

Санкт-Петербургский политехнический университет Петра Великого
Институт прикладной математики и механики
Кафедра «Прикладная математика»

**ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ
«РЕШЕНИЕ ЗАДАЧ ЛИНЕЙНОГО
ПРОГРАММИРОВАНИЯ СИМПЛЕКС-МЕТОДОМ»
ПО ДИСЦИПЛИНЕ «МЕТОДЫ ОПТИМИЗАЦИИ»**

Выполнили
студенты группы 3630102/70401

Зуев В. А.
Курова А. Н.
Мельникова А. Н.
Стоян А. С.

Руководитель
к. ф.-м. н., доц.

Родионова Елена Александровна

Санкт-Петербург
2020

1 Постановка задачи

Поставлена задача линейного программирования:

$$\begin{cases} x_1 - 2x_2 + 2x_3 \leq 6 \\ x_1 + 2x_2 + x_3 + x_4 = 24 \\ 2x_1 + x_2 - 4x_3 + x_4 = 30 \\ -x_1 + 4x_2 - 2x_4 \geq -6 \\ x_1 \geq 0 \end{cases} \quad (1)$$

$$8x_1 + 8x_2 - 4x_3 - 2x_4 \longrightarrow \min$$

1. Привести задачу к виду, необходимому для применения симплекс-метода.
2. Построить к данной задаче двойственную и также привести к виду, необходимому для применения симплекс-метода.
3. Решить обе задачи симплекс-методом с выбором начального приближения методом искусственного базиса.
4. Решить обе задачи методом перебора крайних точек.
5. Разработать схему восстановления прямой задачи по решению двойственной.

Алгоритмы, требуемые для решения задачи, реализовать в таком виде, чтобы их можно было использовать в качестве подпрограмм в следующих лабораторных работах.

2 Исследование применимости метода

Алгоритм симплекс-метода, приведённый в пособии [1] и описанный ниже, применим к задачам линейного программирования на нахождение минимума. Метод работает с задачами в канонической форме при всяких вещественных значениях компонент $A \in \mathbb{R}_{m \times n}$, $b \in \mathbb{R}_m$, $c \in \mathbb{R}_n$ с условием: матрица A имеет ранг m (следовательно, существует хотя бы один опорный вектор).

После приведения задачи (1) (или двойственной к ней) к каноническому виду получается система с матрицей 9×4 ранга 4, следовательно, условие выполнено.

3 Описание алгоритмов

3.1 Алгоритм перевода из общей в каноническую форму

Вход: система

1. Проверяем знаки в системе
2. Если « \leq », то к левой части добавляем $w[i]$, если « \geq », то из левой части вычитаем $w[i]$, $w[i] \geq 0$.
3. Знаки неравенства в системе заменяем на равенство.
4. Производим замену переменных: если $x[i] \leq 0$, то $x'[i] = -x[i] \geq 0$; если $x[i]$ любого знака, то $x[i] = u[i] - v[i]$, $v[i], u[i] \geq 0$.

3.2 Алгоритм построения двойственной задачи

Для простоты алгоритма будем рассматривать задачу минимума:

$$\begin{aligned} (x[N], c[N]) &\longrightarrow \min_{x[N]}, x[N] \in S, x[N] \geq 0 \\ S &:= \{x[N] | A[M, N] \cdot x[N] \geq b[M]\}, x[N] \geq 0 \end{aligned} \quad (2)$$

Если перед нами стоит задача максимума, то домножим вектор коэффициентов матрицы цели на -1 .

1. Транспонируем заданную матрицу
2. Новый вектор коэффициентов, стоящий в системе справа, равен вектору коэффициентов функции цели (2).
3. Новый вектор коэффициентов функции цели равен вектору коэффициентов, стоящему в системе (2) справа.
4. Если ограничение на $x[i] \geq 0$, то i -ая строка новой системы имеет знак « \geq ». Если нет ограничения на знак, то i -ая строка новой системы имеет знак « $=$ ».
5. Если ограничение i -ой строки в исходной системе « \geq » (тк рассматриваем задачу минимума), то ограничение на знак новой переменной $y[i] \geq 0$. Если ограничение i -ой строки в исходной системе « $=$ », то $y[i]$ любого знака.
6. Если исходная задача на поиск минимума, то двойственная на поиск максимума.

