1 Задачи оптимизации

Задачи оптимизации — широкий класс задач, связанных с применением численных методов.

Задача оптимизации:
Есть некоторое множество возможных решений, называемых альтернативными. Каждой альтернативе можно дать некоторую количественную оценку на основе некоторого критерия оптимальности. Решение задачи оптимизации состоит в определении той альтернативы, для которой критерий оптимальности даёт наибольшую или наименьшую количественную оценку.

Виды оптимизации:

- одномерная — множество альтернатив описывается одним числовым параметром;
- многомерная — множество альтернатив описывается несколькими параметрами;
- бесконечномерная — каждая альтернатива характеризуется бесконечным числом параметров;
- конечномерная;
- многокритериальная — в задаче оптимизации несколько критериев оптимальности.

Критерий представляется в виде целевой функции:

$$f_0(x) \to \min, \quad x \in \Omega$$

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

2 Задача математического программирования

С математической точки зрения задача конечномерной оптимизации заключается в определении наибольшего (или наименьшего) значения функции $f(x_1, x_2,$ на заданном множестве Ω и точки $x^* \in \Omega$, в которой достигается это значение функции f.

2.1 Определения

- Целевая функция: $f(x_1, x_2, \ldots, x_n)$
- Допустимое множество: $\Omega \in \mathbb{R}^n$
- Допустимые решения: $\forall x = (x_1, x_2, \ldots, x_n) \in \Omega$
- Оптимальное решение: $x^* \in \Omega$

2.2 Постановка задачи математического программирования в общем виде

$$\begin{cases} f_0(x_1, x_2, \ldots, x_n) \to \max(\min) \\ f_i(x_1, x_2, \ldots, x_n) = 0, \quad \forall i = 1, \ldots, m \end{cases}$$

Функция на заданном участке может не достигать наименьшего (наибольшего) значения, если

- она не ограничена снизу (сверху);
- она ограничена снизу, но не достигает точной нижней грани *(это как?)*.

В этих случаях задача не имеет решения.

3 Примеры задач оптимизации в экономике

Рассмотрим **транспортную задачу**. Предположим, что фирма имеет n магазинов, в которые поступает товар, хранимый на m складах. Известна стоимость c_{ij} перевозки товара с i-го склада в j-й магазин, количество a_i единиц товара на i-ом складе и заказанный объём b_j товара для доставки в j-й магазин. Требуется составить план перевозок товара со складов в магазины так, чтобы суммарная стоимость перевозок была минимальной.

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

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

Магазины должны быть обеспечены товаром в точном соответствии с заказом. Поэтому планируемые объёмы перевозок должны удовлетворять условиям:

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

Однако с любого склада нельзя вывести товара больше, чем там его находится. Следовательно, должны выполняться условия:

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

Объединяя перечисленные условия с условием неотрицательности количества товара: $x_{ij} \ge 0, \quad i = \overline{1, m}, \quad j = \overline{1, n}$, получим постановку задачи оптимизации.

Необходимо отметить, что задача имеет решение только в том случае, когда сумма заказов не превышает количества товара на складах:

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

4 Примеры задач оптимального проектирования (оптимизация в проектировании технических уст

Рассмотрим задачу проектирования бака горючего в виде прямого кругового цилиндра заданного объёма V, будем минимизировать количество стали.

В качестве параметров оптимизации выберем радиус R и высоту H цилиндра. Тогда затраты материала на изготовление бака будет определять площадь его поверхности. Ограничением будет объём бака:

$$\begin{cases} 2\pi R(H + R) \to \min \\ \pi R^2 H = V \\ R > 0, \quad H > 0 \end{cases}$$

Выразим H через R:

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

Подставим в целевую функцию и получим задачу минимизации функции одного переменного:

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

Вычислим производную и приравняем её к нулю:

$$S'(R) = -2\frac{V}{R^2} + 4\pi R = 0$$

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

Зная R_*, найдём второй параметр:

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

Получили, что цилиндрический бак горючего будет иметь наименьшую площадь поверхности в том случае, когда его высота совпадает с диаметром его основания.

5 Примеры задач оптимизации в геометрии

5.1 Пример 1

Рассмотрим задачу определения сторон прямоугольника, вписанного в окружность радиуса R и имеющего наибольшую площадь S:

Известно, что площадь прямоугольника равна половине произведения его диагоналей на синус угла между ними:

$$S = 2R^2 \sin\phi$$

Эта площадь будет наибольшей при $\sin\phi = 1$ или при $\phi = \pi/2$. Отсюда получаем, что искомый прямоугольник — это квадрат со стороной $\sqrt{2}R$ площади $2R^2$.

