере энерговыработки.

4.3. Формирование банковского портфеля максимальной доходности

4.3.1. Основные характеристики ценных бумаг

Одним из важных прикладных аспектов методов математического программирования и, в частности, нелинейного программирования является оптимизация банковской деятельности. Особое внимание уделяется оптимизации работы с ценными бумагами. В этой связи возникает актуальная задача формирования состава ценных бумаг банковского портфеля, обеспечивающих максимальную доходность банковских операций при работе с этими ценными бумагами при сохранении заданной величины риска. Таким образом, основными понятиями при решении данной задачи являются:

доходность банковского портфеля;

риск банковских операций с ценными бумагами, входящими в портфель.

Не вдаваясь в подробности банковских операций, рассмотрим основные характеристики ценных бумаг.

Банковский портфель представляет собой набор активов (пассивов), являющихся титулами собственности или иных благ (акции, векселя, валюта, ваучеры, аккредитивы и т.д.).

Ожидаемая доходность банковского портфеля, есть взвешенная средняя ожидаемой доходности каждого из активов, входящих в

портфель, где весами служат доли инвестиций в каждый из активов от всей суммы, вложенной в портфель:

$$R_{pT}(x_1 \ldots x_n) = \sum_{i=1}^{n} R_{iT} W_i; \quad i = \overline{1,n}, \tag{4.96}$$

где R_{pT} – доходность всего инвестиционного портфеля за период времени T; R_{iT} – доходность единицы i-го актива; W_i – доля инвестиций в i-й актив, от общей суммы инвестирования в портфель, причем $\sum_{i=1}^{n} W_i = 1$, x_i – количество единиц i-й ценной бумаги.

Доходность единицы i-го актива за временной промежуток T определяется следующим образом:

$$R_{iT} = \ln\left(P_i(t) / P_i(t-T)\right). \tag{4.97}$$

Как следует из формулы (4.96), доходность инвестиционного портфеля будет зависеть от двух параметров: доходности отдельного актива, входящего в портфель и доли инвестиции в каждый актив.

Объем инвестиций, вложенных в портфель, будет равен сумме произведений стоимости единицы актива на его количество в портфеле:

$$V = \sum_{i=1}^{n} V_i \cdot x_i .$$

Объем инвестиций, вложенных в отдельный актив портфеля, будет равен произведению стоимости единицы актива на количество единиц актива в портфеле:

$$V(x_i) = V_i x_i, \tag{4.98}$$

где $V(x_i)$ – объем инвестиций, вложенных в i-ю ценную бумагу; V_i – цена единицы i-й ценной бумаги.

Долю инвестиций в каждый актив можно выразить следующим образом:

$$W_i = \frac{V(x_i)}{V_{port}} = \frac{V_i \cdot x_i}{V_{port}}. \tag{4.99}$$

Теперь формулу доходности портфеля (4.96) можно переписать в следующем виде:

$$R_{pT}(x_1 \ldots x_n) = \sum_{i=1}^{n} R_{iT} \cdot \frac{V_i x_i}{V_{port}}. \qquad (4.100)$$

Как уже отмечалось выше, важной характеристикой банковского портфеля является величина риска. Риск потери капитала, в связи с неблагоприятной ситуацией, складывающейся на рынке – один из наиболее актуальных, в настоящее время, параметров, характеризующих финансовый инструмент.

Одним из важнейших видов рисков, является рыночный риск. Рыночный риск представляет собой возможность отрицательного изменения стоимости активов в результате колебания процентных ставок, курсов валют, цен акций, облигаций и товарных контрактов.

В современной теории риск-менеджмента наиболее распространена модель риска под названием *value at risk* или *рисковая стоимость – VaR*.

Концепция value at risk (*VaR*) призвана дать четкий и однозначный ответ на вопрос, возникающий при проведении операций на финансовых рынках: какой максимальный убыток мы рискуем понести за определенный период времени с заданной вероятностью для данного портфеля?

Из этого следует, что величина *VaR* для портфеля заданной структуры *определяется как наибольший ожидаемый убыток и рассчитывается на определенный период времени в будущем (временной горизонт) с заданной вероятностью непревышения некоторого значения VaR (доверительный интервал) при данном предположении о характере поведения рынка (метод расчета)*.

Доверительный интервал и временной горизонт являются ключевыми параметрами при расчете величины *VaR*.

Так, например, значение рисковой стоимости (*VaR*) в 10 млн дол. для временного горизонта в один день и доверительного интервала в 99 % будет означать следующее:

вероятность того, что в течение следующих 24 ч мы потеряем меньше 10 млн дол., составляет 99 %;

вероятность того, что наши убытки превысят 10 млн дол. в течение ближайших суток, равна 1 %;

убытки, превышающие 10 млн дол., ожидаются в среднем один раз в 100 дней торгов.

Таким образом, рисковая стоимость является денежным показателем, отражающим ожидаемые потери с заданной степенью достоверности. Очевидно, рисковая стоимость для i-го актива является разностью между текущим значением цены актива $V_i(t)$ и её прогнозным значением $V_{i,(1-\alpha)}(t+\tau)$, полученным для момента $(t+\tau)$ с доверительной вероятностью $(1-\alpha)$. Таким образом, можно записать:

$$VaR_{i,1-\alpha}(t+\tau) = V(t) - V_{i,(1-\alpha)}(t+\tau). \qquad (4.101)$$

В настоящее время наибольшее распространение получил ковариационный метод (variance-covariance) расчета величины VaR. В его основе лежит допущение о нормальном законе распределения изменений цен активов, входящих в портфель.

При нормально распределенной случайной величине, доверительный интервал (1 – α) всегда характеризуется квантилью $(k_{1-\alpha})$, которая показывает положение искомого значения случайной величины относительно среднего, выраженного в количестве среднеквадратичных отклонений этой случайной величины σ_i от среднего значения.

Для наиболее часто используемых значений доверительного интервала в 95 и 99 % соответствующие квантили будут равны 1,65 и 2,33 стандартных отклонений.

В случае принятия гипотезы о нормальном законе распределения стоимости i-го актива прогнозируемое значение цены для однодневного временного среза $(\tau = 1)$ и доверительной вероятности $(1-\alpha)$ определяется по формуле:

$$V_{i,(1-\alpha)}(t+1) = V(t)\exp(-k_{1-\alpha}\sigma_i(t)). \qquad (4.102)$$

Последняя формула записана в предположении, что математическое ожидание *однодневных доходностей* равно нулю.

Стандартное отклонение $\sigma_i(t)$ может быть оценено по ограниченной выборке цен (историческому периоду наблюдений) i-го актива.

Интересующая нас величина VaR_i для временного среза в один день и доверительной вероятности $(1-\alpha)$ может быть определена следующим образом:

$$VaR_{i,1-\alpha}(t+1) = V_i(t)\{[\exp(-k_{1-\alpha}\sigma_i(t))]-1\}. \qquad (4.103)$$

При малых значениях величины $-k_{1-\alpha}\sigma_i(t)$ выражение $\{[\exp(-k_{1-\alpha}\sigma_i(t))]-1\}$ можно заменить на выражение $-k_{1-\alpha}\sigma_i(t)$. Эта линейная аппроксимация для малых значений $\sigma_i(t)$ основана на разложении исходной функции в ряд Тейлора. Весьма часто знак «–» опускают и оперируют абсолютным значением величины *VaR*.

В результате для i-го актива, состоящего из нескольких инструментов, величина рисковой стоимости с временным горизонтом в один день и доверительным интервалом $(1-\alpha)$ может быть рассчитана по следующей формуле:

$$VaR_i = VaR_{i,1-\alpha}(t+1) = V_i(t)k_{1-\alpha}\sigma_i(t),$$

где $V_i(t)$ – текущая стоимость позиции i-го актива (произведение текущей цены на количество единиц актива).

Для рисковой стоимости с временным горизонтом *T* дней и доверительным интервалом $(1-\alpha)$ последняя формула принимает вид

$$VaR_i = VaR_{i,1-\alpha}(t+1) = V_i(t)k_{1-\alpha}\sigma_i(t)\sqrt{T}, \qquad (4.104)$$

Соответствующая формула для расчета *VaR* всего банковского портфеля имеет следующий вид:

$$VaR = \sqrt{\overline{IVaR}^{\mathrm{T}}\Omega\overline{IVaR}}, \qquad (4.105)$$

где $\overline{IVaR}$ – вектор столбец индивидуальных рисков позиций; Ω – корреляционная матрица доходностей факторов риска. В развернутом виде формула (4.105) принимает вид

$$VaR = \sqrt{\sum_{i=1}^{n} VaR_i^2 + 2\sum_{i=1}^{n}\sum_{j=i+1}^{n}\rho_{ij}VaR_iVaR_j}, \qquad (4.106)$$

где VaR_i – рассчитывается по формуле (4.104), а ρ_{ij} – коэффициенты корреляции доходности финансового инструмента, которые рассчитываются следующим образом:

$$(\sigma_{\text{дох}})_i = \sqrt{\sum_{k=1}^{N}\sum_{l=1}^{N} R_{ik}^2}\,; \quad i = \overline{1,n}, \tag{4.107}$$

$$\rho_{ij} = \frac{\sum_{k=1}^{N}\sum_{l=1}^{N} R_{ik} R_{jl}}{(\sigma_{\text{дох}})_i (\sigma_{\text{дох}})_j}; \quad i, j = \overline{1,n}\,. \tag{4.108}$$

В последней формуле R_{ik}, R_{jl} – реализовавшиеся доходности i-ого и j-го финансовых инструментов в моменты наблюдения k и l, N – число наблюдений.

4.3.2. Постановка задачи формирования портфеля максимальной доходности при фиксированной величине риска

Один из методов управления рисками – это наложение ограничений (лимитов) на величину риска *VaR*. При этом участники фондового рынка задаются определенной величиной *VaR* и стараются так сформировать свой портфель, чтобы его доходность была максимальной.

Учитывая формулу расчета доходности (4.100), запишем критерий, который необходимо максимизировать:

$$J(x_1 ... x_n) = R_p(x_1 ... x_n) = \sum_{i=1}^{n} R_i W_i = \sum_{i=1}^{n} R_i \cdot \frac{V_i x_i}{V_{port}}\,. \tag{4.109}$$

Инвестор, обладая ограниченными средствами, в состоянии инвестировать в рынок, объем денежных средств не превышающий V_{port}. В результате получаем ограничение:

$$\sum_{i=1}^{n} V_i x_i \le V_{port}\,. \tag{4.110}$$

Второе ограничение появляется, из лимита на величину риска, при превышении которой, позиция принудительно закрывается:

$$VaR \le VaR_{zad}\,. \qquad (4.111)$$

Подставив в (4.111) соотношения (4.106), (4.104), (4.98), получим рисковую стоимость для временного горизонта T:

$$VaR = \sqrt{\sum_{i=1}^{n}(k_{1-\alpha}\sqrt{T}V_i x_i \sigma_t)^2 + 2\sum_{i=1}^{n}\sum_{j=i}^{n}\rho_{ij}\sqrt{T}\sqrt{T}(k_{1-\alpha}V_i x_i \sigma_t)(k_{1-\alpha}V_j x_j \sigma_t)} \le$$

$$\le VaR_{zad}\,. \qquad (4.112)$$

Количество отдельных видов каждого актива, не может быть отрицательным и является всегда целым, поэтому верно следующее неравенство:

$$x_i \ge 0\,, \quad x_i \in Z; \quad i = \overline{1,n}\,.$$

Таким образом, задача формирования инвестиционного портфеля максимальной доходности, с заданными объемом инвестиций и значением риска выглядит следующим образом.

