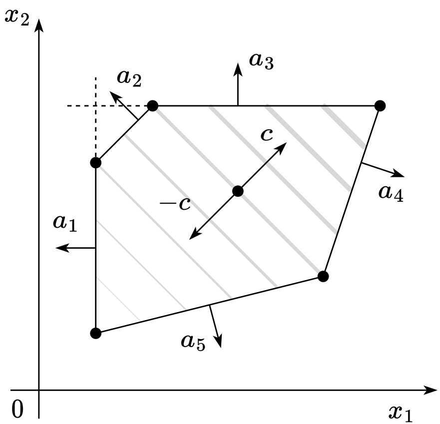
Linear Programming

Daniil Merkulov

Optimization for ML. Faculty of Computer Science. HSE University

Examples of linear programs

What is Linear Programming?

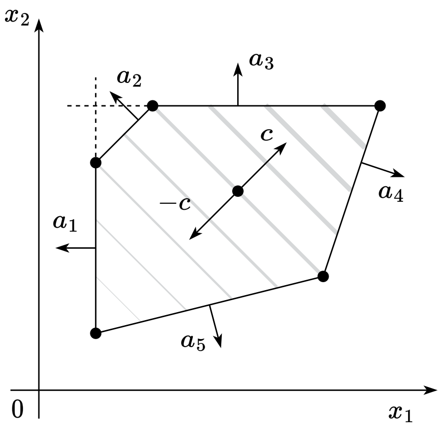


Generally speaking, all problems with linear objective and linear equalities/inequalities constraints could be considered as Linear Programming. However, there are some formulations.

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

for some vectors $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ and matrix $A \in \mathbb{R}^{m \times n}$. Where the inequalities are interpreted component-wise.

What is Linear Programming?



Generally speaking, all problems with linear objective and linear equalities/inequalities constraints could be considered as Linear Programming. However, there are some formulations.

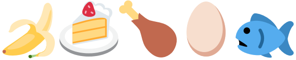
$$
\begin{aligned}
&\min_{x \in \mathbb{R}^n} c^\top x \\
\text{s.t. } & Ax \leq b
\end{aligned}
\tag{LP.Basic}
$$

for some vectors $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ and matrix $A \in \mathbb{R}^{m \times n}$. Where the inequalities are interpreted component-wise.

Standard form. This form seems to be the most intuitive and geometric in terms of visualization. Let us have vectors $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ and matrix $A \in \mathbb{R}^{m \times n}$.

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

Example: Diet problem



$$
\begin{array}{l}
\text{Proteins} \\
\text{Carbs} \\
\text{Fats} \\
\text{Calories} \\
\text{Vitamin D}
\end{array}
\begin{bmatrix}
\text{Amount per 100g} \\
W \in \mathbb{R}^{n \times p}
\end{bmatrix}
$$

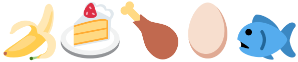
$c \in \mathbb{R}^p$, price per 100g

$r \in \mathbb{R}^n$, nutrient requirements

$x \in \mathbb{R}^p$, amount of products, 100g

$$
\begin{aligned}
&\min_{x \in \mathbb{R}^p} c^T x \\
&Wx \succeq r \\
&x \succeq 0
\end{aligned}
$$

Example: Diet problem



$$
\begin{array}{c}
\text{Proteins} \\ \text{Carbs} \\ \text{Fats} \\ \text{Calories} \\ \text{Vitamin D}
\end{array}
\left[
\begin{array}{c}
\\ \\ \text{Amount per 100g} \\ W \in \mathbb{R}^{n \times p} \\ \\
\end{array}
\right]
$$

$c \in \mathbb{R}^p$, price per 100g

$r \in \mathbb{R}^n$, nutrient requirements

$x \in \mathbb{R}^p$, amount of products, 100g

$$
\begin{aligned}
&\min_{x \in \mathbb{R}^p} c^T x \\
&Wx \succeq r \\
&x \succeq 0
\end{aligned}
$$

Imagine, that you have to construct a diet plan from some products: bananas, cakes, chicken, eggs, fish. Each of the products has its vector of nutrients. Thus, all the food information could be processed through the matrix W. Let us also assume, that we have the vector of requirements for each of nutrients $r \in \mathbb{R}^n$. We need to find the cheapest configuration of the diet, which meets all the requirements:

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

Open In Colab

Minimization of convex function as LP

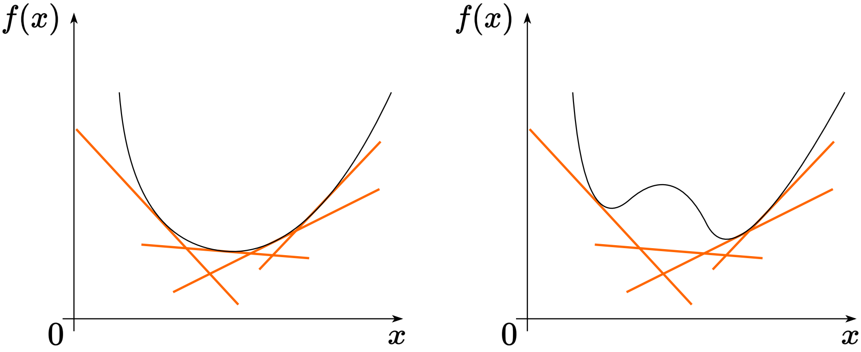


Рисунок 1: How LP can help with general convex problem

- The function is convex iff it can be represented as a pointwise maximum of linear functions.

Minimization of convex function as LP

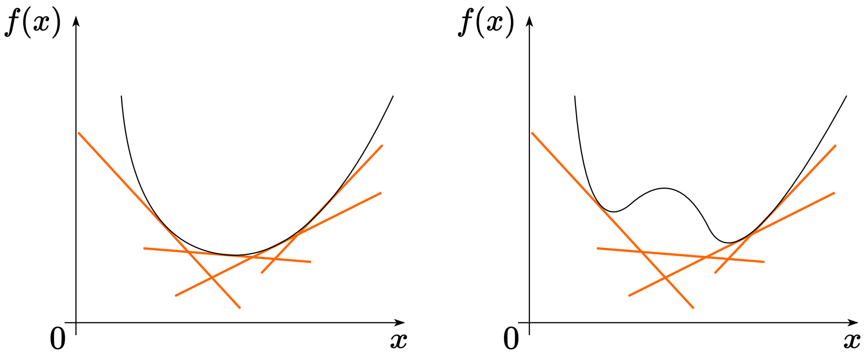


Рисунок 1: How LP can help with general convex problem

- The function is convex iff it can be represented as a pointwise maximum of linear functions.
- In high dimensions, the approximation may require too many functions.

Minimization of convex function as LP

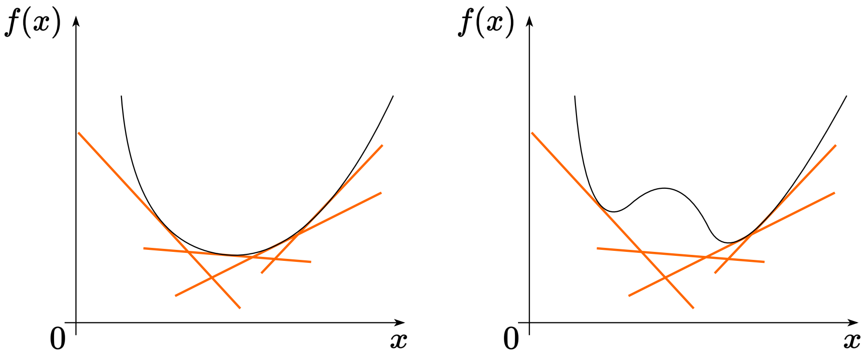


Рисунок 1: How LP can help with general convex problem

- The function is convex iff it can be represented as a pointwise maximum of linear functions.
- In high dimensions, the approximation may require too many functions.
- More efficient convex optimizers (not reducing to LP) exist.

Example: Transportation problem

The prototypical transportation problem deals with the distribution of a commodity from a set of sources to a set of destinations. The object is to minimize total transportation costs while satisfying constraints on the supplies available at each of the sources, and satisfying demand requirements at each of the destinations.

Рисунок 2: Western Europe Map. Open In Colab

Example: Transportation problem

Customer / Source	Arnhem [€/ton]	Gouda [€/ton]	Demand [tons]
London	n/a	2.5	125
Berlin	2.5	n/a	175
Maastricht	1.6	2.0	225
Amsterdam	1.4	1.0	250
Utrecht	0.8	1.0	225
The Hague	1.4	0.8	200
Supply [tons]	550 tons	700 tons	

$$\text{minimize:} \quad \text{Cost} = \sum_{c \in \text{Customers}} \sum_{s \in \text{Sources}} T[c,s] x[c,s]$$

Example: Transportation problem

Customer / Source	Arnhem [€/ton]	Gouda [€/ton]	Demand [tons]
London	n/a	2.5	125
Berlin	2.5	n/a	175
Maastricht	1.6	2.0	225
Amsterdam	1.4	1.0	250
Utrecht	0.8	1.0	225
The Hague	1.4	0.8	200
Supply [tons]	550 tons	700 tons	

$$\text{minimize:} \quad \text{Cost} = \sum_{c \in \text{Customers}} \sum_{s \in \text{Sources}} T[c,s]x[c,s]$$

$$\sum_{c \in \text{Customers}} x[c,s] \leq \text{Supply}[s] \qquad \forall s \in \text{Sources}$$

Example: Transportation problem

Customer / Source	Arnhem [€/ton]	Gouda [€/ton]	Demand [tons]
London	n/a	2.5	125
Berlin	2.5	n/a	175
Maastricht	1.6	2.0	225
Amsterdam	1.4	1.0	250
Utrecht	0.8	1.0	225
The Hague	1.4	0.8	200
Supply [tons]	550 tons	700 tons	

$$\text{minimize:} \quad \text{Cost} = \sum_{c \in \text{Customers}} \sum_{s \in \text{Sources}} T[c,s]x[c,s]$$

$$\sum_{c \in \text{Customers}} x[c,s] \leq \text{Supply}[s] \qquad \forall s \in \text{Sources}$$

$$\sum_{s \in \text{Sources}} x[c,s] = \text{Demand}[c] \qquad \forall c \in \text{Customers}$$

This can be represented in the following graph:

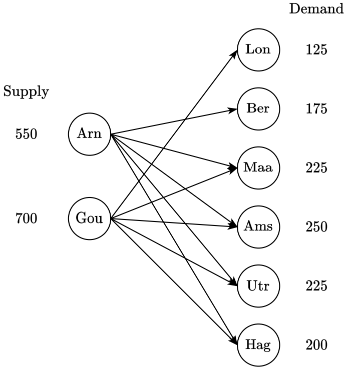


Рисунок 3: Graph associated with the problem

How to derive LP?

Basic transformations

- Max-min

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

Basic transformations

- Max-min

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

- Equality to inequality

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

Basic transformations

- Max-min

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

- Equality to inequality

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

- Inequality to equality by increasing the dimension of the problem by m.

$$Ax \leq b \leftrightarrow \begin{cases} Ax + z = b \\ z \geq 0 \end{cases}$$

Basic transformations

- Max-min

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

- Equality to inequality

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

- Inequality to equality by increasing the dimension of the problem by m.

$$Ax \leq b \leftrightarrow \begin{cases} Ax + z = b \\ z \geq 0 \end{cases}$$

- Unsigned variables to nonnegative variables.

