Преподаватель: Власов Павел Александрович (ищи его в 1025л)

E-mail: pvlx@mail.ru

Лабораторныеработы: Пн, 19:10

Лекция 1

7 сентября 2015

1 Задачи методов оптимизации

1.1 Классификация задач оптимизации

Задача оптимизации имеет обычно имеет следующий вид:

$$\begin{cases} f(x) \to extr \\ x \in G \end{cases},$$

здесь f называется целевой функцией или критерием оптимальности, G называется множеством допустимых решений.

Замечание:

1. Задача min-целевой функции одной переменной:
$$\begin{cases} f(x) \to min \\ x \in [a,b] \subseteq \mathbb{R} \end{cases}$$
2. Задача безусловной оптимизации функции нескольких переменных:
$$\begin{cases} f(x) \to min \\ x \in G \subseteq \mathbb{R}^n \end{cases}$$
3. Задача условной оптимизации функции нескольких переменных:
$$\begin{cases} f(x) \to min \\ \phi(x) = 0 \\ x \in G \subseteq \mathbb{R}^n \end{cases}$$

1. Если $f : G \to \mathbb{R}$ - скалярная функция, то соответсвующая задача называется задачей математическим программированием
2. Если $f : G \to \mathbb{R}^m, m \geq 2$, то соответсвующая задача называется задачей многокретириальной оптимизации

Для задач математического программирования программирования дальнейшая классификация:

Вид функции	Стр. множества G	Название задачи
Линейная	Выпуклый многоугольник в $\mathbb{R}^n$	Задача линейного программирования
Квадратичная	Выпуклый многоугольник в $\mathbb{R}^n$	Задача квадратичного программирования
Выпуклая	Выпуклое множеств в $\mathbb{R}^n$	Задача выпуклого программирования
Произвольная	Конечное множество	Задача дискретного программирования
Произвольная	Подмножество множества $\mathbb{Z}^n$	Задача целочисленного программирования
Произвольная	Подмножество в $\{0, 1\}^n$	Задача логического программирования

Замечание:

1. В этом семестре будем заниматься задачами линейного и целочисленного программирования. Задачами выпуклого и квадратичного программирования будем заниматься в следующем семестре.

2. Как правило задачи оптимизации, в которых мн-во G конечно или счетно, относят к разделу методов оптимизации, которые называется *исследованием операций.*

1.2 Венгерский метод решения задач о назначениях

1.2.1 Постановка задачи о назначениях

Содержательная постановка.

В распоряжении работодателя имеется n работ и n исполнителей. Стоимость выполнения i-ой работы j-ым исполнителем составляет $c_{ij} \geqslant 0$ единиц. Необходимо распределить все работы между исполнителями таким образом, чтобы:

1. Каждый исполнитель исполнял ровно одну работу

2. Суммарная стоимость выполнения всех работ была бы минимальной

Введем управляемые переменные:

$$x_{ij} = \begin{cases} 1, & \text{если } i-\text{ую работу исполняет } j-\text{ый исполнитель} \\ 0, & \text{иначе} \end{cases}$$

Замечание:

Переменную x_{ij}, $i, j = \overline{1, n}$, удобно записывать в матрицу:

$$x = \begin{bmatrix} x_{11} & \dots & x_{1n} \\ \vdots & & \vdots \\ x_{n1} & \dots & x_{nn} \end{bmatrix},$$

которая называется матрицей назначений.

Стоимости c_{ij} записывают в матрицу,

$$C = \begin{bmatrix} C_{11} & \dots & C_{1n} \\ \vdots & & \vdots \\ C_{n1} & \dots & C_{nn} \end{bmatrix},$$

которая называется матрицей стоимостей.

Тогда целевая функция

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

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

Условие того, что j-ый исполнитель выполняет ровно одну работу

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

Условие того, что i-ую работу выполняет ровно один исполнитель

Математическая постановка задачи о назначении:

$$\begin{cases} f(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \to min \\ \sum_{i=1}^{n} x_{ij} = 1, \qquad j = \overline{1,n} \\ \sum_{j=1}^{n} x_{ij} = 1, \qquad i = \overline{1,n} \\ x_{ij} \in \{0,1\}, \qquad i,j = \overline{1,n} \end{cases}$$

Замечание:

Множество допустимых решений задачи о назначении является конечным и состоит из $n!$ элементов. Одним из возможных методов решения этой задачи является прямой перебор допустимых решений однако при больших n он практически не реализуем ввиду большой сложности.