Сформулируем данную задачу в виде задачи оптимизации. Выберем параметрами оптимизации длины сторон квадрата: a и b. Тогда ограничение на параметры следует из теоремы Пифагора.

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

6 Классы задач нелинейного программирования

Задача **нелинейного** программирования: Либо целевая функция нелинейна, либо есть нелинейное ограничение. Для таких задач отсутствует универсальный метод решения, поэтому выделяются более узкие классы задач, допускающих специальные методы решения.

6.1 Задача квадратичного программирования

Все ограничения линейны, а целевая функция квадратичная.

6.2 Задача сепарабельного программирования

Сепарабельная функция — это сумма функций одной переменной вида:

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

Задача нелинейного программирования, в которой целевая функция и левые части ограничений — сепарабельные функции, является сепарабельной.

6.3 Задача геометрического программирования

Задача нелинейного программирования, в которой целевая функция и левые части ограничений — позиномы.

6.3.1 Позином

$$\sum_{i=1}^{m} c_i p_i(x), \quad c_i > 0, \quad i = \overline{1, m},$$

где $p_i(x)$ — моном.

6.3.2 Моном

$$\prod_{j=1}^{n} x_j^{a_{ij}}, \quad a_{ij} \in \mathbb{R}.$$

6.4 Задача кусочно-линейного программирования

Путём преобразований можно свести к линейным, затем решаются соответствующими методами.

7 Задачи выпуклого программирования. Выпуклое множество. Выпуклая функция

Задача **выпуклого** программирования: Допустимое множество Ω выпуклое, и целевая функция $f_0(x)$ выпуклая.

7.1 Выпуклое множество

Множество $X \subset \mathbb{R}^n$ называется выпуклым, если для $\forall x, y \in X$ и $\forall \alpha \in (0, 1)$ выполняется условие:

$$\alpha x + (1 - \alpha) y \in X.$$

Пересечение любого семейства выпуклых множеств является выпуклым множеством.

7.2 Выпуклая функция

Функция $f(x)$, определённая на выпуклом множестве X, называется выпуклой, если:

$$\forall x, y \in X, \quad \forall \alpha \in (0,1) \Rightarrow f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y).$$

Если неравенство выполняется строго, то $f(x)$ строго выпуклая. Пример: квадратичная функция с положительно определённой квадратичной формой.

Если функции $f_1(x), f_2(x), \ldots, f_k(x)$ определены на выпуклом множестве и являются выпуклыми, то и функция:

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

выпуклая.

Для выпуклой функции любая точка локального минимума есть точка наименьшего значения. Для выпуклой и дифференцируемой функции любая стационарная точка (точка с нулевым значением градиента) есть точка наименьшего значения. Строго выпуклая функция может иметь только одну точку наименьшего значения.

7.3 Сильно выпуклая функция

Функция $f(x)$ называется сильно выпуклой, если:

$$\forall x, y \in X, \quad \forall \alpha \in (0,1), \quad \exists \gamma > 0 \Rightarrow f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y) - \alpha(1-$$

Сильно выпуклая функция всегда достигает наименьшего значения.

8 Задачи стохастического и динамического программирования

8.1 Стохастическое программирование

Стохастическое программирование — это подход, позволяющий учитывать неопределённость в оптимизационных методах. В подобных задачах хотя бы один параметр оптимизации должен являться случайной величиной.

8.1.1 Постановка

Пусть целевая функция есть $f_0(x, \omega)$, где ω — некоторые случайные параметры, а M — оператор матожидания.

$$\begin{cases} M[f_0(x,\omega)] \to \min \\ M[f_i(x,\omega)] \leq 0, \quad i = \overline{1,m} \\ x = (x_1, x_2, \ldots, x_n) \end{cases}$$

8.2 Динамическое программирование

Динамическое программирование — это подход, при котором задача разбивается на группу последовательных этапов.

8.2.1 Принципы работы метода

Состоянием системы x_i на этапе i будем называть некоторый набор данных, получаемый в процессе работы этапа i.

- *Последовательность этапов*. Все этапы выполняются друг за другом в жёстко заданной последовательности.
- *Передача состояния от текущего этапа к следующему*. Генерируемое в процессе работы этапа i состояние передаётся этапу $i + 1$.
- *Рекуррентная природа вычислений*. Каждый этап i представляет собой некоторую функцию, которая получает на вход вектор состояния предыдущего этапа x_{i-1}, по определённому правилу выделяет некоторые альтернативы, выбирает из них лучшие и формирует новое состояние: $x_i = \Psi(x_{i-1})$.
- *Принцип оптимальности Беллмана*. Оптимальная стратегия на каждом отдельном этапе определяется исключительно на основании текущего состояния, независимо от состояний и оптимальных стратегий на предшествующих этапах.

