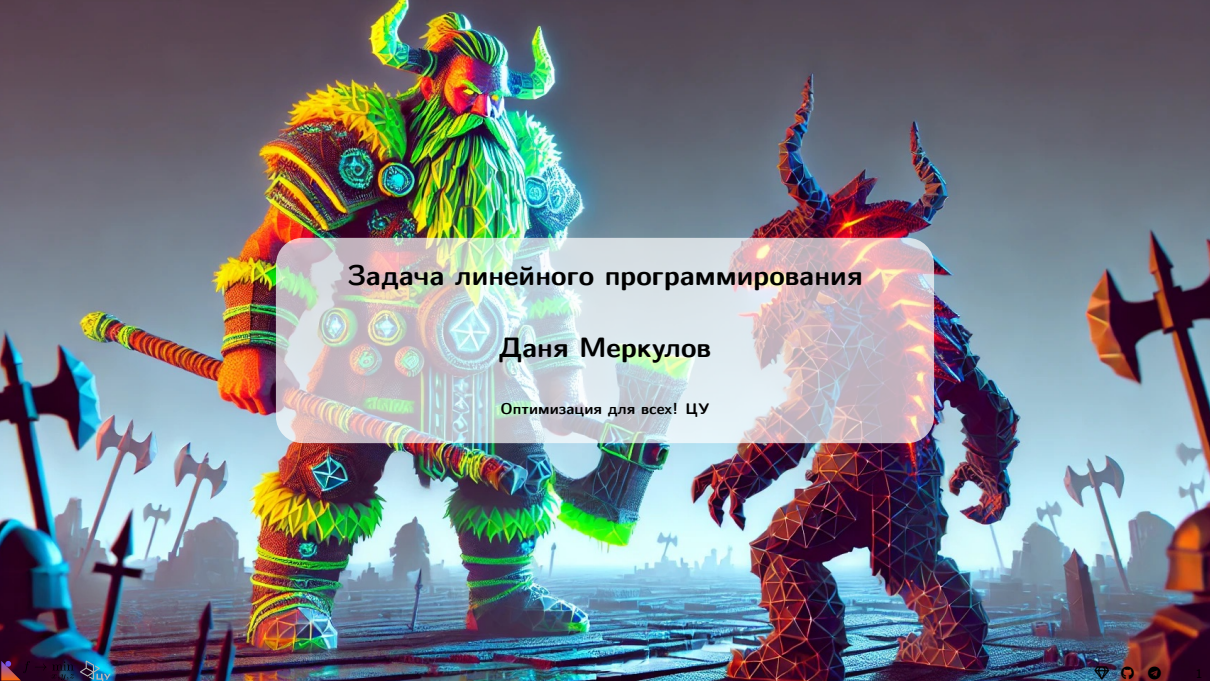


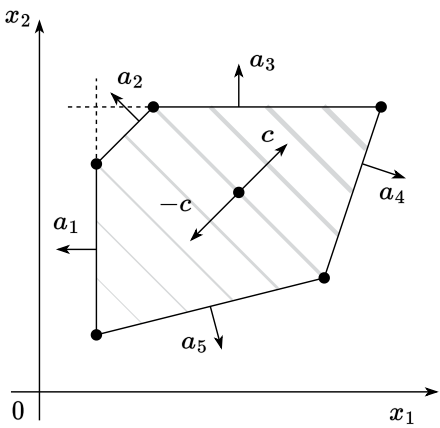
Задача линейного программирования

Даня Меркулов

Оптимизация для всех! ЦУ

Примеры задач линейного программирования

Что такое линейное программирование?

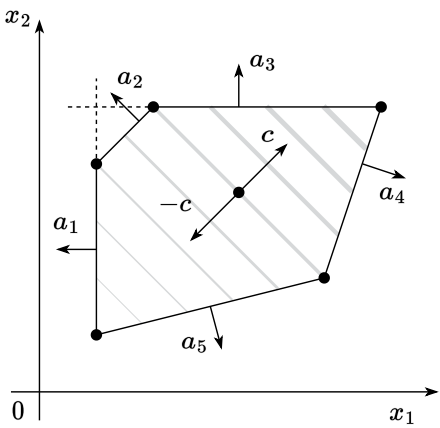


В общем случае все задачи с линейной целевой функцией и линейными ограничениями-неравенствами можно считать задачами линейного программирования. Однако существует несколько стандартных формулировок.

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \le b \qquad \text{(LP.Basic)}$$

для некоторых векторов $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ и матрицы $A \in \mathbb{R}^{m \times n}$. Где неравенства интерпретируются покомпонентно.

Что такое линейное программирование?



В общем случае все задачи с линейной целевой функцией и линейными ограничениями-неравенствами можно считать задачами линейного программирования. Однако существует несколько стандартных формулировок.

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} c^\top x \\ &\text{s.t. } Ax \le b \end{aligned} \tag{LP.Basic}$$

для некоторых векторов $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ и матрицы $A \in \mathbb{R}^{m \times n}$. Где неравенства интерпретируются покомпонентно.

Стандартная форма. Эта форма кажется наиболее интуитивной и геометрической в плане визуализации. Пусть заданы векторы $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ и матрица $A \in \mathbb{R}^{m \times n}$.

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} c^\top x \\ &\text{s.t. } Ax = b \\ &\quad x_i \ge 0,\ i = 1, \dots, n \end{aligned} \tag{LP.Standard}$$

Пример: задача о диете

Белки

Жиры

Углеводы

Калории

Витамин D

$$\left[\quad \text{Содержание на } 100\text{г} \quad W \in \mathbb{R}^{n \times p} \quad\right]$$

$c \in \mathbb{R}^p$, цена за 100г

$r \in \mathbb{R}^n$, нормы потребления

$x \in \mathbb{R}^p$, количество продуктов, 100г

$$\min_{x \in \mathbb{R}^p} c^T x$$

$$Wx \succeq r$$

$$x \succeq 0$$

Пример: задача о диете

Белки

Жиры

Углеводы

Калории

Витамин D

Содержание на 100г

$W \in \mathbb{R}^{n \times p}$

$c \in \mathbb{R}^p$, цена за 100г

$r \in \mathbb{R}^n$, нормы потребления

$x \in \mathbb{R}^p$, количество продуктов, 100г

$$\min_{x \in \mathbb{R}^p} c^T x$$

$$Wx \succeq r$$

$$x \succeq 0$$

Представьте, что вам нужно составить план диеты из некоторых продуктов: бананы, пироги, курица, яйца, рыба. Каждый из продуктов имеет свой вектор питательных веществ. Таким образом, все питательные вещества можно представить в виде матрицы W. Предположим, что у нас есть вектор требований для каждого питательного вещества $r \in \mathbb{R}^n$. Нам нужно найти самую дешёвую конфигурацию диеты, которая удовлетворяет всем требованиям:

$$\begin{aligned} &\min_{x \in \mathbb{R}^p} c^\top x \\ \text{s.t. } &Wx \succeq r \\ &x_i \geq 0,\ i = 1, \dots, n \end{aligned}$$

Open In Colab

Минимизация выпуклой функции как задача линейного программирования

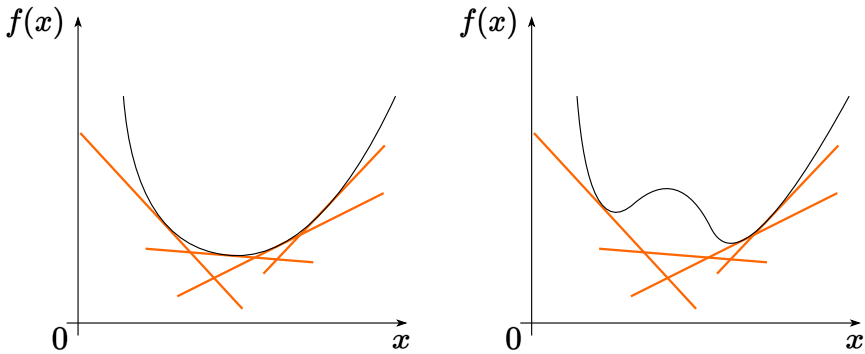


Рис. 1: Как задача линейного программирования может помочь с общей задачей выпуклой оптимизации

- Функция выпукла, если она может быть представлена как максимум линейных функций.

Минимизация выпуклой функции как задача линейного программирования

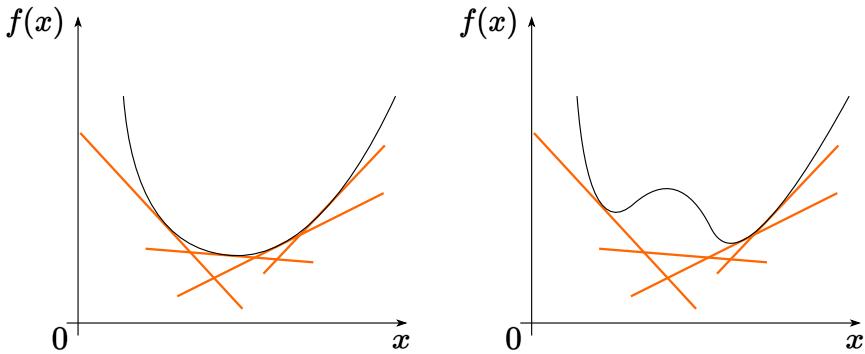


Рис. 1: Как задача линейного программирования может помочь с общей задачей выпуклой оптимизации

- Функция выпукла, если она может быть представлена как максимум линейных функций.
- В пространствах большой размерности аппроксимация может потребовать большого количества функций.