1.2.2 Предварительные соображение о методе решений задачи о назначении:

Соображение 1

Выполним над элементами матрицы стоимостей C следующие преобразования:

1. Из всех элементов $j-$ого столбца вычтем некоторое число $\alpha_j, \qquad j = \overline{1,n}$
2. Из всех элементов $i-$ой строки вычтем некоторое число $\beta_j, \qquad i = \overline{1,n}$

Обозначим полученную матрицу $\widetilde{C}$

$$f_{\widetilde{C}}(x) = \sum_{i=1}^{n}\sum_{j=1}^{n}\tilde{c}_{ij}x_{ij} = \sum_{i=1}^{n}\sum_{j=1}^{n}(c_{ij} - \alpha_j - \beta_i)x_{ij} = \sum_{i=1}^{n}\sum_{j=1}^{n}c_{ij}x_{ij} - \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_j x_{ij} - \sum_{i=1}^{n}\sum_{j=1}^{n}\beta_i x_{ij} = f_c(x) - \gamma$$

$$\gamma = -\sum_{j=1}^{n}\alpha_j - \sum_{i=1}^{n}\beta_i$$

В результате:

$$f_{\tilde{c}}(x) = f_c(x) + const$$

То есть задачи о назначениях с матрицей C и с матрицей $\tilde{C}$ эквивалентно (то есть оптимальные значение функций f_c и $f_{\tilde{c}}$ достигаются при одном и том же x_{ij}.

Соображение 2

Предположим, что в матрице C нашлись n нулей, никакие два их которых не стоят на одной строке и одном столбце.

В этом случае можно сразу записать решение рассматриваемой задачи в матрице x, ставим единицы в тех позициях, в которых в матрице C стоят нули. Остальные элементы матрицы x полагаем равными 0.

$$f(x) = \sum_{i,j} c_{ij}x_{ij} \geqslant 0$$

$$f(x_{opt}) = 0$$

Определение: Набор нулей матрицы C будем называть *системой независимых нулей* (СНН), если никакие 2 нуля этой системы не стоят в одной строке и одном столбце

Замечание: Для решения задачи о назначениях достаточно преобразовать матрицу C к эквивалентному виду, в котором будет содержаться система из n независимых 0.

Замечание: “Слабые места”

1. Возможно после выполнения указанных преобразований мы получим матрицу, из нулей которой в принципе невозможно сформировать систему независимых нулей.

$$\begin{bmatrix} 3 & 1 & 5 & 7 \\ 1 & 1 & 3 & 4 \\ 6 & 1 & 5 & 5 \\ 7 & 1 & 9 & 8 \end{bmatrix} \to \begin{bmatrix} 2 & 0 & 2 & 3 \\ 0 & 0 & 0 & 0 \\ 5 & 0 & 2 & 1 \\ 6 & 0 & 6 & 4 \end{bmatrix}$$

   В этой матрице максимальное число в системе независимых нулей равно 2

2. Возможно в некоторой текущей матрице стоимостей существует набор независимых нулей. Но его построение затруднительно.

В любом из этих случаев построенная система независимых нулей (в которой меньше чем n нулей) может быть улучшена.

Будем считать, что после вычитания наименьших элементов из строк и столбцов матрицы C первоначальная система независимых нулей строится по следующему правилу: просматриваем элементы матрицы C по столбцам в поисках нулей. Если в одном столбце и одной строке с найденным нулем не стоит 0^*, то отмечаем найденые нули звездочкой.

1.2.3 1-ый способ улучшения текущей СНН

Идея: убрать из СНН несколько нулей так, чтобы добавить большее число нулей.

Отметим «+» столбцы, в которых стоят 0^*. Эти столбцы и их элементы будем называть *выделенными*.

Если среди невыделенных элементов есть 0, то можно попытаться улучшить текущую СНН, включив в нее этот 0. Отметим этот 0 штрихом.