8.3 Примечание

В присланных билетах по-другому, там больше формул. Я посчитал, что это излишне.

9 Задачи математического программирования с дискретным множеством параметров оптимизации

Задача **дискретного** программирования подразумевает дискретность области определения всех или нескольких параметров.

В линейном целочисленном программировании исследуется модель вида:

$$\begin{cases} \sum\limits_{j=1}^{n} c_j x_j \to \max(\min) \\ \sum\limits_{j=1}^{n} a_{ij} x_j \le b_i, \quad i = \overline{1,m} \\ x_j \ge 0, \quad j = \overline{1,n} \\ x_j - \text{целые}, \quad j = \overline{1,n'} \end{cases}$$

При $n = n'$ имеет место *задача линейного целочисленного программирования*; при $n > n'$ — задача линейного частично-целочисленного программирования.

Условие целочисленности можно заменить требованием дискретности:

$$x_j \in \{d_1^j, d_2^j, \ldots, d_{k_j}^j\}, \quad j = \overline{1,n'}.$$

В этом случае имеет место *задача линейного программирования с дискретными переменными*.

9.1 Частный случай: задача ЛЦП с булевыми переменными

Частным случаем задачи линейного целочисленного программирования является задача ЛЦП с булевыми переменными:

$$\begin{cases} \sum\limits_{j=1}^{n} c_j x_j \to \max(\min) \\ \sum\limits_{j=1}^{n} a_{ij} x_j \le b_i, \quad i = \overline{1,m} \\ x_j \in \{0,1\}, \quad j = \overline{1,n} \end{cases}$$

10 Виды задач линейного программирования (ЛП)

10.1 Общего вида

$$\begin{cases} \sum\limits_{j=1}^{n} c_j x_j \to \max \\ \sum\limits_{j=1}^{n} a_{ij} x_j \le b_i, \quad i = \overline{1,m_1} \\ \sum\limits_{j=1}^{n} a_{ij} x_j = b_i, \quad i = \overline{m_1+1,m_2} \\ \sum\limits_{j=1}^{n} a_{ij} x_j \ge b_i, \quad i = \overline{m_2+1,m} \end{cases}$$

10.2 1. Неотрицательных переменных

$$\begin{cases} \sum\limits_{j=1}^{n} c_j x_j \to \max \\ \sum\limits_{j=1}^{n} a_{ij} x_j \le b_i, \quad i = \overline{1, m_1} \\ \sum\limits_{j=1}^{n} a_{ij} x_j = b_i, \quad i = \overline{m_1 + 1, m} \\ x_j \ge 0, \quad j = \overline{1, n} \end{cases}$$

10.3 2. Стандартная форма

$$\begin{cases} \sum\limits_{j=1}^{n} c_j x_j \to \max \\ \sum\limits_{j=1}^{n} a_{ij} x_j \le b_i, \quad i = \overline{1, m} \\ x_j \ge 0, \quad j = \overline{1, n} \end{cases}$$

10.4 3. Каноническая форма

$$\begin{cases} \sum\limits_{j=1}^{n} c_j x_j \to \max \\ \sum\limits_{j=1}^{n} a_{ij} x_j = b_i, \quad i = \overline{1, m} \\ x_j \ge 0, \quad j = \overline{1, n} \end{cases}$$

10.5 4. Матричная стандартная форма

$$\begin{cases} (c, x) \to \max \\ Ax \le b \\ x \ge 0 \end{cases}$$

11 Преобразования форм задач ЛП

11.1 1. Изменение направления оптимизации

Исходная задача	Эквивалентная задача
$\sum\limits_{j=1}^{n} c_j x_j \to \min$	$-\sum\limits_{j=1}^{n} c_j x_j \to \max$

Оптимальные решения исходной задачи и задачи, полученной в результате преобразования, будут совпадать, однако оптимальные значения целевых функций будут иметь противоположные знаки.