Минимизация выпуклой функции как задача линейного программирования

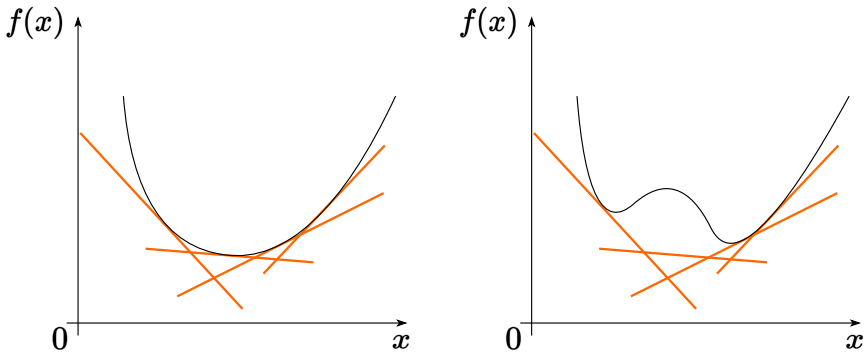


Рис. 1: Как задача линейного программирования может помочь с общей задачей выпуклой оптимизации

- Функция выпукла, если она может быть представлена как максимум линейных функций.
- В пространствах большой размерности аппроксимация может потребовать большого количества функций.
- Существуют более эффективные выпуклые оптимизаторы (не сводящиеся к LP).

Пример: Задача транспортировки

Типичная задача транспортировки заключается в распределении товара от набора источников к набору пунктов назначения. Цель состоит в минимизации общих затрат на транспортировку при соблюдении ограничений на количество товара на каждом источнике и удовлетворении требований к спросу на каждом пункте назначения.

Рис. 2: Карта Западной Европы. Open In Colab

Пример: Задача транспортировки

Пункт назначения / Источник	Арнем [€/тонна]	Гауда [€/тонна]	Спрос [тонн]
Лондон	n/a	2.5	125
Берлин	2.5	n/a	175
Маастрихт	1.6	2.0	225
Амстердам	1.4	1.0	250
Утрехт	0.8	1.0	225
Гаага	1.4	0.8	200
Поставка [тонн]	550	700	

Минимизировать: $\text{Стоимость} = \sum_{c \in \text{Пункты назначения}} \sum_{s \in \text{Источники}} T[c,s]x[c,s]$

Пример: Задача транспортировки

Пункт назначения / Источник	Арнем [€/тонна]	Гауда [€/тонна]	Спрос [тонн]
Лондон	n/a	2.5	125
Берлин	2.5	n/a	175
Маастрихт	1.6	2.0	225
Амстердам	1.4	1.0	250
Утрехт	0.8	1.0	225
Гаага	1.4	0.8	200
Поставка [тонн]	550	700	

Минимизировать: $\text{Стоимость} = \sum_{c \in \text{Пункты назначения}} \sum_{s \in \text{Источники}} T[c,s]x[c,s]$

$$\sum_{c \in \text{Пункты назначения}} x[c,s] \leq \text{Поставка}[s] \qquad \forall s \in \text{Источники}$$

Пример: Задача транспортировки

Пункт назначения / Источник	Арнем [€/тонна]	Гауда [€/тонна]	Спрос [тонн]
Лондон	n/a	2.5	125
Берлин	2.5	n/a	175
Маастрихт	1.6	2.0	225
Амстердам	1.4	1.0	250
Утрехт	0.8	1.0	225
Гаага	1.4	0.8	200
Поставка [тонн]	550	700	

Это можно представить в виде следующего графа:

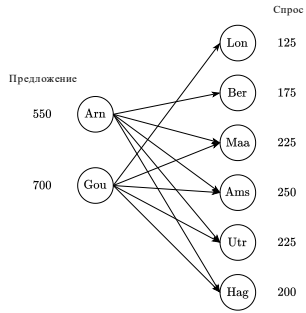


Рис. 3: Граф, связанный с задачей

Минимизировать: $\text{Стоимость} = \sum_{c \in \text{Пункты назначения}} \sum_{s \in \text{Источники}} T[c,s]x[c,s]$

$$\sum_{c \in \text{Пункты назначения}} x[c,s] \leq \text{Поставка}[s] \qquad \forall s \in \text{Источники}$$

$$\sum_{s \in \text{Источники}} x[c,s] = \text{Спрос}[c] \qquad \forall c \in \text{Пункты назначения}$$

Как получить задачу линейного программирования?

Основные преобразования

- Максимум-минимум

$$\min_{x \in \mathbb{R}^n} c^\top x \quad \text{s.t. } Ax \leq b \qquad \leftrightarrow \qquad \max_{x \in \mathbb{R}^n} -c^\top x \quad \text{s.t. } Ax \leq b$$

Основные преобразования

- Максимум-минимум

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} c^\top x \\ &\text{s.t. } Ax \leq b \end{aligned} \quad \leftrightarrow \quad \begin{aligned} &\max_{x \in \mathbb{R}^n} -c^\top x \\ &\text{s.t. } Ax \leq b \end{aligned}$$

- Равенство к неравенству

$$Ax = b \leftrightarrow \begin{cases} Ax \leq b \\ Ax \geq b \end{cases}$$

Основные преобразования

- Максимум-минимум

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} c^\top x \\ \text{s.t. } &Ax \le b \end{aligned} \quad \leftrightarrow \quad \begin{aligned} &\max_{x\in\mathbb{R}^n} -c^\top x \\ \text{s.t. } &Ax \le b \end{aligned}$$

- Равенство к неравенству

$$Ax = b \leftrightarrow \begin{cases} Ax \le b \\ Ax \ge b \end{cases}$$

- Неравенство к равенству, увеличивая размерность задачи на m.

$$Ax \le b \leftrightarrow \begin{cases} Ax + z = b \\ z \ge 0 \end{cases}$$

Основные преобразования

- Максимум-минимум

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} c^\top x \\ \text{s.t. } &Ax \le b \end{aligned} \quad \leftrightarrow \quad \begin{aligned} &\max_{x\in\mathbb{R}^n} -c^\top x \\ \text{s.t. } &Ax \le b \end{aligned}$$

- Равенство к неравенству

$$Ax = b \leftrightarrow \begin{cases} Ax \le b \\ Ax \ge b \end{cases}$$

- Неравенство к равенству, увеличивая размерность задачи на m.

$$Ax \le b \leftrightarrow \begin{cases} Ax + z = b \\ z \ge 0 \end{cases}$$

- Неотрицательные переменные

$$x \leftrightarrow \begin{cases} x = x_+ - x_- \\ x_+ \ge 0 \\ x_- \ge 0 \end{cases}$$

Пример: задача аппроксимации Чебышева

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_\infty \leftrightarrow \min_{x \in \mathbb{R}^n} \max_i |a_i^T x - b_i|$$

Можно эквивалентно записать как задачу линейного программирования с заменой максимальной координаты вектора:

Пример: задача аппроксимации Чебышева

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_\infty \leftrightarrow \min_{x \in \mathbb{R}^n} \max_i |a_i^T x - b_i|$$

Можно эквивалентно записать как задачу линейного программирования с заменой максимальной координаты вектора:

$$\begin{aligned} &\min_{t \in \mathbb{R}, x \in \mathbb{R}^n} t \\ \text{s.t. } &a_i^T x - b_i \leq t, \ i = 1, \dots, n \\ &-a_i^T x + b_i \leq t, \ i = 1, \dots, n \end{aligned}$$

ℓ_1 задача аппроксимации

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_1 \leftrightarrow \min_{x \in \mathbb{R}^n} \sum_{i=1}^{n} |a_i^T x - b_i|$$

Можно эквивалентно записать как задачу линейного программирования с заменой суммы координат вектора:

ℓ_1 задача аппроксимации

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_1 \leftrightarrow \min_{x \in \mathbb{R}^n} \sum_{i=1}^{n} |a_i^T x - b_i|$$

Можно эквивалентно записать как задачу линейного программирования с заменой суммы координат вектора:

$$\begin{aligned} &\min_{t \in \mathbb{R}^n, x \in \mathbb{R}^n} \mathbf{1}^T t \\ \text{s.t. } &a_i^T x - b_i \leq t_i, \ i = 1, \dots, n \\ &-a_i^T x + b_i \leq t_i, \ i = 1, \dots, n \end{aligned}$$

Задача смешивания: от нелинейных ограничений к ЛП [1]

Производственное предприятие получает заказ на 100 литров раствора с определённой концентрацией (например, 4% сахарного раствора). На складе есть:

Компонент	Сахар (%)	Стоимость ($/л)
Концентрат A (Добрый кола)	10.6	1.25
Концентрат B (Север кола)	4.5	1.02
Вода (Псыж)	0.0	0.62

Цель: Найти смешивание с минимальной стоимостью, которое удовлетворит заказ.

Целевая функция

Задача смешивания: от нелинейных ограничений к ЛП [1]

Производственное предприятие получает заказ на 100 литров раствора с определённой концентрацией (например, 4% сахарного раствора). На складе есть:

Компонент	Сахар (%)	Стоимость ($/л)
Концентрат А (Добрый кола)	10.6	1.25
Концентрат В (Север кола)	4.5	1.02
Вода (Псыж)	0.0	0.62

Цель: Найти смешивание с минимальной стоимостью, которое удовлетворит заказ.

Целевая функция

Минимизировать стоимость:

$$\text{Cost} = \sum_{c \in C} x_c P_c$$

где x_c — объём используемого компонента c, и P_c — его цена.

Задача смешивания: от нелинейных ограничений к ЛП [1]

Производственное предприятие получает заказ на 100 литров раствора с определённой концентрацией (например, 4% сахарного раствора). На складе есть:

Компонент	Сахар (%)	Стоимость ($/л)
Концентрат A (Добрый кола)	10.6	1.25
Концентрат B (Север кола)	4.5	1.02
Вода (Псыж)	0.0	0.62

Цель: Найти смешивание с минимальной стоимостью, которое удовлетворит заказ.