В этом случае строка, в которой располагается этот 0, уже не может содержать других элементов СНН. Поэтому если в одной строке с $0'$ есть 0^*, то необходимо снять выделение со столбца, в котором стоит 0^* и выделить строку, в которой стоит $0'$.

Снова среди невыделенных элементов ищем 0 и отмечаем его штрихом. В одной строке с этим $0'$ нет 0^*. Это значит, что можно построить L-цепочку:

от текущего $0' \to_{\text{по стобцу}} 0^* \to_{\text{по строке}} 0' \to 0^* \to \ldots \to 0'$

L-цепочка должна быть непродолжаемой и начинаться и заканчиваться $0'$.

В пределах этой L-цепочки 0^* заменяем на просто 0, а $0'$ на 0^*.

Лекция 2

14 сентября 2015

Слабое место 1

В текущей матрице C не существуют СНН из n нулей тогда и только тогда, когда среди невыделенных элементов нет 0.

$$C = \begin{bmatrix} 0^{*+} & 0 & 1 & 4 \\ 1 & 2 & 0^{*+} & 1 \\ 2 & 2 & 5 & 0^{*+} \\ 3 & 1 & 0 & 5 \end{bmatrix} \to \begin{bmatrix} 0^* & 0' & 1 & 4 \\ 1 & 2 & 0^* & 1 \\ 2 & 2 & 5 & 0^* \\ 3 & 1 & 0 & 5 \end{bmatrix} \to$$

Выберем среди невыделенных элементов наименьший элемент $h > 0$. В этом примере $h = 1$. Вычтем его из невыделенных столбцов

$$\to \begin{bmatrix} -1^* & -1 & 1 & 4 \\ 0 & 1 & 0^* & 1 \\ 1 & 1 & 5 & 0^* \\ 2 & 0 & 0 & 5 \end{bmatrix} \to$$

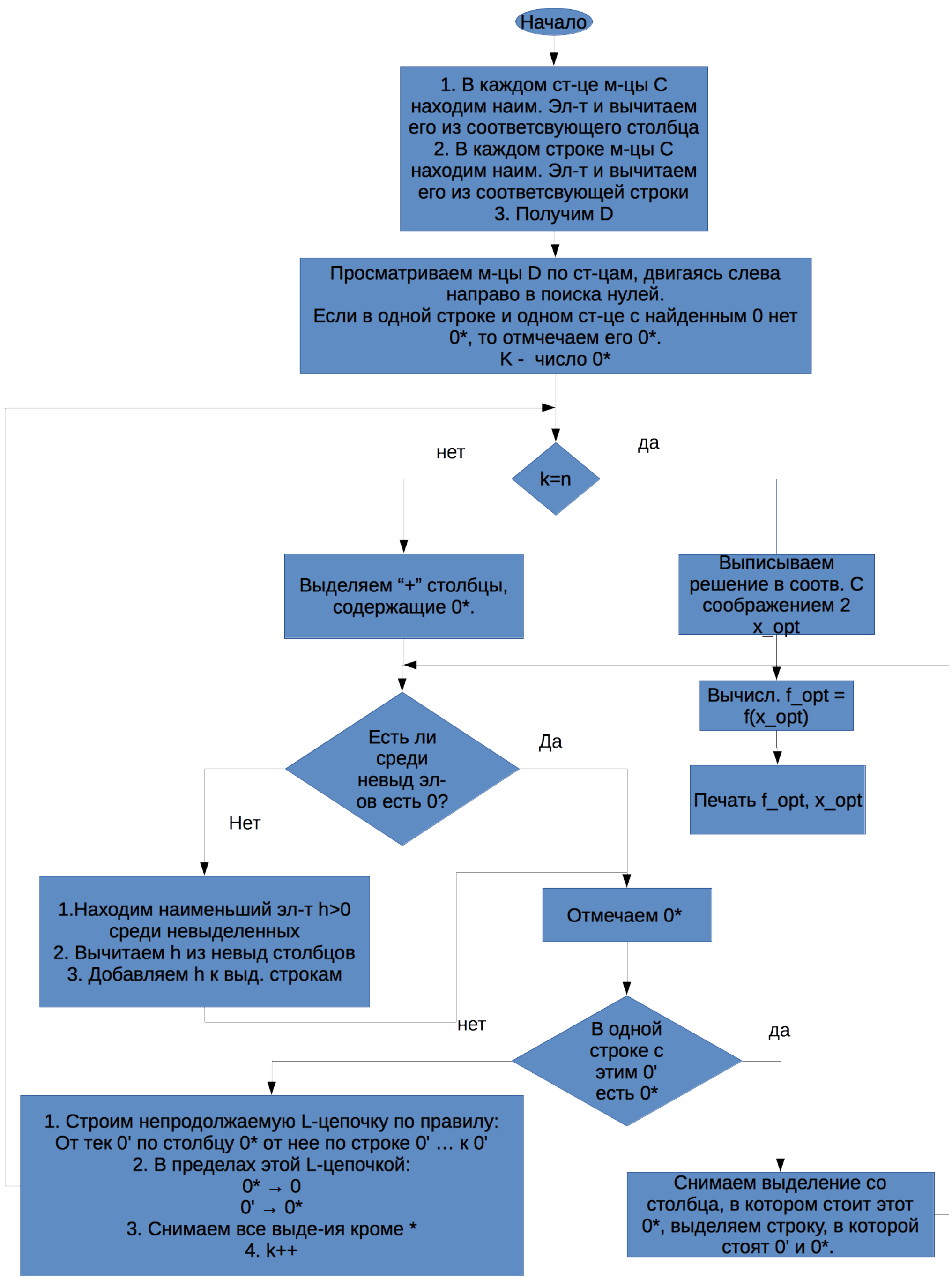
Наличие отрицательных значений не позволяет использовать пункт 2. Прибавим h к выделенным строкам:

$$\rightarrow \begin{bmatrix} 0^* & 0' & 2 & 5 \\ 0 & 1 & 0^* & 1 \\ 1 & 1 & 5 & 0^* \\ 2 & 0 & 0 & 5 \end{bmatrix} \rightarrow \begin{bmatrix} 0^* & 0' & 2 & 5 \\ 0' & 1 & 0^* & 1 \\ 1 & 1 & 5 & 0^* \\ 2 & 0 & 0 & 5 \end{bmatrix} \rightarrow$$

В одной строке с нулем нет 0^*. Строим L-цепочку.

$$\rightarrow \begin{bmatrix} 0 & 0^* & 2 & 5 \\ 0^* & 1 & 0 & 1 \\ 1 & 1 & 5 & 0^* \\ 2 & 0 & 0^* & 5 \end{bmatrix} \Rightarrow x_{opt} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

1.3 Венгерский метод решения задачи о назначениях



Пример:

$$C=\begin{bmatrix}7&2&8&2&5\\1&2&3&1&2\\7&2&5&3&6\\2&2&12&3&6\\6&7&11&1&9\end{bmatrix}\to\begin{bmatrix}6&0&5&1&3\\0&0&0&0&0\\6&0&2&2&4\\5&0&9&2&4\\3&5&8&0&7\end{bmatrix}\to\begin{bmatrix}6&0^*&5&1&3\\0^*&0&0&0&0\\6&0&2&2&4\\5&0&9&2&4\\3&5&8&0^*&7\end{bmatrix}\to k=3<5=h\to$$

$$\to\begin{bmatrix}6&0^*&5&1&3\\0^*&0&0&0&0\\6&0&2&2&4\\5&0&9&2&4\\3&5&8&0^*&7\end{bmatrix}\to\begin{bmatrix}6&0^*&5&1&3\\0^*&0&0&0&0\\6&0&2&2&4\\5&0&9&2&4\\3&5&8&0^*&7\end{bmatrix}\to$$