11.2 2. Придание ограничениям-неравенствам противоположного направления

Исходная задача	Эквивалентная задача
$\sum\limits_{j=1}^{n} a_{ij}x_j \geq b_i$	$-\sum\limits_{j=1}^{n} a_{ij}x_j \leq -b_i$

11.3 3. Наложение на переменные требования неотрицательности

Исходная задача	Эквивалентная задача
x_j не ограничена в знаке	Замена переменной: $x_j = x_j^1 - x_j^2, \quad x_j^1, x_j^2 \geq 0$

11.4 4. Замена уравнений неравенствами

Исходная задача	Эквивалентная задача
$\sum\limits_{j=1}^{n} a_{ij}x_j = b_i$	$\begin{cases} \sum\limits_{j=1}^{n} a_{ij}x_j \leq b_i \\ -\sum\limits_{j=1}^{n} a_{ij}x_j \leq -b_i \end{cases}$

11.5 5. Замена нестрогих неравенств уравнениями

Исходная задача	Эквивалентная задача
$\sum\limits_{j=1}^{n} a_{ij}x_j \leq b_i$	$\begin{cases} \sum\limits_{j=1}^{n} a_{ij}x_j + x_{n+1} = b_i \\ x_{n+1} \geq 0 \end{cases}$
$\sum\limits_{j=1}^{n} a_{ij}x_j \geq b_i$	$\begin{cases} \sum\limits_{j=1}^{n} a_{ij}x_j - x_{n+1} = b_i \\ x_{n+1} \geq 0 \end{cases}$

12 Базисные и свободные переменные в задаче ЛП

Рассмотрим задачу ЛП в канонической форме:

$$\begin{cases} (c, x) \to \max \\ Ax = b \\ x \geq 0 \end{cases}$$

Пусть $A \in \mathbb{R}^{m \times n}$. Предполагается, что $m < n$ и что $\operatorname{rang} A = m$. Тогда можно выбрать из этой матрицы базисный минор $B \in \mathbb{R}^{m \times m}$. Оставшиеся столбцы матрицы обозначим как $N \in \mathbb{R}^{m \times (n-m)}$.

Такая операция разделит неизвестные на базисные и свободные:

$$x^B = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix}, \quad x^N = \begin{pmatrix} x_{m+1} \\ \vdots \\ x_n \end{pmatrix}.$$

С учётом введённых обозначений ограничения можно описать в виде матричного уравнения:

$$Bx^B + Nx^N = b$$

13 Графический метод решения задач ЛП. Теорема о количестве решений ЗЛП

Если в задаче ЛП всего две переменных, то для её решения можно использовать геометрическую интерпретацию. В этом случае ограничение типа равенства представляет собой прямую, а каждое ограничение типа неравенства — полуплоскость.

Если задача записана в стандартной форме:

$$\begin{cases} c_1x_1 + c_2x_2 \to \max \\ a_{i1}x_1 + a_{i2}x_2 \le b_i, \quad i = \overline{1,m} \\ x_1 \ge 0, \quad x_2 \ge 0 \end{cases}$$

то допустимое множество Ω будет пересечением некоторого множества полуплоскостей и будет представлять собой:

- выпуклый многоугольник;
- отрезок;
- незамкнутую область;
- единственную точку;
- пустое множество.

Линии уровня целевой функции составляют семейство параллельных прямых, общим нормальным вектором которых является вектор $c = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$ коэффициентов целевой функции. При перемещении прямой вдоль этого вектора можно найти её крайнее положение, когда она ещё пересекает многоугольник (прямая f^*).

Каждая точка пересечения этой прямой с многоугольником изображает оптимальное решение задачи, а значение функции, соответствующее этой линии уровня, есть максимальное значение целевой функции.

13.1 Теорема

Теорема 1. *Если задача линейного программирования имеет оптимальное решение, то оно совпадает с одной (двумя) из угловых точек допустимого множества.*

14 Базисное решение ЗЛП

Рассмотрим задачу ЛП в канонической форме:

$$\begin{cases} (c, x) \to \max \\ Ax = b \\ x \geq 0 \end{cases}$$

Пусть $A \in \mathbb{R}^{m \times n}$. Предполагается, что $m < n$ и что $\operatorname{rang} A = m$. Тогда можно выбрать из этой матрицы базисный минор $B \in \mathbb{R}^{m \times m}$. Оставшиеся столбцы матрицы обозначим как $N \in \mathbb{R}^{m \times (n-m)}$.

Такая операция разделит неизвестные на базисные и свободные:

$$x_B = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix}, \quad x_N = \begin{pmatrix} x_{m+1} \\ \vdots \\ x_n \end{pmatrix}.$$

С учётом введённых обозначений ограничения можно описать в виде матричного уравнения:

$$Bx_B + Nx_N = b$$

Матрица B представляет собой базисный минор, поэтому невырождена. Значит $\exists B^{-1}$. Умножим равенство на B^{-1}, чтобы выразить вектор базисных переменных:

$$x_B = B^{-1}(b - Nx_N)$$

Таким образом, вектор

$$x = \begin{pmatrix} x_B \\ x_N \end{pmatrix} = \begin{pmatrix} B^{-1}(b - Nx_N) \\ x_N \end{pmatrix}$$

удовлетворяет системе ограничений, а при $x \geq 0$ является допустимым решением. Выделим случай $x_N = 0$. Здесь $x = x_B$, и такое решение называется **базисным решением**. Базисное решение не является единственным, так как зависит от выбора минора, однако число базисных решений конечно.