Целевая функция

Минимизировать стоимость:

$$\text{Cost} = \sum_{c \in C} x_c P_c$$

где x_c — объём используемого компонента c, и P_c — его цена.

Ограничение на объём

Задача смешивания: от нелинейных ограничений к ЛП [1]

Производственное предприятие получает заказ на 100 литров раствора с определённой концентрацией (например, 4% сахарного раствора). На складе есть:

Компонент	Сахар (%)	Стоимость ($/л)
Концентрат A (Добрый кола)	10.6	1.25
Концентрат B (Север кола)	4.5	1.02
Вода (Псыж)	0.0	0.62

Цель: Найти смешивание с минимальной стоимостью, которое удовлетворит заказ.

Целевая функция

Минимизировать стоимость:

$$\text{Cost} = \sum_{c \in C} x_c P_c$$

где x_c — объём используемого компонента c, и P_c — его цена.

Ограничение на объём

Убедитесь, что общий объём V:

$$V = \sum_{c \in C} x_c$$

Ограничение на состав

Задача смешивания: от нелинейных ограничений к ЛП [1]

Производственное предприятие получает заказ на 100 литров раствора с определённой концентрацией (например, 4% сахарного раствора). На складе есть:

Компонент	Сахар (%)	Стоимость ($/л)
Концентрат А (Добрый кола)	10.6	1.25
Концентрат В (Север кола)	4.5	1.02
Вода (Псыж)	0.0	0.62

Цель: Найти смешивание с минимальной стоимостью, которое удовлетворит заказ.

Целевая функция

Минимизировать стоимость:

$$\text{Cost} = \sum_{c \in C} x_c P_c$$

где x_c — объём используемого компонента c, и P_c — его цена.

Ограничение на объём

Убедитесь, что общий объём V:

$$V = \sum_{c \in C} x_c$$

Ограничение на состав

Убедитесь, что содержание сахара 4%:

$$\bar{A} = \frac{\sum_{c \in C} x_c A_c}{\sum_{c \in C} x_c}$$

[1] Source

Задача смешивания: от нелинейных ограничений к ЛП [1]

Производственное предприятие получает заказ на 100 литров раствора с определённой концентрацией (например, 4% сахарного раствора). На складе есть:

Компонент	Сахар (%)	Стоимость ($/л)
Концентрат А (Добрый кола)	10.6	1.25
Концентрат В (Север кола)	4.5	1.02
Вода (Псыж)	0.0	0.62

Цель: Найти смешивание с минимальной стоимостью, которое удовлетворит заказ.

Целевая функция

Минимизировать стоимость:

$$\text{Cost} = \sum_{c \in C} x_c P_c$$

где x_c — объём используемого компонента c, и P_c — его цена.

Ограничение на объём

Убедитесь, что общий объём V:

$$V = \sum_{c \in C} x_c$$

Ограничение на состав

Убедитесь, что содержание сахара 4%:

$$\bar{A} = \frac{\sum_{c \in C} x_c A_c}{\sum_{c \in C} x_c}$$

Линеаризованная версия:

$$0 = \sum_{c \in C} x_c (A_c - \bar{A})$$

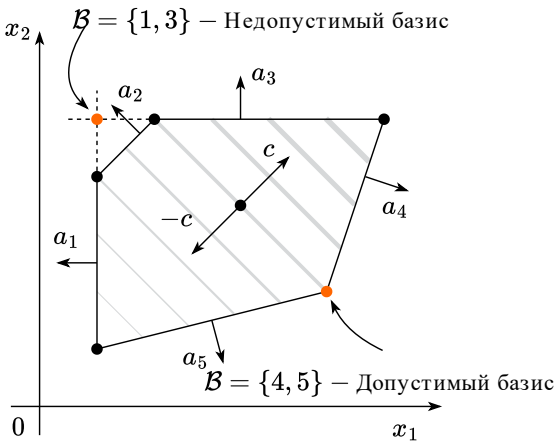
Это можно решить с помощью линейного программирования.

Код

[1] Source

Симплекс-метод

Геометрия симплекс-метода

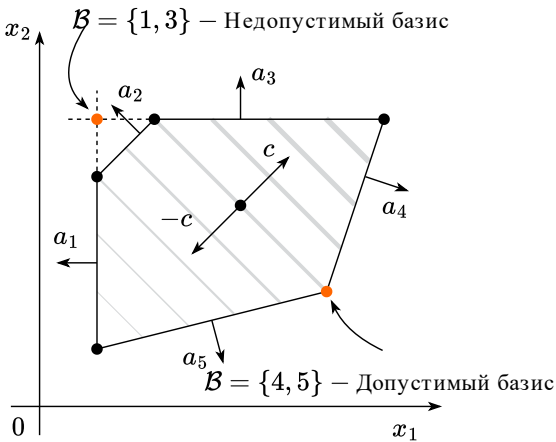


Рассмотрим следующую простую формулировку задачи линейного программирования, которая, фактически, является двойственной к стандартной форме:

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \le b \qquad \text{(LP.Inequality)}$$

- Определение: **базис** $\mathcal{B}$ — это подмножество n (целых) чисел между 1 и m, такое что $\text{rank} A_{\mathcal{B}} = n$.

Геометрия симплекс-метода

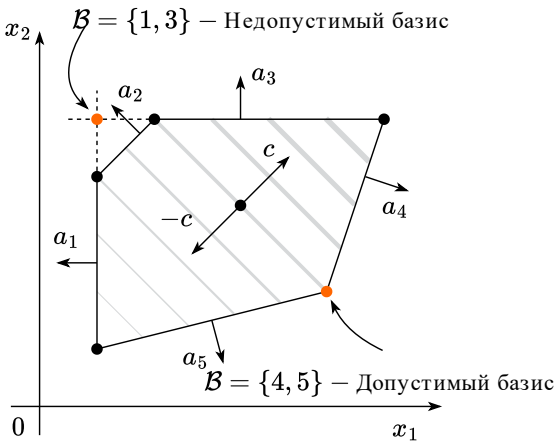


Рассмотрим следующую простую формулировку задачи линейного программирования, которая, фактически, является двойственной к стандартной форме:

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \le b \qquad \text{(LP.Inequality)}$$

- Определение: **базис** $\mathcal{B}$ — это подмножество n (целых) чисел между 1 и m, такое что $\text{rank} A_{\mathcal{B}} = n$.
- Обратите внимание, что мы можем связать подматрицу $A_{\mathcal{B}}$ и соответствующую правую часть $b_{\mathcal{B}}$ с базисом $\mathcal{B}$.

Геометрия симплекс-метода

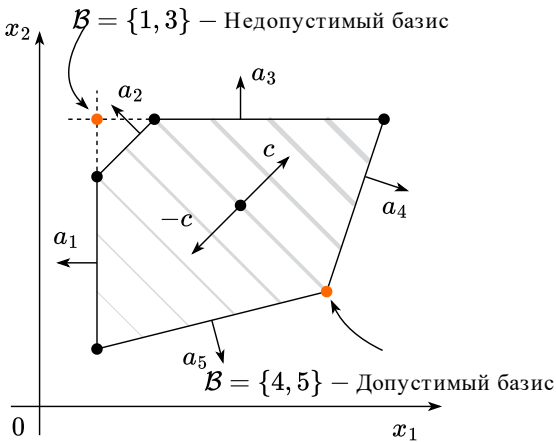


Рассмотрим следующую простую формулировку задачи линейного программирования, которая, фактически, является двойственной к стандартной форме:

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \leq b \qquad \text{(LP.Inequality)}$$

- Определение: **базис** $\mathcal{B}$ — это подмножество n (целых) чисел между 1 и m, такое что $\text{rank} A_{\mathcal{B}} = n$.
- Обратите внимание, что мы можем связать подматрицу $A_{\mathcal{B}}$ и соответствующую правую часть $b_{\mathcal{B}}$ с базисом $\mathcal{B}$.
- Также мы можем получить точку пересечения всех этих гиперплоскостей из базиса: $x_{\mathcal{B}} = A_{\mathcal{B}}^{-1} b_{\mathcal{B}}$.

Геометрия симплекс-метода

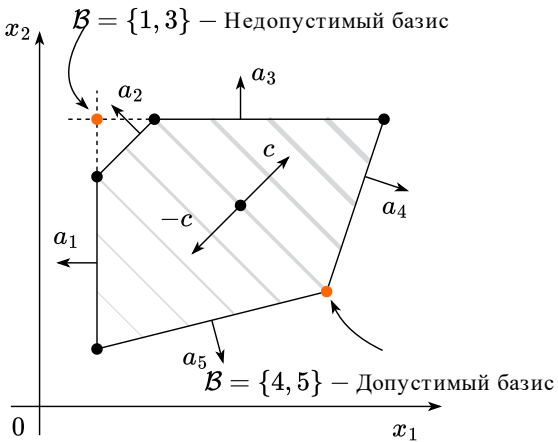


Рассмотрим следующую простую формулировку задачи линейного программирования, которая, фактически, является двойственной к стандартной форме:

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \leq b \qquad \text{(LP.Inequality)}$$

- Определение: **базис** $\mathcal{B}$ — это подмножество n (целых) чисел между 1 и m, такое что $\text{rank} A_{\mathcal{B}} = n$.
- Обратите внимание, что мы можем связать подматрицу $A_{\mathcal{B}}$ и соответствующую правую часть $b_{\mathcal{B}}$ с базисом $\mathcal{B}$.
- Также мы можем получить точку пересечения всех этих гиперплоскостей из базиса: $x_{\mathcal{B}} = A_{\mathcal{B}}^{-1} b_{\mathcal{B}}$.
- Если $Ax_{\mathcal{B}} \leq b$, то базис $\mathcal{B}$ является **допустимым**.