Найти

$$\max\left\{J(x_1 \ldots x_n) = R_p(x_1 \ldots x_n) = \sum_{i=1}^{n} R_i \frac{V_i x_i}{V_{port}}\right\}$$

при ограничениях

$$\begin{cases} \sum_{i=1}^{n} V_i x_i \le V_{port}; \\ \sqrt{\sum_{i=1}^{n}(k_{1-\alpha}\sqrt{T}V_i x_i \sigma_i)^2 + 2\sum_{i=1}^{n}\sum_{j=i}^{n}\sqrt{T}\sqrt{T}\rho_{ij}(k_{1-\alpha}V_i x_i \sigma_i)(k_{1-\alpha}V_j x_j \sigma_j)} \le VaR_{zad}; \\ x_i \ge 0, i = \overline{1,n}; \\ x_i \in Z, i = \overline{1,n}. \end{cases}$$

Введем обозначения:

$$c_i = R_i \frac{V_i}{V_{port}};\; d_{ij} = \rho_{ij}\sqrt{T}\sqrt{T}k_{1-\alpha}^2 V_i V_j \sigma_j \sigma_i;\; \rho_{ii} = 1;\; i, j = \overline{1,n}\,. \quad (4.113)$$

С учетом введенных обозначений, данную задачу можно записать в виде

найти

$$\max\left\{J(x_1...x_n)=\sum_{i=1}^{n}c_i x_i\right\} \qquad (4.114)$$

при ограничениях

$$\begin{cases}\sum_{i=1}^{n}V_i x_i \le V_{port};\\ \sqrt{\sum_{i=1}^{n}d_{ii}x_i^2+2\sum_{i=1}^{n}\sum_{j=i}^{n}d_{ij}x_i x_j}\le VaR_{zad};\\ x_i\ge 0,\ i=\overline{1,n};\\ x_i\in Z,\ i=\overline{1,n}.\end{cases} \qquad (4.115)$$

4.3.3. Решение задачи формирования оптимального портфеля с использованием множителей Лагранжа

Задача формирования банковского портфеля максимальной доходности с заданной величиной риска *VaR*, является задачей с линейным критерием и смешанными ограничениями, среди которых присутствуют как линейные, так и нелинейное ограничение. Так как максимизируемый критерий (4.114) и ограничения (4.115) (если пренебречь условием целочисленности аргументов оптимизации $x_i,\ i=\overline{1,n}$) являются выпуклыми дифференцируемыми функциями, то поставленная задача может быть решена с помощью функций Лагранжа [22]. Учитывая, что данный метод предполагает минимизацию функции, запишем критерий задачи формирования инвестиционного портфеля в виде

$$J(x_1...x_n)=-\sum_{i=1}^{n}c_i x_i\ , \qquad (4.116)$$

при этом ограничения примут следующий вид:

$$\begin{cases} \sum_{i=1}^{n} V_i x_i - V_{port} \le 0; \\ \sqrt{\sum_{i=1}^{n} d_{ii} x_i^2 + 2\sum_{i=1}^{n}\sum_{j=i}^{n} d_{ij} x_i x_j} - VaR_{zad} \le 0; \\ x_i \ge 0, i = \overline{1,n}. \end{cases} \quad (4.117)$$

Поскольку метод множителей Лагранжа не был рассмотрен в гл. 3, остановимся на этом методе более подробно.

Обозначим левые части неравенств (4.117) через $g_j(\bar{x}),\ j = \overline{1,m}$, а критерий минимизации (4.116) через функцию $f(\bar{x})$, тогда задача будет иметь следующий вид:

найти

$$\min f(\bar{x}) \quad (4.118)$$

при ограничениях

$$g_j(\bar{x}) \le 0, \quad j = \overline{1,m}. \quad (4.119)$$

Введем дополнительные переменные $z_j,\ j = \overline{1,m}$, и перейдем от ограничений неравенств (4.119) к ограничениям равенствам:

$$g_j(\bar{x}) + z^2{}_j = 0, \quad j = \overline{1,m}. \quad (4.120)$$

Запишем функцию Лагранжа задачи (4.118), (4.120):

$$\widetilde{L}(\bar{x}, \bar{\lambda}, \bar{z}) = f(\bar{x}) + \sum_{j=1}^{m} \lambda_j [g_j(\bar{x}) + z^2{}_j]. \quad (4.121)$$

Система уравнений для ее стационарных точек имеет вид

$$\frac{\partial \widetilde{L}}{\partial x_i} = \frac{\partial f}{\partial x_i} + \sum_{j=1}^{m} \lambda_j \frac{\partial g_j}{\partial x_i} = 0,\ i = \overline{1,n}; \quad (4.122)$$

$$\frac{\partial \widetilde{L}}{\partial z_j} = 2\lambda_j z_j = 0,\ j = \overline{1,.m}; \quad (4.123)$$

$$\frac{\partial \widetilde{L}}{\partial \lambda_j} = g_j(\overline{x}) + z^2{}_j = 0, \quad \overline{j = 1, m}; \tag{4.124}$$

Условия (4.122) – (4.124) являются необходимыми условиями минимума задачи (4.118), (4.119). Очевидно, равенства с величинами $z^2{}_j \geq 0$ эквивалентны неравенствам (4.119).

Исключим из этой системы вспомогательные переменные z_j. Умножив каждое равенство из (4.123) на $z_i/2$, получим: $\lambda_j z^2{}_j = 0$ или, как нетрудно убедиться из соотношения (4.124),

$$\lambda_j g_j = 0. \tag{4.125}$$

С учетом последних соотношений, необходимые условия минимума для задачи (4.118) – (4.119) принимают вид

$$\frac{\partial L(\overline{x}, \overline{\lambda})}{\partial x_i} = \frac{\partial f(\overline{x})}{\partial x_i} + \sum_{j=1}^{m} \lambda_j \frac{\partial g_j(x)}{\partial x_i} = 0, \quad i = \overline{1, n}; \tag{4.126}$$

$$\frac{\partial L(\overline{x}, \overline{\lambda})}{\partial \lambda_j} = g_j(\overline{x}) \leq 0, \quad j = \overline{1, m}; \tag{4.127}$$

$$\lambda_j g_j(\overline{x}) = 0, \quad j = \overline{1,.m}. \tag{4.128}$$

Существует следующая теорема.

Теорема. Пусть $\overline{x}^*, \overline{\lambda}^*$ – решение системы (4.126) – (4.128). Тогда, если точка $\overline{x}^*$ является решением задачи (4.126) – (4.128), то $\lambda^*{}_i \geq 0$ для всех $i = 1, \ldots, n$.

С учетом вышеприведенной теоремы и выражений (4.126) – (4.128) можно сформулировать следующее необходимые условия минимума в задаче (4.118) – (4.119) с допустимым множеством, удовлетворяющим условию регулярности: если $\overline{x}^*$ является решением задачи (4.118) – (4.119), то для чисел λ^*_j, $j = 1, \ldots, n$ выполняются соотношения:

$$\lambda^*{}_j \geq 0, \quad j = \overline{1, m}; \tag{4.129}$$

$$\frac{\partial L(\overline{x}^*)}{\partial x_i} = \frac{\partial f(\overline{x})}{\partial x_i} + \sum_{j=1}^{m} \lambda^*{}_j \frac{\partial g_j(\overline{x}^*)}{\partial x_i} = 0, \quad i = \overline{1,n}; \qquad (4.130)$$

$$\lambda^*{}_j g_j(\overline{x}^*) = 0, \quad j = \overline{1,m}; \qquad (4.131)$$

$$g_j(\overline{x}^*) \le 0, \quad j = \overline{1,m}, \qquad (4.132)$$

которые называются условиями Куна – Такера [1].

Эти условия являются также и достаточными условиями минимума в задаче (4.118) – (4.119).

Запишем функцию Лагранжа непосредственно для нашей задачи:

$$L(x,\lambda) = -\sum_{i=1}^{n} c_i x_i + \lambda_1 \left(\sum_{i=1}^{n} V_i x_i - V_{port} \right) +$$

$$+ \lambda_2 \left(\sqrt{\sum_{i=1}^{n} d_{ii} x_i^2 + 2\sum_{i=1}^{n} \sum_{j=i}^{n} d_{ij} x_i x_j} - VaR_{zad} \right). \qquad (4.133)$$

Запишем условия Куна – Такера:

$$\begin{cases} g_i(\overline{x}) \le 0, \quad \lambda_i \ge 0, \quad i = 1,2; \\ \lambda_1 \left(\sum_{i=1}^{n} V_i x_i - V_{port} \right) = 0; \\ \lambda_2 \left(\sqrt{\sum_{i=1}^{n} d_{ii} x_i^2 + 2\sum_{i=1}^{n} \sum_{j=i}^{n} d_{ij} x_i x_j} - VaR_{zad} \right) = 0; \\ c_k + \lambda_1 V_k + \frac{1}{2} \lambda_2 \left(\dfrac{2 d_{kk} x_k + \sum_{\substack{i=1 \\ i \ne k}}^{n} d_{ki} x_i}{\sqrt{\sum_{i=1}^{n} d_{ii} x_i^2 + 2\sum_{i=1}^{n} \sum_{j=i}^{n} d_{ij} x_i x_j}} \right) = 0; \ k = 1,\ldots,n. \end{cases} \qquad (4.134)$$

Получили систему нелинейных уравнений. В отличие от систем линейных уравнений не существует решения нелинейных систем общего вида для прямых методов. Лишь в отдельных случаях системы подобного вида удается разрешить непосредственно в явном виде.

Для решения систем нелинейных уравнений обычно используют итерационные методы. Наиболее простым с точки зрения программной реализации и, кроме того, обладающим оптимальной скоростью сходимости, является метод решения систем нелинейных уравнений Ньютона.

В основе метода Ньютона лежат разложения функций в ряд Тейлора, причем члены, содержащие вторые (и более высоких порядков) производные, отбрасываются.

Пусть приближенные значения неизвестных системы (4.134) (например, полученные на предыдущей итерации) равны соответственно $a_1, ..., a_n, b_1, b_2$. Задача состоит в нахождении приращений (поправок) к этим значениям $\Delta x_1, ..., \Delta x_n, \Delta\lambda_1, \Delta\lambda_2$, благодаря которым решение системы (4.134) запишется в виде:

$$x_1 = a_1 + \Delta x_1, ..., x_n = a_n + \Delta x_n, \lambda_1 = b_1 + \Delta\lambda_1, \lambda_2 = b_2 + \Delta\lambda_2 . \quad (4.135)$$

Полученная в результате система линейных уравнений решается методом Гаусса относительно поправок $\Delta x_1, ..., \Delta x_n, \Delta\lambda_1, \Delta\lambda_2$.

Алгоритм решения задачи формирования оптимального портфеля максимальной доходности, с заданной величиной риска *VaR* может быть представлен следующей последовательностью действий.

1. Задаем начальные приближения a_i, b (i =1, …, n; j = 1, 2) для системы линейных уравнений (4.134).

2. Задаем ε – точность, с которой необходимо получить решение задачи.