3.3 Алгоритм симплекс-метода и связанные процедуры

Алгоритм симплекс-метода

1. Ввод: $x_0 [N]$
 $A [M, N]$
 $b [M]$
 $c [N]$
 N_k
 $B [N_0, M]$
- где A, b, c - параметры задачи $c^T x \rightarrow \min, x \in S$,
 $S := \{x \geq 0 \mid Ax = b\}$
 $x_0 [N]$ - опорный вектор к множеству S
 N_0 - индексы базисных столбцов $x_0, N_0 \subset N$
 $B [N_0, M] : B [N_0, M] \cdot A [M, N_0] = E$

2. Введён $N_k^0 := \{i \in N_k \mid x_k[i] = 0\}$ (вначале $k=0, x_k = x_0, N_k = N_0$)
 $N_k^+ := \{i \in N_k \mid x_k[i] > 0\}$

$$bg := \left(\begin{array}{cccc} 1 & \dots & 1 & 1 \\ \vdots & & & \\ 1 & \underbrace{\left(bg_{i-1,j} + bg_{i,j-1} \right)}_{m - |N_k^+|} & & \end{array} \right) \Bigg\} |N_k^+|$$

- повернутый на 45° участок
 треугольника Паскаля. В каждой
 ячейке $bg_{ij} = C_i^j$, в левом нижнем
 углу $C^{m - |N_k^+|}_{|N_k^+|}$

3. Циклы для каждого $binom_idx$ от 0 до $C^{m - |N_k^+|}_{|N_k^+|}$:

- 3а. Ищем индексы $N_k \subset N, L_k \subset N : N = N_k \cup L_k ; N_k \supset N_k^+ ; N_k = m$;
 N_k отличается от N_{k-1} только одним индексом.

Для этого запускаем процедуру `subsetByIndex`, наложив её параметры

$$bg = bg, i = binom_idx, M = N \setminus N_k^+ = N_k^0$$

$$\text{Её результатом } \tilde{N}_k^0 \Rightarrow N_k := N_k^+ \cup \tilde{N}_k^0$$

$$L_k := N \setminus N_k$$

- 3б. Если определитель $\det(A [M, N_k]) = 0$, переходим к следующему 'binom_idx',
 т.к. текущие индексы не могут быть базисными индексами базисных столбцов 0.в.

- 3в. Если $binom_idx = 0$, нам известна $B [N_0, M]$. В ином случае ищем $B [N_k, M]$,
 отобразив на $U_{k-1} [N_{k-1}]$:

3.в.1. Положим $i_k :=$ "N-индекс в N_k , отсутствующий по сравнению с N_{k-1} "

3.6. II. Строим

$$F[N_k, N_{k-1}] := \begin{bmatrix} 1 & 0 & \dots & 0 & -u_k[i_1]/u_k[i_k] & 0 & \dots & 0 \\ & \ddots & & & \vdots & & & \\ 0 & & \ddots & 1 & -u_k[j_{k-1}]/u_k[i_k] & 0 & & \\ & & & & \vdots & & & \\ & & & 0 & 1/u_k[i_k] & 0 & & \\ & & & 0 & -u_k[j_{k+1}]/u_k[i_k] & 1 & & \\ & & & & \vdots & 0 & \ddots & 0 \\ & & & & & & & \\ 0 & \dots & 0 & & -u_k[m]/u_k[i_k] & 0 & \dots & 0 & 1 \end{bmatrix}$$

3.8. III. $B[N_k, M] := F[N_k, N_{k-1}] \cdot B[N_{k-1}, M]$

3.2. Находим векторы

$$y_k[M] := B^T[N_k, M] \cdot c[N_k]$$

$$d_k[N] := c[N] - A^T[M, N] \cdot y_k[M]$$

3.9. Если $d_k[i] \geq 0 \quad \forall i \in L_k$, то x_k -решение (завершаем алгоритм).