Геометрия симплекс-метода

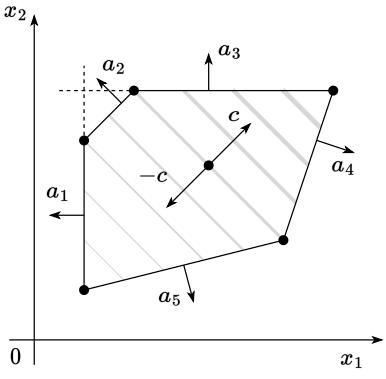


Рассмотрим следующую простую формулировку задачи линейного программирования, которая, фактически, является двойственной к стандартной форме:

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \leq b \qquad \text{(LP.Inequality)}$$

- Определение: **базис** $\mathcal{B}$ — это подмножество n (целых) чисел между 1 и m, такое что $\text{rank} A_{\mathcal{B}} = n$.
- Обратите внимание, что мы можем связать подматрицу $A_{\mathcal{B}}$ и соответствующую правую часть $b_{\mathcal{B}}$ с базисом $\mathcal{B}$.
- Также мы можем получить точку пересечения всех этих гиперплоскостей из базиса: $x_{\mathcal{B}} = A_{\mathcal{B}}^{-1} b_{\mathcal{B}}$.
- Если $Ax_{\mathcal{B}} \leq b$, то базис $\mathcal{B}$ является **допустимым**.
- Базис $\mathcal{B}$ оптимален, если $x_{\mathcal{B}}$ является решением задачи LP. Неравенства.

Если решение задачи линейного программирования существует, то оно лежит в вершине

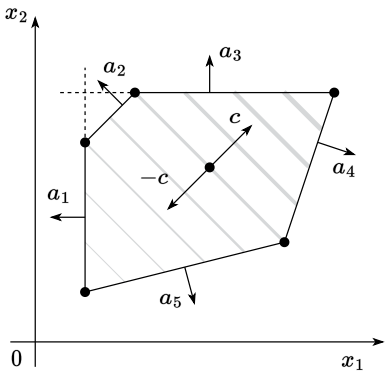


Theorem

1. Если стандартная задача линейного программирования имеет непустую допустимую область, то существует по крайней мере одна допустимая точка базиса.

Верхнеуровневая идея симплекс-метода следующая:

Если решение задачи линейного программирования существует, то оно лежит в вершине

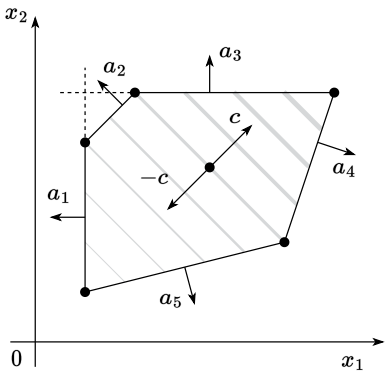


Theorem

1. Если стандартная задача линейного программирования имеет непустую допустимую область, то существует по крайней мере одна допустимая точка базиса.
2. Если стандартная задача линейного программирования имеет решения, то по крайней мере одно из таких решений является оптимальной точкой базиса.

Верхнеуровневая идея симплекс-метода следующая:

Если решение задачи линейного программирования существует, то оно лежит в вершине

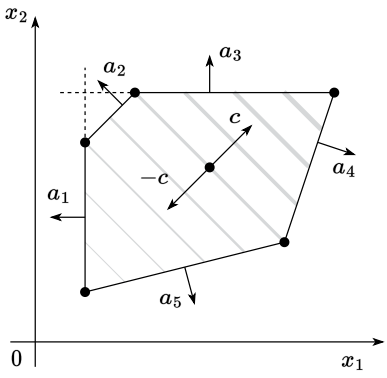


Theorem

1. Если стандартная задача линейного программирования имеет непустую допустимую область, то существует по крайней мере одна допустимая точка базиса.
2. Если стандартная задача линейного программирования имеет решения, то по крайней мере одно из таких решений является оптимальной точкой базиса.
3. Если стандартная задача линейного программирования допустима и ограничена, то она имеет оптимальное решение.

Верхнеуровневая идея симплекс-метода следующая:

Если решение задачи линейного программирования существует, то оно лежит в вершине

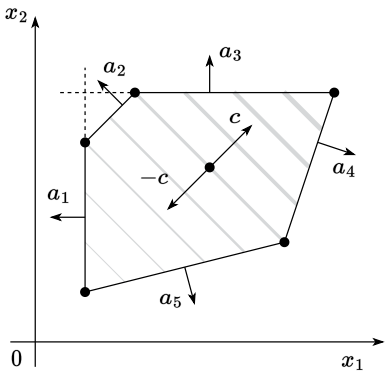


Theorem

1. Если стандартная задача линейного программирования имеет непустую допустимую область, то существует по крайней мере одна допустимая точка базиса.
2. Если стандартная задача линейного программирования имеет решения, то по крайней мере одно из таких решений является оптимальной точкой базиса.
3. Если стандартная задача линейного программирования допустима и ограничена, то она имеет оптимальное решение.

Верхнеуровневая идея симплекс-метода следующая:

Если решение задачи линейного программирования существует, то оно лежит в вершине



Theorem

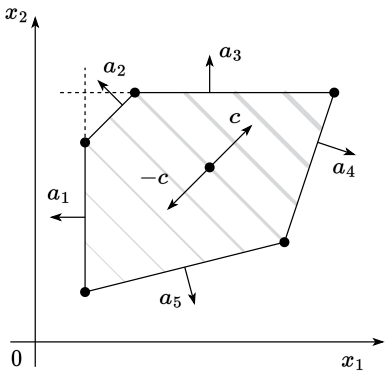
1. Если стандартная задача линейного программирования имеет непустую допустимую область, то существует по крайней мере одна допустимая точка базиса.
2. Если стандартная задача линейного программирования имеет решения, то по крайней мере одно из таких решений является оптимальной точкой базиса.
3. Если стандартная задача линейного программирования допустима и ограничена, то она имеет оптимальное решение.

Для доказательства см. теорему 13.2 в Numerical Optimization by Jorge Nocedal and Stephen J. Wright

Верхнеуровневая идея симплекс-метода следующая:

- Убедитесь, что вы находитесь в вершине.

Если решение задачи линейного программирования существует, то оно лежит в вершине



Theorem

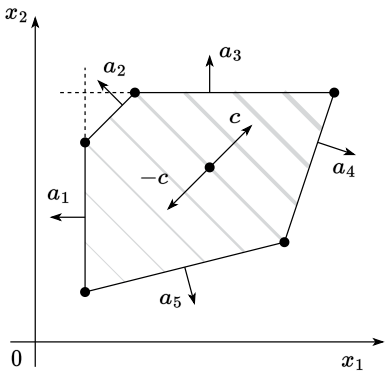
1. Если стандартная задача линейного программирования имеет непустую допустимую область, то существует по крайней мере одна допустимая точка базиса.
2. Если стандартная задача линейного программирования имеет решения, то по крайней мере одно из таких решений является оптимальной точкой базиса.
3. Если стандартная задача линейного программирования допустима и ограничена, то она имеет оптимальное решение.

Для доказательства см. теорему 13.2 в Numerical Optimization by Jorge Nocedal and Stephen J. Wright

Верхнеуровневая идея симплекс-метода следующая:

- Убедитесь, что вы находитесь в вершине.
- Проверьте оптимальность.

Если решение задачи линейного программирования существует, то оно лежит в вершине



Theorem

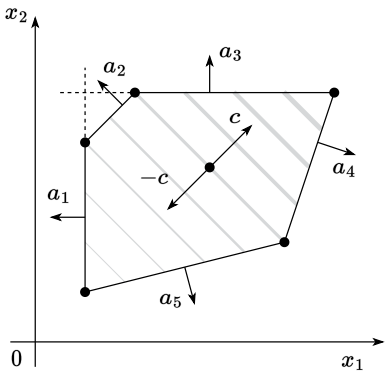
1. Если стандартная задача линейного программирования имеет непустую допустимую область, то существует по крайней мере одна допустимая точка базиса.
2. Если стандартная задача линейного программирования имеет решения, то по крайней мере одно из таких решений является оптимальной точкой базиса.
3. Если стандартная задача линейного программирования допустима и ограничена, то она имеет оптимальное решение.

Для доказательства см. теорему 13.2 в Numerical Optimization by Jorge Nocedal and Stephen J. Wright

Верхнеуровневая идея симплекс-метода следующая:

- Убедитесь, что вы находитесь в вершине.
- Проверьте оптимальность.
- Если необходимо, переключите вершину (измените базис).

Если решение задачи линейного программирования существует, то оно лежит в вершине



Theorem

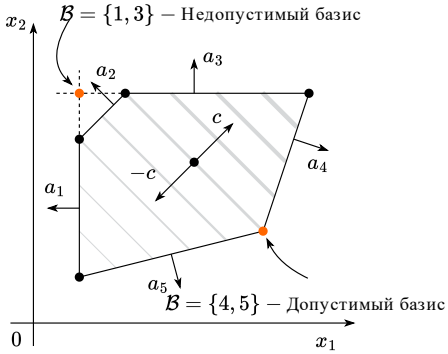
1. Если стандартная задача линейного программирования имеет непустую допустимую область, то существует по крайней мере одна допустимая точка базиса.
2. Если стандартная задача линейного программирования имеет решения, то по крайней мере одно из таких решений является оптимальной точкой базиса.
3. Если стандартная задача линейного программирования допустима и ограничена, то она имеет оптимальное решение.

