DM545/DM871
Linear and Integer Programming

Slides: Full Deck

Marco Chiarandini

Department of Mathematics and Computer Science

15. september 2026

47 in total registered in ItsLearning

DM545 (5 ECTS)
33 enrolled

- Math-economics
  (Bachelor, 5th semester)
  Section M1
  6 enrolled
- Others?
  Section H1
  27 enrolled

DM871 (5 ECTS)
13 enrolled

- Section H1, 13 enrolled
  - Computer Science
    (Master)
  - Mathematics
    (Bachelor/Master)
  - Applied Mathematics
    (Bachelor/Master)
  - Data Science (Master)
  - Others?

Learn about **mathematical optimization**:

- linear programming (continuous/numerical linear optimization)
- integer linear programming (discrete/combinatorial linear optimization)

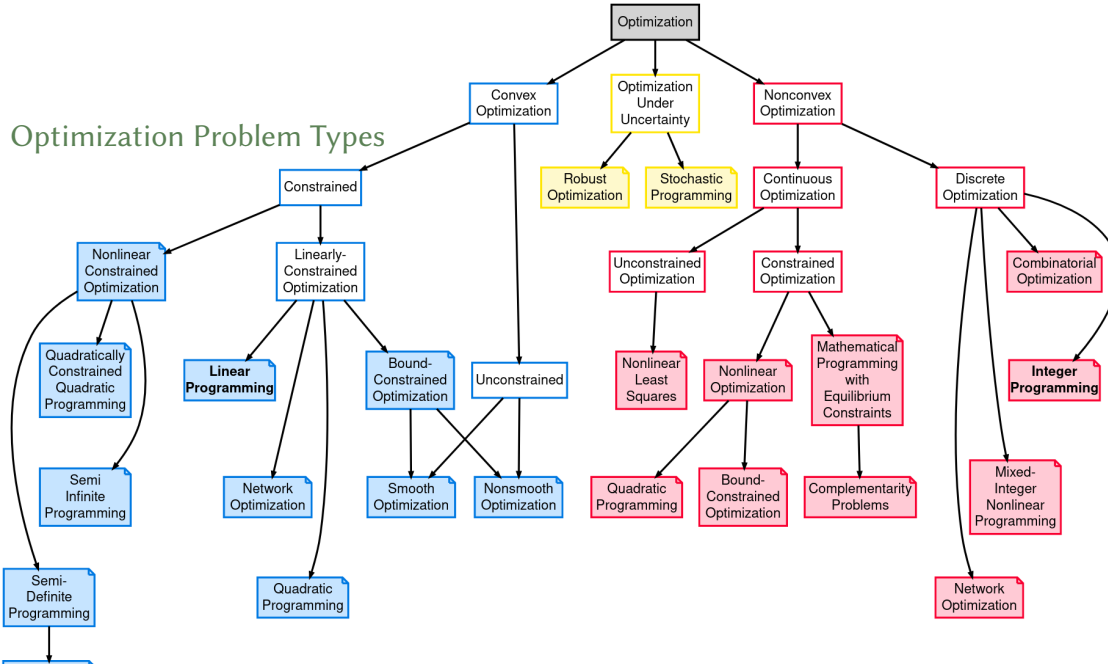
Hence, an alternative course title could be: **Linear Optimization**

Optimization is an important tool in **decision making** and in analyzing physical systems.

In mathematical terms, an **optimization problem** is the problem of finding the **best solution** from the set of all **feasible solutions**.

The first step in the optimization process is constructing an appropriate **mathematical model**

Optimization Problem Types



Linear Programming

1 Introduction - Linear Programming, Notation & Modeling
2 Linear Programming, Simplex Method
3 Exception Handling
4 Duality Theory
5 Sensitivity
6 Revised Simplex Method

Integer Linear Programming

7 Modeling Examples, Good Formulations, Relaxations
8 Well Solved Problems
9 Cutting Planes & Branch and Bound
10 Network Optimization Models (Max Flow, Min cost flow, Matching)
11 More on Modeling

Teacher: Marco Chiarandini (imada.sdu.dk/u/marco/)
Instructor: (Marco Chiarandini ?)