15 Лемма о крайних точках

Лемма 1. *Допустимое решение $x = (x_1\, x_2\, \ldots\, x_n)^T$ задачи линейного программирования в канонической форме является точкой множества допустимых решений тогда и только тогда, когда система всех столбцов матрицы A, отвечающих ненулевым значениям x_i, $i = \overline{1,n}$, линейно независима.*

Доказательство. Докажем от противного. Можно считать, что вектор x имеет вид $x = (x_1\, x_2\, \ldots\, x_r\, 0\, \ldots\, 0)^T$, $x_i \neq 0$, $i = \overline{1,r}$.

Если вектор x не является крайней точкой допустимого множества, то $\exists u \neq 0$ вектор и такое число $\delta > 0$, что вектор $x + tu$, $|t| < \delta$, принадлежит допустимому множеству.

Т.е. $A(x + tu) = b$, $x + tu \geq 0$. Последнее неравенство можно записать в виде $x \geq -tu$, тогда имеем, что $\forall i = \overline{r+1,n}$, $u_i = 0$, т.к. при $t > 0$ имеем $u_i \geq 0$, а при $t < 0$ — $u_i \leq 0$.

Поскольку x принадлежит допустимому множеству, выполняется $Ax = b$. С учётом равенства $A(x + tu) = b$ получаем, что $Au = 0$. С учётом вышепоказанного это выражение записывается так:

$$a_1u_1 + a_2u_2 + \cdots + a_ru_r = 0 \quad (*)$$

По условию $u \neq 0$, значит по критерию линейной зависимости столбцы $a_1, a_2, \ldots$ линейно зависимы.

Теперь пусть столбцы $a_1, a_2, \ldots, a_r$ линейно зависимы. Тогда $\exists u_1, u_2, \ldots, u_r$ ($\sum\limits_{i=0}^{r} |u_i| \neq 0$), для которых выполняется равенство $(*)$. Рассмотрим вектор $u = (u_1\, \ldots\, u_r\, 0\, \ldots\, 0)^T$. Равенство $(*)$ означает, что $Au = 0$. Следовательно, $A(x+tu) = Ax = b$ для любого t, т.е. вектор $x+tu$ удовлетворяет ограничениям задачи ЛП. Покажем, что при малых значениях t этот вектор удовлетворяет и условию неотрицательности.

Пусть $\rho = \max\limits_{i=\overline{1,r}} |u_i| > 0$. Пусть $\delta = \min\limits_{i=\overline{1,r}} \frac{|x_i|}{\rho} > 0$. При этом при $|t| < \delta$ выполняется:

$$x_i + tu_i \geq x_i - |t||u_i| > x_i - \delta|u_i| \geq x_i - \delta\rho \geq 0$$

Следовательно, $x + u \geq 0$, и вектор x не является крайней точкой допустимого множества. □

16 Теорема о допустимом базисном решении

Теорема 2. *Допустимое решение задачи линейного программирования в канонической форме является допустимым базисным решением (ДБР) тогда и только тогда, когда оно является крайней точкой Ω.*

Доказательство. У ДБР ненулевыми могут быть только те компоненты, которые отвечают базисным переменным, т.е. система столбцов матрицы A ограничений, отвечающих ненулевым компонентам вектора, есть часть системы базисных столбцов, а потому она линейно независима. Следовательно, ДБР — крайняя точка допустимого множества.

У крайней точки x допустимого множества количество ненулевых значений не превышает ранга матрицы A, а, значит, количества ограничений m. Соответствующие столбцы матрицы A линейно независимы, и их можно дополнить до набора базисных столбцов. Всем оставшимся столбцам будут соответствовать нулевые компоненты вектора x. Следовательно, вектор x является базисным решением. Т.к. он принадлежит допустимому множеству, он есть ДБР. □

17 Симплекс-метод. Идея. Процесс. Базис ЗЛП

Поиск оптимального решения задачи линейного программирования в канонической форме следует проводить среди допустимых базисных решений, или, что то же самое, среди крайних точек допустимого множества. Полный перебор всех таких точек неэффективен, так что можно процесс поиска строить, переходя от одного базисного решения к другому, которое будет являться "соседним". Это идея симплекс-метода.