3e. $j_k :=$ "первый индекс из L_k : $d_k[j_k] < 0$

$$u_k[N_k] := B[N_k, M] \cdot A[M, j_k]$$

3ж. Если $u_k[N_k]$ не содержит положительных компонент, останавливаем процесс: целевая функция $c^T[N] \cdot x[N]$ не ограничена снизу

3з. Если $N_k^+ = N_k$ (о.в. невырожденный) или $u_k[N_k \setminus N_k^+] \leq 0$:

3.3.I. $\theta_k := \min_{i \in N_k, u_k[i] > 0} \frac{x_k[i]}{u_k[i]}$

3.3.II. Дополним $u_k[N_k]$ до $u_k[N]$ так:

$$u_k[j_k] = -1$$

$$u_k[L_k \setminus j_k] = 0$$

$$x_{k+1}[N] := x_k[N] - \theta_k u_k[N]$$

3.3.III. Устанавливаем $k := k+1$ и переходим к шагу (2).

Вспомогательные процедуры

• Subset By Index:

1. Ввод: таблица бинарных коэффициентов

$$bg = \left[\begin{array}{c} 1 \dots 1 \\ \vdots \\ (bg_{ij}) \\ \vdots \\ 1 \end{array} \right]_{k+1}^{n-k+1} \quad bg_{ij} = bg_{i,j} + bg_{i,j-1} \quad n \geq k$$

индекс $idx \in \{0; C_n^k - 1\}$ - номер подмножества

упорядоченное мн-во (массив) эл-тов $M: |M| = n$

2. Обходим матрицу 'bg', начиная в левом нижнем углу. Объявим

$t := idx$

$i := n - k$

~~$j := k$~~

$res := \{\emptyset\}$

3. В цикле до тех пор, пока $j > 0$:

3.а. Если $t < bg[i, j-1]$:

3.а.1. добавляем $M_{[i+j-1]}$ в 'res'

3.а.2. $j := j - 1$ (переходим в клетку правее)

3.б. иначе:

3.б.1. $t := t - bg[i, j-1]$

$t := t - 1$

4. Возвращаем 'res'

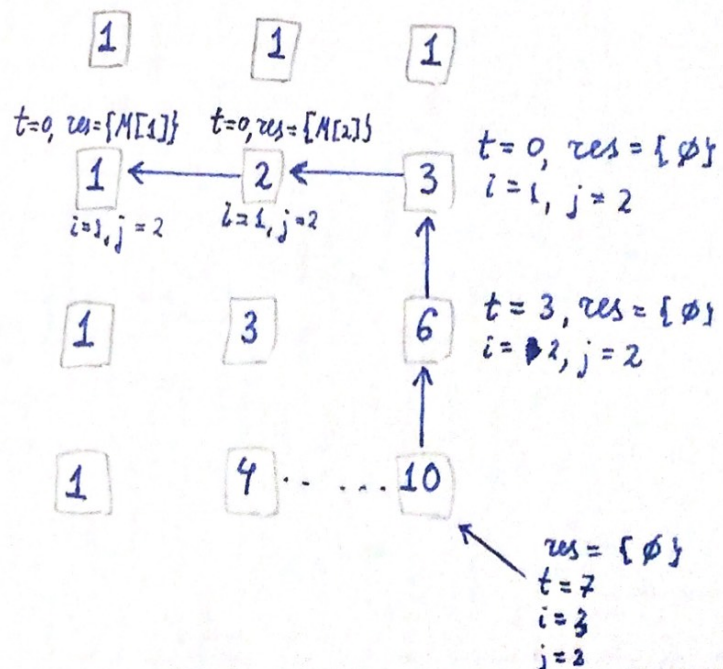
• Пример работы:

~~$n = 5, k = 3, idx = 7$~~

$n = 5$

$k = 2$

$idx = 7$



Алгоритм выбора начального приближения

Вход: $\begin{cases} A[M, N] \\ b[M] \\ c[N] \end{cases}$ - параметры задачи $c^T[N] \cdot x[N] \rightarrow \min,$
 $x \in S := \{x[N] \geq 0 \mid A[M, N] \cdot x[N] = b[M]\}$