Schedule, alternative views:

- mitsdu.sdu.dk, SDU Mobile
- Official course description (læserplanen)
- ItsLearning

Schedule (15 weeks, up to week 51):

- Introductory classes: 28 hours (14 classes), scheduled 32
- Training classes: 14 hours (7 classes), scheduled 28

- Announcements in ItsLearning
- Ask peers and teacher in class
- ⇔ External Web Page
  (link `https://dm871.github.io`)
- Write to Marco (`marco@imada.sdu.dk`) for confidential and individual issues
- Feedback from classes
- Midway evaluation

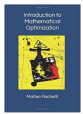
⇝ **It is good to ask questions!!**

⇝ **Let me know if you think we should do things differently!**

Main references:

[LN] Lecture Notes (continously updated)

[F] M. Fischetti, **Introduction to Mathematical Optimization**, Independently published, 2019



Linear Programming:

[VA] R. Vanderbei. Linear Programming: Foundations and Extensions. Springer US, 2008

Integer Programming:

[Wo] L.A. Wolsey. Integer programming. John Wiley & Sons, New York, USA, 1998

Others

[HL] Frederick S Hillier and Gerald J Lieberman, Introduction to Operations Research, 9th edition, 2010

External Web Page is the main reference for list of contents

It contains:

- List of units with topics and references
- Slides
- Exercises and some solutions
- Links
- Tutorials for programming tasks

- One **3 h Written Exam**
  January 11, 2027
  all helping tools allowed, but not GenAI
  - style: short open answers about calculations and modeling
  - (language: Danish and/or English)

- (only DM871) Home implementation assignment in the second half of the course

Graded with internal censorship

- Do the plus exercises in advance
- Prepare the starred exercises in advance to get out the most from the session
- Participate actively in class
- Try the others, unsolved in class, after the session
- Best if carried out in small groups
- (Starred) Exercises are examples of exam questions

Exercise classes are about material presented the same week.

Linear Algebra:
manipulation of matrices and vectors with some theoretical background

Linear Algebra

- Matrices and vectors - Matrix algebra
- Dot (scalar, Euclidean inner) product
- Geometric insights
- Systems of Linear Equations - Row echelon form, Gaussian elimination
- Matrix inversion and determinants
- Rank and linear dependency

Unit 0: sheet0.pdf

- gives you the ability to create new and useful artifacts
- allows you to have more control of your world as more and more of it becomes digital
- is just fun.

It can also help you to **understand math**.

- listening to lectures
- watching an instructor work through a derivation
- working through numerical examples by hand
- learn **by doing**, **interacting with Python**.

- Python 3.14+, numpy
- ipython, jupyter, jupyterLab (= interactive python) or **Visual Code** or Spyder3.
- Commercial software: Gurobi, Cplex, Xpress $\approx$ 100 000 Dkk
  Open source: Highs, SCIP
- Python APIs: gurobipy, Pyomo, Python-MIP, PySCIPOpt (Python interfaces to SCIP Optimization Suite)

Unit 0: `sheet0_programming.pdf`

- Read SDU rules on the topic https://mitsdu.dk/en/aiatsdu. They apply here.
  In short: give credit when using
- Positive aspects:
  better explanations, examples, immediate feedback
- Negative aspects:
  economical costs, impact on environment, sovereignty risk, the social aspect, personal skills
- First learn, then use, and always check
  Comapnies recommend to learn to use but want experts who can verify the outcomes

Use it responsibly, to enhance your learning!
Be aware that the exam form requires you to have achieved the learning goals not GenAI.