Среди невыделеных элемнтов нет 0 тогда h = 2

$$\to\begin{bmatrix}4&0^*&3&1&1\\0^*&2&0'&2&0\\4&0&0'&2&2\\3&0&7&2&2\\1&5&6&0^*&5\end{bmatrix}\to\text{Делаем L-цепочку}\to\begin{bmatrix}4&0^*&3&1&1\\0^*&2&0&2&0\\4&0&0'&2&2\\3&0&7&2&2\\1&5&6&0^*&5\end{bmatrix}\to k=4\to$$

$$\to\begin{bmatrix}4&0^*&3&1&1\\0^*&2&0&2&0'\\4&0&0'&2&2\\3&0&7&2&2\\1&5&6&0^*&5\end{bmatrix}\to\begin{bmatrix}4&0^*&3&1&1\\0^*&2&0&2&0\\4&0&0'&2&2\\3&0&7&2&2\\1&5&6&0^*&5\end{bmatrix}\to\begin{bmatrix}4&0^*&3&1&1\\0^*&2&0&2&0'\\4&0&0'&2&2\\3&0&7&2&2\\1&5&6&0^*&5\end{bmatrix}\to k=1\to$$

$$\to\begin{bmatrix}3&0^*&3&1&0'\\0^*&3&1&3&0'\\3&0&0'&2&1\\2&0&7&2&1\\0&5&6&0^*&4\end{bmatrix}\to\begin{bmatrix}3&0^*&3&1&0'\\0^*&3&1&3&0'\\3&0&0'&2&1\\2&0&7&2&1\\0&5&6&0^*&4\end{bmatrix}\to\begin{bmatrix}3&0^*&3&1&0'\\0^*&3&1&3&0'\\3&0&0^*&2&1\\2&0&7&2&1\\0'&5&6&0^*&4\end{bmatrix}\to$$

$$\to\begin{bmatrix}3&0^*&3&1&0'\\0^*&3&1&3&0'\\3&0'&0^*&2&1\\2&0&7&2&1\\0'&5&6&0^*&4\end{bmatrix}\to\text{Строим L-цепочку}\to\begin{bmatrix}3&0&3&1&0^*\\0^*&3&1&3&0\\3&0&0^*&3&1\\2&0^*&7&2&1\\0&5&6&0^*&4\end{bmatrix}\Rightarrow$$

$$\Rightarrow x_{opt}=\begin{bmatrix}0&0&0&0&1\\1&0&0&0&0\\0&0&1&0&0\\0&1&0&0&0\\0&0&0&1&0\end{bmatrix}$$

$$f_{opt}=5+1+5+2+1=14$$

<u>Замечание:</u>

1. Встречается среди задач о назначениях, в которых величины c_{ij} интерпретируются как прибыль от выполнения $i-$ой работы $j-$ым работником. В этом случае задача о назначениях является задачей максимизации:

$$\begin{cases} f=\sum_{i=1}^{n}\sum_{j=1}^{n}c_{ij}x_{ij}\to max\\ \text{те же ограничения}\end{cases}\tag{1}$$

2. В этом случае исходной задаче (1) будет эквивалетна задача:

$$\begin{cases} f_l=f=\sum_{i=1}^{n}\sum_{j=1}^{n}(-c_{ij})\,x_{ij}\to min\\ \text{те же ограничения}\end{cases}\tag{2}$$

которая не может быть решена с использованием венгерского метода ввиду неположительных $-c_{ij}$.

3. Выберем среди c_{ij} наиб. элементы a и прибавим его всем элементам м-це $(-c)$:

$$-\tilde{c_{ij}} = a - c_{ij}$$

Тогда задача

$$\begin{cases} f = \sum_{i=1}^{n} \sum_{j=1}^{n} \tilde{c_{ij}} x_{ij} \to min \\ \text{те же ограничения} \end{cases}$$

эквивалентна задаче (2) и очевидно задаче (1) и может быть решена венгерским методом.

2 Линейное программирование

Задача:

$$\begin{cases} f = \sum_{j=1}^{n} c_j x_j \to extr \\ \sum_{j=1}^{n} a_{ij} x_j \{\geqslant / \leqslant / =\} b_i \quad i = \overline{1; m} \end{cases}$$

2.1 Основные определения

2.1.1 Задача распределительного типа

Задача распределительного типа является характерной задачей линейного программирования

Пример: Кондитерская фабрика выпускает 2 вида карамели (обозначим их А и Б), да производства которых используются сахарный песок и фруктовое пюре. Данные о затратах:

	Нормы затрат		Общий объем запасов (т)
	А	Б	
Сахарный песок	1	3	13
Фруктовое пюре	3	2	18
Стоимость изготовления 1т. готового продукта	1	2	

Кроме того известно, что рыночный спрос на карамель Б-вида не превышает 4т.