1. Строим $\bar{A} := (A[M, N] \mid E[M, M])$, где $E[M, M]$ - единичная матрица
 $\bar{c} := (\underbrace{0 \dots 0}_n \mid \underbrace{1 \dots 1}_m)$

2. Если $b[M]$ содержит отрицательные компоненты, умножаем соответствующие строки системы $(\bar{A} \mid \bar{b})$ на -1 .

3. Строим $\bar{x}_0 := \begin{pmatrix} 0 \\ \vdots \\ 0 \\ b[M] \end{pmatrix}^n$ - опорный вектор к $\bar{S} = \{\bar{x} \geq 0 \mid \bar{A}\bar{x} = \bar{b}\}$

$$\bar{b}_0 := E[M, M]$$

4.

4а. Решаем задачу $\min_{x \in \bar{S}} \bar{c}^T \cdot \bar{x}$ симплекс-методом.

Пусть решение - \bar{x}_* , а \bar{N}_* - соответствующее множество индексов базисных столбцов \bar{A} .

$$x_* := \bar{x}_*[N], N_* := \bar{N}_* \cap N$$

4б. Если $N_* = N_+ := \{i \mid x_*[i] > 0\}$, то x_* можно взять в качестве искомого начального приближения.

4в. Если $N_* \subsetneq N_+$ ($x[N]$ - несовместимый 0.б.):

4в. I. Выделяем подмножество индексов $L := N \setminus N_+$

4в. II. Строим бинарную таблицу bd : в ~~левом~~^{правом} нижнем углу
табл $C_{|L|}^{|M|-|N_+|}$ (см. шаг (2) симплекс-метода).

4в. III. Повторяем шаг (3) алгоритма симплекс-метода, используя эту бинарную таблицу и индексы L в качестве множества индексов, которыми дополняем A_k .
Затем возвращаемся к (4б).

3.4 Алгоритм перебора опорных векторов

Опорные векторы можно искать прямо по определению, перебирая все возможные базисы и находя соответствующие ненулевые коэффициенты из решения СЛАУ. Множества входящих в базис столбцов будем определять с помощью метода *subsetByIndex*, описанной в предыдущем разделе. Также будем пользоваться процедурой *inv*, описываемой в следующем разделе, для нахождения обратной матрицы при решении СЛАУ.

Algorithm 1: Метод перебора опорных векторов решения задачи линейного программирования в канонической форме

Data: $A[M, N], b[M], c[N]$ – параметры задачи линейного программирования, поставленной в канонической форме;
 $m = |M|, n = |N|$
Result: опорный вектор $x_*[N]$, минимизирующий целевую функцию $(x[N], c[N])$

- 1 инициализация матрицы $bg[N, M]$ биномиальных коэффициентов;
- 2 $V := \emptyset$ – будущий список опорных векторов;
- 3 **for** i в диапазоне $\{0; C_m^n\}$ **do**
- 4 $N_k := \text{subsetByIndex}(i, bg)$;
- 5 **if** $\det(A[M, N_k]) \neq 0$ **then**
- 6 $x[N_k] := \text{inv}(A[M, N_k], b[M])$;
- 7 Дополняем нулями до $x[N]$;
- 8 Добавляем $x[N]$ в V ;
- 9 **end**
- 10 **end**
- 11 Выбираем x_* – любой вектор из V ;
- 12 **for** $v \in V$ **do**
- 13 **if** $(v, c) < (x_*, c)$ **then**
- 14 $x_* := v$;
- 15 **end**
- 16 **end**

3.5 Алгоритм восстановления решения прямой(двойственной) задачи по решению двойственной(прямой)

Рассмотрим пару двойственных задач, образованную основной задачей линейного программирования и двойственной к ней:

- Прямая (исходная) задача:

$$\begin{aligned} F &= (x[N], c[N]) \longrightarrow \max_{x[N], x[N] \in S} \\ S &= \{x[N] | A[M, N] \cdot x[N] = b[M], x[N] \geq 0\} \end{aligned} \quad (3)$$

- Двойственная к ней задача:

$$\begin{aligned} F_{dual} &= (y[M], b[M]) \longrightarrow \min_{y[M], y[M] \in S_{dual}} \\ S_{dual} &= \{y[M] | A^T[N, M] \cdot y[M] \leq c[N]\} \end{aligned} \quad (4)$$

Каждая из задач (3) и (4) фактически является самостоятельной задачей линейного программирования и может быть решена независимо от другой. Однако при нахождении оптимального плана одной из задач при помощи симплекс метода или метода искусственного базиса находится решение и другой задачи.

Пусть:

- x^* – найденный оптимальный план задачи (3);
- $P_{i_1}, P_{i_2}, \dots, P_{i_m}$ – базис, определяющий план;
- $C_{basis} := (c_{i_1}, c_{i_2}, \dots, c_{i_m})$ – вектор-строка, составленная из коэффициентов при неизвестных в целевой функции задачи (3);
- P^{-1} – матрица, обратная матрице P , составленной из базисных векторов.

Тогда будем находить решение прямой задачи в соответствии с теоремой [2]:

Теорема 3.1 Если основная задача линейного программирования имеет оптимальный план x^* , то $y^* = C_{basis} \cdot P^{-1}$ является оптимальным планом двойственной задачи.

Теперь, помня о том, что задачи (3) и (4) двойственны друг к другу, можем решать любую из них и находить оптимальный план для парной с затратами лишь на обращение матрицы из базисных векторов и на умножение этой матрицы на вектор C_{basis} .

4 Результаты решения задачи

Симплекс метод для задачи (1) дал нам оптимальный план $\tilde{x}_*^T = (0., -3.5, -0.5, 31.5)$.

Метод перебора крайних точек дал нам решение $\tilde{\tilde{x}}_*^T = (0., -3.5, -0.5, 31.5)$.

При этом точное решение задачи $x_*^T = (0., -\frac{7}{2}, -\frac{1}{2}, \frac{63}{2})$ - его можно получить, например, при решении в дробях системы из строки (6) алгоритма перебора крайних точек (1).

Сравнив результаты, полученные при использовании исследуемых методов, между собой и с точным решением, можем сделать вывод, что абсолютная погрешность при нашей реализации очень хорошая. Такую ситуацию имеем за счет того, что достаточно малые числа компьютер просто не хранит. Неточности, попадающие в область машинного нуля, нивелируются.

5 Оценка достоверности полученного результата

Обоснование достоверности решения:

$$x^*[N] = \left(0, -\frac{3}{2}, -\frac{1}{2}, \frac{63}{2}\right)$$

По теореме о необходимых и достаточных условиях решения задачи:

$$\exists y^*[M]$$

$$y_1^*[M_1] \geq 0$$

$$c^*[N_1] - y_1^*[M_1] \cdot A[M_1, N_1] \geq 0$$

$$c^*[N_2] - y_1^*[M_1] \cdot A[M_1, N_2] = 0$$

$$y_1^*[M_1] \cdot (A[M_1, N_1] \cdot x^*[N_1] - b[M_1]) = 0$$

$$(c^*[N_1] - y_1^*[M_1] \cdot A[M_1, N_1]) \cdot x^*[N_1] = 0$$

для задачи вида:

$$\min c^*[N] \cdot x[N], \quad x[N] \in S$$

$$S = \{x[N] \mid A[M_1, N] \cdot x[N] \geq b[M_1], \quad A[M_2, N] \cdot x[N] = b[M_2], \quad x[N_1] \geq 0\}$$

Данная задача:

$$8x_1 + 8x_2 - 4x_3 - 2x_4 \rightarrow \min$$

$$\begin{cases} x_1 - 2x_2 + 2x_3 \leq 6 \\ x_1 + 2x_2 + x_3 + x_4 = 24 \\ 2x_1 + x_2 - 4x_3 + x_4 = 30 \\ -x_1 + 4x_2 - 2x_4 \geq -6 \\ x_1 \geq 0 \end{cases}$$