- A **set** is a collection of objects. eg.: $A = \{1, 2, 3\}$
- $A = \{n \mid n \text{ is a whole number and } 1 \leq n \leq 3\}$
  ('|' reads 'such that')
- $B = \{x \mid x \text{ is a student of this course}\}$
- $x \in A$
  x belongs to A
- set of no members: **empty set**, denoted $\emptyset$
- if a set S is a **(proper) subset** of a set T, we write ($S \subset T$) $T \subseteq S$
  $\{1, 2, 5\} \subset \{1, 2, 4, 5, 6, 30\}$
- for two sets A and B, the **union** $A \cup B$ is $\{x \mid x \in A \text{ or } x \in B\}$
- for two sets A and B, the **intersection** $A \cap B$ is $\{x \mid x \in A \text{ and } x \in B\}$
  $A = \{1, 2, 3, 5\}$ and $B = \{2, 4, 5, 7\}$, then $A \cap B = \{2, 5\}$

- set of real numbers: $\mathbb{R}$
- set of natural numbers: $\mathbb{N} = \{1, 2, 3, 4, \ldots\}$ (positive integers); $\mathbb{N}_0$ to include zero
- set of all integers: $\mathbb{Z} = \{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$; $\mathbb{Z}_0^+$ only positives and zero
- set of rational numbers: $\mathbb{Q} = \{p/q \mid p, q \in \mathbb{Z}, q \neq 0\}$
- set of complex numbers: $\mathbb{C}$
- absolute value (non-negative):

$$|a| = \begin{cases} a & \text{if } a \geq 0 \\ -a & \text{if } a \leq 0 \end{cases}$$

- the set $\mathbb{R}^2$ is the set of ordered pairs (x, y) of real numbers
  (eg, coordinates of a point wrt a pair of axes, the **Cartesian plane**)

- A **matrix** is a rectangular array of numbers or symbols. It can be written as

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

- An $n \times 1$ matrix is a **column vector**, or simply a vector:

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

- the set $\mathbb{R}^n$ is the set of vectors $[x_1, x_2, \ldots, x_n]^T$ of real numbers (eg, coordinates of a point wrt an n-dimensional space, the **Euclidean Space**)

- An **undirected graph G=(V,E)** is a structure made of a pair of sets: V the set of **vertices** and $E \subseteq \{e = \{u, v\} \mid u, v \in V\}$ the set of **edges** (undirected links)
- A **directed graph (digraph)** $D = (V, A)$ is a pair of sets: V the set of **vertices** and $A \subseteq \{uv = (u, v) \mid u, v \in V\}$ the set of **arcs** (directed links)
- Graph drawing (2D embeddings)
- Labelled graphs and weighted graphs
- Walks, Paths, Cycles, Connectedness
- Problems on graphs: eg, determining intrinsic properties such as shortest st-path
- Matrix representations: **adjacency matrix**, **incidence matrix**

- adjacency set (neighborhood set) for $v \in V$: $N(v) = \{u \mid u \in V, uv \in E\}$; vertex degree (number of incident edges), $\delta(v)$; outdegree (number of outgoing edges), $\delta^+(v)$; indegree (number of incoming edges), $\delta^-(v)$;

- subgraph and **induced subgraph** $G' = G[V] \subseteq G$

- cut: set of edges that separates a connected graphs into two connected components $G[X]$, $G[Y]$: $C = \{xy \mid x \in X, y \in Y\}$

- bipartite graphs $G = (U, V, E)$, trees (= connected acycle graphs)

- multi-graphs, hypergraphs, mixed graphs

- Graphs are the basic subject studied by graph theory. The word "graph" was first used in this sense by J.J. Sylvester in 1878 due to a direct relation between mathematics and chemical structure.

Elementary Algebra: the study of symbols and the rules for manipulating symbols. It differs from **arithmetic** in the use of abstractions, such as using letters to stand for numbers that are either unknown or allowed to take on many values

- collecting up terms: eg. $2a + 3b - a + 5b = a + 8b$
- multiplication of variables: eg:

$$a(-b) - 3ab + (-2a)(-4b) = -ab - 3ab + 8ab = 4ab$$

- expansion of bracketed terms: eg:

$$\begin{aligned} -(a - 2b) &= -a + 2b, \\ (2x - 3y)(x + 4y) &= 2x^2 - 3xy + 8xy - 12y^2 \\ &= 2x^2 + 5xy - 12y^2 \end{aligned}$$

- $a^r a^s = a^{r+s}$, $(a^r)^s = a^{rs}$, $a^{-n} = 1/a^n$,
  $a^{m/n} = (a^{1/n})^m$, $a^{1/n} = x \iff x^n = a$, $a^x = n \iff x = \log_a n$

- In Mathematics and Statistics, a **variable** is an alphabetic character representing a **value**, which is unknown. They are used in **symbolic** calculations.
  Commonly given one-character names.