Составить оптимальный план производства, максимизирующий доходы от продаж.

Введем управляемые переменные:

x_1- объем производства карамели А (т)

x_2- объем производства карамели Б (т)

Тогда:

1. Доход от продажи:
$$f = 1 \cdot x_1 + 2 \cdot x_2$$
2. Ограничение на запасы:
   (a) песка:
   $$1 \cdot x_1 + 3 \cdot x_2 \leqslant 13$$
   (b) пюре:
   $$3 \cdot x_1 + 2 \cdot x_2 \leqslant 18$$
   (c) ограничение на рыночный спрос:
   $$x_2 \leqslant 4$$

Математическая модель:

$$\begin{cases} f = 1 \cdot x_1 + 2 \cdot x_2 \\ 1 \cdot x_1 + 3 \cdot x_2 \leqslant 13 \\ 3 \cdot x_1 + 2 \cdot x_2 \leqslant 18 \\ x_2 \leqslant 4 \\ x_1 \geqslant 0, x_2 \geqslant 0 \end{cases}$$

Лекция 3

21 сентября 2015

2.2 Графический метод

Графический метод может быть использован для решения ЗЛП малой размерности ($n \leqslant 2$)

Пример: Рассмаотрим задачу о производстве карамели

$f = x_1 + 2 \cdot x_2 = 0$

Рис 1.

Градиент функции

$$\text{grad} f = \frac{\partial f}{\partial x_1} \overrightarrow{i} + \frac{\partial f}{\partial x_2} \overrightarrow{j} = \overrightarrow{i} + 2 \cdot \overrightarrow{j}$$

Нужно решить систему:

$$C = (1) \cap (2) \Rightarrow \begin{cases} x_1 + 3x_2 = 13 \\ 3x_1 + 2x_2 = 18 \end{cases} \Rightarrow x_1 = 4;\ x_2 = 3$$

Замечание: На примере этой задачи можно уловить некоторые особенности присущие всем задача линейного программирования.

1. Множество допустимых решений является выпуклым.
2. Оптимальное значение целевой функции достигается в крайней точке множества допустимых решений
3. Если оптимальное значение целевой функции достигается в двух различных точках p, q, то оно достигается во всех точках отрезка $[p, q]$.

2.3 Стандартная форма ЗЛП

Общая форма ЗЛП:

$$\begin{cases} f = \sum_{j=1}^{n} c_j x_j \to extr \\ \sum_{j=1}^{n} a_{ij} x_j \{\geqslant / \leqslant / =\} b_i \quad i = \overline{1; m} \end{cases}$$

Определение: *Стандартной формой ЗЛП* называется задача следующего вида:

$$\begin{cases} f = \sum_{j=1}^{n} c_j x_j \to max \\ \sum_{j=1}^{n} a_{i,j} x_j = b_i \\ x_j \geqslant 0, \quad j = \overline{1, n} \end{cases}$$

где $b_i \geqslant 0, i = \overline{1, m}$.

Признаки стандартной формы

1. Целевая функция максимизируется
2. Все ограничения имеют вид ? с нестандартными правыми частями
3. Все переменные больше 0

Любая ЗЛП может быть приведена к стандартной форме:

1. Если $f \to min$ то рассм задачу с функцией
$$f_1 = -f = \sum_{j=1}^{n}(-c_j x_j) -> max$$
2. Если некоторое ограничения имеет вид
$$\sum_{j=1}^{n} b_{i,j} x_j = b_{i,j},$$
где $b_i < 0$, то умножим обе его части на (-1):
$$\sum_{j=1}^{n} (-b_{i,j})\, x_j = -b_{i,j}$$
3. Если некоторое ограничение имеет вид неравенства:
$$\sum_{j=1}^{n} a_{i,j} x_j \leqslant b_i,$$
то введем дополнительную переменную $x_{n+1} \geqslant 0$
$$\sum_{j=1}^{n} a_{i,j} x_j + x_{n+1} = b_i$$
4. Если некоторое ограничение имеет вид неравенства:
$$\sum_{j=1}^{n} a_{i,j} x_j \geqslant b_i,$$
то введем дополнительную переменную $x_{n+1} \geqslant 0$:
$$\sum_{j=1}^{n} a_{i,j} x_j - x_{n+1} = b_i$$
5. Если некоторая переменная x_j подчинена условию неполжительности то есть $x \leqslant 0$, то рассмотрим вместо нее переменную $-x_j$
6. Если некоторая переменная x_j не подчинена ни условию $x_j \geqslant 0$ ни устовию $x_j \leqslant 0$ те не ограничена в знаке то представим ее в виде разности:
$$x_j = x'_j - x''_j,$$
$$x'_j \geqslant 0$$
$$x''_j \geqslant 0$$