Для доказательства см. теорему 13.2 в Numerical Optimization by Jorge Nocedal and Stephen J. Wright

Верхнеуровневая идея симплекс-метода следующая:

- Убедитесь, что вы находитесь в вершине.
- Проверьте оптимальность.
- Если необходимо, переключите вершину (измените базис).
- Повторяйте, пока не сходится.

Оптимальный базис



Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$$

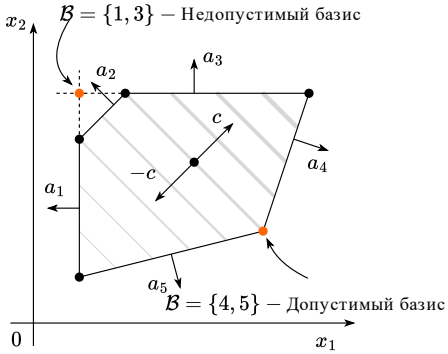
Theorem

Если все компоненты $\lambda_{\mathcal{B}}$ неотрицательны и $\mathcal{B}$ допустим, то $\mathcal{B}$ оптимален.

Доказательство

$$\exists x^* : Ax^* \leq b, c^T x^* < c^T x_{\mathcal{B}}$$

Оптимальный базис



Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$$

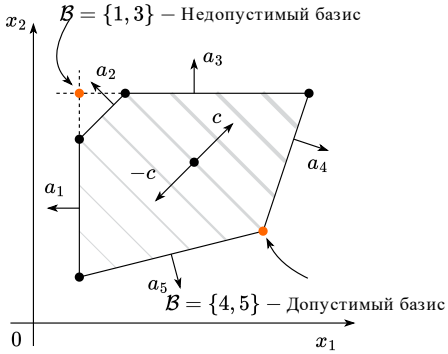
Theorem

Если все компоненты $\lambda_{\mathcal{B}}$ неотрицательны и $\mathcal{B}$ допустим, то $\mathcal{B}$ оптимален.

Доказательство

$$\exists x^* : Ax^* \leq b, c^T x^* < c^T x_{\mathcal{B}}$$
$$A_{\mathcal{B}} x^* \leq b_{\mathcal{B}}$$

Оптимальный базис



Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$$

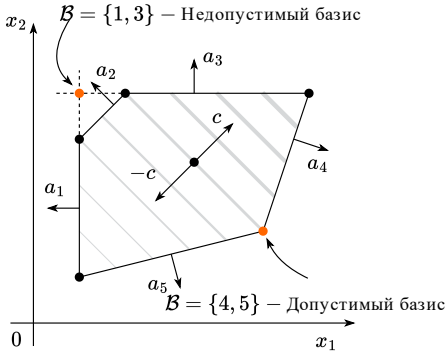
Theorem

Если все компоненты $\lambda_{\mathcal{B}}$ неотрицательны и $\mathcal{B}$ допустим, то $\mathcal{B}$ оптимален.

Доказательство

$$\exists x^* : Ax^* \leq b, c^T x^* < c^T x_{\mathcal{B}}$$
$$A_{\mathcal{B}} x^* \leq b_{\mathcal{B}}$$
$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} x^* \geq \lambda_{\mathcal{B}}^T b_{\mathcal{B}}$$

Оптимальный базис



Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$$

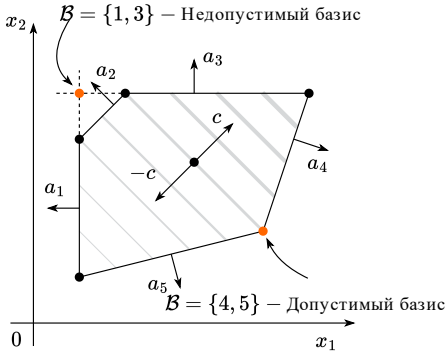
Theorem

Если все компоненты $\lambda_{\mathcal{B}}$ неотрицательны и $\mathcal{B}$ допустим, то $\mathcal{B}$ оптимален.

Доказательство

$$\exists x^* : Ax^* \leq b, c^T x^* < c^T x_{\mathcal{B}}$$
$$A_{\mathcal{B}} x^* \leq b_{\mathcal{B}}$$
$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} x^* \geq \lambda_{\mathcal{B}}^T b_{\mathcal{B}}$$
$$c^T x^* \geq \lambda_{\mathcal{B}}^T A_{\mathcal{B}} x_{\mathcal{B}}$$

Оптимальный базис



Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$$

Theorem

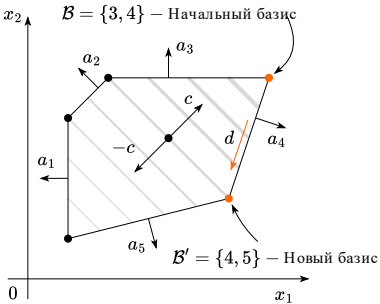
Если все компоненты $\lambda_{\mathcal{B}}$ неотрицательны и $\mathcal{B}$ допустим, то $\mathcal{B}$ оптимален.

Доказательство

$$\exists x^* : Ax^* \leq b, c^T x^* < c^T x_{\mathcal{B}}$$
$$A_{\mathcal{B}} x^* \leq b_{\mathcal{B}}$$
$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} x^* \geq \lambda_{\mathcal{B}}^T b_{\mathcal{B}}$$
$$c^T x^* \geq \lambda_{\mathcal{B}}^T A_{\mathcal{B}} x_{\mathcal{B}}$$
$$c^T x^* \geq c^T x_{\mathcal{B}}$$

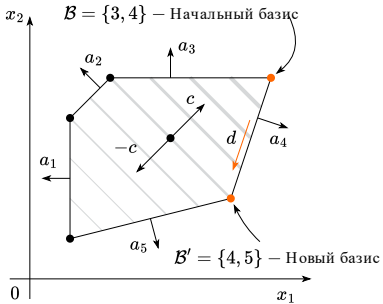
Изменение базиса

- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$



Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

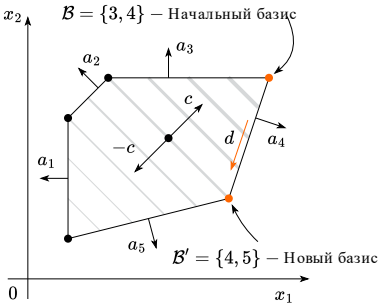
Изменение базиса



- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса

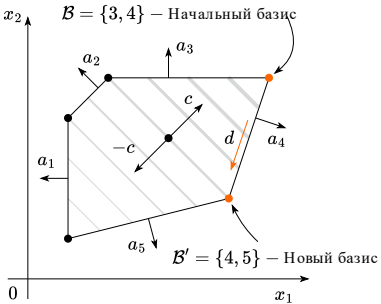


- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases}$$

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса

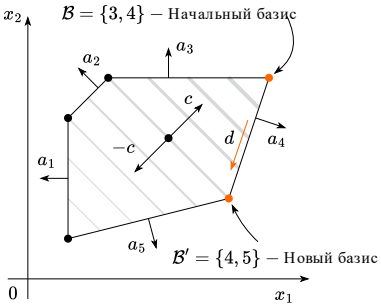


- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d$$

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса

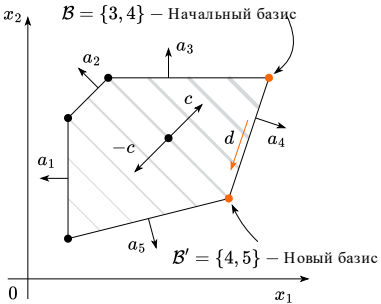


- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d = \lambda_{\mathcal{B}}^T A_{\mathcal{B}} d$$

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса

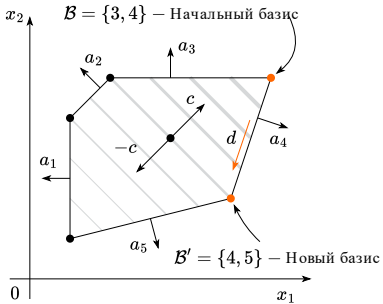


- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d = \lambda_{\mathcal{B}}^T A_{\mathcal{B}} d = \sum_{i=1}^{n} \lambda_{\mathcal{B}}^i (A_{\mathcal{B}} d)^i$$

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса

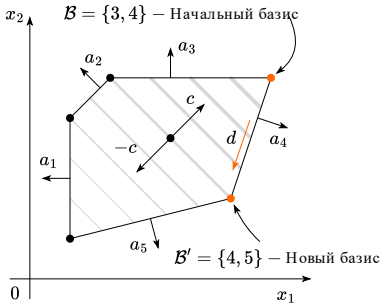


- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d = \lambda_{\mathcal{B}}^T A_{\mathcal{B}} d = \sum_{i=1}^{n} \lambda_{\mathcal{B}}^i (A_{\mathcal{B}} d)^i = -\lambda_{\mathcal{B}}^k < 0$$

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса



- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

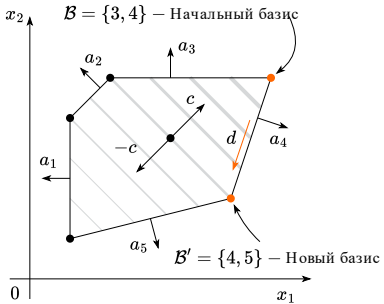
$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d = \lambda_{\mathcal{B}}^T A_{\mathcal{B}} d = \sum_{i=1}^{n} \lambda_{\mathcal{B}}^i (A_{\mathcal{B}} d)^i = -\lambda_{\mathcal{B}}^k < 0$$

- Для всех $j \notin \mathcal{B}$ рассчитаем размер шага проекции:

$$\mu_j = \frac{b_j - a_j^T x_{\mathcal{B}}}{a_j^T d}$$

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса



Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d = \lambda_{\mathcal{B}}^T A_{\mathcal{B}} d = \sum_{i=1}^{n} \lambda_{\mathcal{B}}^i (A_{\mathcal{B}} d)^i = -\lambda_{\mathcal{B}}^k < 0$$