- in contrast, a **parameter** or **constant** or **given** is a known real number

- in contrast, in **Computer Science**, a **variable** is a storage location paired with an associated identifier, which contains a value that may be known or unknown.
  Commonly given long, explanatory names.

- a **function** f on a set $\mathcal{X}$ into a set $\mathcal{Y}$ is a rule that assigns a **unique** element $f(x)$ in $\mathcal{Y}$ to each element x in $\mathcal{X}$.

$$y = f(x)$$

y dependent variable — x independent variable

- scalar (element of a field, eg, $\mathbb{R}$, vector of a single component) and vectors

- a **linear function** is a polynomial of degree one or less.
  If only one variable:

  $$f(x) := ax + b$$

  where a and b are constants, often real numbers.
  The graph is a nonvertical line (a is the slope and b the intercept).
  For any finite number of variables: $f(x_1, \ldots, x_k) = b + a_1 x_1 + \ldots + a_k x_k$ and the graph is a hyperplane of dimension k. (actually these are affine functions: linear + translation)

Operations Research (aka, Management Science, Analytics):
is the discipline that uses a **scientific approach to decision making**.

It seeks to determine how best to design and operate a system,
usually under conditions requiring the allocation of scarce resources,
by means of **mathematics** and **computer science**.

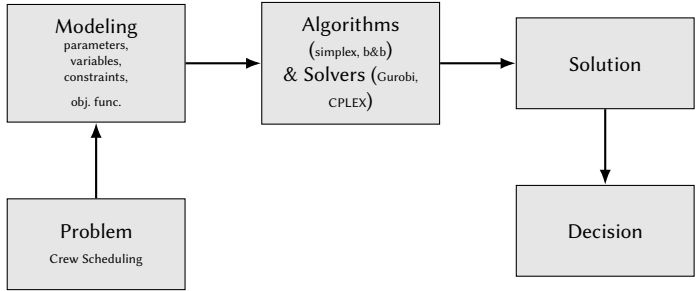
Quantitative methods for planning and analysis.

It encompasses a wide range of problem-solving techniques and methods applied in the pursuit of improved decision-making and efficiency:

- simulation,
- **mathematical optimization**,
- queueing theory and other stochastic-process models,
- Markov decision processes
- econometric methods,
- data envelopment analysis,
- machine learning,
- expert systems

- Production Planning and Inventory Control
- Budget Investment
- Blending and Refining
- Manpower Planning
  - Crew Rostering (airline crew, rail crew, nurses)
- Packing Problems
  - Knapsack Problem
- Cutting Problems
  - Cutting Stock Problem
- Routing
  - Vehicle Routing Problem (trucks, planes, trains ...)
- Locational Decisions
  - Facility Location
- Scheduling/Timetabling
  - Examination timetabling/ train timetabling
- .... + many more

- Planning decisions must be made
- The problems relate to quantitative issues
  - Cheapest
  - Shortest route
  - Fewest number of people
- Not all plans are feasible - there are constraining rules
  - Limited amount of available resources
- It can be extremely difficult to figure out what to do



1. Observe the System
2. Formulate the Problem
3. Formulate Mathematical Model
4. Verify Model
5. Select Alternative
6. Show Results to Company
7. Implementation

Central Idea

Build a mathematical model describing exactly what one wants, and what the "rules of the game" are. However, **what is a mathematical model and how?**

- Find out exactly what the decision maker needs to know:
  - which investment?
  - which product mix?
  - which job j should a person i do?
- Define **Parameters**, that is the given, known elements of the problem.
- Define **Decision Variables** of suitable type (continuous, integer, binary) corresponding to the needs
- Formulate **Objective Function** computing the benefit/cost
- Formulate mathematical **Constraints** indicating the interplay between the different variables.