Пример:

$$\begin{cases} f = x_1 + 2x_2 - 3x_3 -> min \\ 2x_1 + x_2 - x_3 \geqslant -3 \\ x_1 - x_2 + x_3 \geqslant 4 \\ x_1 - 2x_2 + 5x_3 = -1 \\ x_1 >= 0 \\ x_2 <= 0 \\ x_3 \text{ не ограничена в знаке} \end{cases}$$

$$\begin{cases} f = -x_1 - 2x_2 + 3x_3 - > min \\ 2x_1 + x_2 - x_3 - x_4 = -3 \\ x_1 - x_2 + x_3 - x_5 = 4 \\ -x_1 + 2x_2 - 5x_3 = -1 \\ x_1 >= 0 \\ x_2 >= 0 (x_2^{'} = -x_2) \\ x_3 = x_3' - x_3'' (x_3' >= 0, x_3'' >= 0 \\ x_4 >= 0 \\ x_5 >= 0 \end{cases}$$

$$\begin{cases} f = -x_1 + 2x_2' + 3x_3' - 3x_3'' \to min \\ -2x_1 + x_2' + x_3' - x_3'' + x_4 = 3 \\ x_1 - x_2' + x_3' - x_3'' - x_5 = 4 \\ -x_1 + 2x_2 - 5x_3 = -1 \\ x_1 \geqslant 0 \\ x_2' \geqslant 0, \ (x_2^{'} = -x_2) \\ x_3' \geqslant 0, x_3'' \geqslant 0 \\ x_4 \geqslant 0, x_5 \geqslant 0 \\ ? \end{cases}$$

Рассмотрим ЗЛП в ст. р

$$\begin{cases} f = \sum_{j=1}^{n} c_j x_j \to max \\ \sum_{j=1}^{n} a_{i,j} x_j = b_i, i = \overline{1,m} \\ x_j >= 0, j = \overline{1,n} \end{cases}$$

Введем обозначения:

$$A = (a_{i,j})_{i=\overline{1;m}}$$

$$b = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$$

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

$$C = \begin{pmatrix} C_1 \\ \vdots \\ C_n \end{pmatrix}$$

$$\begin{cases} f = c^T x \to max \\ Ax = b \geqslant 0 \\ x > 0 \end{cases}$$

Замечание: Всюду в дальнейшим будем предполагать, что

1. Ранг матрицы $A = \text{rg}(A|b)$ - по теореме Кронекера-Капелли. Если условие не выполняется, то СЛАУ $Ax = b$ не совместна, а это значит, что множество дополнительных решений пусто.
2. $\text{rg}(A|b) = m$, где m-число ограничений. m-число строк в $(A|b) \Rightarrow \text{rg}(A|b) \leqslant$ m. Если $\text{rg}(A|b) <$ m, то система ограничений является линейнозависимой то есть некоторые ограничения можно убрать не изменив множество допустимых решений

3. $m < n$, где n - число переменных. Если $m > n$, то $\text{rg}(A|b) > n$ - число столбцов. Если m = n в силу пунктов 1 и 2 СЛАУ $Ax = b$ имеет единственное решение, следовательно задачи оптимизации вырождения.

2.4 О применимости графического метода

Графический метод может быть применен в случае, когда $n \leqslant 2$. Иногда его можно использовать и в случае $n > 2$.