- Для всех $j \notin \mathcal{B}$ рассчитаем размер шага проекции:

$$\mu_j = \frac{b_j - a_j^T x_{\mathcal{B}}}{a_j^T d}$$

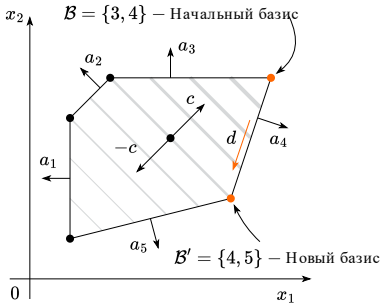
- Определим новую вершину, которую мы добавим в новый базис:

$$t = \arg\min_j \{\mu_j \mid \mu_j > 0\}$$

$$\mathcal{B}' = \mathcal{B}\setminus\{k\} \cup \{t\}$$

$$x_{\mathcal{B}'} = x_{\mathcal{B}} + \mu_t d = A_{\mathcal{B}'}^{-1} b_{\mathcal{B}'}$$

Изменение базиса



Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d = \lambda_{\mathcal{B}}^T A_{\mathcal{B}} d = \sum_{i=1}^{n} \lambda_{\mathcal{B}}^i (A_{\mathcal{B}} d)^i = -\lambda_{\mathcal{B}}^k < 0$$

- Для всех $j \notin \mathcal{B}$ рассчитаем размер шага проекции:

$$\mu_j = \frac{b_j - a_j^T x_{\mathcal{B}}}{a_j^T d}$$

- Определим новую вершину, которую мы добавим в новый базис:

$$t = \arg\min_j \{\mu_j \mid \mu_j > 0\}$$

$$\mathcal{B}' = \mathcal{B}\setminus\{k\} \cup \{t\}$$

$$x_{\mathcal{B}'} = x_{\mathcal{B}} + \mu_t d = A_{\mathcal{B}'}^{-1} b_{\mathcal{B}'}$$

- Обратите внимание, что изменение базиса приводит к уменьшению целевой функции

Поиск начального допустимого базисного решения

Нам нужно решить следующую задачу:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} c^\top x \\ &\text{s.t. } Ax \leq b \end{aligned} \tag{1}$$

Предложенный алгоритм требует начального допустимого базисного решения и соответствующего базиса.

Поиск начального допустимого базисного решения

Нам нужно решить следующую задачу:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} c^\top x \\ &\text{s.t. } Ax \leq b \end{aligned} \tag{1}$$

Предложенный алгоритм требует начального допустимого базисного решения и соответствующего базиса.

Начнём с переформулировки задачи:

$$\begin{aligned} &\min_{y \in \mathbb{R}^n, z \in \mathbb{R}^n} c^\top (y - z) \\ &\text{s.t. } Ay - Az \leq b \\ &y \geq 0, z \geq 0 \end{aligned} \tag{2}$$

Поиск начального допустимого базисного решения

Нам нужно решить следующую задачу:

$$\min_{x \in \mathbb{R}^n} c^\top x \quad \text{s.t. } Ax \leq b \tag{1}$$

Предложенный алгоритм требует начального допустимого базисного решения и соответствующего базиса.

Начнём с переформулировки задачи:

$$\min_{y \in \mathbb{R}^n, z \in \mathbb{R}^n} c^\top (y - z) \quad \text{s.t. } Ay - Az \leq b, \; y \geq 0, z \geq 0 \tag{2}$$

Зная решение задачи (Уравнение 2), можно восстановить решение задачи (Уравнение 1), и наоборот.

$$x = y - z \qquad \Leftrightarrow \qquad y_i = \max(x_i, 0), \quad z_i = \max(-x_i, 0)$$

Теперь мы попытаемся сформулировать новую задачу линейного программирования, решение которой будет допустимой точкой базиса для Задачи 2. Это означает, что мы сначала запускаем симплекс-метод для задачи Phase-1, а затем запускаем задачу Phase-2 с известным начальным решением. Обратите внимание, что допустимое базисное решение для Phase-1 должно быть установлено некоторой легкой процедурой.

Поиск начального допустимого базисного решения

$$\min_{y \in \mathbb{R}^n, z \in \mathbb{R}^n} c^\top (y - z)$$

s.t. $Ay - Az \leq b$ (Фаза-2 (главная задача ЛП))

$y \geq 0, z \geq 0$

Поиск начального допустимого базисного решения

$$\min_{y\in\mathbb{R}^n, z\in\mathbb{R}^n} c^\top (y-z)$$

s.t. $Ay - Az \le b$ (Фаза-2 (главная задача ЛП))

$y \ge 0, z \ge 0$

$$\min_{\xi\in\mathbb{R}^m, y\in\mathbb{R}^n, z\in\mathbb{R}^n} \sum_{i=1}^{m} \xi_i$$

s.t. $Ay - Az \le b + \xi$ (Фаза-1)

$y \ge 0, z \ge 0, \xi \ge 0$

Поиск начального допустимого базисного решения

$$\min_{y\in\mathbb{R}^n, z\in\mathbb{R}^n} c^\top (y-z)$$

s.t. $Ay - Az \le b$ (Фаза-2 (главная задача ЛП))

$y \ge 0, z \ge 0$

$$\min_{\xi\in\mathbb{R}^m, y\in\mathbb{R}^n, z\in\mathbb{R}^n} \sum_{i=1}^{m} \xi_i$$

s.t. $Ay - Az \le b + \xi$ (Фаза-1)

$y \ge 0, z \ge 0, \xi \ge 0$

- Если Фаза-2 (главная задача ЛП) имеет допустимое решение, то оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю).
  Доказательство: тривиальная проверка.

Поиск начального допустимого базисного решения

$$\min_{y\in\mathbb{R}^n, z\in\mathbb{R}^n} c^\top (y - z)$$

s.t. $Ay - Az \le b$ (Фаза-2 (главная задача ЛП))

$y \ge 0, z \ge 0$

$$\min_{\xi\in\mathbb{R}^m, y\in\mathbb{R}^n, z\in\mathbb{R}^n} \sum_{i=1}^{m} \xi_i$$

s.t. $Ay - Az \le b + \xi$ (Фаза-1)

$y \ge 0, z \ge 0, \xi \ge 0$

- Если Фаза-2 (главная задача ЛП) имеет допустимое решение, то оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю).
  Доказательство: тривиальная проверка.
- Если оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю), то мы получаем допустимый базис для Фаза-2.
  Доказательство: тривиальная проверка.

Поиск начального допустимого базисного решения

$$\min_{y\in\mathbb{R}^n, z\in\mathbb{R}^n} c^\top (y - z)$$

s.t. $Ay - Az \le b$ (Фаза-2 (главная задача ЛП))

$y \ge 0, z \ge 0$

$$\min_{\xi\in\mathbb{R}^m, y\in\mathbb{R}^n, z\in\mathbb{R}^n} \sum_{i=1}^{m} \xi_i$$

s.t. $Ay - Az \le b + \xi$ (Фаза-1)

$y \ge 0, z \ge 0, \xi \ge 0$

- Если Фаза-2 (главная задача ЛП) имеет допустимое решение, то оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю).
  Доказательство: тривиальная проверка.
- Если оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю), то мы получаем допустимый базис для Фаза-2.
  Доказательство: тривиальная проверка.

Поиск начального допустимого базисного решения

$$\min_{y \in \mathbb{R}^n, z \in \mathbb{R}^n} c^\top (y - z)$$

s.t. $Ay - Az \leq b$ (Фаза-2 (главная задача ЛП))

$y \geq 0, z \geq 0$

- Если Фаза-2 (главная задача ЛП) имеет допустимое решение, то оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю).
  Доказательство: тривиальная проверка.
- Если оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю), то мы получаем допустимый базис для Фаза-2.
  Доказательство: тривиальная проверка.

$$\min_{\xi \in \mathbb{R}^m, y \in \mathbb{R}^n, z \in \mathbb{R}^n} \sum_{i=1}^{m} \xi_i$$

s.t. $Ay - Az \leq b + \xi$ (Фаза-1)

$y \geq 0, z \geq 0, \xi \geq 0$

- Теперь мы знаем, что если мы можем решить задачу Фаза-1, то мы либо найдём начальную точку для симплекс-метода в исходном методе (если переменные ξ_i равны нулю), либо проверим, что исходная задача не имеет допустимого решения (если переменные ξ_i не равны нулю).