In manufacturing industry, **factory planning**: find the best product mix.

Example

A factory makes two products **standard** and **deluxe**. Eg, yogurt, sleeping beds, etc.

A unit of standard gives a profit of 6(k) Dkk.
A unit of deluxe gives a profit of 8(k) Dkk.

The warming|grinding and cooling|polishing times in terms of hours per week for a **unit** of each type of product are given below:

	Standard	Deluxe
(Machine 1) Warming|Grinding	5	10
(Machine 2) Cooling|Polishing	4	4

Warming|Grinding capacity: 60 hours per week
Cooling|Polishing capacity: 40 hours per week

Q: How much of each product, **standard** and **deluxe**, should we produce to maximize the profit?

Decision Variables

$x_1 \geq 0$ units of product standard
$x_2 \geq 0$ units of product deluxe

Object Function

$\max\ 6x_1 + 8x_2$ maximize profit

Constraints

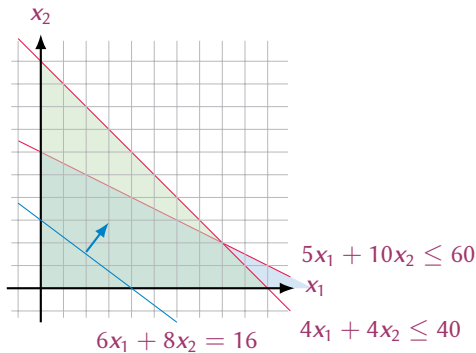
$5x_1 + 10x_2 \leq 60$ machine 1 capacity
$4x_1 + 4x_2 \leq 40$ machine 2 capacity

Machines/Materials A and B
Products 1 and 2

$$
\begin{array}{rrrrrr}
\max & 6x_1 & + & 8x_2 & & \\
 & 5x_1 & + & 10x_2 & \leq & 60 \\
 & 4x_1 & + & 4x_2 & \leq & 40 \\
 & & & x_1 & \geq & 0 \\
 & & & x_2 & \geq & 0
\end{array}
$$

a_{ij}	1	2	b_i
A	5	10	60
B	4	4	40
c_j	6	8	

Graphical Representation:



Managing a production facility

$j = 1, 2, \ldots, n$	products
$i = 1, 2, \ldots, m$	materials
b_i	units of raw material at disposal
a_{ij}	units of raw material i to produce one unit of product j
σ_j	market price of unit of jth product
ρ_i	prevailing market value for material i
$c_j = \sigma_j - \sum_{i=1}^{m} \rho_i a_{ij}$	profit per unit of product j
x_j	amount of product j to produce

$$
\begin{array}{rrcrcrcrcl}
\max & c_1 x_1 & + & c_2 x_2 & + & c_3 x_3 & + & \ldots & + & c_n x_n & = & z \\
\text{subject to} & a_{11} x_1 & + & a_{12} x_2 & + & a_{13} x_3 & + & \ldots & + & a_{1n} x_n & \leq & b_1 \\
& a_{21} x_1 & + & a_{22} x_2 & + & a_{23} x_3 & + & \ldots & + & a_{2n} x_n & \leq & b_2 \\
& \ldots & & & & & & & & & & \\
& a_{m1} x_1 & + & a_{m2} x_2 & + & a_{m3} x_3 & + & \ldots & + & a_{mn} x_n & \leq & b_m \\
& & & & & & & x_1, x_2, \ldots, x_n & & & \geq & 0
\end{array}
$$