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

Example: Chebyshev approximation problem

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

Could be equivalently written as an LP with the replacement of the maximum coordinate of a vector:

Example: Chebyshev approximation problem

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

Could be equivalently written as an LP with the replacement of the maximum coordinate of a vector:

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

ℓ_1 approximation problem

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

Could be equivalently written as an LP with the replacement of the sum of coordinates of a vector:

ℓ_1 approximation problem

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

Could be equivalently written as an LP with the replacement of the sum of coordinates of a vector:

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

Blending problem: from non-linear constraints to LP [1]

A manufacturing facility receives an order for 100 liters of a solution with a specific composition (e.g., 4% sugar solution). The facility has on hand:

Component	Sugar (%)	Cost ($/l)
Concentrate A (Dobry cola)	10.6	1.25
Concentrate B (Sever cola)	4.5	1.02
Water (Psyzh)	0.0	0.62

Goal: Find the minimum-cost blend to meet the order.

Objective Function

Blending problem: from non-linear constraints to LP [1]

A manufacturing facility receives an order for 100 liters of a solution with a specific composition (e.g., 4% sugar solution). The facility has on hand:

Component	Sugar (%)	Cost ($/l)
Concentrate A (Dobry cola)	10.6	1.25
Concentrate B (Sever cola)	4.5	1.02
Water (Psyzh)	0.0	0.62

Goal: Find the minimum-cost blend to meet the order.

Objective Function

Minimize cost:

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

where x_c is the volume of component c used, and P_c is its price.

Blending problem: from non-linear constraints to LP [1]

A manufacturing facility receives an order for 100 liters of a solution with a specific composition (e.g., 4% sugar solution). The facility has on hand:

Component	Sugar (%)	Cost ($/l)
Concentrate A (Dobry cola)	10.6	1.25
Concentrate B (Sever cola)	4.5	1.02
Water (Psyzh)	0.0	0.62

Goal: Find the minimum-cost blend to meet the order.

Objective Function

Minimize cost:

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

where x_c is the volume of component c used, and P_c is its price.

Volume Constraint

Blending problem: from non-linear constraints to LP [1]

A manufacturing facility receives an order for 100 liters of a solution with a specific composition (e.g., 4% sugar solution). The facility has on hand:

Component	Sugar (%)	Cost ($/l)
Concentrate A (Dobry cola)	10.6	1.25
Concentrate B (Sever cola)	4.5	1.02
Water (Psyzh)	0.0	0.62

Goal: Find the minimum-cost blend to meet the order.

Objective Function

Minimize cost:

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

where x_c is the volume of component c used, and P_c is its price.

Volume Constraint

Ensure total volume V:

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

Composition Constraint

Blending problem: from non-linear constraints to LP [1]

A manufacturing facility receives an order for 100 liters of a solution with a specific composition (e.g., 4% sugar solution). The facility has on hand:

Component	Sugar (%)	Cost ($/l)
Concentrate A (Dobry cola)	10.6	1.25
Concentrate B (Sever cola)	4.5	1.02
Water (Psyzh)	0.0	0.62

Goal: Find the minimum-cost blend to meet the order.

Objective Function

Minimize cost:

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

where x_c is the volume of component c used, and P_c is its price.

Volume Constraint

Ensure total volume V:

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

Composition Constraint

Ensure 4% sugar content:

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

[1] Source

Blending problem: from non-linear constraints to LP [1]

A manufacturing facility receives an order for 100 liters of a solution with a specific composition (e.g., 4% sugar solution). The facility has on hand:

Component	Sugar (%)	Cost ($/l)
Concentrate A (Dobry cola)	10.6	1.25
Concentrate B (Sever cola)	4.5	1.02
Water (Psyzh)	0.0	0.62

Goal: Find the minimum-cost blend to meet the order.

Objective Function

Minimize cost:

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

where x_c is the volume of component c used, and P_c is its price.

Volume Constraint

Ensure total volume V:

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

Composition Constraint

Ensure 4% sugar content:

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

Linearized version:

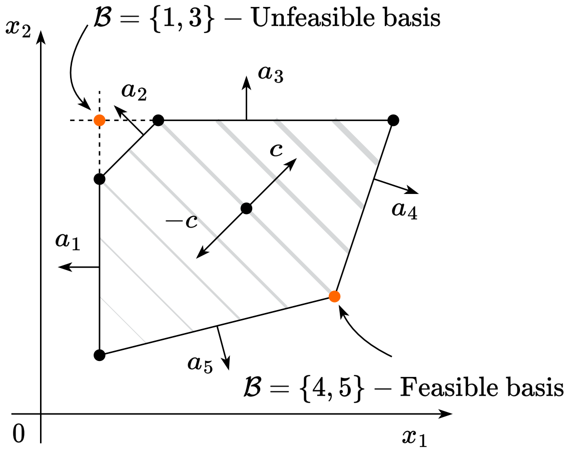
$$0 = \sum_{c \in C} x_c (A_c - \bar{A})$$

This can be solved using linear programming.
Source code

1 Source

Simplex method

Geometry of simplex method

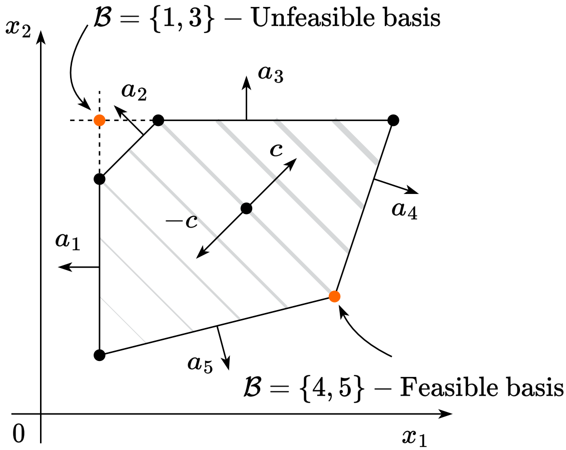


We will consider the following simple formulation of LP, which is, in fact, dual to the Standard form:

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

- Definition: a **basis** $\mathcal{B}$ is a subset of n (integer) numbers between 1 and m, so that $\text{rank} A_{\mathcal{B}} = n$.

Geometry of simplex method

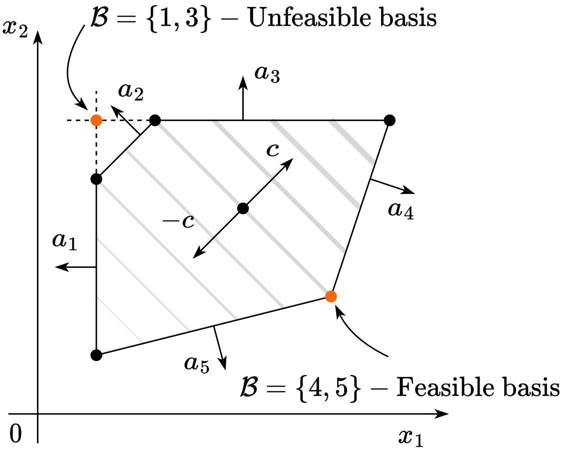


We will consider the following simple formulation of LP, which is, in fact, dual to the Standard form:

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

- Definition: a **basis** $\mathcal{B}$ is a subset of n (integer) numbers between 1 and m, so that $\text{rank} A_{\mathcal{B}} = n$.
- Note, that we can associate submatrix $A_{\mathcal{B}}$ and corresponding right-hand side $b_{\mathcal{B}}$ with the basis $\mathcal{B}$.

Geometry of simplex method

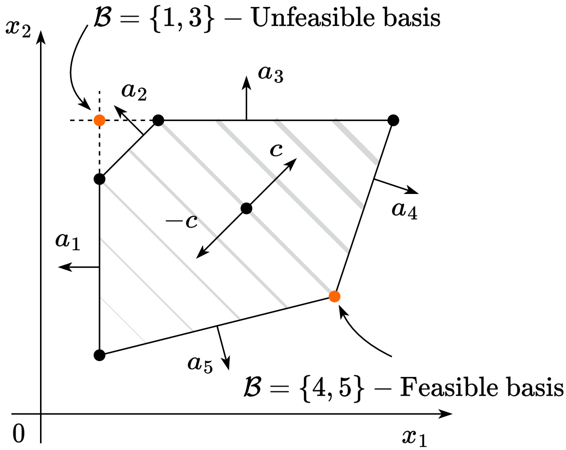


We will consider the following simple formulation of LP, which is, in fact, dual to the Standard form:

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

- Definition: a **basis** $\mathcal{B}$ is a subset of n (integer) numbers between 1 and m, so that $\text{rank} A_{\mathcal{B}} = n$.
- Note, that we can associate submatrix $A_{\mathcal{B}}$ and corresponding right-hand side $b_{\mathcal{B}}$ with the basis $\mathcal{B}$.
- Also, we can derive a point of intersection of all these hyperplanes from the basis: $x_{\mathcal{B}} = A_{\mathcal{B}}^{-1} b_{\mathcal{B}}$.

Geometry of simplex method

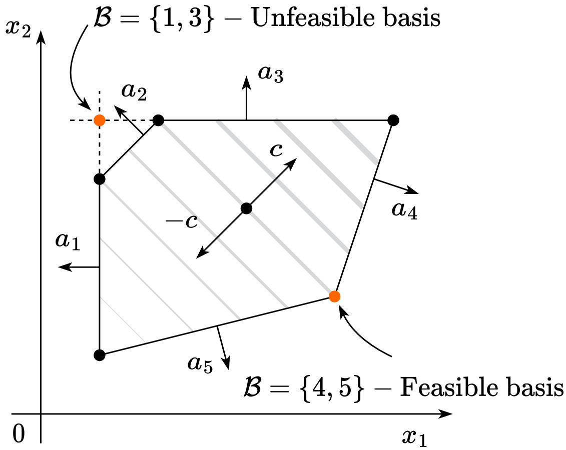


We will consider the following simple formulation of LP, which is, in fact, dual to the Standard form:

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

- Definition: a **basis** $\mathcal{B}$ is a subset of n (integer) numbers between 1 and m, so that rank$A_{\mathcal{B}} = n$.
- Note, that we can associate submatrix $A_{\mathcal{B}}$ and corresponding right-hand side $b_{\mathcal{B}}$ with the basis $\mathcal{B}$.
- Also, we can derive a point of intersection of all these hyperplanes from the basis: $x_{\mathcal{B}} = A^{-1}_{\mathcal{B}} b_{\mathcal{B}}$.
- If $Ax_{\mathcal{B}} \leq b$, then basis $\mathcal{B}$ is **feasible**.

Geometry of simplex method

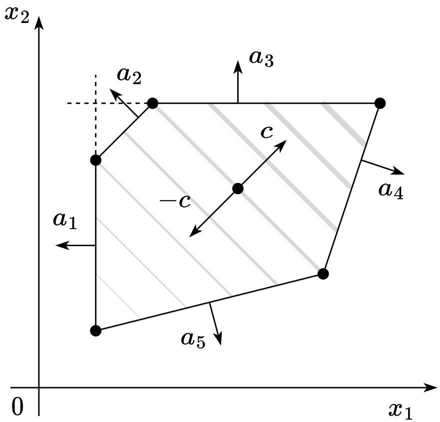


We will consider the following simple formulation of LP, which is, in fact, dual to the Standard form:

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

- Definition: a **basis** $\mathcal{B}$ is a subset of n (integer) numbers between 1 and m, so that $\text{rank} A_{\mathcal{B}} = n$.
- Note, that we can associate submatrix $A_{\mathcal{B}}$ and corresponding right-hand side $b_{\mathcal{B}}$ with the basis $\mathcal{B}$.
- Also, we can derive a point of intersection of all these hyperplanes from the basis: $x_{\mathcal{B}} = A^{-1}_{\mathcal{B}} b_{\mathcal{B}}$.
- If $Ax_{\mathcal{B}} \leq b$, then basis $\mathcal{B}$ is **feasible**.
- A basis $\mathcal{B}$ is optimal if $x_{\mathcal{B}}$ is an optimum of the LP.Inequality.