3. Решаем систему линейных уравнений (4.134) методом Гаусса относительно приращений Δx_i $(i = \underline{1, n})$ и $\Delta\lambda_j$ $(j = 1, 2)$.

4. После нахождения решения задаем начальные приближения равными следующим величинам: $a_{ti} = a_{t-1,i} + \Delta x_i, b_{t,j} = b_{t-1,i} + \Delta\lambda_j$, здесь t – номер итерации.

5. Проверяем выполнение неравенства $\Delta x_i \le \varepsilon$ и $\Delta\lambda_j \le \varepsilon$, если эти условия выполняются, то переходим к шагу 6. Если нет, то переходи к шагу 1, где начальные приближения задаем равными величинам, полученным в п. 4.

6. Принимаем $x_i^* = a_i$, $i = \overline{1,n}$, и выходим из цикла решения задачи.

4.3.4. Пример задачи формирования оптимального портфеля

В качестве тестового примера была взята ситуация из реальной жизни одного из участников фондового рынка России в 2003 – 2004 гг. (табл. 4.2 – 4.6).

Объем денежных средств, которые трейдер в состоянии инвестировать в рынок ценных бумаг, равен 60 000 000 руб. Временной горизонт, на который делается расчет инвестиционного портфеля, равен одной неделе или семи дням. Доверительный интервал равен 95 %. Величина допустимого риска (*VaR*) равна 1 800 000 руб., т.е. вероятность того, что наши убытки не превысят 1 800 000 руб. в течение недели равна 5 %.

С использованием истории цен на акции с начала 2002 г. по конец первого квартала 2004 г. были рассчитаны доходность (см. формулу (4.97)) и индивидуальный риск (см. формулу (4.104)) акций каждого эмитента.

При составлении выражения для риска диверсифицированного портфеля была рассчитана ковариационная матрица доходностей акций (см. формулы (4.107), (4.108)), на диагонали которой стоят дисперсии доходностей бумаг.

Ниже представлены результаты расчета, по приведенной выше методике, банковского портфеля за различные промежутки времени. В таблицах указана доля акций каждого эмитента в портфеле $W_i = \dfrac{V_i x_i}{V_{port}}$, доли доходности, которые вносит каждая из бумаг в портфель и общая доходность портфеля, вычисленная по формуле (4.100).

Таблица 4.2

Оптимальный портфель максимальной доходности,
рассчитанный по историческим данным в период с 1 по 31 марта 2003 г., %

Акция	GSPBEX	LKOH	YUKO	SNGS	SNGSP	SIBN	TATN	TATNP	EESP	EESRP	MSNG	IRGZ	RTKM	RTKMP	AVAZ	GMKN	SBER	AFLT
Доля акций, данного эмитента в портфеле	0.00	7.76	0.96	8.26	10.22	2.35	5.07	3.68	3.44	5.41	5.04	9.71	6.11	7.19	2.07	9.89	2.41	10.44
Доля доходности, которую акция вносит в доходность портфеля	0.00	0.28	0.13	-0.04	0.21	0.52	0.63	0.23	-0.01	0.11	0.09	0.34	0.35	0.96	0.04	0.93	0.68	0.01
Доходность портфеля	5.44																	

Таблица 4.3

Оптимальный портфель максимальной доходности,
рассчитанный по историческим данным в период с 1 по 30 июня 2003 г., %

Акция	GSPBEX	LKOH	YUKO	SNGS	SNGSP	SIBN	TATN	TATNP	EESP	EESRP	MSNG	IRGZ	RTKM	RTKMP	AVAZ	GMKN	SBER	AFLT
Доля акций, данного эмитента в портфеле	12.90	6.28	1.23	5.98	8.18	5.82	4.98	3.85	0.00	2.25	0.88	7.89	5.69	6.82	3.73	9.56	4.99	8.97
Доля доходности, которую акция вносит в доходность портфеля	0.18	0.05	0.07	0.06	0.19	0.46	0.37	0.01	0.00	0.31	0.16	0.37	0.29	0.83	-0.35	0.56	0.56	0.10
Доходность портфеля	4.22																	

Таблица 4.4

Оптимальный портфель максимальной доходности,
рассчитанный по историческим данным в период с 1 по 30 сентября 2003 г., %

Акция	GSPBEX	LKOH	YUKO	SNGS	SNGSP	SIBN	TATN	TATNP	EESP	EESRP	MSNG	IRGZ	RTKM	RTKMP	AVAZ	GMKN	SBER	AFLT
Доля акций, данного эмитента в портфеле	0.00	3.83	0.51	2.44	6.51	3.38	6.73	5.57	3.18	8.28	3.86	9.44	5.51	8.25	5.23	12.28	4.23	10.78
Доля доходности, которую акция вносит в доходность портфеля	0.00	0.17	0.05	0.10	0.22	0.23	0.70	0.26	1.19	3.19	1.02	1.02	0.61	1.33	-0.18	2.38	0.57	1.00
Доходность портфеля	13.86																	

Таблица 4.5

Оптимальный портфель максимальной доходности,
рассчитанный по историческим данным в период с 1 по 31 декабря 2003 г., %

Акция	GSPBEX	LKOH	YUKO	SNGS	SNGSP	SIBN	TATN	TATNP	EESP	EESRP	MSNG	IRGZ	RTKM	RTKMP	AVAZ	GMKN	SBER	AFLT
Доля акций, данного эмитента в портфеле	17.51	4.65	0.00	5.86	5.95	2.34	3.62	3.99	3.94	5.35	4.26	5.87	6.10	6.10	3.71	8.86	3.67	8.20
Доля доходности, которую акция вносит в доходность портфеля	2.33	0.35	0.00	0.76	0.66	0.16	0.29	0.23	1.14	1.64	0.99	0.74	0.81	0.99	0.01	2.79	0.30	1.38
Доходность портфеля	15.57																	

Таблица 4.6

Оптимальный портфель максимальной доходности,
рассчитанный по историческим данным в период с 1 марта 2003 г. по 31 марта 2004 г., %

Акция	GSPBEX	LKOH	YUKO	SNGS	SNGSP	SIBN	TATN	TATNP	EESP	EESRP	MSNG	IRGZ	RTKM	RTKMP	AVAZ	GMKN	SBER	AFLT
Доля акций, данного эмитента в портфеле	22.38	5.34	0.00	4.77	5.60	2.95	4.53	5.08	0.17	2.60	2.31	7.21	6.58	6.41	5.90	5.41	5.09	7.68
Доля доходности, которую акция вносит в доходность портфеля	4.45	0.75	0.00	0.84	0.85	0.29	0.46	0.40	0.05	0.71	0.45	1.06	0.90	0.83	0.30	1.62	0.57	1.85
Доходность портфеля	16.38																	

4.4. Использование методов нелинейного программирования при оценке параметров формирующего фильтра

Важные прикладные аспекты использования методов нелинейного программирования связанны с задачами идентификации параметров систем различной природы и назначения. Одной из таких задач идентификации может служить задача определения параметров формирующего фильтра, выходом которого является случайный процесс, корреляционная функция которого близка корреляционной функции моделируемого случайного процесса. Рассмотрим, прежде всего, основные свойства формирующих фильтров.

4.4.1. Основные свойства формирующих фильтров

Как известно [56], формирующим фильтром называется линейная динамическая система (в общем случае нестационарная), на вход которой подаётся нормально распределенный «белый шум» единичной интенсивности, а выходом является случайный процесс, корреляционная функция которого $K_{\text{y}}(t_1, t_2)$.

В общем случае уравнение формирующего фильтра имеет вид

$$a_n(t)\frac{d^n y(t)}{dt^n} + a_{n-1}(t)\frac{d^{n-1} y(t)}{dt^{n-1}} + \ldots + a_0(t)y(t) =$$

$$= b_m(t)\frac{d^m x(t)}{dt^m} + b_{m-1}(t)\frac{d^{m-1} x(t)}{dt^{m-1}} + \ldots + b_0(t)x(t), \qquad (4.136)$$

где $a_0(t), \ldots, a_n(t), b_0(t), \ldots, b_m(t)$ – коэффициенты формирующего фильтра ($m \le n$); $y(t)$ – случайный процесс на выходе фильтра; $x(t)$ – нормально распределенный «белый шум» интенсивности N_x.

Уравнение (4.136) можно записать в операторной форме:

$$D_n(S,t)y(t) = M_m(S,t)x(t), \qquad (4.137)$$

где $D_n(S,t)$ и $M_m(S,t)$ – линейные операторы:

$$D_n(S,t) = a_n(t)S^n + \ldots + a_0(t); \tag{4.138}$$

$$M_m(S,t) = b_m(t)S^m + \ldots + b_0(t). \tag{4.139}$$

Как известно, корреляционная функция $K_{\text{y}}(t_1,t_2)$ произвольного случайного процесса $y(t)$ определяется следующим образом [10]:

$$K_{\text{y}}(t_1,t_2) = M[(y(t_1) - m_{\text{y}}(t_1))(y(t_2) - m_{\text{y}}(t_2))], \tag{4.140}$$

где $m_{\text{y}}(t_i)$ – математическое ожидание сечения случайной функции $y(t)$ в момент t_i.

Известно, что корреляционная функция обладает следующими свойствами.

1. $K_{\text{y}}(t_1,t_2) = K_{\text{y}}(t_2,t_1)$ – свойство симметрии.
2. $K_{\text{y}}^2(t_1,t_2) \geq K_{\text{y}}(t_1,t_1)K_{\text{y}}(t_2,t_2)$. (4.141)
3. $K_{\text{y}}(t_1,t_2)$ – положительно определенная функция.

Предположим, что импульсная переходная функция динамической системы (4.136), представляющая собой реакцию этой системы на бесконечно большой импульс, поданный в момент времени τ, известна:

$$\begin{gathered} k(t,\tau) = \frac{M_m(S,t)}{D_n(S,t)}\delta(t-\tau), \quad t \geq \tau; \\ k(t,\tau) = 0, \quad t < \tau. \end{gathered} \tag{4.142}$$

Тогда, для любого момента времени t справедливо выражение:

$$y(t) = \int_{-\infty}^{t} x(\tau)k(t,\tau)d\tau. \tag{4.143}$$

Подставляя (4.143) в формулу (4.140), получим:

$$K_{\text{y}}(t_1,t_1) = M[y(t_1)y(t_2)] =$$

$$= \int_{-\infty}^{t_1} k(t_1, \tau_1) \left[\int_{-\infty}^{t_2} k(t_2, \tau_2) K_x(\tau_1, \tau_2) d\tau_2 \right] d\tau_1, \qquad (4.144)$$

где $K_x(t_1, t_2)$ – корреляционная функция случайного процесса $x(t)$.