Поиск начального допустимого базисного решения

$$\min_{y\in\mathbb{R}^n, z\in\mathbb{R}^n} c^\top(y-z)$$
$$\text{s.t. } Ay - Az \le b \qquad \text{(Фаза-2 (главная задача ЛП))}$$
$$y \ge 0, z \ge 0$$

- Если Фаза-2 (главная задача ЛП) имеет допустимое решение, то оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю).
  Доказательство: тривиальная проверка.
- Если оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю), то мы получаем допустимый базис для Фаза-2.
  Доказательство: тривиальная проверка.

$$\min_{\xi\in\mathbb{R}^m, y\in\mathbb{R}^n, z\in\mathbb{R}^n} \sum_{i=1}^{m} \xi_i \qquad \text{(Фаза-1)}$$
$$\text{s.t. } Ay - Az \le b + \xi$$
$$y \ge 0, z \ge 0, \xi \ge 0$$

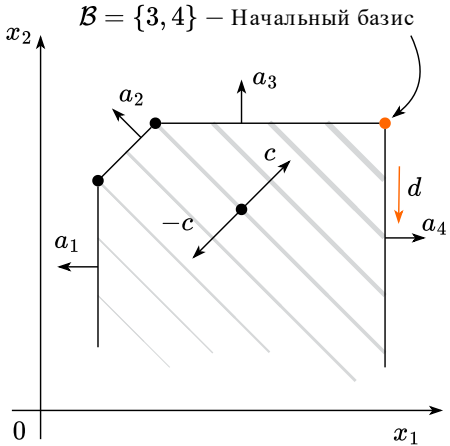
- Теперь мы знаем, что если мы можем решить задачу Фаза-1, то мы либо найдём начальную точку для симплекс-метода в исходном методе (если переменные ξ_i равны нулю), либо проверим, что исходная задача не имеет допустимого решения (если переменные ξ_i не равны нулю).
- Но как решить задачу Фаза-1? Она имеет допустимое базисное решение (задача имеет $2n+m$ переменных, и точка ниже гарантирует, что $2n+m$ неравенств удовлетворяются как равенства (активны).)

$$z = 0 \quad y = 0 \quad \xi_i = \max(0, -b_i)$$

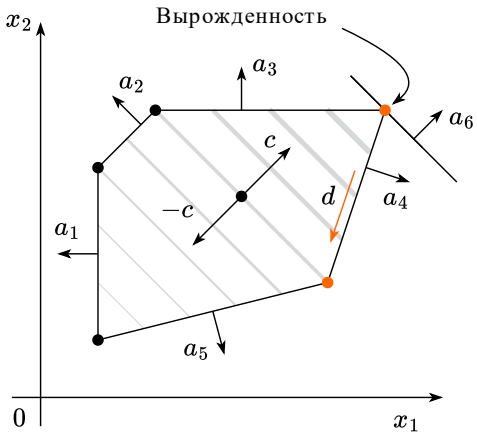
Сходимость симплекс-метода

Неограниченное множество допустимых решений

В этом случае все μ_j будут отрицательными.

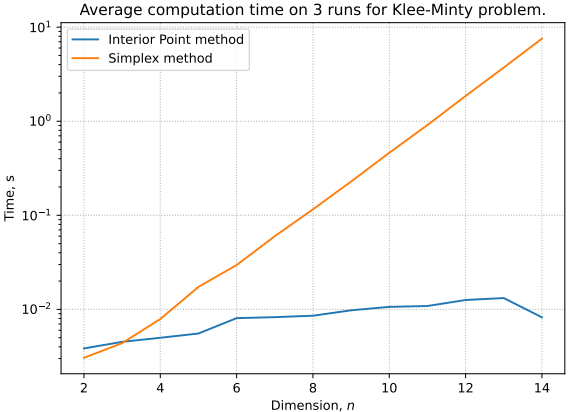


Вырожденность



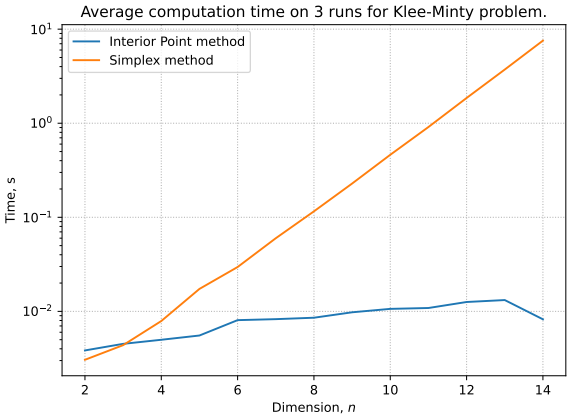
Случаи вырожденности требуют особого рассмотрения. В отсутствие вырожденности на каждой итерации гарантируется монотонное убывание значения целевой функции.

Экспоненциальная сходимость



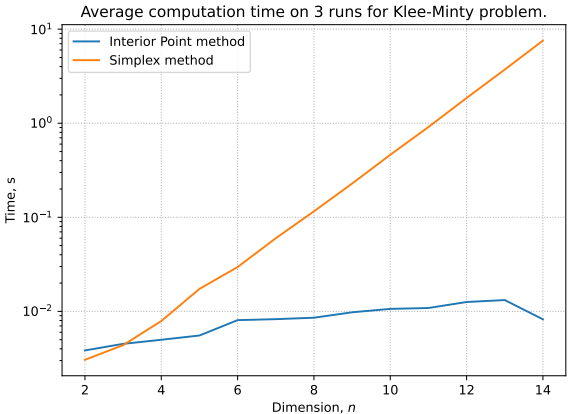
- Множество разнообразных прикладных задач может быть сформулировано в виде задач линейного программирования.

Экспоненциальная сходимость



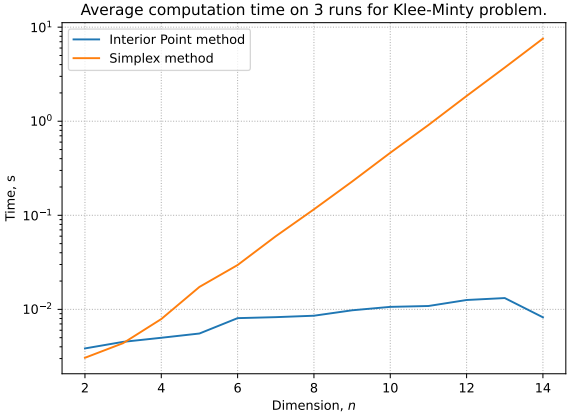
- Множество разнообразных прикладных задач может быть сформулировано в виде задач линейного программирования.
- Симплекс-метод прост в своей основе, но в худшем случае может работать экспоненциально долго.

Экспоненциальная сходимость



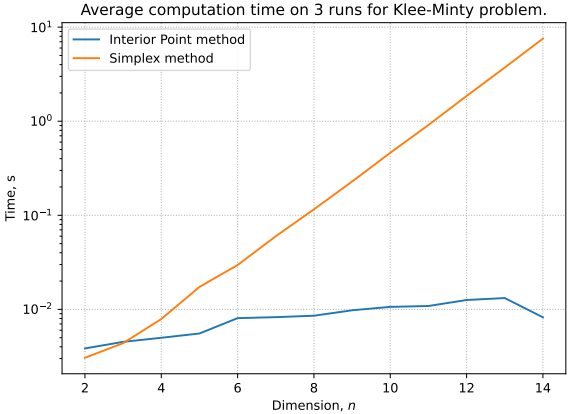
- Множество разнообразных прикладных задач может быть сформулировано в виде задач линейного программирования.
- Симплекс-метод прост в своей основе, но в худшем случае может работать экспоненциально долго.
- Метод эллипсоидов Хачияна (1979) стал первым алгоритмом с доказанной полиномиальной сложностью для задач ЛП. Однако он обычно работает медленнее, чем симплекс-метод в реальных задачах.

Экспоненциальная сходимость



- Множество разнообразных прикладных задач может быть сформулировано в виде задач линейного программирования.
- Симплекс-метод прост в своей основе, но в худшем случае может работать экспоненциально долго.
- Метод эллипсоидов Хачияна (1979) стал первым алгоритмом с доказанной полиномиальной сложностью для задач ЛП. Однако он обычно работает медленнее, чем симплекс-метод в реальных задачах.
- Основной прорыв — метод Кармаркара (1984) для решения задач ЛП с использованием метода внутренней точки.

Экспоненциальная сходимость



- Множество разнообразных прикладных задач может быть сформулировано в виде задач линейного программирования.
- Симплекс-метод прост в своей основе, но в худшем случае может работать экспоненциально долго.
- Метод эллипсоидов Хачияна (1979) стал первым алгоритмом с доказанной полиномиальной сложностью для задач ЛП. Однако он обычно работает медленнее, чем симплекс-метод в реальных задачах.
- Основной прорыв — метод Кармаркара (1984) для решения задач ЛП с использованием метода внутренней точки.
- Методы внутренней точки являются последним словом в этой области. Тем не менее, для типовых задач ЛП качественные реализации симплекс-метода и методов внутренней точки показывают схожую производительность.

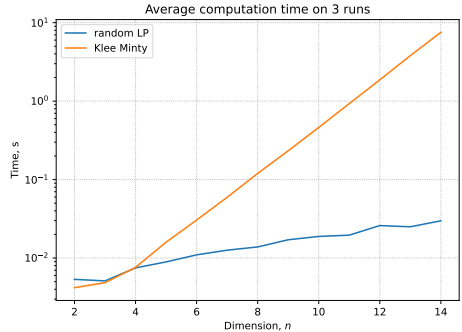
Пример Klee Minty

Так как число вершин конечно, сходимость алгоритма гарантирована (за исключением вырожденных случаев, которые здесь не рассматриваются). Тем не менее, сходимость может быть экспоненциально медленной из-за потенциально большого числа вершин. Существует пример, в котором симплекс-метод вынужден пройти через все вершины многогранника.

В следующей задаче симплекс-метод должен проверить $2^n - 1$ вершин с $x_0 = 0$.

$$\max_{x \in \mathbb{R}^n} 2^{n-1}x_1 + 2^{n-2}x_2 + \cdots + 2x_{n-1} + x_n$$

$$\begin{aligned}
\text{s.t. } & x_1 \leq 5 \\
& 4x_1 + x_2 \leq 25 \\
& 8x_1 + 4x_2 + x_3 \leq 125 \\
& \ldots \\
& 2^n x_1 + 2^{n-1}x_2 + 2^{n-2}x_3 + \ldots + x_n \leq 5^n \\
& x \geq 0
\end{aligned}$$



Смешанное целочисленное программирование

Сложность СЦЛП

Рассмотрим следующую задачу смешанного целочисленного программирования (СЦЛП):

$$
\begin{aligned}
z = 8x_1 + 11x_2 + 6x_3 + 4x_4 &\to \max_{x_1, x_2, x_3, x_4} \\
\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 &\leq 14 \\
x_i &\in \{0, 1\} \quad \forall i
\end{aligned}
\tag{3}
$$

Сложность СЦЛП

Рассмотрим следующую задачу смешанного целочисленного программирования (СЦЛП):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{3}$$
$$x_i \in \{0,1\} \quad \forall i$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{4}$$
$$x_i \in [0,1] \quad \forall i$$

Сложность СЦЛП

Рассмотрим следующую задачу смешанного целочисленного программирования (СЦЛП):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{3}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ and } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{4}$$
$$x_i \in [0,1] \quad \forall i$$

Сложность СЦЛП

Рассмотрим следующую задачу смешанного целочисленного программирования (СЦЛП):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{3}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ and } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{4}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ and } z = 22.$$

Сложность СЦЛП

Рассмотрим следующую задачу смешанного целочисленного программирования (СЦЛП):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{3}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ and } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{4}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ and } z = 22.$$

- Округление $x_3 = 0$: даёт $z = 19$.