The solution of LP if exists lies in the corner

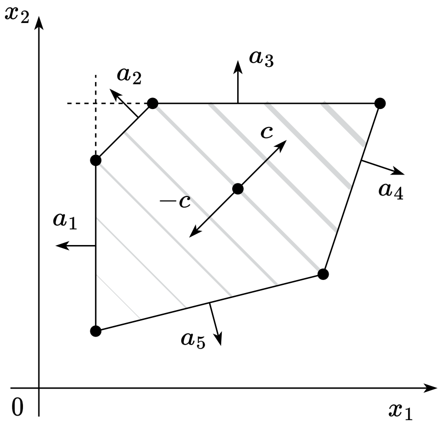


Theorem

1. If Standard LP has a nonempty feasible region, then there is at least one basic feasible point

The high-level idea of the simplex method is following:

The solution of LP if exists lies in the corner

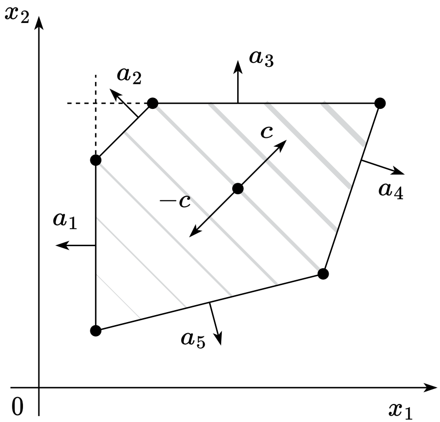


Theorem

1. If Standard LP has a nonempty feasible region, then there is at least one basic feasible point
2. If Standard LP has solutions, then at least one such solution is a basic optimal point.

The high-level idea of the simplex method is following:

The solution of LP if exists lies in the corner

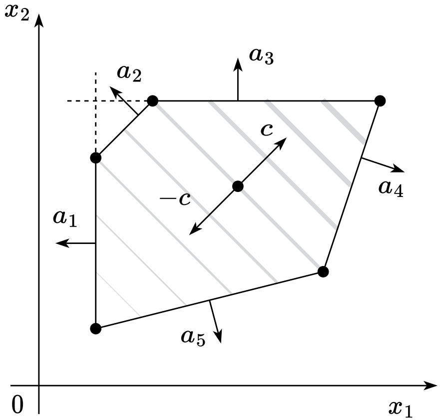


Theorem

1. If Standard LP has a nonempty feasible region, then there is at least one basic feasible point
2. If Standard LP has solutions, then at least one such solution is a basic optimal point.
3. If Standard LP is feasible and bounded, then it has an optimal solution.

The high-level idea of the simplex method is following:

The solution of LP if exists lies in the corner

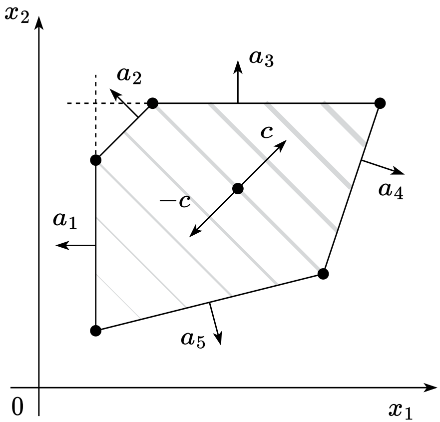


Theorem

1. If Standard LP has a nonempty feasible region, then there is at least one basic feasible point
2. If Standard LP has solutions, then at least one such solution is a basic optimal point.
3. If Standard LP is feasible and bounded, then it has an optimal solution.

The high-level idea of the simplex method is following:

The solution of LP if exists lies in the corner



Theorem

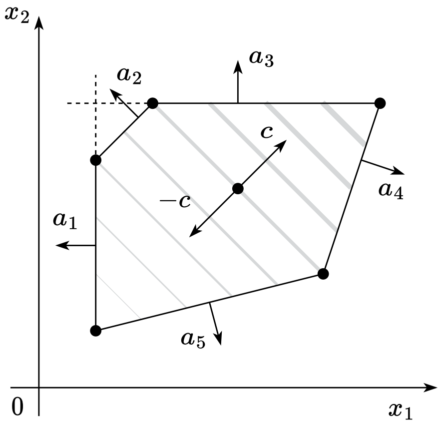
1. If Standard LP has a nonempty feasible region, then there is at least one basic feasible point
2. If Standard LP has solutions, then at least one such solution is a basic optimal point.
3. If Standard LP is feasible and bounded, then it has an optimal solution.

For proof see Numerical Optimization by Jorge Nocedal and Stephen J. Wright theorem 13.2

The high-level idea of the simplex method is following:

- Ensure, that you are in the corner.

The solution of LP if exists lies in the corner



Theorem

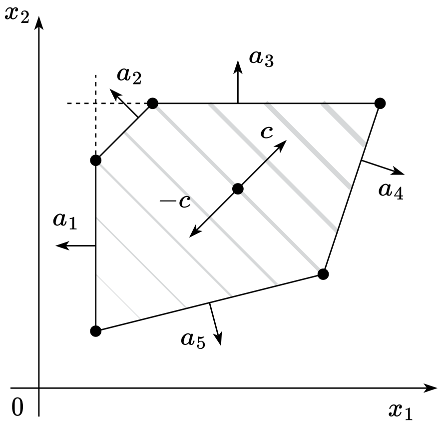
1. If Standard LP has a nonempty feasible region, then there is at least one basic feasible point
2. If Standard LP has solutions, then at least one such solution is a basic optimal point.
3. If Standard LP is feasible and bounded, then it has an optimal solution.

For proof see Numerical Optimization by Jorge Nocedal and Stephen J. Wright theorem 13.2

The high-level idea of the simplex method is following:

- Ensure, that you are in the corner.
- Check optimality.

The solution of LP if exists lies in the corner



Theorem

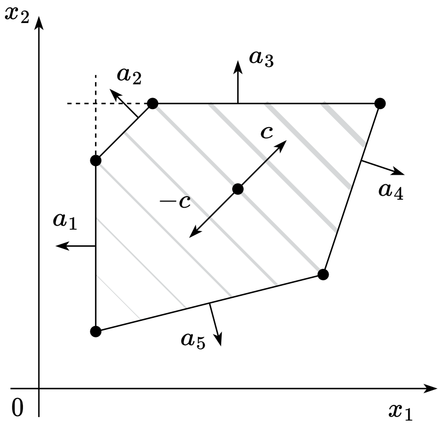
1. If Standard LP has a nonempty feasible region, then there is at least one basic feasible point
2. If Standard LP has solutions, then at least one such solution is a basic optimal point.
3. If Standard LP is feasible and bounded, then it has an optimal solution.

For proof see Numerical Optimization by Jorge Nocedal and Stephen J. Wright theorem 13.2

The high-level idea of the simplex method is following:

- Ensure, that you are in the corner.
- Check optimality.
- If necessary, switch the corner (change the basis).

The solution of LP if exists lies in the corner



Theorem

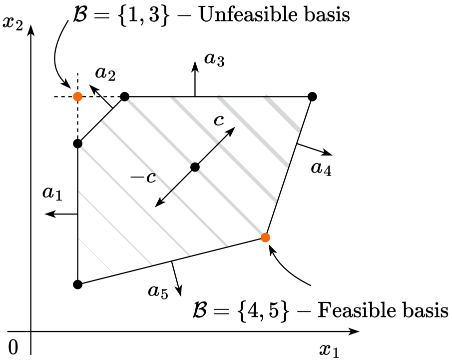
1. If Standard LP has a nonempty feasible region, then there is at least one basic feasible point
2. If Standard LP has solutions, then at least one such solution is a basic optimal point.
3. If Standard LP is feasible and bounded, then it has an optimal solution.

For proof see Numerical Optimization by Jorge Nocedal and Stephen J. Wright theorem 13.2

The high-level idea of the simplex method is following:

- Ensure, that you are in the corner.
- Check optimality.
- If necessary, switch the corner (change the basis).
- Repeat until converge.

Optimal basis



Since we have a basis, we can decompose our objective vector c in this basis and find the scalar coefficients $\lambda_{\mathcal{B}}$:

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

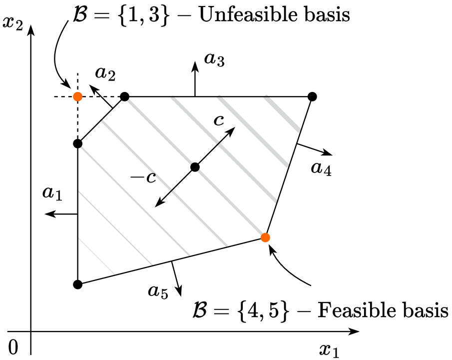
Theorem

If all components of $\lambda_{\mathcal{B}}$ are non-positive and $\mathcal{B}$ is feasible, then $\mathcal{B}$ is optimal.

Proof

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

Optimal basis



Since we have a basis, we can decompose our objective vector c in this basis and find the scalar coefficients $\lambda_\mathcal{B}$:

$$\lambda^T_\mathcal{B} A_\mathcal{B} = c^T \leftrightarrow \lambda^T_\mathcal{B} = c^T A^{-1}_\mathcal{B}$$

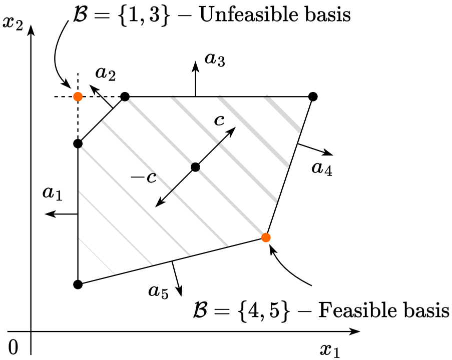
Theorem

If all components of $\lambda_\mathcal{B}$ are non-positive and $\mathcal{B}$ is feasible, then $\mathcal{B}$ is optimal.

Proof

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

Optimal basis



Since we have a basis, we can decompose our objective vector c in this basis and find the scalar coefficients $\lambda_{\mathcal{B}}$:

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

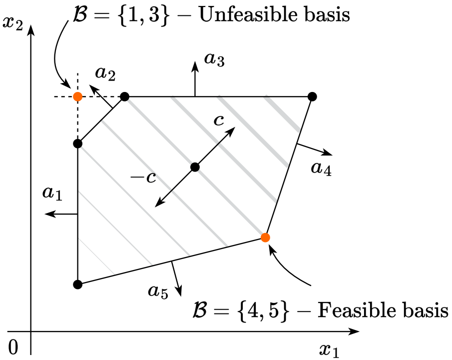
Theorem

If all components of $\lambda_{\mathcal{B}}$ are non-positive and $\mathcal{B}$ is feasible, then $\mathcal{B}$ is optimal.