При $t_1 = t_2 = t$ получим дисперсию процесса $y(t)$ в момент времени t:

$$\sigma_y^2(t) = \int_{-\infty}^{t} k(t, \tau_1) \int_{-\infty}^{t} k(t, \tau_2) K_x(\tau_1, \tau_2)\, d\tau_2 d\tau_1 . \qquad (4.145)$$

Найдем корреляционную функцию сигнала $y(t)$ на выходе динамической системы (4.136), на вход которой подается «белый шум» $x(t)$. Так как спектральная плотность «белого шума» постоянна и равна интенсивности, т.е.

$$S_x(\omega) = S_x(0) = N_x, \quad -\infty < \omega < \infty , \qquad (4.146)$$

то корреляционную функцию случайного процесса $x(t)$ можно записать в виде

$$K_x(\tau) = \frac{N_x}{2\pi} \int_{-\infty}^{\infty} e^{j\omega\tau} d\omega = N_x \delta(\tau) . \qquad (4.147)$$

Таким образом, корреляционная функция стационарного «белого шума» для интервала времени $\tau = \tau_1 - \tau_2$ будет иметь вид

$$K_x(\tau_1 - \tau_2) = N_x \delta(\tau_1 - \tau_2) . \qquad (4.148)$$

Подставляя (4.148) при $N_x = 1$ в формулу (4.144), получим:

$$K_y(t_1, t_2) = \int_{-\infty}^{t_1} k(t_1, \tau_1) k(t_2, \tau_1) d\tau_1 , \quad t_1 \le t_2 ; \qquad (4.149а)$$

$$K_y(t_1, t_2) = \int_{-\infty}^{t_2} k(t_1, \tau_2) k(t_2, \tau_2) d\tau_2 , \quad t_1 \ge t_2 . \qquad (4.149б)$$

Полагая в (4.149а) и (4.149б) $t_1 = t_2 = t$, найдем дисперсию выходного сигнала:

$$\sigma_{\text{y}}^2 = \int_{-\infty}^{t} k^2(t,\tau)d\tau\,. \qquad (4.150)$$

Для стационарных линейных динамических систем импульсная передаточная функция $k(t,\tau)$ зависит только от промежутка времени, истекшего с момента подачи возмущающего импульса, т.е.

$$k(t,\tau) = k(t-\tau)\,. \qquad (4.151)$$

Применим к обеим частям равенства (4.149б) линейный оператор $D_n(S,t)$, определяемый выражением (4.138). Тогда, с учетом выражения (4.142), получим:

$$D_n(S,t)K_{\text{y}}(t_1,t_2) = \int_{-\infty}^{t_2} k(t_2,\tau)M(S,t_1)\delta(t_1-\tau)d\tau; \quad t_2 < t_1\,. \quad (4.152)$$

Так как δ-функция в правой части (4.152) не равна нулю только при $\tau = t_1$, а верхний предел $t_2 < t_1$, то справедливо выражение:

$$D_n(S,t)K_{\text{y}}(t_1,t_2) = 0, \quad t_2 < t_1, \qquad (4.153)$$

или, подставляя значение линейного оператора (4.138), получим линейное однородное уравнение, которому удовлетворяет корреляционная функция $K_{\text{y}}(t_1,t_2)$ при $t_2 < t_1$:

$$a_n(t_1)\frac{d^n}{dt_1^n}K_{\text{y}}(t_1,t_2) + \ldots + a_0(t_1)K_{\text{y}}(t_1,t_2) = 0; \quad t_2 < t_1\,. \quad (4.154)$$

Рассмотрим выражение (4.149а). Очевидно, это выражение можно трактовать как уравнение свертки [56] линейной динамической системы с импульсной переходной функцией $k(t_1,\tau)$, на вход которой подается сигнал $k(t_2,\tau)$, а выходом является корреляционная функция $K_{\text{y}}(t_1,\tau)$. Таким образом, можно записать:

$$a_n(t_1)\frac{d^n}{dt_1^n}K_{\text{y}}(t_1,t_2) + \ldots + a_0(t_1)K_{\text{y}}(t_1,t_2) =$$

$$= b_m(t_1)\frac{d^m}{dt_1^m}k(t_1,t_2) + \ldots + b_0(t_1)k(t_1,t_2)\,, \quad t_2 \ge t_1\,. \quad (4.155)$$

Рассуждая аналогичным образом, получим еще два уравнения:

$$a_n(t_2)\frac{d^n}{dt_2^n}K_{\text{у}}(t_1,t_2)+...+a_0(t_2)K_{\text{у}}(t_1,t_2)=0\,,\quad t_1<t_2\,. \quad (4.156)$$

$$a_n(t_2)\frac{d^n}{dt_2^n}K_{\text{у}}(t_1,t_2)+...+a_0(t_2)K_{\text{у}}(t_1,t_2)=$$

$$=b_m(t_2)\frac{d^m}{dt_2^m}k(t_1,t_2)+...+b_0(t_2)k(t_1,t_2)\,,\quad t_1\ge t_2\,. \quad (4.157)$$

Известно [57], что импульсная переходная функция $k(t,\tau)$ линейной стационарной динамической системы при $M(S,\tau)=1$ совпадает с функцией Грина и равна:

$$k(t,\tau)=G(t,\tau)\,, \quad (4.158)$$

где

$$G(t,\tau)=\frac{(-1)^{n-1}}{\Delta(\tau)}\det\begin{pmatrix}\Phi_1(t) & \Phi_2(t) & ... & \Phi_n(t)\\ \Phi_1(\tau) & \Phi_2(\tau) & ... & \Phi_n(\tau)\\ \Phi_1'(\tau) & \Phi_2'(\tau) & ... & \Phi_n'(\tau)\\ \Phi_1^{(n-2)}(\tau) & \Phi_1^{(n-2)}(\tau) & ... & \Phi_1^{(n-2)}(\tau)\end{pmatrix}; \quad (4.159)$$

$\Delta(\tau)$ – определитель Вронского:

$$\Delta(\tau)=\det\begin{pmatrix}\Phi_1(\tau) & ... & \Phi_n(\tau)\\ \Phi_1'(\tau) & ... & \Phi_n'(\tau)\\ ... & ... & ...\\ \Phi_1^{(n-1)}(\tau) & ... & \Phi_1^{(n-1)}(\tau)\end{pmatrix}, \quad (4.160)$$

$\Phi_1(\tau),...,\Phi_2(\tau)$ – фундаментальное решение однородного дифференциального уравнения:

$$a_n(t)\frac{d^n y(t)}{dt^n}+a_{n-1}(t)\frac{d^{n-1}y(t)}{dt^{n-1}}+...+a_0(t)y(t)=0\,. \quad (4.161)$$

4.4.2. Корреляционная функция формирующего фильтра второго порядка

Используя свойства корреляционной функции формирующего фильтра, полученные в предыдущем разделе, найдем в явном виде выражение для корреляционной функции случайного процесса на выходе стационарного формирующего фильтра второго порядка при $M(S,\tau)=1$.

В этом случае уравнение формирующего фильтра имеет вид

$$\frac{d^2}{dt^2}y(t)+a_1\frac{d}{dt}y(t)+a_0y(t)=\xi(t). \qquad (4.162)$$

На основании свойства (4.154), корреляционная функция случайного процесса $y(t)$ на выходе формирующего фильтра (4.162), удовлетворяет уравнению:

$$\frac{d^2}{dt_1^{\,2}}K_{\text{y}}(t_1,t_2)+a_1\frac{d}{dt_1}K_{\text{y}}(t_1,t_2)+a_0K_{\text{y}}(t_1,t_2)=0; \quad t_1>t_2.$$

Пусть λ_1,λ_2 – корни характеристического уравнения

$$\lambda^2+a_1\lambda+a_0=0; \quad \lambda_{1,2}=\frac{1}{2}\left(-a_1\pm\sqrt{a_1^2-4a_0}\right). \qquad (4.163)$$

Тогда фундаментальное решение будет иметь вид:

1) действительные корни характеристического уравнения

$$\Phi_1(t)=e^{\lambda_1 t}; \quad \Phi_2(t)=e^{\lambda_2 t}; \qquad (4.164)$$

2) кратные корни характеристического уравнения

$$\Phi_1(t)=e^{\lambda t}; \quad \Phi_2(t)=te^{\lambda t}; \qquad (4.165)$$

3) комплексные корни характеристического уравнения

$$\Phi_{1,2}(t)=e^{\alpha_1 t}(\cos j\beta\pm\sin j\beta). \qquad (4.166)$$

Получим значение корреляционной функции для каждого вида корней характеристического уравнения.

1. Случай действительных корней. В этом случае, согласно соотношениям (4.159) и (4.160), определитель Вронского и импульсная переходная функция будут иметь вид

$$\Delta(\tau)=\begin{vmatrix} e^{\lambda_1\tau} & e^{\lambda_2\tau} \\ \lambda_1 e^{\lambda_2\tau} & \lambda_2 e^{\lambda_1\tau} \end{vmatrix}=e^{(\lambda_1+\lambda_2)\tau}(\lambda_2-\lambda_1)\,; \qquad (4.167)$$

$$G(t,\tau)=\frac{-1}{e^{(\lambda_1+\lambda_2)\tau}(\lambda_2-\lambda_1)}\begin{vmatrix} e^{\lambda_1 t} & e^{\lambda_2 t} \\ e^{\lambda_1\tau} & e^{\lambda_2\tau} \end{vmatrix}=\frac{e^{\lambda_1(t-\tau)}-e^{\lambda_2(t-\tau)}}{\lambda_1-\lambda_2}=k(t,\tau)\,. \qquad (4.168)$$

Подставляя в формулу (4.149а) выражение для импульсной переходной функции (4.168) и учитывая, что для стационарного случайного процесса корреляционная функция зависит только от разности моментов времени Δt, получим:

$$K_{\text{у}}(\Delta t)=\frac{[\lambda_2\exp(\lambda_1\Delta t)-\lambda_1\exp(\lambda_2\Delta t)]}{2a_1a_0(\lambda_2-\lambda_1)}. \qquad (4.169)$$

2. Случай кратных корней. Для случая кратных корней определитель Вронского и импульсная переходная функция будут иметь вид

$$\Delta(\tau)=\begin{vmatrix} e^{\lambda\tau} & \tau e^{\lambda\tau} \\ \lambda e^{\lambda\tau} & e^{\lambda\tau}(1+\lambda\tau) \end{vmatrix}=e^{2\lambda\tau}\,; \qquad (4.170)$$

$$G(t,\tau)=\frac{-1}{e^{2\lambda\tau}}\begin{vmatrix} e^{\lambda t} & te^{\lambda t} \\ e^{\lambda\tau} & \tau e^{\lambda\tau} \end{vmatrix}=\frac{-e^{\lambda(t+\tau)}(t-\tau)}{e^{2\lambda\tau}}=(t-\tau)e^{\lambda(t-\tau)}=k(t,\tau)\,. \qquad (4.171)$$

Соответствующая корреляционная функция будет равна:

$$K_{\text{у}}(\Delta t)=\exp(\lambda\Delta t)\left(\frac{1}{2a_1a_0}-\frac{\lambda\Delta t}{2a_1a_0}\right). \qquad (4.172)$$

3. Случай комплексно-сопряженных корней. Определитель Вронского, импульсная переходная функция и корреляционная функция, соответственно, будут иметь вид

$$\Delta(\tau)=\begin{vmatrix} a^{\alpha\tau}(\cos\beta\tau+j\sin\beta\tau) & a^{\alpha\tau}(\cos\beta\tau-j\sin\beta\tau) \\ a^{\alpha\tau}\beta(\alpha+j)(\cos\beta\tau+j\sin\beta\tau) & a^{\alpha\tau}\beta(\alpha-j)(\cos\beta\tau-j\sin\beta\tau) \end{vmatrix}=$$

$$=-2j\beta e^{2\alpha\tau}; \qquad (4.173)$$

$$G(t,\tau)=\frac{-1}{-2j\beta e^{2\alpha\tau}}\begin{vmatrix} e^{\alpha\tau}(\cos\beta t+j\sin\beta t) & e^{\alpha\tau}(\cos\beta t-j\sin\beta t) \\ e^{\alpha\tau}(\cos\beta\tau+j\sin\beta\tau) & e^{\alpha\tau}(\cos\beta\tau-j\sin\beta\tau) \end{vmatrix}=$$

$$=\frac{e^{\alpha(t-\tau)}\sin(\beta(t-\tau))}{\beta}; \qquad (4.174)$$

$$K_{\text{y}}(\Delta t)=e^{\alpha(\Delta t)}\left[\frac{1}{2a_0a_1}\cos(\beta\Delta t)-\frac{\alpha}{\beta}\frac{1}{2a_0a_1}\sin(\beta\Delta t)\right]. \qquad (4.175)$$

Нетрудно показать, что, вне зависимости от вида корней характеристического уравнения, дисперсия стационарного случайного процесса $y(t)$ на выходе формирующего фильтра будет одинаковой:

$$K_{\text{y}}(0)=\sigma_{\text{y}}^2=\frac{1}{2a_1a_0}. \qquad (4.176)$$

4.4.3. Задача идентификации коэффициентов формирующего фильтра как задача нелинейного программирования

Как правило, для моделирования случайных процессов используются стационарные формирующие фильтры вида (4.162). При этом возникает задача идентификации коэффициентов формирующего фильтра, на выходе которого наблюдается стационарный случайный процесс $y(t)$, корреляционная функция $K_{\text{y}}(\Delta t)$ которого близка корреляционной функции $K(\Delta t)$ моделируемого случайного процесса.