Проиллюстрируем процесс работы симплекс-метода на примере. Пусть требуется максимизировать $f(x_1, x_2, x_3) = \alpha x_3, \quad \alpha > 0$. При такой постановке необходимо искать точку многогранника с максимальным значением координаты x_3. Выберем одну из вершин в качестве начальной. На каждом шаге переходим в одну из соседних (т.е. вершин, соединённых с текущей ребром многогранника), выбирая среди них такую, у которой x_3 больше. Если такой соседней вершины нет, то текущая вершина есть оптимальное решение.

Процесс поиска оптимального решения можно представить себе как последовательный переход от одного допустимого базисного решения к "соседнему дающему большее значение целевой функции. "Соседство"состоит в том, что новое допустимое базисное решение получается из исходного заменой одного из свободных переменных; иначе говоря, из базиса выводится одно переменное и в него включается другое, которое до этого было свободным.

Совокупность базисных переменных будем называть **базисом задачи линейного программирования**.

18 Выводимая из базиса переменная в ЗЛП. Вырожденная ЗЛП

Пусть есть задача ЛП в канонической форме. Пусть первые r переменных будут базисными, а остальные — свободными. Разрешим систему ограничений относительно базисных переменных, а также выразим целевую функцию через свободные переменные:

$$f(x) = \gamma_0 - (\gamma_{r+1}x_{r+1} + \cdots + \gamma_n x_n)$$

$$\begin{cases} x_1 = \beta_1 - (\alpha_{1r+1}x_{r+1} + \cdots + \alpha_{1n}x_n) \\ x_i = \beta_i - (\alpha_{ir+1}x_{r+1} + \cdots + \alpha_{in}x_n) \\ x_r = \beta_r - (\alpha_{rr+1}x_{r+1} + \cdots + \alpha_{rn}x_n) \end{cases}$$

Возьмём все свободные переменные равными нулю. Тогда получим базисное решение $x_i = \beta_i, \quad i = \overline{1,r}$, $\vec{x_b} = (\beta_1, \beta_2, \ldots, \beta_r, 0, \ldots, 0)^T$. Значение целевой функции, в свою очередь, равно $f(\vec{x_b}) = \gamma_0$.

Легко заметить, что значение целевой функции можно увеличить только в случае, если имеются $\gamma_j < 0$. Пусть такой имеется, тогда за счёт увеличения x_j можно увеличить значение целевой функции. Возьмём все свободные переменные равными нулю, кроме x_j:

$$f(\vec{x}) = \gamma_0 - \gamma_j x_j$$

$$x_i = \beta_i - \alpha_{ij}x_j, \quad i = \overline{1,r}$$

Если все коэффициенты $\alpha_{ij} \leq 0$, то увеличение x_j может быть неограниченным, это приводит к неограниченному возрастанию целевой функции. В таком случае считается, что задача не имеет решения.

Если же $\alpha_{ij} > 0$, то увеличение x_j приведёт к тому, что базисная переменная x_i будет уменьшаться, пока не станет равна нулю. Это определяется уравнением: $x_i = \beta_i - \alpha_{ij}x_j = 0$. Отсюда $x_j = \frac{\beta_i}{\alpha_{ij}}$.

Если несколько $\alpha_{ij} > 0$, то первой в ноль обратится переменная x_l, для которой отношение $\frac{\beta_l}{\alpha_{lj}}$ минимально. Таким образом, нужная переменная выбирается из соотношения

$$\frac{\beta_l}{\alpha_{lj}} = \min_i \left\{ \frac{\beta_i}{\alpha_{ij}} \right\} = \rho$$

Элемент α_{lj} называется разрешающим: он указывает переменную x_l, которую выводят из базиса.

Может оказаться, что $\rho = 0$, тогда целевая функция сохраняет предыдущее значение, хотя переменные меняются. Такой случай называют **вырожденным**.