Proof

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

Optimal basis



Since we have a basis, we can decompose our objective vector c in this basis and find the scalar coefficients $\lambda_{\mathcal{B}}$:

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

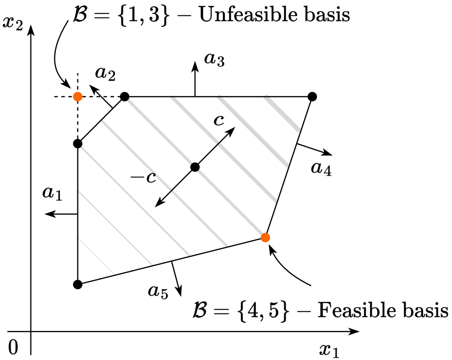
Theorem

If all components of $\lambda_{\mathcal{B}}$ are non-positive and $\mathcal{B}$ is feasible, then $\mathcal{B}$ is optimal.

Proof

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

Optimal basis



Since we have a basis, we can decompose our objective vector c in this basis and find the scalar coefficients $\lambda_{\mathcal{B}}$:

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

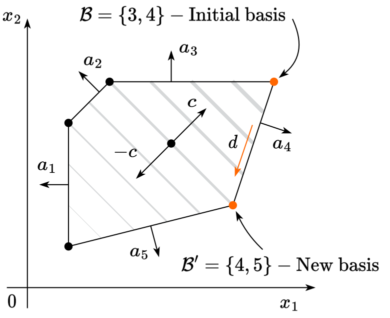
Theorem

If all components of $\lambda_{\mathcal{B}}$ are non-positive and $\mathcal{B}$ is feasible, then $\mathcal{B}$ is optimal.

Proof

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

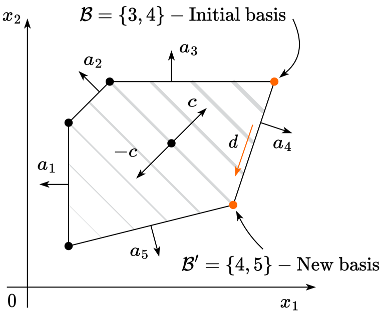
Changing basis



Suppose, some of the coefficients of $\lambda_{\mathcal{B}}$ are positive. Then we need to go through the edge of the polytope to the new vertex (i.e., switch the basis)

- Suppose, we have a basis $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$

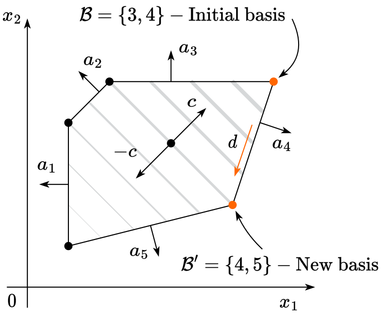
Changing basis



Suppose, some of the coefficients of $\lambda_{\mathcal{B}}$ are positive. Then we need to go through the edge of the polytope to the new vertex (i.e., switch the basis)

- Suppose, we have a basis $\mathcal{B}$: $\lambda^T_{\mathcal{B}} = c^T A^{-1}_{\mathcal{B}}$
- Let's assume, that $\lambda^k_{\mathcal{B}} > 0$. We'd like to drop k from the basis and form a new one:

Changing basis

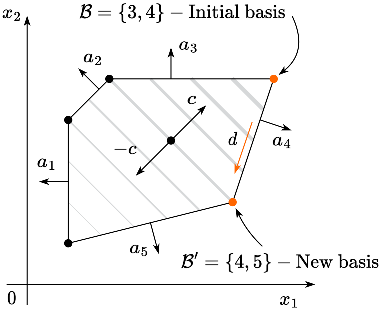


Suppose, some of the coefficients of $\lambda_{\mathcal{B}}$ are positive. Then we need to go through the edge of the polytope to the new vertex (i.e., switch the basis)

- Suppose, we have a basis $\mathcal{B}$: $\lambda^T_{\mathcal{B}} = c^T A^{-1}_{\mathcal{B}}$
- Let's assume, that $\lambda^k_{\mathcal{B}} > 0$. We'd like to drop k from the basis and form a new one:

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

Changing basis

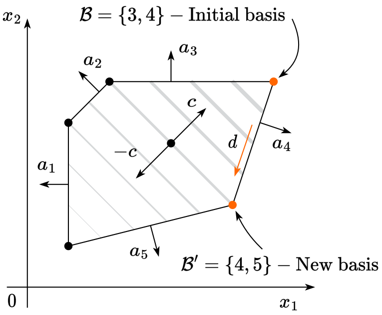


Suppose, some of the coefficients of $\lambda_{\mathcal{B}}$ are positive. Then we need to go through the edge of the polytope to the new vertex (i.e., switch the basis)

- Suppose, we have a basis $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Let's assume, that $\lambda_{\mathcal{B}}^k > 0$. We'd like to drop k from the basis and form a new one:

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

Changing basis

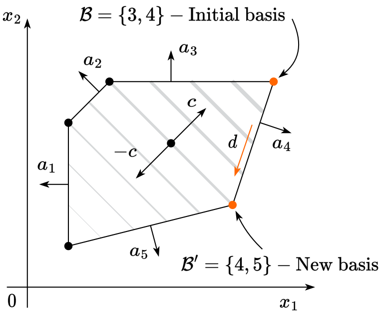


Suppose, some of the coefficients of $\lambda_{\mathcal{B}}$ are positive. Then we need to go through the edge of the polytope to the new vertex (i.e., switch the basis)

- Suppose, we have a basis $\mathcal{B}$: $\lambda^T_{\mathcal{B}} = c^T A_{\mathcal{B}}^{-1}$
- Let's assume, that $\lambda^k_{\mathcal{B}} > 0$. We'd like to drop k from the basis and form a new one:

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

Changing basis

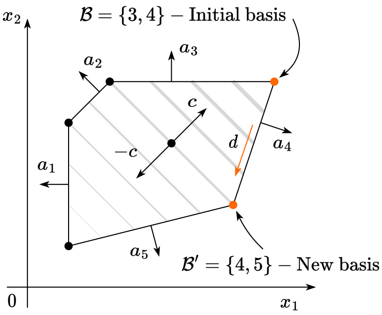


Suppose, some of the coefficients of $\lambda_{\mathcal{B}}$ are positive. Then we need to go through the edge of the polytope to the new vertex (i.e., switch the basis)

- Suppose, we have a basis $\mathcal{B}$: $\lambda^T_{\mathcal{B}} = c^T A^{-1}_{\mathcal{B}}$
- Let's assume, that $\lambda^k_{\mathcal{B}} > 0$. We'd like to drop k from the basis and form a new one:

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

Changing basis

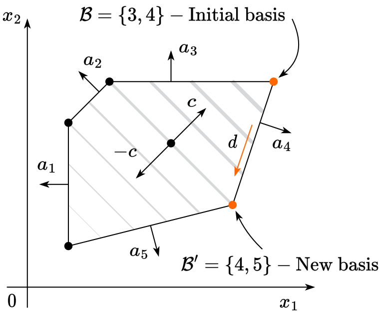


Suppose, some of the coefficients of $\lambda_{\mathcal{B}}$ are positive. Then we need to go through the edge of the polytope to the new vertex (i.e., switch the basis)

- Suppose, we have a basis $\mathcal{B}$: $\lambda^T_{\mathcal{B}} = c^T A^{-1}_{\mathcal{B}}$
- Let's assume, that $\lambda^k_{\mathcal{B}} > 0$. We'd like to drop k from the basis and form a new one:

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

Changing basis



Suppose, some of the coefficients of $\lambda_{\mathcal{B}}$ are positive. Then we need to go through the edge of the polytope to the new vertex (i.e., switch the basis)

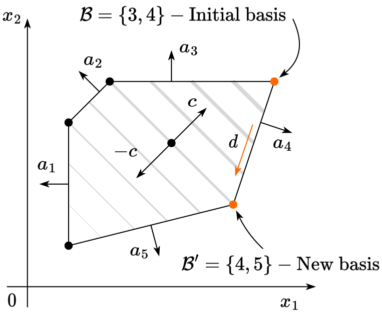
- Suppose, we have a basis $\mathcal{B}$: $\lambda^T_{\mathcal{B}} = c^T A^{-1}_{\mathcal{B}}$
- Let's assume, that $\lambda^k_{\mathcal{B}} > 0$. We'd like to drop k from the basis and form a new one:

$$
\begin{cases} A_{\mathcal{B} \setminus \{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d = \lambda^T_{\mathcal{B}} A_{\mathcal{B}} d = \sum\limits_{i=1}^n \lambda^i_{\mathcal{B}} (A_{\mathcal{B}} d)^i = -\lambda^k_{\mathcal{B}} < 0
$$

- For all $j \notin \mathcal{B}$ calculate the projection stepsize:

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

Changing basis



Suppose, some of the coefficients of $\lambda_{\mathcal{B}}$ are positive. Then we need to go through the edge of the polytope to the new vertex (i.e., switch the basis)

- Suppose, we have a basis $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Let's assume, that $\lambda_{\mathcal{B}}^k > 0$. We'd like to drop k from the basis and form a new one:

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

- For all $j \notin \mathcal{B}$ calculate the projection stepsize:

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

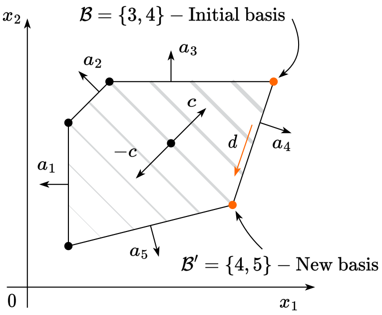
- Define the new vertex, that you will add to the new basis:

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

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

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

Changing basis



Suppose, some of the coefficients of $\lambda_\mathcal{B}$ are positive. Then we need to go through the edge of the polytope to the new vertex (i.e., switch the basis)

- Suppose, we have a basis $\mathcal{B}$: $\lambda^T_\mathcal{B} = c^T A^{-1}_\mathcal{B}$
- Let's assume, that $\lambda^k_\mathcal{B} > 0$. We'd like to drop k from the basis and form a new one:

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

- For all $j \notin \mathcal{B}$ calculate the projection stepsize:

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

- Define the new vertex, that you will add to the new basis:

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

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

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

- Note, that changing basis implies objective function decreasing

$$c^T x_{\mathcal{B}'} = c^T (x_\mathcal{B} + \mu_t d) = c^T x_\mathcal{B} + \mu_t c^T d$$

Finding an initial basic feasible solution

We aim to solve the following problem:

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

The proposed algorithm requires an initial basic feasible solution and corresponding basis.

Finding an initial basic feasible solution

We aim to solve the following problem:

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

The proposed algorithm requires an initial basic feasible solution and corresponding basis.

We start by reformulating the problem:

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

Finding an initial basic feasible solution

We aim to solve the following problem:

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

The proposed algorithm requires an initial basic feasible solution and corresponding basis.

We start by reformulating the problem:

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

Given the solution of Problem 2 the solution of Problem 1 can be recovered and vice versa

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

Now we will try to formulate a new LP problem, which solution will be a basic feasible point for Problem 2. This means, that we first run the Simplex method for the Phase-1 problem and run the Phase-2 problem with the known starting point. Note, that the basic feasible solution for Phase-1 should be somehow easily established.