В качестве меры близости корреляционных функций истинного и модельного случайных процессов выбран квадратичный критерий вида

$$J(a_0, a_1) = \int_0^{\infty} (K(\tau) - K_{\text{у}}(a_0, a_1, \tau))^2 \, d\tau . \qquad (4.177)$$

На практике корреляционная функция истинного процесса $K(\Delta t)$ задается в виде таблиц или графиков. В этом случае интегральный критерий (4.177) удобно заменить суммой квадратов невязок между корреляционными функциями истинного и модельного процессов:

$$J(a_0, a_1) = \sum_{i=1}^{N} (K(i) - K_{\text{у}}(a_0, a_1, i))^2 . \qquad (4.178)$$

Очевидно, вид критерия (4.178) зависит от типа корней характеристического уравнения λ_1 и λ_2, определяемых по формуле (4.163):

1) действительные корни

$$J(a_0, a_1) = \sum_{i=0}^{N} \left(K_{\text{зад}}\left(i\frac{T}{N}\right) - \frac{\left[\lambda_2 \exp\left(\lambda_1 i \frac{T}{N}\right) - \lambda_1 \exp\left(\lambda_2 i \frac{T}{N}\right)\right]}{2a_1 a_0 (\lambda_2 - \lambda_1)} \right)^2 ; \qquad (4.179)$$

2) кратные корни

$$J(a_0, a_1) = \sum_{i=0}^{N} \left(K_{\text{зад}}\left(i\frac{T}{N}\right) - \exp\left(\lambda i \frac{T}{N}\right) \left(\frac{1}{2a_1 a_0} - \frac{\lambda i \frac{T}{N}}{2a_1 a_0} \right) \right)^2 ; \qquad (4.180)$$

3) комплексно-сопряженные корни

$$J(a_0, a_1) = \sum_{i=0}^{N} \left(K_{\text{зад}}\left(i\frac{T}{N}\right) - e^{\alpha(\Delta t i)} \left[\frac{1}{2a_0 a_1} \cos\left(\beta i \frac{T}{N}\right) - \right.\right.$$

$$\left.\left. - \frac{\alpha}{\beta} \frac{1}{2a_0 a_1} \sin\left(\beta i \frac{T}{N}\right) \right] \right)^2 , \qquad (4.181)$$

где N, T – общее число точек и полный интервал времени вычисления корреляционной функции; α, β – действительная и мнимая части корней характеристического уравнения.

Очевидно, задача минимизации критерия (4.178) является задачей нелинейного программирования без ограничений. Методы решения таких задач подробно описаны выше.

В конкретном случае данная задача нелинейного программирования решается двумя методами:

методом Ньютона – Гаусса;

методом покоординатного спуска с одномерной минимизацией по каждой координате.

Ниже приведены алгоритмы решения задачи этими методами.

4.4.3.1. Алгоритм решения задачи методом Ньютона – Гаусса

Метод Ньютона – Гаусса предназначен для решения задач нелинейного программирования с квадратичным критерием, вида (4.178) [63], [4].

В основе данного метода лежит линеаризация нелинейной функции $K_{\text{у}}(a_0, a_1, i)$ относительно оценок параметров на предыдущем шаге итерационного процесса. Основная итерационная формула имеет вид

$$\hat{\bar{c}}(j) = \hat{\bar{c}}(j-1) + (\Phi^{\text{т}}(j)\Phi(j))^{-1}\Phi^{\text{т}}(j)\bar{e}(j), \qquad (4.182)$$

где $\hat{\bar{c}}(j) = \begin{bmatrix} a_0(j) \\ a_1(j) \end{bmatrix}$ – вектор оцениваемых коэффициентов на j-м шаге итерационного процесса; $\Phi(j)$ – матрица производных, вычисленная в точке оценки на предыдущем шаге итерационного процесса:

$$\Phi(j)=\begin{bmatrix} \dfrac{\partial K_{\text{y}}(a_0,a_1,1)}{\partial a_0} & \dfrac{\partial K_{\text{y}}(a_0,a_1,1)}{\partial a_1} \\ \cdots & \cdots \\ \cdots & \cdots \\ \dfrac{\partial K_{\text{y}}(a_0,a_1,N)}{\partial a_0} & \dfrac{\partial K_{\text{y}}(a_0,a_1,N)}{\partial a_1} \end{bmatrix}_{\substack{a_0=a_0(j-1),\\ a_1=a_1(j-1)}} ; \quad (4.183)$$

$\overline{e}(j)$ – вектор невязок между истинным значением корреляционной функции $K(i)$ и корреляционной функцией $K_{\text{y}}(a_0(j-1),,$ $a_1(j-1),i)$, соответствующей случайному процессу на выходе формирующего фильтра на $(j-1)$-м шаге итерационного процесса:

$$\overline{e}(j)=\begin{bmatrix} K(1)-K_{\text{y}}(a_0(j-1),a_1(j-1),1) \\ \vdots \\ K(1)-K_{\text{y}}(a_0(j-1),a_1(j-1),1) \end{bmatrix}. \quad (4.184)$$

На рис. 4.11 приведена структурная схема решения задачи методом Ньютона – Гаусса.

Забегая вперед, можно отметить, что использование метода Ньютона – Гаусса обеспечивает очень быструю сходимость критерия к нулю (за две – три итерации) при задании начальных приближений вблизи точки оптимума. В противном случае, может наблюдаться расходимость итерационного процесса.

Учитывая это, можно предложить следующий способ задания начальных приближений.

Воспользуемся формулой расчета дисперсии случайного процесса на выходе формирующего фильтра (4.176). Допуская, что дисперсия модельного и истинного процесса совпадают, выразим начальное приближение коэффициента $a_0(0)$ как функцию начального приближения коэффициента $a_1(0)$ и значения дисперсии исходного процесса.

$$a_0(0)=\frac{1}{2a_1(0)\sigma^2}, \quad (4.185)$$

где σ^2 – значение дисперсии исходного процесса.

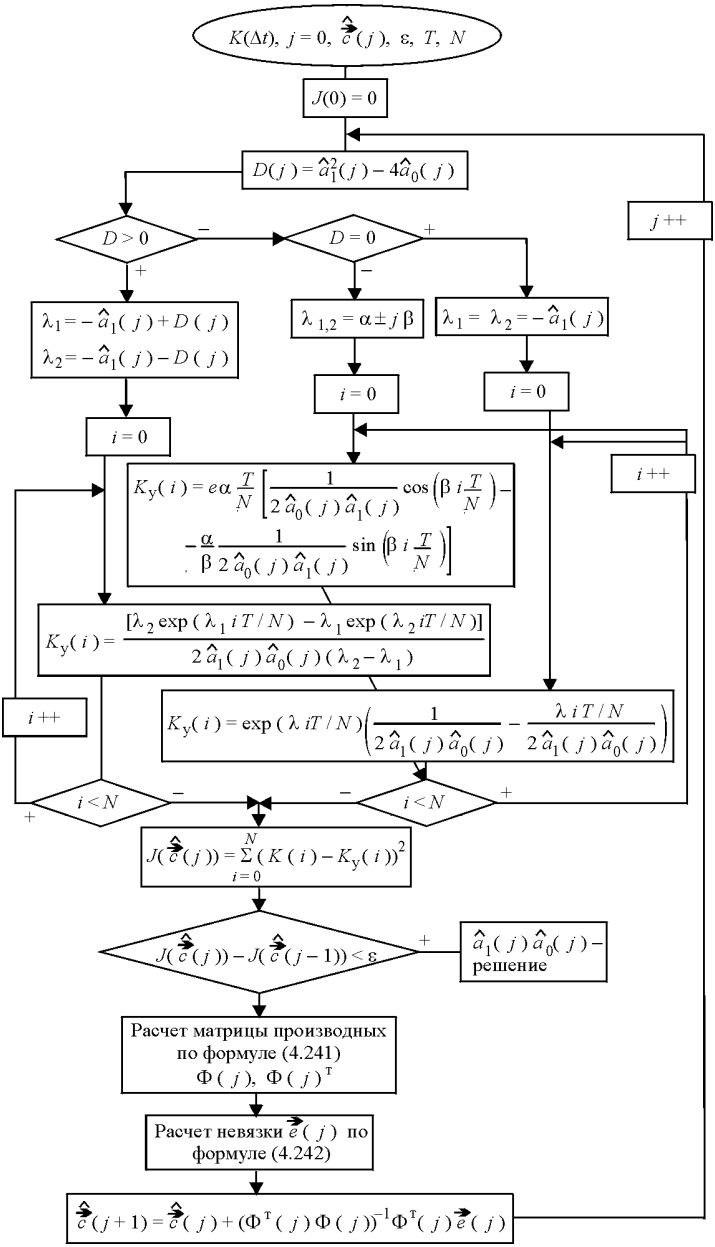


Рис. 4.11. Алгоритм решения задачи, метод Ньютона – Гаусса

Пусть значения корреляционной функции исходного процесса и корреляционной функции модельного процесса также совпадают на конце интервала расчета t_f корреляционных функций, т.е.

$$K(t_f) = K_{\text{у}}(t_f). \tag{4.186}$$

Здесь $K(t_f)$, $K_{\text{у}}(t_f)$ – заданная и модельная корреляционные функции, соответственно.