Сложность СЦЛП

Рассмотрим следующую задачу смешанного целочисленного программирования (СЦЛП):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{3}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ and } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{4}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ and } z = 22.$$

- Округление $x_3 = 0$: даёт $z = 19$.
- Округление $x_3 = 1$: Недопустимо.

Сложность СЦЛП

Рассмотрим следующую задачу смешанного целочисленного программирования (СЦЛП):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{3}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ and } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{4}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ and } z = 22.$$

- Округление $x_3 = 0$: даёт $z = 19$.
- Округление $x_3 = 1$: Недопустимо.

Сложность СЦЛП

Рассмотрим следующую задачу смешанного целочисленного программирования (СЦЛП):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{3}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ and } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{4}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ and } z = 22.$$

- Округление $x_3 = 0$: даёт $z = 19$.
- Округление $x_3 = 1$: Недопустимо.

! СЦЛП намного сложнее, чем ЛП

- Наивное округление решения, полученного для ЛП-релаксации исходной задачи смешанно-целочисленного программирования, может привести к недопустимому или неоптимальному решению.

Сложность СЦЛП

Рассмотрим следующую задачу смешанного целочисленного программирования (СЦЛП):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{3}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ and } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{4}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ and } z = 22.$$

- Округление $x_3 = 0$: даёт $z = 19$.
- Округление $x_3 = 1$: Недопустимо.

! СЦЛП намного сложнее, чем ЛП

- Наивное округление решения, полученного для ЛП-релаксации исходной задачи смешанно-целочисленного программирования, может привести к недопустимому или неоптимальному решению.
- Общая СЦЛП является NP-hard.

Сложность СЦЛП

Рассмотрим следующую задачу смешанного целочисленного программирования (СЦЛП):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{3}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ and } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{4}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ and } z = 22.$$

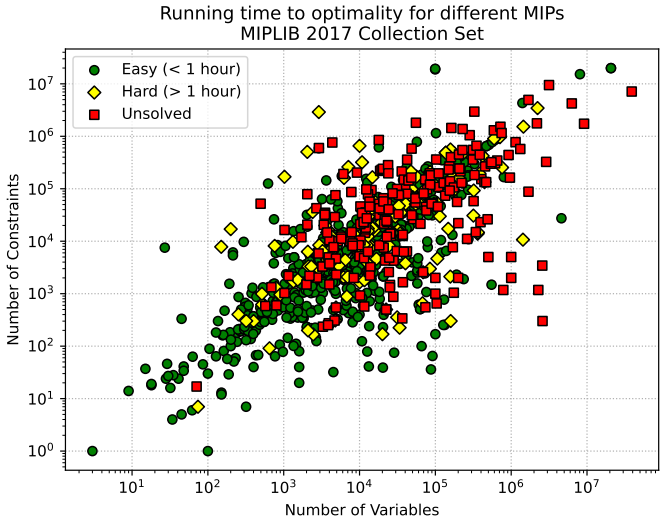
- Округление $x_3 = 0$: даёт $z = 19$.
- Округление $x_3 = 1$: Недопустимо.

! СЦЛП намного сложнее, чем ЛП

- Наивное округление решения, полученного для ЛП-релаксации исходной задачи смешанно-целочисленного программирования, может привести к недопустимому или неоптимальному решению.
- Общая СЦЛП является NP-hard.
- Однако, если матрица коэффициентов СЦЛП является *полностью невырожденной матрицей*, то она может быть решена за полиномиальное время.

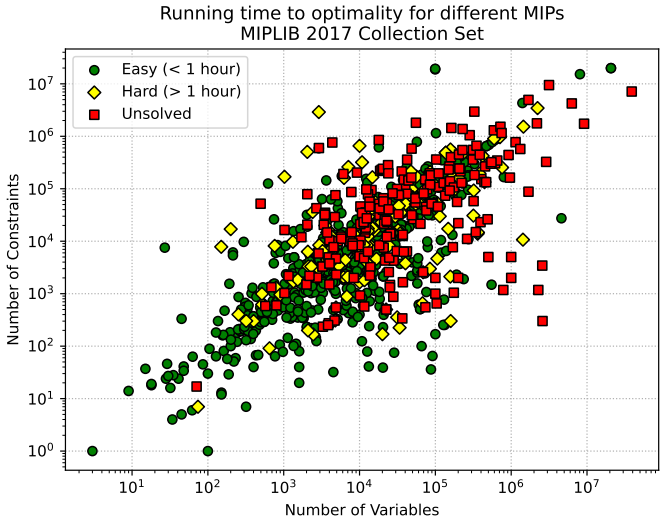
Непредсказуемая сложность СЦЛП

- Трудно предсказать, что будет решено быстро, а что потребует много времени



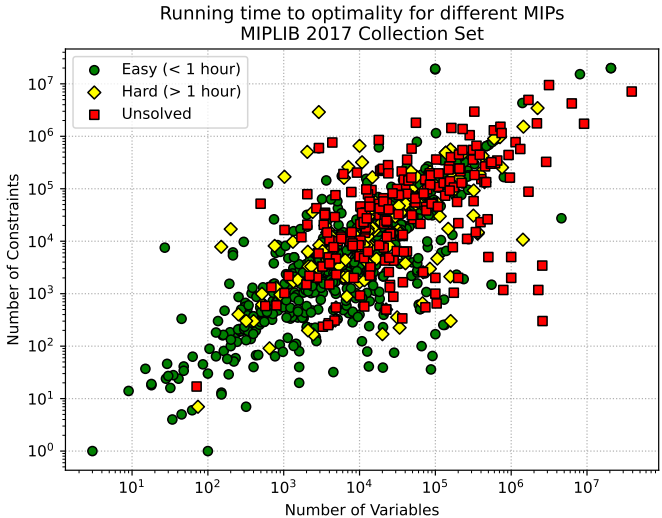
Непредсказуемая сложность СЦЛП

- Трудно предсказать, что будет решено быстро, а что потребует много времени
- 🔗Датасет



Непредсказуемая сложность СЦЛП

- Трудно предсказать, что будет решено быстро, а что потребует много времени
- 🔗Датасет
- Код



Прогресс аппаратного vs программного обеспечения

Что бы вы выбрали, если предположить, что вопрос поставлен корректно (вы можете скомпилировать ПО для любого оборудования, и задача в обоих случаях одна и та же)? Мы рассмотрим период с 1992 по 2023 год.

Аппаратное обеспечение

Решение СЦЛП с использованием старого ПО на современном оборудовании

Программное обеспечение

Решение СЦЛП с использованием современного ПО на старом оборудовании

Прогресс аппаратного vs программного обеспечения

Что бы вы выбрали, если предположить, что вопрос поставлен корректно (вы можете скомпилировать ПО для любого оборудования, и задача в обоих случаях одна и та же)? Мы рассмотрим период с 1992 по 2023 год.

Аппаратное обеспечение

Решение СЦЛП с использованием старого ПО на современном оборудовании

$\approx 1.664.510$ x ускорение

Закон Мура утверждает, что вычислительная мощность удваивается каждые 18 месяцев.

Программное обеспечение

Решение СЦЛП с использованием современного ПО на старом оборудовании

$\approx 2.349.000$ x ускорение

Р. Бикси провёл масштабный эксперимент по сравнению производительности всех версий CPLEX с 1992 по 2007 год и измерил общее прогресс ПО (29000 раз), позже (в 2009 году) он стал одним из основателей Gurobi Optimization, которое дало дополнительное ≈ 81 ускорение на СЦЛП.

2 R. Bixby report Recent study

Прогресс аппаратного vs программного обеспечения

Что бы вы выбрали, если предположить, что вопрос поставлен корректно (вы можете скомпилировать ПО для любого оборудования, и задача в обоих случаях одна и та же)? Мы рассмотрим период с 1992 по 2023 год.

Аппаратное обеспечение

Решение СЦЛП с использованием старого ПО на современном оборудовании

$\approx 1.664.510$ x ускорение

Закон Мура утверждает, что вычислительная мощность удваивается каждые 18 месяцев.

Программное обеспечение

Решение СЦЛП с использованием современного ПО на старом оборудовании

$\approx 2.349.000$ x ускорение

Р. Бикси провёл масштабный эксперимент по сравнению производительности всех версий CPLEX с 1992 по 2007 год и измерил общее прогресс ПО (29000 раз), позже (в 2009 году) он стал одним из основателей Gurobi Optimization, которое дало дополнительное ≈ 81 ускорение на СЦЛП.

Оказывается, что если вам нужно решить СЦЛП, лучше использовать старый компьютер и современные методы, чем наоборот, самый новый компьютер и методы начала 1990-х годов![2]

[2] R. Bixby report · Recent study

Источники

- Теория оптимизации (MATH4230) курс @ CUHK, профессор Тейюн Цень