Finding an initial basic feasible solution

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

Finding an initial basic feasible solution

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

$$
\begin{aligned}
\min_{\xi \in \mathbb{R}^m, y \in \mathbb{R}^n, z \in \mathbb{R}^n} \; & \sum_{i=1}^m \xi_i \\
\text{s.t. } & Ay - Az \leq b + \xi \\
& y \geq 0, z \geq 0, \xi \geq 0
\end{aligned}
\tag{Phase-1}
$$

Finding an initial basic feasible solution

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

$$
\begin{aligned}
&\min_{\xi \in \mathbb{R}^m, y \in \mathbb{R}^n, z \in \mathbb{R}^n} \sum_{i=1}^m \xi_i \\
\text{s.t. } & Ay - Az \leq b + \xi \\
& y \geq 0, z \geq 0, \xi \geq 0
\end{aligned}
\tag{Phase-1}
$$

- If the Phase-2 (Main LP) problem has a feasible solution, then the Phase-1 optimum is zero (i.e. all slacks ξ_i are zero).
  Proof: trivial check.

Finding an initial basic feasible solution

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

$$
\begin{aligned}
&\min_{\xi \in \mathbb{R}^m, y \in \mathbb{R}^n, z \in \mathbb{R}^n} \sum_{i=1}^m \xi_i \\
\text{s.t. } & Ay - Az \leq b + \xi \\
& y \geq 0, z \geq 0, \xi \geq 0
\end{aligned}
\tag{Phase-1}
$$

- If the Phase-2 (Main LP) problem has a feasible solution, then the Phase-1 optimum is zero (i.e. all slacks ξ_i are zero).
  Proof: trivial check.
- If Phase-1 optimum is zero (i.e. all slacks ξ_i are zero), then we get a feasible basis for Phase-2.
  Proof: trivial check.

Finding an initial basic feasible solution

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

$$
\begin{aligned}
&\min_{\xi \in \mathbb{R}^m, y \in \mathbb{R}^n, z \in \mathbb{R}^n} \sum_{i=1}^{m} \xi_i \\
\text{s.t. } & Ay - Az \leq b + \xi \\
& y \geq 0, z \geq 0, \xi \geq 0
\end{aligned}
\tag{Phase-1}
$$

- If the Phase-2 (Main LP) problem has a feasible solution, then the Phase-1 optimum is zero (i.e. all slacks ξ_i are zero).
  Proof: trivial check.
- If Phase-1 optimum is zero (i.e. all slacks ξ_i are zero), then we get a feasible basis for Phase-2.
  Proof: trivial check.

Finding an initial basic feasible solution

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

$$
\begin{aligned}
\min_{\xi \in \mathbb{R}^m, y \in \mathbb{R}^n, z \in \mathbb{R}^n} \; & \sum_{i=1}^m \xi_i \\
\text{s.t. } & Ay - Az \leq b + \xi \\
& y \geq 0, z \geq 0, \xi \geq 0
\end{aligned}
\tag{Phase-1}
$$

- If the Phase-2 (Main LP) problem has a feasible solution, then the Phase-1 optimum is zero (i.e. all slacks ξ_i are zero).
  Proof: trivial check.
- If Phase-1 optimum is zero (i.e. all slacks ξ_i are zero), then we get a feasible basis for Phase-2.
  Proof: trivial check.

- Now we know, that if we can solve a Phase-1 problem then we will either find a starting point for the simplex method in the original method (if slacks are zero) or verify that the original problem was infeasible (if slacks are non-zero).

Finding an initial basic feasible solution

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

$$
\begin{aligned}
\min_{\xi \in \mathbb{R}^m, y \in \mathbb{R}^n, z \in \mathbb{R}^n} \; & \sum_{i=1}^m \xi_i \\
\text{s.t. } & Ay - Az \leq b + \xi \\
& y \geq 0, z \geq 0, \xi \geq 0
\end{aligned}
\tag{Phase-1}
$$

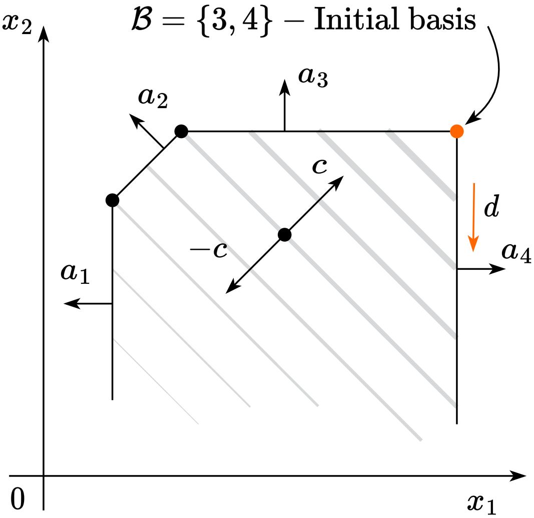
- If the Phase-2 (Main LP) problem has a feasible solution, then the Phase-1 optimum is zero (i.e. all slacks ξ_i are zero).
  Proof: trivial check.
- If Phase-1 optimum is zero (i.e. all slacks ξ_i are zero), then we get a feasible basis for Phase-2.
  Proof: trivial check.

- Now we know, that if we can solve a Phase-1 problem then we will either find a starting point for the simplex method in the original method (if slacks are zero) or verify that the original problem was infeasible (if slacks are non-zero).
- But how to solve Phase-1? It has a basic feasible solution (the problem has $2n + m$ variables and the point below ensures $2n + m$ inequalities are satisfied as equalities (active).)

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

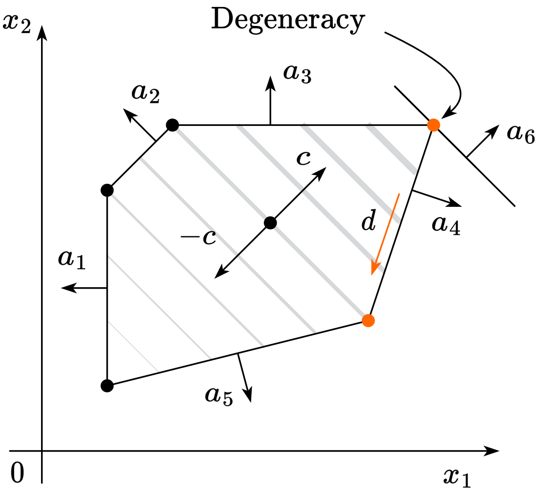
Convergence of the Simplex method

Unbounded budget set



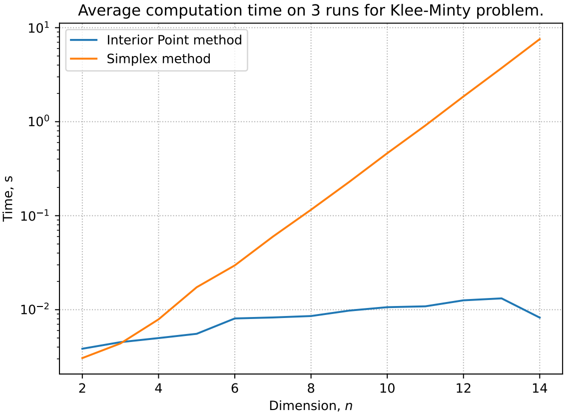
In this case, all μ_j will be negative.

Degeneracy



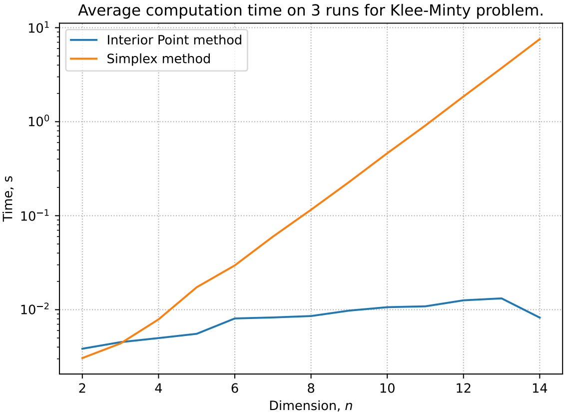
One needs to handle degenerate corners carefully. If no degeneracy exists, one can guarantee a monotonic decrease of the objective function on each iteration.

Exponential convergence



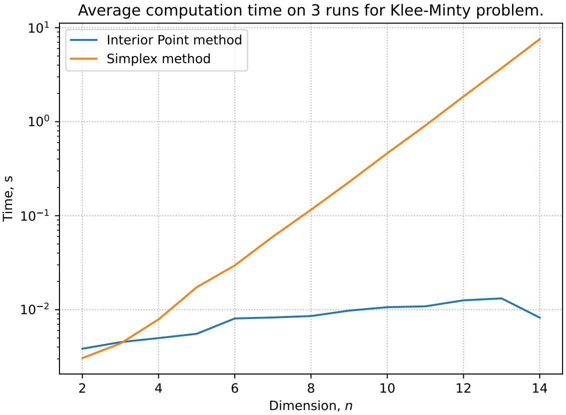
- A wide variety of applications could be formulated as linear programming.

Exponential convergence



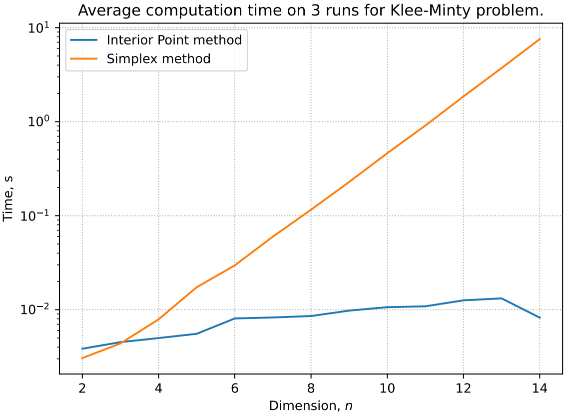
- A wide variety of applications could be formulated as linear programming.
- The simplex method is simple but could work exponentially long.

Exponential convergence



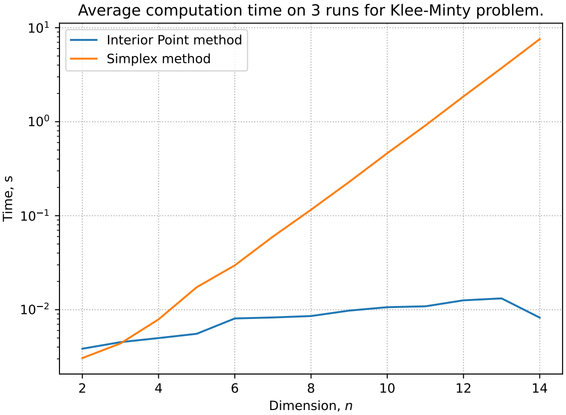
- A wide variety of applications could be formulated as linear programming.
- The simplex method is simple but could work exponentially long.
- Khachiyan's ellipsoid method (1979) is the first to be proven to run at polynomial complexity for LPs. However, it is usually slower than simplex in real problems.

Exponential convergence



- A wide variety of applications could be formulated as linear programming.
- The simplex method is simple but could work exponentially long.
- Khachiyan's ellipsoid method (1979) is the first to be proven to run at polynomial complexity for LPs. However, it is usually slower than simplex in real problems.
- Major breakthrough - Narendra Karmarkar's method for solving LP (1984) using the interior point method.