Учитывая (4.185), запишем значение корней характеристического уравнения (4.163), как функции одного параметра $a_1(0)$:

1) действительные корни характеристического уравнения:

$$\lambda_{1,2}(a_1) = \frac{-a_1 \pm \sqrt{a_1^2 - \dfrac{2}{a_1 K(0)}}}{2}, \quad D > 0; \tag{4.187}$$

2) кратные корни характеристического уравнения:

$$\lambda_1(a_1) = \lambda_2 = \frac{-a_1}{2}, \quad D = 0; \tag{4.188}$$

3) комплексно-сопряженные корни характеристического уравнения:

$$\lambda_{1,2}(a_1) = \frac{-a_1 \pm \sqrt{a_1^2 - \dfrac{2}{a_1 K(0)}}}{2}, \quad D < 0;$$

$$\alpha(a_1) = -\frac{a_1}{2}; \quad \beta(a_1) = \frac{2}{a_1 K(0)} - a_1^2. \tag{4.189}$$

Подставляя (4.187) – (4.189) в формулы расчета корреляционных функций (4.169), (4.172), (4.175) при $\Delta t = t_f$ и принимая во внимание (4.186), для различных типов корней характеристического уравнения получим:

1) для действительных корней характеристического уравнения:

$$K(t_f) - K(0)\frac{[\lambda_2(a_1)\exp(\lambda_1(a_1)t_1) - \lambda_1(a_1)\exp(\lambda_2(a_1)t_1)]}{(\lambda_2(a_1) - \lambda_1(a_1))} = 0; \tag{4.190}$$

2) для кратных корней характеристического уравнения:

$$K(t_f) - K(0)\exp(\lambda(a_1)t_f)(1 - \lambda(a_1)t_f) = 0; \tag{4.191}$$

3) для комплексно-сопряженных корней характеристического уравнения:

$$K(t_f) - K(0)e^{\alpha(a_1)t_f}\left[\cos(\beta(a_1)t_f) - \frac{\alpha(a_1)}{\beta(a_1)}\sin(\beta(a_1)t_f)\right] = 0. \tag{4.192}$$

Разрешая уравнения (4.190) – (4.192) относительно $a_1(0)$ ($a_0(0)$ определяется по формуле (4.185)) получим значения $a_1(0), a_0(0)$, которые могут быть использованы в качестве начальных приближений.

4.4.3.2. Алгоритм решения задачи методом покоординатного спуска

Как было отмечено в предыдущем разделе, метод Ньютона – Гаусса оказывается чувствителен к заданию начальных приближений. Неудачное задание начальных приближений часто приводит к расхождению итерационного процесса.

В этой связи, методом более устойчивым к выбору начальных приближений является метод покоординатного спуска, подробно рассмотренный в гл. 3.

На рис. 4.12 приведена структурная схема решения задачи определения коэффициентов формирующего фильтра с использованием этого метода.

В качестве метода одномерной оптимизации при нахождении точки экстремума по каждой координате используется метод «золотого сечения», также рассмотренный в гл. 3.

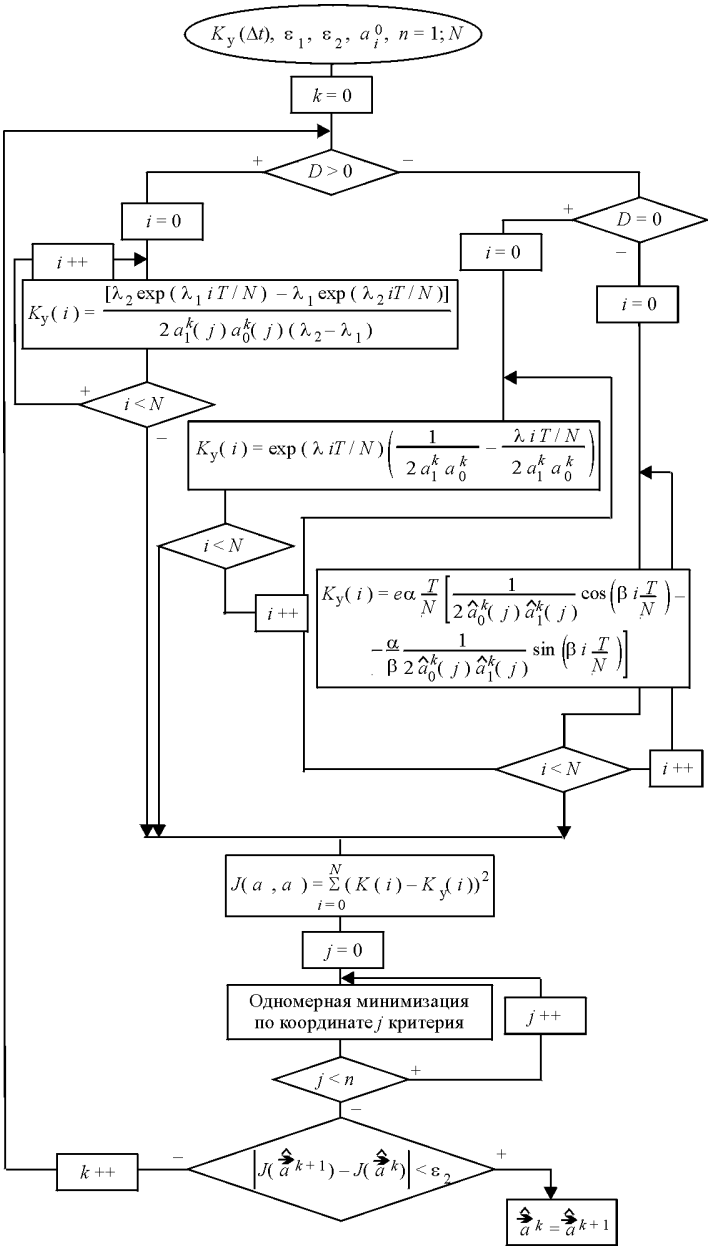


Рис. 4.12. Блок-схема алгоритма метода покоординатного спуска (один цикл)

4.4.4. Пример расчета коэффициентов формирующего фильтра

В качестве примера рассмотрим расчет коэффициентов формирующего фильтра, выходом которого является случайный процесс с корреляционной функцией близкой к корреляционной функции, изображенной на рис. 4.13.

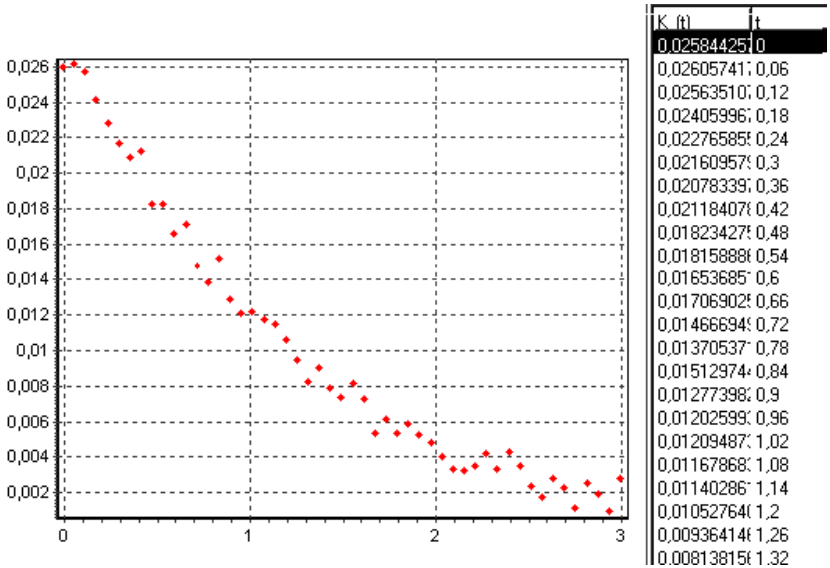


Рис. 4.13. Вид корреляционной функции исходного процесса

Данная задача решалась двумя методами:

методом Ньютона – Гаусса;

методом покоординатного спуска.

В процессе решения получены результаты, приведенные в табл. 4.7.

Как видно из результатов расчета, метод Ньютона – Гаусса обеспечивает быструю сходимость алгоритма вблизи точки оптимума (варианты расчетов 1 и 2), тогда как при больших отклонениях начальных приближений ($a_0(0), a_1(0)$) от оптимальных наблюдается расходимость метода (варианты расчетов 3 и 4).

Метод покоординатного спуска обеспечивает более устойчивую сходимость при различных начальных приближениях (варианты расчетов 5 – 8).

Таблица 4.7

Сравнительные характеристики методов оптимизации

Метод	Начальные приближения	Характеристики			
		точ-ность	итера ции	a_0	a_1
Нью-тона – Гаусса	1. $a_0(0) = 5{,}45$; $a_1(0) = 3{,}87$	$10e-6$	2	4,999	4,0005
	2. $a_0(0) = 6{,}00$; $a_1(0) = 6{,}00$	$10e-6$	5	5,0001	3,998
	3. $a_0(0) = 8{,}00$; $a_1(0) = 8{,}00$	$10e-6$	Метод расходится		
	4. $a_0(0) = 15{,}00$; $a_1(0) = 15{,}00$	$10e-6$	Метод расходится		
Поко-орди-натно-го спуска	5. $a_0(0) = 5{,}45$; $a_1(0) = 3{,}87$	$10e-6$	5	5,01	3,999
	6. $a_0(0) = 6{,}00$; $a_1(0) = 6{,}00$	$10e-6$	9	5,000	4,000
	7. $a_0(0) = 8{,}00$; $a_1(0) = 8{,}00$	$10e-6$	9	4,998	3,989
	8. $a_0(0) = 15{,}00$; $a_1(0) = 15{,}00$	$10e-6$	11	5,000	4,000

На рис. 4.14 изображены графики исходной и модельной корреляционной функций случайных процессов.

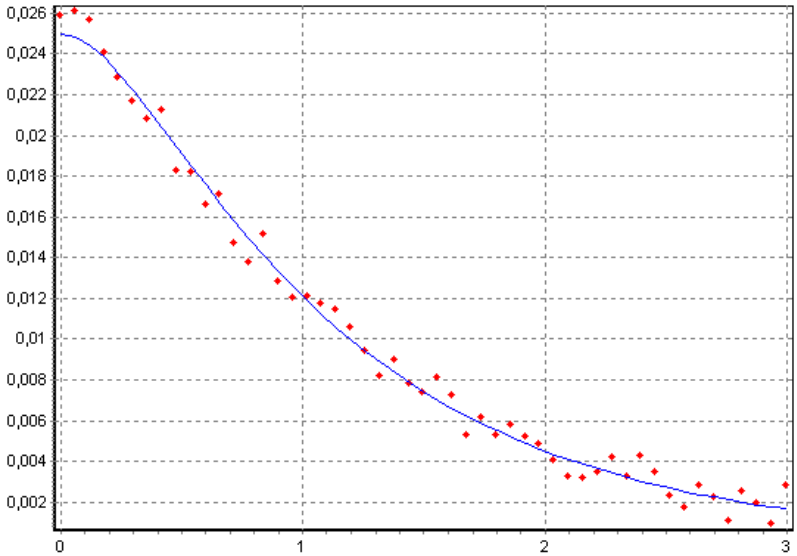


Рис. 4.14. Вид корреляционных функций исходного и моделируемого с помощью формирующего фильтра случайных процессов

4.5. Оптимизация режима работы ядерного реактора в переменном суточном графике нагрузки с учетом возможности утилизации энергии

В разд. 2.2 и 4.2 рассматривались задачи оптимизации работы системы реакторов в переменном суточном графике нагрузки. При этом АЭС с реакторами серийных типов (ВВЭР, РБМК) по ряду причин не в состоянии обеспечить переменный суточный график нагрузки в полном диапазоне без серьезного ущерба для экономических показателей и надежности энергоустановки. Допустимый диапазон суточных колебаний мощности и скорость набора нагрузки для этих реакторов значительно ниже, чем требуется по условиям работы энергосистем с разуплотненным графиком нагрузки.