тогда в наших обозначениях

$$A[M_1, N] = \begin{pmatrix} -1 & 2 & -2 & 0 \\ -1 & 4 & 0 & -2 \end{pmatrix} \quad b[M_1] = (-6, -6)$$

$$A[M_2, N] = \begin{pmatrix} 1 & 2 & 1 & 1 \\ 2 & 1 & -4 & 1 \end{pmatrix} \quad b[M_2] = (24, 30)$$

$$c^*[N_1] = (8)$$

$$x^*[N_1] = (0)$$

$$c^*[N_2] = (8, -4, -2)$$

$$x^*[N_2] = \left(-\frac{3}{2}, -\frac{1}{2}, \frac{63}{2}\right)$$

$$A[M, N_1] = \begin{pmatrix} 1 \\ 2 \\ -1 \\ -1 \end{pmatrix}$$

$$A[M, N_2] = \begin{pmatrix} 2 & 1 & 1 \\ 1 & -4 & 1 \\ 2 & -2 & 0 \\ 4 & 0 & -2 \end{pmatrix}$$

теперь будем подставлять:

$$\begin{aligned} y_1^*[M_1] (A[M_1, N] \cdot x^*[N] - b[M_1]) &= -y_1^*[M_1] \cdot b[M_1] = -y_1 \cdot (-6) - y_2 \cdot (-6) = 6y_1 + 6y_2 = 0 \Rightarrow y_1 = -y_2, \text{ но } y_1^*[M_1] \geq 0 \\ &\stackrel{0}{=} -[(y_1, y_2) \begin{pmatrix} -6 \\ -6 \end{pmatrix}] = 6y_1 + 6y_2 = 0 \Rightarrow y_1 = y_2 = 0 \end{aligned}$$

$$C^T[N_2] - y_2^T[M] \cdot A[M, N_2] = 0$$

$$y_2^T[M] A[M, N_2] = C^T[N_2]$$

$$(8 \ -4 \ -2) - (0 \ 0 \ y_3 \ y_4) \begin{pmatrix} 2 & 1 & 1 \\ 1 & -4 & 1 \\ 2 & -2 & 0 \\ 4 & 0 & -2 \end{pmatrix} = 0$$

$$(8 \ -4 \ -2) - (2y_3 + 4y_4 \ -2y_3 \ -2y_4) = 0$$

$$\begin{cases} 8 = 2y_3 + 4y_4 \\ -4 = -2y_3 \\ -2 = -2y_4 \end{cases} \Rightarrow \begin{cases} y_3 = 2 \\ y_4 = 1 \end{cases}$$

$$(C^T[N_1] - y_1^T[M] \cdot A[M, N_1]) \cdot x_1[N_1] = 0$$

$$(y_1 \ y_2 \ y_3 \ y_4 \ y_5) \begin{pmatrix} 1 \\ 2 \\ -1 \\ -1 \end{pmatrix} (0) = (0)$$

$0=0$ - верно

$$C^T[N_1] - y_1^T[M] A[M, N_1] \geq 0$$

$$8 - (0 \ 0 \ 2 \ 1) \begin{pmatrix} 1 \\ 2 \\ -1 \\ -1 \end{pmatrix} \geq 0$$

$$8 + 2 + 1 \geq 0$$

$11 \geq 0$ - верно

\Rightarrow все условия соблюдены

$$\exists y_*(M) = (0 \ 0 \ 2 \ 1)$$

Список литературы

- [1] Петухов Л. В. Методы оптимизации. Задачи выпуклого программирования: учеб. пособие / Л. В. Петухов, Г. А. Серёгин, Е. А. Родионова. – СПб.: Изд-во Политехн. ун-та, 2014. – 99 с.
- [2] Акулич И.Л. Математическое программирование в примерах и задачах: учеб. пособие. – СПб.: Изд-во 'Лань', 2011. – 352с.: ил.