Exponential convergence



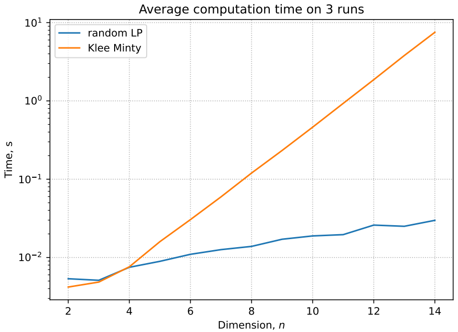
- A wide variety of applications could be formulated as linear programming.
- The simplex method is simple but could work exponentially long.
- Khachiyan's ellipsoid method (1979) is the first to be proven to run at polynomial complexity for LPs. However, it is usually slower than simplex in real problems.
- Major breakthrough - Narendra Karmarkar's method for solving LP (1984) using the interior point method.
- Interior point methods are the last word in this area. However, good implementations of simplex-based methods and interior point methods are similar for routine applications of linear programming.

Klee Minty example

Since the number of edge points is finite, the algorithm should converge (except for some degenerate cases, which are not covered here). However, the convergence could be exponentially slow, due to the high number of edges. There is the following iconic example when the simplex method should perform exactly all vertexes.

In the following problem, the simplex method needs to check $2^n - 1$ vertexes with $x_0 = 0$.

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



Duality in Linear Programming

Duality

Primal problem:

$$
\begin{aligned}
&\min_{x \in \mathbb{R}^n} c^\top x \\
\text{s.t. } & Ax = b \\
& x_i \geq 0, \; i = 1, \dots, n
\end{aligned}
\tag{3}
$$

Duality

Primal problem:

$$
\begin{aligned}
&\min_{x \in \mathbb{R}^n} c^\top x \\
\text{s.t. } & Ax = b \\
& x_i \geq 0, \; i = 1, \dots, n
\end{aligned}
\tag{3}
$$

KKT for optimal x^*, ν^*, λ^*:

$$
\begin{aligned}
&L(x, \nu, \lambda) = c^T x + \nu^T (Ax - b) - \lambda^T x \\
&- A^T \nu^* + \lambda^* = c \\
&Ax^* = b \\
&x^* \succeq 0 \\
&\lambda^* \succeq 0 \\
&\lambda_i^* x_i^* = 0
\end{aligned}
$$

Duality

Primal problem:

$$
\begin{aligned}
&\min_{x \in \mathbb{R}^n} c^\top x \\
\text{s.t. } & Ax = b \\
& x_i \geq 0, \; i = 1, \dots, n
\end{aligned}
\tag{3}
$$

KKT for optimal x^*, ν^*, λ^*:

$$
\begin{aligned}
&L(x, \nu, \lambda) = c^T x + \nu^T (Ax - b) - \lambda^T x \\
&-A^T \nu^* + \lambda^* = c \\
&Ax^* = b \\
&x^* \succeq 0 \\
&\lambda^* \succeq 0 \\
&\lambda_i^* x_i^* = 0
\end{aligned}
$$

Has the following dual:

$$
\begin{aligned}
&\max_{\nu \in \mathbb{R}^m} -b^\top \nu \\
\text{s.t. } & -A^T \nu \preceq c
\end{aligned}
\tag{4}
$$

Find the dual problem to the problem above (it should be the original LP). Also, write down KKT for the dual problem, to ensure, they are identical to the primal KKT.

Strong duality in linear programming

(i) If either problem Уравнение 3 or Уравнение 4 has a (finite) solution, then so does the other, and the objective values are equal.

Strong duality in linear programming

(i) If either problem Уравнение 3 or Уравнение 4 has a (finite) solution, then so does the other, and the objective values are equal.

(ii) If either problem Уравнение 3 or Уравнение 4 is unbounded, then the other problem is infeasible.

Strong duality in linear programming

(i) If either problem Уравнение 3 or Уравнение 4 has a (finite) solution, then so does the other, and the objective values are equal.

(ii) If either problem Уравнение 3 or Уравнение 4 is unbounded, then the other problem is infeasible.

Strong duality in linear programming

(i) If either problem Уравнение 3 or Уравнение 4 has a (finite) solution, then so does the other, and the objective values are equal.

(ii) If either problem Уравнение 3 or Уравнение 4 is unbounded, then the other problem is infeasible.

PROOF. For (i), suppose that Уравнение 3 has a finite optimal solution x^*. It follows from KKT that there are optimal vectors λ^* and ν^* such that (x^*, ν^*, λ^*) satisfies KKT. We noted above that KKT for Уравнение 3 and Уравнение 4 are equivalent. Moreover, $c^T x^* = (-A^T \nu^* + \lambda^*)^T x^* = -(\nu^*)^T A x^* = -b^T \nu^*$, as claimed.

A symmetric argument holds if we start by assuming that the dual problem Уравнение 4 has a solution.

Strong duality in linear programming

(i) If either problem Уравнение 3 or Уравнение 4 has a (finite) solution, then so does the other, and the objective values are equal.

(ii) If either problem Уравнение 3 or Уравнение 4 is unbounded, then the other problem is infeasible.

PROOF. For (i), suppose that Уравнение 3 has a finite optimal solution x^*. It follows from KKT that there are optimal vectors λ^* and ν^* such that (x^*, ν^*, λ^*) satisfies KKT. We noted above that KKT for Уравнение 3 and Уравнение 4 are equivalent. Moreover, $c^T x^* = (-A^T\nu^* + \lambda^*)^T x^* = -(\nu^*)^T A x^* = -b^T\nu^*$, as claimed.

A symmetric argument holds if we start by assuming that the dual problem Уравнение 4 has a solution.

To prove (ii), suppose that the primal is unbounded, that is, there is a sequence of points x_k, $k = 1, 2, 3, \ldots$ such that

$$c^T x_k \downarrow -\infty, \quad Ax_k = b, \quad x_k \geq 0.$$

Strong duality in linear programming

(i) If either problem Уравнение 3 or Уравнение 4 has a (finite) solution, then so does the other, and the objective values are equal.

(ii) If either problem Уравнение 3 or Уравнение 4 is unbounded, then the other problem is infeasible.

PROOF. For (i), suppose that Уравнение 3 has a finite optimal solution x^*. It follows from KKT that there are optimal vectors λ^* and ν^* such that (x^*, ν^*, λ^*) satisfies KKT. We noted above that KKT for Уравнение 3 and Уравнение 4 are equivalent. Moreover, $c^T x^* = (-A^T \nu^* + \lambda^*)^T x^* = -(\nu^*)^T A x^* = -b^T \nu^*$, as claimed.

A symmetric argument holds if we start by assuming that the dual problem Уравнение 4 has a solution.

To prove (ii), suppose that the primal is unbounded, that is, there is a sequence of points x_k, $k = 1, 2, 3, \ldots$ such that

$$c^T x_k \downarrow -\infty, \quad Ax_k = b, \quad x_k \geq 0.$$

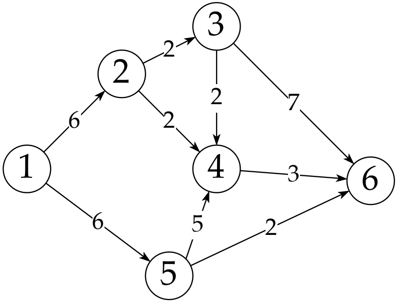
Suppose too that the dual Уравнение 4 is feasible, that is, there exists a vector $\bar{\nu}$ such that $-A^T \bar{\nu} \leq c$. From the latter inequality together with $x_k \geq 0$, we have that $-\bar{\nu}^T A x_k \leq c^T x_k$, and therefore

$$-\bar{\nu}^T b = -\bar{\nu}^T A x_k \leq c^T x_k \downarrow -\infty,$$

yielding a contradiction. Hence, the dual must be infeasible. A similar argument can be used to show that the unboundedness of the dual implies the infeasibility of the primal.

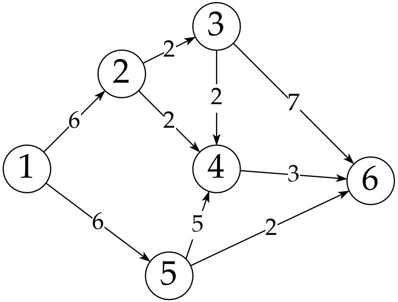
Max-flow min-cut

Max-flow problem example



The nodes are routers, the edges are communications links; associated with each node is a capacity — node 1 can communicate to node 2 at as much as 6 Mbps, node 2 can communicate to node 4 at up to 2 Mbps, etc.

Max-flow problem example

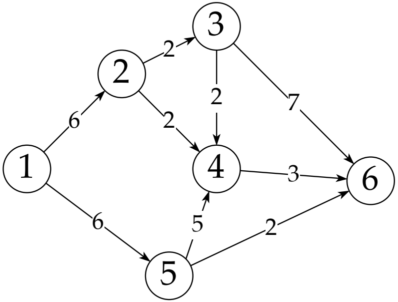


The nodes are routers, the edges are communications links; associated with each node is a capacity — node 1 can communicate to node 2 at as much as 6 Mbps, node 2 can communicate to node 4 at up to 2 Mbps, etc.

Question:

- A network of nodes and edges represents communication links, each with a specified capacity.

Max-flow problem example

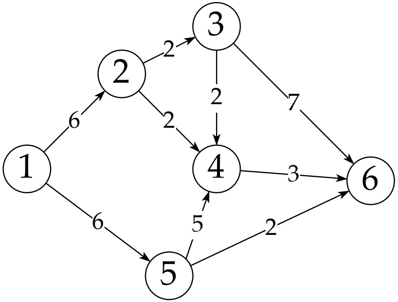


The nodes are routers, the edges are communications links; associated with each node is a capacity — node 1 can communicate to node 2 at as much as 6 Mbps, node 2 can communicate to node 4 at up to 2 Mbps, etc.

Question:

- A network of nodes and edges represents communication links, each with a specified capacity.
- Example: Can node 1 (source) communicate with node 6 (sink) at 6 Mbps? 12 Mbps? What is the maximum rate?

Max-flow problem example

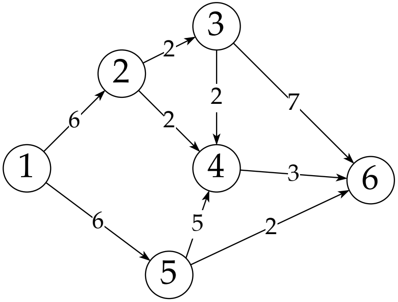


The nodes are routers, the edges are communications links; associated with each node is a capacity — node 1 can communicate to node 2 at as much as 6 Mbps, node 2 can communicate to node 4 at up to 2 Mbps, etc.

Question:

- A network of nodes and edges represents communication links, each with a specified capacity.
- Example: Can node 1 (source) communicate with node 6 (sink) at 6 Mbps? 12 Mbps? What is the maximum rate?

Max-flow problem example



The nodes are routers, the edges are communications links; associated with each node is a capacity — node 1 can communicate to node 2 at as much as 6 Mbps, node 2 can communicate to node 4 at up to 2 Mbps, etc.