В качестве одного из эффективных решений названной проблемы предполагается создание на базе АЭС энергокомплексов, включающих установки, способные воспринять и полезно использовать избыток энергии, вырабатываемой АЭС в период снижения ее потребления в энергосистеме. В качестве устройств, использующих избыточную энергию, могут рассматриваться утилизаторы, обеспечивающие производство другого ценного продукта (водорода, синтетического или жидкого топлива) или энергоснабжение потребителей низкопотенциальным теплом. Не останавливаясь детально на конкретных схемах утилизации энергии, рассмотрим возможности оптимизации работы комплекса «реактор-утилизатор» с точки зрения эффективности использования ядерного топлива [68].

Физическая предпосылка оптимизации заключается в том, что, с одной стороны, если переменный график работы энергосистемы полностью отрабатывается реактором, то это может привести к резервированию дополнительного запаса реактивности, а следовательно, и увеличенному расходу топлива. С другой стороны, если реактор работает в базовом режиме, а переменный график обеспечивается работой утилизатора, то можно ожидать, что при низкой эффективности утилизатора также будет иметь место увеличенный расход ядерного топлива. Таким образом, возникает задача об оптимальном режиме работы энергокомплекса.

4.5.1. Постановка задачи

Имеется ядерный энергоблок номинальной тепловой мощностью W_H [МВт]. Известно, что по условиям работы энергосистемы потребуется эксплуатация этого энергоблока в переменном суточном графике нагрузки. При этом задаются следующие параметры графика нагрузки:

1) время работы на пониженной мощности τ [сут];

2) уровень пониженной мощности αW_H $(0 \le \alpha \le 1)$.

Для повышения эффективности работы энергоблока предполагается возможность утилизации части энергии с коэффициентом полезного действия $\eta = Q_A / Q_B$, где Q_A – полезная энергия, отдаваемая утилизатором; Q_B – полная энергия, отпущенная на утилизацию. Каков должен быть режим работы ядерного реактора, чтобы расход топлива на единицу отпущенной потребителю энергии был минимален?

Рассмотрим следующую ситуацию. Пусть в момент времени $t = 0$ мощность реактора снижается до уровня $\varepsilon \cdot W_H$, где $\alpha \le \varepsilon \le 1$. Поскольку в систему требуется поставить энергию, соответствующую работе на более низком уровне мощности $\alpha \cdot W_H$, то излишек энергии в количестве $Q = \tau \cdot W_H \cdot (\varepsilon - \alpha)$ передается утилизатору (рис. 4.15). Из этой энергии потребителю будет отпущена часть, равная $Q_A = \eta \cdot \tau \cdot W_H \cdot (\varepsilon - \alpha)$. Таким образом, реактор за одни сутки (один цикл) вырабатывает энергию в количестве:

$$Q_P = W_H \cdot (1 - \tau) + W_H \cdot \tau \cdot \varepsilon,$$

а потребителю будет отпущена лишь часть энергии:

$$E_n = W_H \cdot (1 - \tau) + \alpha \cdot W_H \cdot \tau + (\varepsilon - \alpha) \cdot \eta \cdot \tau \cdot W_H .$$

Расход топлива при работе реактора в таком режиме – G. Эта величина определяется отношением полной энергии, произведенной реактором за сутки к глубине выгорания топлива:

$$G = \frac{W_H \cdot (1 - \tau) + W_H \cdot \varepsilon \cdot \tau}{Pt(\varepsilon)} .$$

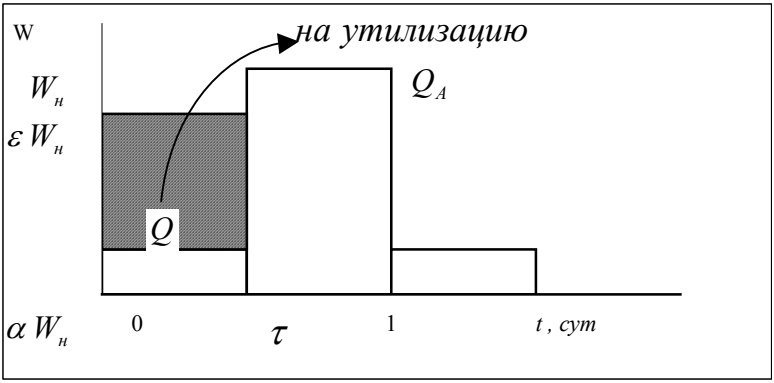


Рис. 4.15. Режим работы энергокомплекса с утилизацией энергии

Тогда расход топлива на единицу энергии, отпущенной потребителю, есть:

$$g(\varepsilon)=\frac{G}{E_n}=\frac{W_H\cdot(1-\tau)+W_H\cdot\varepsilon\cdot\tau}{Pt(\varepsilon)\cdot[(1-\tau)\cdot W_H+\alpha\cdot\tau\cdot W_H+(\varepsilon-\alpha)\cdot\eta\cdot\tau\cdot W_H]}, \quad (4.193)$$

где $Pt(\varepsilon)$ – глубина выгорания топлива при работе реактора в переменном графике нагрузки с ежесуточной разгрузкой до уровня мощности $\varepsilon\cdot W_H$ ($\varepsilon<1$).

Глубина выгорания топлива $Pt(\varepsilon)$ зависит от резервируемого запаса реактивности, дающего возможность снизить мощность реактора до величины $\varepsilon\cdot W_H$. Для реактора с непрерывной перегрузкой топлива и для корпусных реакторов в конце кампании между глубиной выгорания топлива и величиной запаса реактивности справедливо соотношение:

$$Pt(\varepsilon)=Pt_b-\frac{\Delta p(\varepsilon)}{q}, \quad (4.194)$$

где Pt_b – глубина выгорания топлива при работе реактора в базовом режиме на номинальной мощности, (МВт · сут)/тU; q – коэффициент пропорциональности, зависящий от физических свойств активной зоны реактора, тU/(МВт · сут); $\Delta p(\varepsilon)$ – запас реактивно-

сти на преодоление нестационарного ксенонового отравления при снижении мощности до уровня $\varepsilon \cdot W_H$.

К сожалению, в явном виде зависимость $\Delta p(\varepsilon)$ получить не удается. Однако можно показать, что данная зависимость с погрешностью не более 3 % аппроксимируется функцией вида

$$\Delta p(\varepsilon) = \frac{1-\varepsilon}{B \cdot \varepsilon + C} \cdot \frac{\gamma_y}{\nu_f}, \qquad (4.195)$$

где γ_y – выход йода на одно деление; ν_f – среднее число вторичных нейтронов на акт деления; $B = 0{,}52$, $C = 0{,}977$ – константы аппроксимации.

Подставляя эту зависимость в соотношения (4.193) и (4.194), получим явный вид минимизируемой функции:

$$g(\varepsilon) = \frac{1-\tau+\varepsilon\cdot\tau}{\left(Pt_b - \frac{1-\varepsilon}{B\cdot\varepsilon+C}\cdot\frac{\gamma_y}{\nu_f\cdot q}\right)\cdot(1-\tau+\alpha\cdot\tau+(\varepsilon-\alpha)\cdot\eta\cdot\tau)}. \qquad (4.196)$$

Оптимизационная задача ставится следующим образом: до какого уровня от номинала ε^* следует снижать мощность реактора, чтобы при заданных параметрах энергокомплекса (КПД утилизатора и плотности потока нейтронов в реакторе) расход топлива на единицу отпущенной энергии был минимален?

В математическом плане данная задача относится к классу задач нелинейного программирования:

найти

$$\min g(\varepsilon) \qquad (4.197)$$

при ограничении $\alpha \le \varepsilon \le 1$.

4.5.2. Анализ оптимального режима

Для получения численных результатов был рассмотрен энергоблок со следующими характеристиками (близкими к характеристикам реактора РБМК-1000):

номинальная тепловая мощность реактора $W_H = 3200$ МВт;

коэффициент $q = 1{,}02 \cdot 10^{-5}$ тU/(МВт · сут);

время разгрузки $\tau = 1/3$ сут.

Константы аппроксимации *B* и *C* однозначно определяются уровнем плотности потока нейтронов. Ниже рассматривались следующие варианты:

1) $\varphi = 5 \cdot 10^{13} \ \dfrac{\text{нейтр.}}{\text{см}^2 \cdot \text{с}}$;

2) $\varphi = 3 \cdot 10^{13} \ \dfrac{\text{нейтр.}}{\text{см}^2 \cdot \text{с}}$.

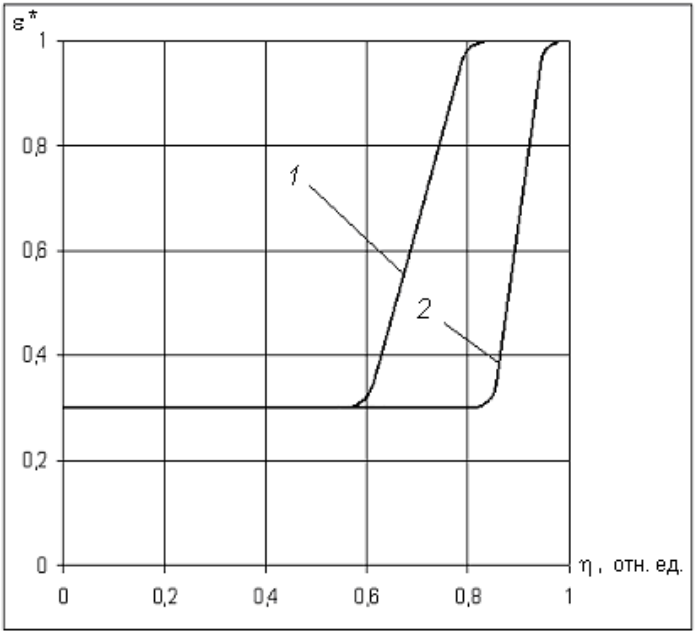


Рис. 4.16. Зависимость оптимальной степени снижения мощности реактора от КПД утилизатора при различных плотностях потока нейтронов

$\varphi = 5 \cdot 10^{13} \ \dfrac{\text{нейтр.}}{\text{см}^2 \cdot \text{с}}$ (*1*); $\varphi = 3 \cdot 10^{13} \ \dfrac{\text{нейтр.}}{\text{см}^2 \cdot \text{с}}$ (*2*)

На рис. 4.16 показана зависимость оптимальной степени снижения мощности энергоблока от КПД утилизатора при $\alpha = 0{,}3$ и со-

ответствующих φ. Из результатов, представленных на рисунке, следует, что при КПД утилизатора менее 60 % – оптимально вообще от него отказаться и отрабатывать переменный суточный график нагрузки путем соответствующего изменения мощности реактора. Напротив, если предполагается использовать утилизатор с КПД более 80 %, то оптимальным режимом является базисный режим работы реактора с передачей всего излишка вырабатываемой энергии на утилизатор. Если КПД утилизации находится в пределах 60 – 80 %, то оптимальным является режим с частичной разгрузкой энергоблока и частичной утилизацией энергии.

Ход кривой $\varepsilon^*(\eta)$ имеет простое физическое объяснение. С ростом η увеличивается доля полезной энергии при утилизации Q_A, поэтому выгодно передавать больше энергии на утилизацию. С увеличением плотности потока нейтронов растет запас реактивности на компенсацию ксенонового отравления и снижать уровень мощности в реакторе становится менее выгодным (ε^* растет).

