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:

$$\begin{cases} x_1 - 2x_2 + 2x_3 \leq 6 \\ x_1 + 2x_2 + x_3 + x_4 = 24 \\ 2x_1 + x_2 - 4x_3 + 1 = 30 \\ -x_1 + 4x_2 + 2x_4 \geq -6 \\ x_i \geq 0 \, \forall i \in \{1; 3\} \end{cases} \tag{1}$$

$$8x_1 + 3x_2 + 4x_3 + 2x_4 \longrightarrow min$$

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 $A \in \mathbb{R}_{m \times n}, b \in \mathbb{R}_m, c \in R_n$ 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 симплекс-метода и связанные процедуры

Алгоритм симплекс-метода

1. Вход: $x_0[N]$, $A[M,N]$, $b[M]$, $c[N]$, N_k, $B[N_0,M]$

где A, b, c – параметры задачи $c^T x \to \min$, $x \in S$,
$S := \{x \geq 0 \mid Ax = b\}$

$x_0[N]$ – 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. Введём $N_k^0 := \{i \in N_k \mid x_k[i] = 0\}$ (вначале $k=0$, $x_k = x_0$, $N_k = N_0$)

$N_k^+ := \{i \in N_k \mid x_k[i] > 0\}$

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

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

3. В цикле для каждого binomIdx от 0 до $C_{|N_k^+|}^{m-|N_k^+|}$:

3а. Ищем индексы $N_k \subset N$, $L_k \subset N$: $N = N_k \cap L_k$; $N_k \supset N_k^+$; $N_k = m$;
N_k отличается от N_{k-1} только одним индексом.
Для этого запускаем процедуру subsetByIndex, положив её параметры
$bg = bg$, $i = \text{binomIdx}$, $M = N \setminus N_k^+ = N_k^0$
] Её результат $\tilde{N}_k^0 \Rightarrow N_k := N_k^+ \cup \tilde{N}_k^0$

$L_k := N \setminus N_k$

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

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

3.в.I. Положим $i_k :=$ "№ индекса в N_k, изменённый по сравнению с N_{k-1}"

3.в.II. Строим

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

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

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

3.е. $\angle\, j_k :=$ " первый индекс из L_k : $d_k[j_k] < 0$

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

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

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

3.з.I. $\theta_k := \min\limits_{i \in N_k,\, u_k[i] > 0} \dfrac{x_k[i]}{u_k[i]}$

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

Вспомогательные процедуры

- Subset By Index:

1. Вход: таблица биномиальных коэффициентов $bg = \underbrace{\begin{bmatrix} 1 & \cdots & 1 \\ \vdots & (bg_{ij}) & \\ 1 & & \end{bmatrix}}_{k+1} \Big\} n-k+1$, $bg_{ij} = bg_{i-1\,j} + bg_{i\,j-1}$, $n \geq k$

   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

   $t := idx$
   $i := n - k$
   $j := k$
   $res := \{\varnothing\}$

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:

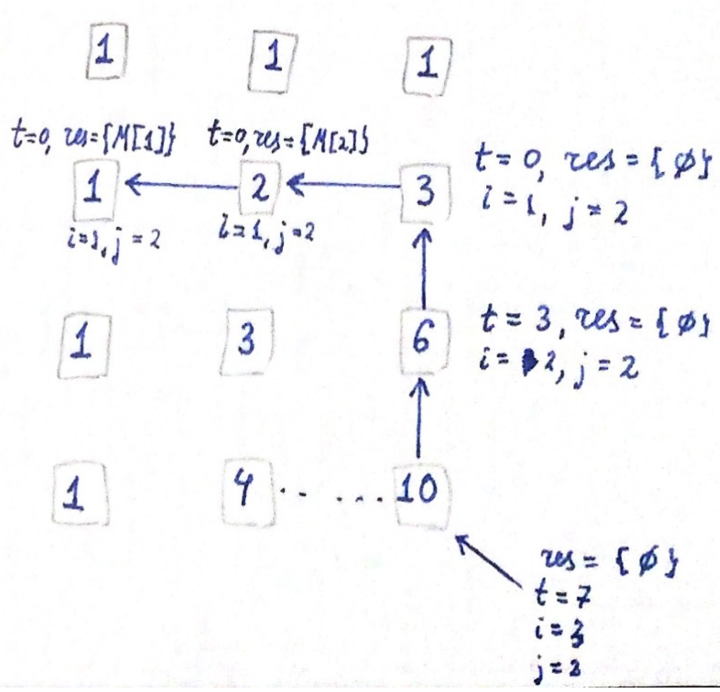
   3.б.1. $t := t - bg[i, j-1]$
   $t := t - 1$

4. Возвращаем \`res\`

- Пример работы:

~~$n = 5$, $k = 2$, $idx = 7$~~

$n = 5$
$k = 2$
$idx = 7$



Алгоритм выбора начального приближения

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

1. Строим $\bar{A} := (A[M,N] \vdots E[M,M])$, где $E[M,M]$ - идентичная матрица

$\bar{c} := (\underbrace{0 \ldots 0}_{n} \; \underbrace{1 \ldots 1}_{m})$

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

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

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

4.

4а. Решаем задачу $\min\limits_{x \in \bar{S}} \bar{c}^T \bar{x}$ 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 $N_* \supset N_+$ ($x[N]$- несобственный о.в.):

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

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

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

Затем возвращаемся к (4б).

3.2 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 $(x[N], c[N])$
инициализация матрицы $bg[N, M]$ биномиальных коэффициентов;
$V := \emptyset$ – будущий список опорных векторов;
for i *в диапазоне* $\{0; C_m^n\}$ **do**
    $N_k :=$ subsetByIndex(i,bg);
    if $det(A[M, N_k]) \neq 0$ **then**
        $x[N_k] :=$ inv($A[M, N_k], b[M]$);
        Дополняем нулями до $x[N]$;
        Добавляем $x[N]$ в V;
    end
end
Выбираем x_* – любой вектор из V;
for $v \in V$ **do**
    if $(v, c) < (x_*, c)$ **then**
        $x_* := v$;
    end
end

3.3 Алгоритм нахождения обратной матрицы

4 Результаты решения задачи

5 Оценка достоверности полученного результата

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.