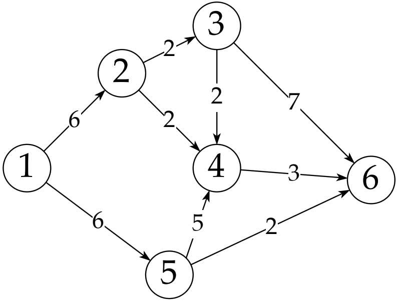
Question:

- A network of nodes and edges represents communication links, each with a specified capacity.
- Example: Can node 1 (source) communicate with node 6 (sink) at 6 Mbps? 12 Mbps? What is the maximum rate?

Capacity Matrix:

$$C = \begin{bmatrix} 0 & 6 & 0 & 0 & 6 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 5 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Max-flow problem example



The nodes are routers, the edges are communications links; associated with each node is a capacity — node 1 can communicate to node 2 at as much as 6 Mbps, node 2 can communicate to node 4 at up to 2 Mbps, etc.

Question:

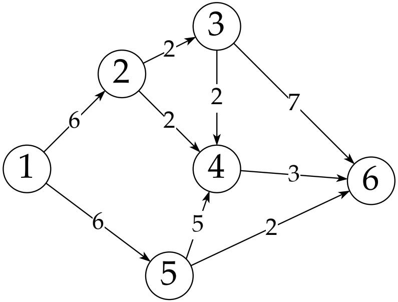
- A network of nodes and edges represents communication links, each with a specified capacity.
- Example: Can node 1 (source) communicate with node 6 (sink) at 6 Mbps? 12 Mbps? What is the maximum rate?

Capacity Matrix:

$$C = \begin{bmatrix} 0 & 6 & 0 & 0 & 6 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 5 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Flow Matrix: $X[i, j]$ represents flow from node i to node j.

Max-flow problem example



The nodes are routers, the edges are communications links; associated with each node is a capacity — node 1 can communicate to node 2 at as much as 6 Mbps, node 2 can communicate to node 4 at up to 2 Mbps, etc.

Question:

- A network of nodes and edges represents communication links, each with a specified capacity.
- Example: Can node 1 (source) communicate with node 6 (sink) at 6 Mbps? 12 Mbps? What is the maximum rate?

Capacity Matrix:

$$
C = \begin{bmatrix} 0 & 6 & 0 & 0 & 6 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 5 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
$$

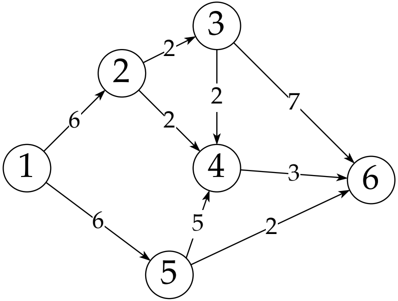
Flow Matrix: $X[i,j]$ represents flow from node i to node j.

Constraints:

$$
0 \preceq X \qquad X \preceq C
$$

Flow Conservation: $\displaystyle\sum_{j=2}^{N} X(i,j) = \sum_{k=1}^{N-1} X(k,i), \; i = 2, \dots, N-1$

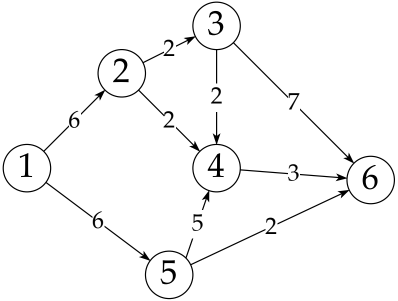
Max-flow problem example



Given the setup, when everything, that is produced by the source will go to the sink. The flow of the network is simply the sum of everything coming out of the source:

$$\sum_{i=2}^{N} X(1, i) \qquad \text{(Flow)}$$

Max-flow problem example



$$
\begin{aligned}
\text{maximize } & \langle X, S \rangle \\
\text{s.t. } & -X \preceq 0 \\
& X \preceq C \\
& \langle X, L_n \rangle = 0, \; n = 2, \dots, N-1,
\end{aligned}
\tag{Max-Flow Problem}
$$

L_n consists of a single column (n) of ones (except for the last row) minus a single row (also n) of ones (except for the first column).

$$
S = \begin{bmatrix} 0 & 1 & \cdots & 1 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}, \quad L_2 = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & -1 & \cdots & -1 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}.
$$

Given the setup, when everything, that is produced by the source will go to the sink. The flow of the network is simply the sum of everything coming out of the source:

$$
\sum_{i=2}^{N} X(1, i) \tag{Flow}
$$

Deriving dual to the Max-flow

Deriving dual to the Max-flow

$$
\begin{aligned}
\underset{\Lambda, \nu}{\text{minimize}} \ & \langle \Lambda, C \rangle \\
\text{s.t. } & \Lambda + Q \succeq S \\
& \Lambda \succeq 0
\end{aligned}
\qquad \text{(Max-Flow Dual Problem)}
$$

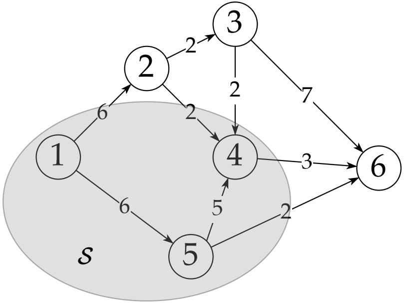
where

$$
Q = \begin{bmatrix}
0 & \nu_2 & \nu_3 & \cdots & \nu_{N-1} & 0 \\
0 & 0 & \nu_3 - \nu_2 & \cdots & \nu_{N-1} - \nu_2 & -\nu_2 \\
0 & \nu_2 - \nu_3 & 0 & \cdots & \nu_{N-1} - \nu_3 & -\nu_3 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & \nu_2 - \nu_{N-1} & \nu_3 - \nu_{N-1} & \cdots & 0 & -\nu_{N-1} \\
0 & 0 & 0 & \cdots & 0 & 0
\end{bmatrix}.
$$

Min-cut problem example

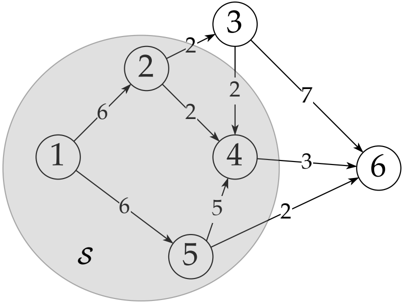
A cut of the network separates the vertices into two sets: one containing the source (we call this set $\mathcal{S}$, and one containing the sink. The capacity of the cut is the total value of the edges coming out of $\mathcal{S}$ — we are separating the sets by "cutting off the flow" along these edges.

$\mathcal{S} = \{1, 4, 5\}$



The edges in the cut are $1 \to 2, 4 \to 6$, and $5 \to 6$ the capacity of this cut is $6 + 3 + 2 = 11$.

$\mathcal{S} = \{1, 2, 4, 5\}$



The edges in the cut are $2 \to 3, 4 \to 6$, and $5 \to 6$ the capacity of this cut is $2 + 3 + 2 = 7$.

Min-cut is the dual to max-flow

What is the minimum value of the smallest cut? We will argue that it is the same as the optimal value of the solution d^* of the dual program (Max-Flow Dual Problem).

Min-cut is the dual to max-flow

What is the minimum value of the smallest cut? We will argue that it is the same as the optimal value of the solution d^* of the dual program (Max-Flow Dual Problem).

First, suppose that $\mathcal{S}$ is a valid cut. From $\mathcal{S}$, we can easily find a dual feasible point that matches its capacity: for $n = 1, \dots, N$, take

$$\nu_n = \begin{cases} 1, & n \in \mathcal{S}, \\ 0, & n \notin \mathcal{S}, \end{cases} \qquad \text{and} \qquad \lambda_{i,j} = \begin{cases} \max(\nu_i - \nu_j, 0), & i \neq 1,\ j \neq N, \\ 1 - \nu_j, & i = 1, \\ \nu_i, & j = N. \end{cases}$$

Min-cut is the dual to max-flow

What is the minimum value of the smallest cut? We will argue that it is the same as the optimal value of the solution d^* of the dual program (Max-Flow Dual Problem).

First, suppose that $\mathcal{S}$ is a valid cut. From $\mathcal{S}$, we can easily find a dual feasible point that matches its capacity: for $n = 1, \dots, N$, take

$$\nu_n = \begin{cases} 1, & n \in \mathcal{S}, \\ 0, & n \notin \mathcal{S}, \end{cases} \qquad \text{and} \qquad \lambda_{i,j} = \begin{cases} \max(\nu_i - \nu_j, 0), & i \neq 1, \, j \neq N, \\ 1 - \nu_j, & i = 1, \\ \nu_i, & j = N. \end{cases}$$

Notice that these choices obey the constraints in the dual and that $\lambda_{i,j}$ will be 1 if $i \to j$ is cut, and 0 otherwise, so

$$\text{capacity}(S) = \sum_{i,j} \lambda_{i,j} C_{i,j}.$$

Min-cut is the dual to max-flow

What is the minimum value of the smallest cut? We will argue that it is the same as the optimal value of the solution d^* of the dual program (Max-Flow Dual Problem).

First, suppose that $\mathcal{S}$ is a valid cut. From $\mathcal{S}$, we can easily find a dual feasible point that matches its capacity: for $n = 1, \ldots, N$, take

$$\nu_n = \begin{cases} 1, & n \in \mathcal{S}, \\ 0, & n \notin \mathcal{S}, \end{cases} \qquad \text{and} \qquad \lambda_{i,j} = \begin{cases} \max(\nu_i - \nu_j, 0), & i \neq 1,\, j \neq N, \\ 1 - \nu_j, & i = 1, \\ \nu_i, & j = N. \end{cases}$$

Notice that these choices obey the constraints in the dual and that $\lambda_{i,j}$ will be 1 if $i \to j$ is cut, and 0 otherwise, so

$$\text{capacity}(S) = \sum_{i,j} \lambda_{i,j} C_{i,j}.$$

Every cut is feasible, so

$$d^\star \leq \text{MINCUT}.$$

Min-cut is the dual to max-flow

Now we show that for every solution ν^*, λ^* of the dual, there is a cut that has a capacity at most d^*. We generate a cut *at random*, and then show that the expected value of the capacity of the cut is less than d^* — this means there must be at least one with a capacity of d^* or less.

Min-cut is the dual to max-flow

Now we show that for every solution ν^*, λ^* of the dual, there is a cut that has a capacity at most d^*. We generate a cut *at random*, and then show that the expected value of the capacity of the cut is less than d^* — this means there must be at least one with a capacity of d^* or less.