19 Вводимая в базис переменная в ЗЛП. Вектор симплекс-разностей. Оптимальное базисное решение

Пусть есть задача ЛП в канонической форме. Пусть первые r переменных будут базисными, а остальные — свободными. Разрешим систему ограничений относительно базисных переменных, а также выразим целевую функцию через свободные переменные:

$$f(x) = \gamma_0 - (\gamma_{r+1}x_{r+1} + \cdots + \gamma_n x_n) \quad (1)$$

$$\begin{cases} x_1 = \beta_1 - (\alpha_{1r+1}x_{r+1} + \cdots + \alpha_{1n}x_n) \\ x_i = \beta_i - (\alpha_{ir+1}x_{r+1} + \cdots + \alpha_{in}x_n) \\ x_r = \beta_r - (\alpha_{rr+1}x_{r+1} + \cdots + \alpha_{rn}x_n) \end{cases}$$

Возьмём все свободные переменные равными нулю. Тогда получим базисное решение $x_i = \beta_i, \quad i = \overline{1,r}$, $\vec{x_b} = (\beta_1, \beta_2, \ldots, \beta_r, 0, \ldots, 0)^T$. Значение целевой функции, в свою очередь, равно $f(\vec{x_b}) = \gamma_0$.

Легко заметить, что значение целевой функции можно увеличить только в случае, если имеются $\gamma_j < 0$. Если же все $\gamma_j \geq 0$, то значение целевой функции увеличить нельзя. Поэтому признаком **оптимальности решения** поставленной задачи максимизации является неотрицательность всех коэффициентов при свободных переменных в выражении (1).

Пусть такой имеется, тогда за счёт увеличения x_j можно увеличить значение целевой функции. Возьмём все свободные переменные равными нулю, кроме x_j:

$$f(\vec{x}) = \gamma_0 - \gamma_j x_j$$
$$x_i = \beta_i - \alpha_{ij}x_j, \quad i = \overline{1,r}$$

Если все коэффициенты $\alpha_{ij} \leq 0$, то увеличение x_j может быть неограниченным, это приводит к неограниченному возрастанию целевой функции. В таком случае считается, что задача не имеет решения.

Если же $\alpha_{ij} > 0$, то увеличение x_j приводит к тому, что базисная переменная x_i будет уменьшаться, пока не станет равна нулю. Это определяется уравнением: $x_i = \beta_i - \alpha_{ij}x_j = 0$. Отсюда $x_j = \frac{\beta_i}{\alpha_{ij}}$.

Если несколько $\alpha_{ij} > 0$, то первой в ноль обратится переменная x_l, для которой отношение $\frac{b_l}{a_{lj}}$ минимально. Таким образом, нужная переменная выбирается из соотношения

$$\frac{b_l}{a_{lj}} = \min_i \left\{ \frac{b_i}{a_{ij}} \right\} = \rho$$

Элемент a_{lj} называется разрешающим: он указывает переменную x_l, которую выводят из базиса, и свободную переменную x_j, которую вводят в базис.

Теперь у нас есть новый базис. Выразим базисную переменную x_l через свободные переменные. Для этого возьмём одно из уравнений системы ограничений:

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

Из этого уравнения получим:

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

Подставив полученное x_j во все остальные уравнения, получим выражения для нового базиса.

Вектор симплекс-разностей — это вектор $\vec{\gamma}$.

20 Вырожденное допустимое базисное решение. Зацикливание в решении ЗЛП

Вырожденному ДБР соответствуют несколько наборов базисных переменных. В этом случае смена базиса не вызывает изменения ДБР и продвижения к оптимальному решению не происходит. Чтобы избежать этой ситуации, можно изменить вводимую в базис переменную, выбрав другую положительную симплекс-разность, но это не всегда возможно. Может случиться, что после перебора нескольких базисов данного ДБР произойдёт возврат к уже рассматривавшемуся базису. Произойдёт зацикливание.

Эту ситуацию можно исключить, если все базисы определённым образом упорядочить, используя значение ЦФ на соответствующем ДБР и состав переменных, входящих в базис. Тогда базисы не будут повторяться. При наличии положительных симплекс-разностей уточнённый итерационный процесс перехода от одного ДБР к другому приведёт к новому большому базису. Следовательно, итерационный процесс остановится, когда для очередного базиса все симплекс-разности окажутся положительными. Это будет означать получение оптимального решения.

20.1 Дополнение из Загребаева (стр. 35–41)

Можно добавить информацию о методах предотвращения зацикливания, таких как **антициклины** и другие подходы, описанные в работах Загребаева. Эти методы позволяют упорядочить выбор базисных переменных и исключить повторение базисов, что гарантирует сходимость симплекс-метода.