Рассмотрим ЗЛП в стандартной форме:

$$\begin{cases} f = c^T x \to max \\ Ax = b \geqslant 0 \\ x \geqslant 0 \end{cases}$$

Если $m - n \leqslant 2$, то можно использовать графический метод.

Пример:

$$\begin{cases} f = x_1 - 2x_2 + x_3 \to max \\ x_1 + x_2 + x_3 + x_4 = 1 \\ x_1 + 2x_2 - x_3 - x_5 = 2 \\ -x_1 + x_2 + x_3 = 3 \\ x_1, \ldots, x_3 \geqslant 0 \end{cases}$$

$$(A|b) = \left[\begin{array}{ccccc|c} 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 2 & -1 & 0 & -1 & 2 \\ -1 & 1 & -4 & 0 & 0 & 3 \end{array}\right] \sim$$

Если $\text{rg}(A|b) = 3$, то $h - m = 2$, тогда можно использовать

$$\sim \left[\begin{array}{ccccc|c} 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & -2 & -1 & -1 & 1 \\ 0 & 2 & 3 & 1 & 0 & 4 \end{array}\right] \sim \left[\begin{array}{ccccc|c} 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & -2 & -1 & -1 & 1 \\ 0 & 0 & 1 & 3 & 2 & 2 \end{array}\right] \sim$$

$$rg(A|b) = 3$$

$$\sim \left[\begin{array}{ccccc|c} 1 & 1 & 1 & 0 & -2 & -1 \\ 0 & 1 & 0 & 5 & 3 & 5 \\ 0 & 0 & 1 & 3 & 2 & 2 \end{array}\right] \sim \left[\begin{array}{ccccc|c} 1 & 0 & 0 & -7 & -5 & -6 \\ 0 & 1 & 0 & 5 & 3 & 5 \\ 0 & 0 & 1 & 3 & 2 & 2 \end{array}\right]$$

$$\begin{cases} x_1 = -6 + 7x_4 + 5x_5 & \geqslant 0 \\ x_2 = 5 - 5x_4 - 3x_5 & \geqslant 0 \\ x_3 = 2 - 3x_4 - 2x_5 & \geqslant 0 \end{cases}$$

Подставим в выражение для f:

$$\begin{cases} f = 14x_4 + 2x_5 - 14 \\ 7x_4 + 5x_5 \geqslant 6 \\ 3x_4 + 5x_5 \leqslant 2 \\ x_4 >= 0, x_5 \geqslant 0 \\ 3x_4 + 2x_5 \leqslant 2 \end{cases}$$

3 Основные утверждения линейного программирования

3.1 Выпуклые множества

Пусть $x, y \in \mathbb{R}^n$

Определение: Отрезком соединяющим точки x и y называется множество точек:

$$t(\lambda) = \lambda x + (1 - \lambda) y, \qquad \lambda \in [0; 1]$$

Замечание:

$$\lambda = 0 \Rightarrow t(0) = y$$
$$\lambda = 1 \Rightarrow t(1) = x$$

Определение: Множество $G \in \mathbb{R}^n$ называется выпуклым, если $\forall\forall x, y \in G:\ xy \subseteq G$

рис 2

Определение: Точка $a \in G$ выпуклого множества G называется крайней точкой этого множества, если a не содержится строго внутри никакого отрезка целиком лежащего в G.

рис 3

Определение: Выпуклой комбинацией точек $q_1, \ldots q_k \in \mathbb{R}^n$ называется множество точек:

$$\langle q_{1,} \ldots q_k \rangle = \left\{ t \in \mathbb{R}^n : t = \lambda_1 q_1 + \cdots + \lambda_k q_k,\ \sum_{j=1}^{k} \lambda_j = 1, \lambda_j \geqslant 0 \right\}$$

где

Замечание:

1. Очевидно отрезок
2. $$xy = \langle x, y \rangle$$
3. Можно доказать что $\langle q_1, \ldots, q_k \rangle$ является наименьшим выпуклым множестов содержащим $q_1, \ldots, q_k$

Теорема: Пусть

1. G- выпукло
2. G- ограничено
3. G имеет конечное число кр. точек $q_1, \ldots, q_k$

Тогда $G = \langle q_1, \ldots, q_k \rangle$

Конец теоремы