Об эффективности оптимизации режима работы энергокомплекса «реактор – утилизатор энергии» можно судить по величине:

$$S = \frac{g_{\max} - g_{\text{опт}}}{g_{\max}} \cdot 100\,\%,$$

где $g_{\max}$ – максимальный удельный расход топлива для отработки переменного графика нагрузки «антиоптимальным» способом; $g_{\text{опт}}$ – минимальный удельный расход топлива при оптимальном уровне снижения мощности.

Величина S зависит от таких параметров, как номинальная плотность потока нейтронов в реакторе, КПД утилизации, время работы на пониженной мощности и уровнень снижения мощности. Расчеты показывают, что эффективность при различных КПД и потоках изменяется от 0,2 до 7 %. При этом для энергокомплекса с параметрами, близкими к реально возможным ($\varphi \approx$ $\approx (5 \div 7) \cdot 10^{13}\ \dfrac{\text{нейтр.}}{\text{см}^2 \cdot \text{с}}$; $\eta = 60 \div 80\,\%$), оптимальным является компромиссный режим, при этом эффект от оптимизации составляет величину 0,2 – 0,3 %.

СПИСОК ЛИТЕРАТУРЫ

1. *Аоки М.* Введение в методы оптимизации. – М.: Наука, 1977.
2. *Ашиманов С.А.* Линейное программирование. – М.: Наука, 1981.
3. *Бахвалов Н.С.* Численные методы (анализ, алгебра, обыкновенные дифференциальные уравнения). – М.: Наука, 1975.
4. *Бородакий Ю.В., Крицына Н.А., Кулябичев Ю.П., Шумилов Ю.Ю.* Вероятностно-статистические методы обработки данных в информационных системах. – М.: Радио и связь, 2003.
5. *Бородакий Ю.В., Лободинский Ю.Г.* Основы теории систем управления (исследование и проектирование). М.: Радио и связь, 2004.
6. *Бородакий Ю.В., Муравьев С.К., Шумилов Ю.Ю.* Нейросетевые методы оптимальной кластеризации воздушных объектов // Труды XIII Международного научно-технического семинара «Современные технологии в задачах управления, автоматики и обработки информации». – Алушта, 2004. – Ч. II. – С. 216 – 218.
7. *Васильев Ф.П.* Численные методы решения экстремальных задач. – М.: Наука, 1988.
8. *Васильев Ф.П., Иваницкий А.Ю.* Линейное программирование. – М.: Факториал Пресс, 2003.
9. *Вентцель Е.С.* Исследование операций: Задачи, принципы, методология. – М.: Высшая школа, 2001.
10. *Вентцель Е.С.* Теория вероятностей. – М. Высшая школа, 1999.
11. *Владимиров В.И.* Практические задачи по эксплуатации ядерных реакторов. – М.: Атомиздат, 1976.
12. *Гвоздев С.Е.* Математическое программирование (двойственность, транспортные задачи). – Новосибирск: Новосибирский государственный архитектурно-строительный университет, 2001.
13. *Дюбин Г.Н., Суздаль В.Г.* Введение в прикладную теорию игр. – М.: Наука, 1981.
14. *Дюран Б., Оделл П.* Кластерный анализ. – М.: Статистика, 1977.
15. *Ежов А.А., Шумский С.А.* Нейрокомпьютинг и его применения в экономике и бизнесе. – М.: МИФИ, 1998.
16. *Загребаев А.М., Наумов В.И.* О минимизации потери энерговыработки системы реакторов, работающих в переменном графике нагрузки // Атомная энергия. – 1979. – Т. 47. – Вып. 3. – С. 165 – 166.
17. *Зангвилл У.И.* Нелинейное программирование. Единый подход. – М.: Советское радио,1973.
18. *Зимин Г.В., Бурмистров С.К., Букин Б.М.* Справочник офицера ПВО. – М.: Воениздат, 1987.
19. *Золотов В.П. и др.* Общие принципы исследования эффективности боевых действий и вооружения: тексты лекций. – М.: ВАД, 1978.

20. *Иванов Н.М., Лысенко Л.Н.* Баллистика и навигация космических аппаратов. – М.: Дрофа, 2004.

21. Инвестиции / Под ред. *В. Ковалевой, В. Ивановой*. – М.: Проспект, 2004.

22. *Интрилигатор М.* Математические методы оптимизации и экономическая теория. – М.: Прогресс, 1975.

23. *Калиткин Н.Н.* Численные методы. – М.: Наука, 1978.

24. *Канторович Л.В, Горстко А.Б.* Математическое оптимальное программирование в экономике. – М.: Знание, 1968.

25. *Карманов В.Г.* Математическое программирование. – М.: Наука, 1986.

26. *Карманов В.Г.* Математическое программирование: учебное пособие. – 5е изд. – М.: Физматлит, 2004.

27. *Крянев А*. Основы финансового анализа и портфельного инвестирования в рыночной экономике. – М.: МИФИ, 2001.

28. *Кулябичев Ю.П., Анитова Т.В.* Теоретико-игровые методы исследования сложных систем. – М.: МИФИ, 1994.

29. *Кулябичев Ю.П., Крицына Н.А.* Лабораторный практикум «Теоретико-игровые методы исследования сложных систем». – М.: МИФИ, 1984.

30. *Кулябичев Ю.П., Шляхов А.В., Шумилов Ю.Ю.* Сравнительный анализ методов кластеризации // Труды XIII Международного научно-технического семинара «Современные технологии в задачах управления, автоматики и обработки информации». Алушта, 2003. – С. 362 – 363.

31. *Лавров С.С.* Программирование. Математические основы, средства, теория. – СПб.: БХВ – Петербург, 2001.

32. *Лесин В.В., Лисовец Ю.П.* Основы методов оптимизации. – М.: МАИ, 1988.

33. Линейное и нелинейное программирование / Под ред. *И.Н. Лященко.* – М.: Высшая школа, 1975.

34. *Лисицын В.* Основы методов оптимизации. – М.: МАИ, 2003.

35. *Максимов Ю.Я.* Алгоритмы линейного и дискретного программирования. – М.: МИФИ, 1980.

36. *Максимов Ю.Я., Филипповская Е.А.* Алгоритмы решения задач нелинейного программирования. – М.: МИФИ, 1982.

37. *Медынский М.М.* Численные методы нелинейной оптимизации (алгоритмы и программы): учебное пособие. – М.: МАИ, 2003.

38. *Мещеряков Р.В., Шумилов Ю.Ю.* Сравнительный анализ алгоритмов кластеризации // Труды X Международного научно-технического семинара «Современные технологии в задачах управления, автоматизации и обработки информации». – Алушта, 2001.

39. *Муравьев С.К.* Методика построения системы укрупнения и отображения воздушной обстановки // Труды X Международного научно-технического семинара «Современные технологии в задачах управления, автоматизации и обработки информации». – Алушта, 2001. – С. 247 – 248.

40. *Новикова Н.М.* Основы оптимизации (курс лекций). – М., 1998. (www.ccas.ru).

41. *Оуэн Г.* Теория игр. – М.: Мир, 1973.

42. *Пантелеев А.В.* Методы оптимизации в примерах и задачах: учебное пособие для студентов высших технических учебных заведений. – М.: Высшая школа, 2005.

43. *Пападимитриу Х., Стайглиц К.* Комбинаторная оптимизация. – М.: Мир, 1985.

44. *Партхасаратхи Т., Рагхаван Т.* Некоторые вопросы теории игр двух лиц. – М.: Мир, 1974.

45. *Петросян Л.А., Зенкевич Н.А., Семина Е.А.* Теория игр. – М.: Высшая школа, 1998.

46. *Пугачев В.С.* Теория вероятностей и математическая статистика. – М.: Наука, 1979.

47. *Пшеничный Б.Н., Дажилин Ю.М.* Численные методы в экстремальных задачах. – М.: Наука, 1975.

48. *Решетнев М.Ф., Лебедев А.А., Бартенев В.А. и др.* Управление и навигация искусственных спутников Земли на околокруговых орбитах. – М.: Машиностроение, 1988.

49. *Салмин И.Д.* Математические методы решения оптимизационных задач: учебное пособие. – М.: МИФИ, 2004.

50. *Салмин И.Д.* Математическое программирование. – Ч. 1, 2. – М.: МИФИ, 1978.

51. *Сергиенко И.В.* Задачи дискретной оптимизации. – Киев: Наукова думка, 2003.

52. *Сигал И.Х.* Введение в прикладное дискретное программирование. – М.: Физматлит, 2001.

53. *Струченков В.И.* Математическое программирование: Методы, задачи, обучающие компьютерные программы: учебное пособие. – М.: МИФИ, 2004.

54. *Субботин М.Ф.* Введение в теоретическую астрономию. – М.: Наука, 1968.

55. *Сухинин М.Ф.* Численное решение задач линейного программирования и вычисление границ спектра симметричной матрицы. – М.: Физматлит, 2002.

56. Техническая кибернетика. – Кн. 3. – Ч. 1 / Под ред. *В.В. Солодовникова.* – М.: Машиностроение, 1986.

57. *Тихонов А.Н., Васильева А.П., Свешников А.Г.* Дифференциальные уравнения. – М.: Наука, 2002.

58. *Ферапонтов М.М., Крицына Н.А., Деев Д.Л.* Моделирование случайных воздействий на ЭВМ. – М.: МИФИ, 1995.

59. *Фокс А., Пратт М.* Вычислительная геометрия. Применение в проектировании и на производстве. – М.: Мир, 1982.

60. *Хемминг Р.В.* Численные методы: пер. с англ. – М.: Наука, 1972.

61. *Цлаф Л.Я.* Вариационное исчисление и интегральные уравнения: справочное пособие. – СПб.: Лань, 2005.

62. *Шабунин М.И.* Некоторые вопросы математического программирования, Линейное программирование: учебное пособие. – Долгопрудный: МФТИ, 1981.

63. *Эйкхофф В.* Основы идентификации систем управления. – М.: Наука, 1985.

64. *Эльясберг П.Е.* Определение движения по результатам измерений. – М.: Наука, 1976.

65. Энциклопедия финансового риск-менеджмента / Под ред. *А. Лобанова, А. Чугунова.* – М.: Альпина Паблишер, 2003.

66. *Юдин А.Б., Гольдштейн Е.Г.* Задачи и методы линейного программирования. – М.: Советское радио, 1961.

67. *Lemke C.E., Howson J.J.* Equilibrium points of bimatrix games // J. of the Society for Industrial and Mathematics. 1964. V. 12. P. 413 – 423.

Андрей Маркоянович Загребаев
Надежда Александровна Крицына
Юрий Павлович Кулябичев
Юрий Юрьевич Шумилов

МЕТОДЫ МАТЕМАТИЧЕСКОГО ПРОГРАММИРОВАНИЯ В ЗАДАЧАХ ОПТИМИЗАЦИИ СЛОЖНЫХ ТЕХНИЧЕСКИХ СИСТЕМ

Учебное пособие

Редактор М.В. Макарова

Подписано в печать 10.10.2007. Формат 60x84 1/16
Печ.л. 20,75. Уч.-изд.л. 20,75. Тираж 200 экз.
Изд. № 1/2. Заказ №

Московский инженерно-физический институт
(государственный университет).
115409, Москва, Каширское ш., 31

Типография издательства «Тровант».
г. Троицк Московской обл.