21 Алгоритм симплекс-метода

1. Каноническая система ограничений $A\vec{x} = \vec{b}$ приводится к виду, в котором базисные переменные выражены через свободные, а целевая функция к примерно тому же:

$$f(x) = \gamma_0 - (\gamma_{r+1}x_{r+1} + \cdots + \gamma_n x_n)$$

$$\begin{cases} x_1 = \beta_1 - (\alpha_{1r+1}x_{r+1} + \cdots + \alpha_{1n}x_n) \\ x_i = \beta_i - (\alpha_{ir+1}x_{r+1} + \cdots + \alpha_{in}x_n) \\ x_r = \beta_r - (\alpha_{rr+1}x_{r+1} + \cdots + \alpha_{rn}x_n) \end{cases}$$

2. Заполняется симплекс-таблица:

Каждая строка таблицы соответствует уравнению, а последняя строка соответствует целевой функции.

3. В строке коэффициентов целевой функции (не считая γ_0) выбирается $\gamma_j < 0$, максимальное по модулю. Если все $\gamma_j \geq 0$, то оптимальное решение достигнуто, причём значения переменных определяются столбцом свободных членов, а оптимальное значение функции — клеткой, соответствующей свободному члену.

4. В j-ом столбце среди положительных коэффициентов α_{ij} выбирается разрешающий элемент α_{lj}, т.е. элемент, для которого минимально отношение $\frac{\beta_l}{\alpha_{lj}}$. Если положительных коэффициентов нет, то задача не имеет решений.

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

7. Переменная x_j вводится в базис вместо x_l.

Далее идёт переход на шаг 3.

22 Общий метод нахождения начального допустимого базисного решения ЗЛП

Для решения задачи ЛП симплекс-методом необходимо выразить базисные переменные через свободные, причём в этом выражении свободные члены должны быть неотрицательными. В случае, если эта проблема не решается тривиально, используется *метод искусственного базиса*, при котором в задачу вводятся дополнительные искусственные переменные.

22.1 Метод вспомогательной задачи

Пусть дана задача ЛП:

$$\vec{c}\vec{x} \to \max$$

$$\begin{cases} a_{i1}x_1 + \cdots + a_{in}x_n = b_i, \quad i = \overline{1,m} \\ x_j \geq 0, \quad j = \overline{1,n} \end{cases}$$

Вспомогательная задача имеет вид:

$$-x_{n+1} - x_{n+2} - \cdots - x_{n+m} \to \max$$

$$\begin{cases} a_{i1}x_1 + \cdots + a_{in}x_n + x_{n+i} = b_i, \quad i = \overline{1,m} \\ x_j \geq 0, \quad j = \overline{1,n+m} \end{cases}$$

Здесь $\forall b_i \geq 0$. В вспомогательной задаче очевидно опорное решение: $x' = (0, \ldots, 0, b_1, \ldots, b_m)$, где первые n нулей — значения основных переменных, а остальные значения соответствуют искусственным. Искусственные переменные образуют единичный базис, значит можно решать задачу симплекс-методом и получить оптимальное решение вспомогательной задачи:

$$x^* = (x_1^*, \ldots, x_{n+m}^*)$$

В зависимости от того, входят ли искусственные векторы в базис оптимального решения и какие при этом значения имеют искусственные переменные, относительно исходной задачи можно сделать различные выводы.

22.1.1 Случай 1

Среди чисел $x_{n+1}^*, \ldots, x_{n+m}^*$ есть отличные от нуля. В этом случае исходная задача не имеет допустимых решений.

22.1.2 Случай 2

В оптимальном решении все искусственные переменные имеют нулевое значение.

Случай 2а В оптимальном базисе нет искусственных векторов. В этом случае получено опорное решение исходной задачи и известно разложение всех векторов задачи по базису опорного решения. Для решения исходной задачи достаточно:

- в полученной симплекс-таблице отбросить столбцы искусственных переменных;
- восстановить коэффициенты целевой функции исходной задачи;
- пересчитать значение целевой функции и оценки свободных векторов.

Случай 2б В оптимальный базис входят искусственные векторы. Важно знать, существуют ли среди коэффициентов разложения основных векторов при искусственных базисных векторах отличные от нуля.

Здесь для определённости принято, что первые l переменных основные, а остальные — искусственные. Рассматриваем область x_{rs}.

Случай 2б1 Все $x_{rs} = 0$. Тогда любой вектор исходной задачи выражается только через основные вектора, входящие в базис оптимального решения вспомогательной задачи. Таким образом, можно выкинуть строки и столбцы, связанные с искусственными переменными, и действовать как в случае 2а.

Случай 2б2 Существует $x_{rs} \neq 0$. Тогда основной вектор A_s принудительно вводится в базис вместо искусственного вектора A_{i_r} и по обычным правилам перестраивается симплекс-таблица. Процедура повторяется, пока не возникнет ситуация 2а или 2б1. В любом случае будет получено опорное решение исходной задачи и базис этого решения, состоящий только из основных векторов.