Let Z be a uniform random variable on $[0,1]$. Along with $\lambda^*, \nu_2^*, \dots, \nu_{N-1}^*$ generated by solving (Max-Flow Dual Problem), take $\nu_1 = 1$ and $\nu_N = 0$. Create a cut $\mathcal{S}$ with the rule:

$$\text{if } \nu_n^* > Z, \text{ then take } n \in \mathcal{S}.$$

... The probability that a particular edge $i \to j$ is in this cut is

$$
\begin{aligned}
P(i \in \mathcal{S}, j \notin \mathcal{S}) &= P\left(\nu_j^\star \leq Z \leq \nu_i^\star\right) \\
&\leq \begin{cases} \max(\nu_i^\star - \nu_j^\star, 0), & 2 \leq i, j \leq N-1, \\ 1 - \nu_j^\star, & i = 1;\ j = 2, \dots, N-1, \\ \nu_i^\star, & i = 2, \dots, N-1;\ j = N, \\ 1, & i = 1;\ j = N. \end{cases} \\
&\leq \lambda_{i,j}^\star,
\end{aligned}
$$

Min-cut is the dual to max-flow

The last inequality follows simply from the constraints in the dual program (Max-Flow Dual Problem). This cut is random, so its capacity is a random variable, and its expectation is

$$
\begin{aligned}
\mathbb{E}[\text{capacity}(\mathcal{S})] &= \sum_{i,j} C_{i,j} P(i \in \mathcal{S}, j \notin \mathcal{S}) \\
&\leq \sum_{i,j} C_{i,j} \lambda^\star_{i,j} = d^\star .
\end{aligned}
$$

Min-cut is the dual to max-flow

The last inequality follows simply from the constraints in the dual program (Max-Flow Dual Problem). This cut is random, so its capacity is a random variable, and its expectation is

$$
\begin{aligned}
\mathbb{E}[\text{capacity}(\mathcal{S})] &= \sum_{i,j} C_{i,j} P(i \in \mathcal{S}, j \notin \mathcal{S}) \\
&\leq \sum_{i,j} C_{i,j} \lambda^\star_{i,j} = d^\star.
\end{aligned}
$$

Thus there must be a cut whose capacity is at most $d^\star$. This establishes that

$$
\text{MINCUT} \leq d^\star.
$$

Min-cut is the dual to max-flow

The last inequality follows simply from the constraints in the dual program (Max-Flow Dual Problem). This cut is random, so its capacity is a random variable, and its expectation is

$$\begin{aligned}\mathbb{E}[\text{capacity}(\mathcal{S})] &= \sum_{i,j} C_{i,j} P(i \in \mathcal{S}, j \notin \mathcal{S}) \\ &\leq \sum_{i,j} C_{i,j} \lambda^\star_{i,j} = d^\star.\end{aligned}$$

Thus there must be a cut whose capacity is at most $d^\star$. This establishes that

$$\text{MINCUT} \leq d^\star.$$

Combining these two facts of course means that

$$d^\star = \text{MINCUT} = \text{MAXFLOW} = p^\star,$$

where $p^\star$ is the solution of the primal, and equality follows from strong duality for linear programming.

Min-cut is the dual to max-flow

The last inequality follows simply from the constraints in the dual program (Max-Flow Dual Problem). This cut is random, so its capacity is a random variable, and its expectation is

$$
\begin{aligned}
\mathbb{E}[\text{capacity}(\mathcal{S})] &= \sum_{i,j} C_{i,j} P(i \in \mathcal{S}, j \notin \mathcal{S}) \\
&\leq \sum_{i,j} C_{i,j} \lambda^\star_{i,j} = d^\star.
\end{aligned}
$$

Thus there must be a cut whose capacity is at most $d^\star$. This establishes that

$$
\text{MINCUT} \leq d^\star.
$$

Combining these two facts of course means that

$$
d^\star = \text{MINCUT} = \text{MAXFLOW} = p^\star,
$$

where $p^\star$ is the solution of the primal, and equality follows from strong duality for linear programming.

> **Max-flow min-cut theorem.**
>
> The maximum value of an s-t flow is equal to the minimum capacity over all s-t cuts.

Mixed Integer Programming

Complexity of MIP

Consider the following Mixed Integer Programming (MIP):

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

Complexity of MIP

Consider the following Mixed Integer Programming (MIP):

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

Relax it to:

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

. . . # Mixed Integer Programming

Complexity of MIP

Consider the following Mixed Integer Programming (MIP):

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

Complexity of MIP

Consider the following Mixed Integer Programming (MIP): Relax it to:

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

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

Complexity of MIP

Consider the following Mixed Integer Programming (MIP):

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

Optimal solution

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

Relax it to:

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

Complexity of MIP

Consider the following Mixed Integer Programming (MIP):

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

Optimal solution

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

Relax it to:

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

Optimal solution

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

Complexity of MIP

Consider the following Mixed Integer Programming (MIP):

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

Optimal solution

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

Relax it to:

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

Optimal solution

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

- Rounding $x_3 = 0$: gives $z = 19$.

Complexity of MIP

Consider the following Mixed Integer Programming (MIP):

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

Optimal solution

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

Relax it to:

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

Optimal solution

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

- Rounding $x_3 = 0$: gives $z = 19$.
- Rounding $x_3 = 1$: Infeasible.

Complexity of MIP

Consider the following Mixed Integer Programming (MIP):

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

Optimal solution

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

Relax it to:

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

Optimal solution

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

- Rounding $x_3 = 0$: gives $z = 19$.
- Rounding $x_3 = 1$: Infeasible.

Complexity of MIP

Consider the following Mixed Integer Programming (MIP):

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

Optimal solution

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

Relax it to:

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

Optimal solution

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

- Rounding $x_3 = 0$: gives $z = 19$.
- Rounding $x_3 = 1$: Infeasible.

❗ MIP is much harder, than LP

- Naive rounding of LP relaxation of the initial MIP problem might lead to an infeasible or suboptimal solution.

Complexity of MIP

Consider the following Mixed Integer Programming (MIP):

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

Optimal solution

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

Relax it to:

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

Optimal solution

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

- Rounding $x_3 = 0$: gives $z = 19$.
- Rounding $x_3 = 1$: Infeasible.

! MIP is much harder, than LP

- Naive rounding of LP relaxation of the initial MIP problem might lead to an infeasible or suboptimal solution.
- General MIP is NP-hard.

Complexity of MIP

Consider the following Mixed Integer Programming (MIP):

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

Optimal solution

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

Relax it to:

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

Optimal solution

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

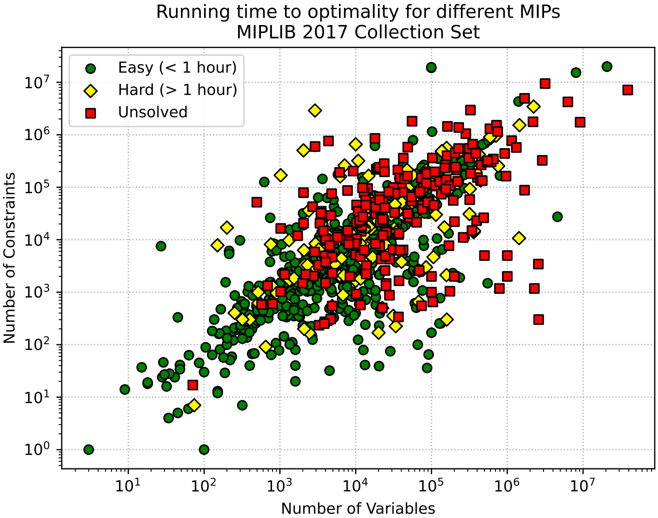
- Rounding $x_3 = 0$: gives $z = 19$.
- Rounding $x_3 = 1$: Infeasible.

❗ MIP is much harder, than LP

- Naive rounding of LP relaxation of the initial MIP problem might lead to an infeasible or suboptimal solution.
- General MIP is NP-hard.
- However, if the coefficient matrix of a MIP is a *totally unimodular matrix*, then it can be solved in polynomial time.

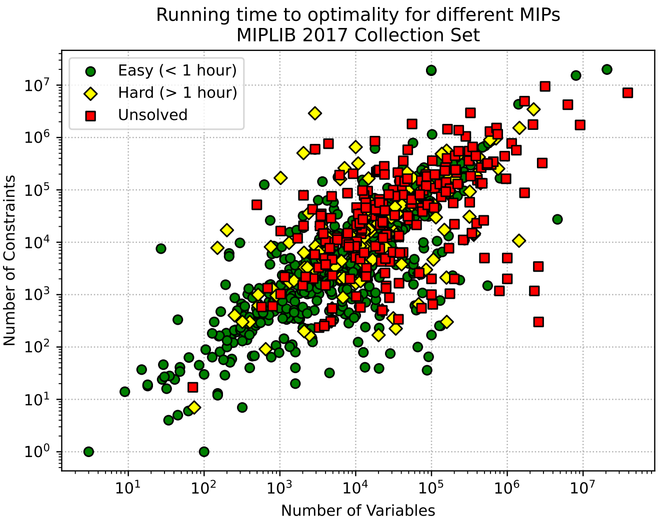
Unpredictable complexity of MIP

- It is hard to predict what will be solved quickly and what will take a long time



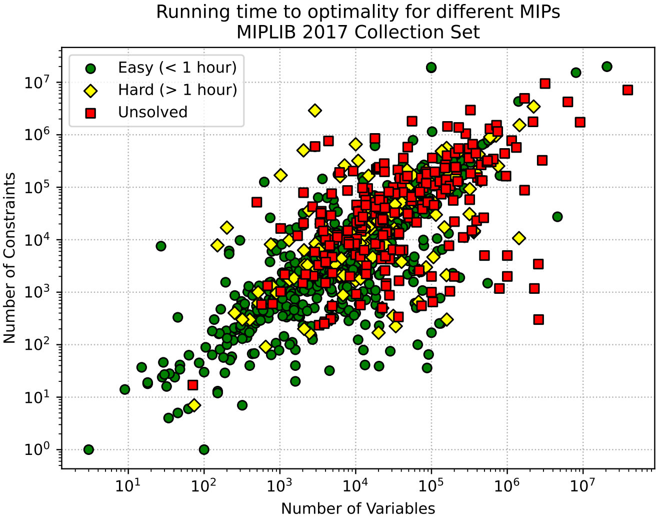
Unpredictable complexity of MIP

- It is hard to predict what will be solved quickly and what will take a long time
- Dataset



Unpredictable complexity of MIP

- It is hard to predict what will be solved quickly and what will take a long time
- Dataset
- Source code



Hardware progress vs Software progress

What would you choose, assuming, that the question is posed correctly (you can compile software for any hardware and the problem is the same for both options)? We will consider the time from 1992 to 2023.

Hardware

Solving MIP with old software on modern hardware

Software

Solving MIP with modern software on old hardware

Hardware progress vs Software progress

What would you choose, assuming, that the question is posed correctly (you can compile software for any hardware and the problem is the same for both options)? We will consider the time from 1992 to 2023.

Hardware

Solving MIP with old software on modern hardware

$\approx 1.664.510$ x speedup

Moore's law states that computational power doubles every 18 months.

Software

Solving MIP with modern software on old hardware

$\approx 2.349.000$ x speedup

R. Bixby conducted an intensive experiment with benchmarking all CPLEX software versions starting from 1992 to 2007 and measured overall software progress (29000 times), later (in 2009) he was a cofounder of Gurobi optimization software, which gave additional ≈ 81 speedup on MILP.

[2] R. Bixby report Recent study

Hardware progress vs Software progress

What would you choose, assuming, that the question is posed correctly (you can compile software for any hardware and the problem is the same for both options)? We will consider the time from 1992 to 2023.

Hardware

Solving MIP with old software on modern hardware

$\approx 1.664.510$ x speedup

Moore's law states that computational power doubles every 18 months.

Software

Solving MIP with modern software on old hardware

$\approx 2.349.000$ x speedup

R. Bixby conducted an intensive experiment with benchmarking all CPLEX software versions starting from 1992 to 2007 and measured overall software progress (29000 times), later (in 2009) he was a cofounder of Gurobi optimization software, which gave additional ≈ 81 speedup on MILP.

It turns out that if you need to solve a MILP, it is better to use an old computer and modern methods than vice versa, the newest computer and methods of the early 1990s![2]

[2] R. Bixby report Recent study

Sources

- Optimization Theory (MATH4230) course @ CUHK by Professor Tieyong Zeng