$$
\begin{array}{rrcrcrcrcrcl}
\max & c_1x_1 & + & c_2x_2 & + & c_3x_3 & + & \dots & + & c_nx_n & = & z \\
\text{s.t.} & a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & + & \dots & + & a_{1n}x_n & \leq & b_1 \\
 & a_{21}x_1 & + & a_{22}x_2 & + & a_{23}x_3 & + & \dots & + & a_{2n}x_n & \leq & b_2 \\
 & \dots & & & & & & & & & & \\
 & a_{m1}x_1 & + & a_{m2}x_2 & + & a_{m3}x_3 & + & \dots & + & a_{mn}x_n & \leq & b_m \\
 & & & & & & & & & x_1, x_2, \dots, x_n & \geq & 0
\end{array}
$$

$$
\begin{aligned}
\max \quad & \sum_{j=1}^{n} c_j x_j \\
& \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \dots, m \\
& x_j \geq 0, \quad j = 1, \dots, n
\end{aligned}
$$

$$
\begin{array}{lrcrcrcrcl}
\max & c_1 x_1 & + & c_2 x_2 & + & c_3 x_3 & + & \dots & + & c_n x_n & = & z \\
\text{s.t.} & a_{11} x_1 & + & a_{12} x_2 & + & a_{13} x_3 & + & \dots & + & a_{1n} x_n & \leq & b_1 \\
& a_{21} x_1 & + & a_{22} x_2 & + & a_{23} x_3 & + & \dots & + & a_{2n} x_n & \leq & b_2 \\
& \dots & & & & & & & & & & \\
& a_{m1} x_1 & + & a_{m2} x_2 & + & a_{m3} x_3 & + & \dots & + & a_{mn} x_n & \leq & b_m \\
& & & & & & & & & x_1, x_2, \dots, x_n & \geq & 0
\end{array}
$$

$$
\mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}, \quad
A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}, \quad
\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad
\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}
$$

$$
\begin{array}{lrcl}
\max & z & = & \mathbf{c}^T \mathbf{x} \\
& A\mathbf{x} & \leq & \mathbf{b} \\
& \mathbf{x} & \geq & 0
\end{array}
$$

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} c_j x_j \\ & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \ldots, m \\ & x_j \geq 0, \quad j = 1, \ldots, n \end{aligned}$$

$$\begin{aligned} \max \quad & 6x_1 + 8x_2 \\ & 5x_1 + 10x_2 \leq 60 \\ & 4x_1 + 4x_2 \leq 40 \\ & x_1, x_2 \geq 0 \end{aligned}$$

$$\begin{aligned} \max \quad & \mathbf{c}^T \mathbf{x} \\ & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

$$\mathbf{x} \in \mathbb{R}^n, \mathbf{c} \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{R}^m$$

$$\max \quad \begin{bmatrix} 6 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\begin{bmatrix} 5 & 10 \\ 4 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 60 \\ 40 \end{bmatrix}$$

$$x_1, x_2 \geq 0$$

Resource Valuation problem: Determine the value of the raw materials in hand such that: (i) it would be convenient selling and (ii) an outside company would be willing to buy them.

z_i	value of a unit of raw material i
$\sum_{i=1}^{m} b_i z_i$	total expenses for buying (or opportunity cost, cost of having instead of selling)
ρ_i	prevailing unit market value of material i
σ_j	prevailing unit product price

Goal: for the outside company to minimize the total expenses;
(for the owing company the minimum amount of opportunity cost to accept for selling)

$$\min \sum_{i=1}^{m} b_i z_i \tag{1}$$

$$\sum_{i=1}^{m} z_i a_{ij} \geq \sigma_j, \quad j = 1 \ldots n \tag{2}$$

$$z_i \geq \rho_i, \quad i = 1 \ldots m \tag{3}$$

(2) otherwise not selling and (3) otherwise selling to someone else

Let

$$y_i = z_i - \rho_i$$

markup that the company would make by selling the raw material instead of producing.

$$\min \sum_{i=1}^{m} y_i b_i + \cancel{\sum_{i} \rho_i b_i}$$

$$\sum_{i=1}^{m} y_i a_{ij} \geq c_j, \quad j = 1 \dots n$$

$$y_i \geq 0, \quad i = 1 \dots m$$

Dual Problem

$$\max \sum_{j=1}^{n} c_j x_j$$

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \dots, m$$

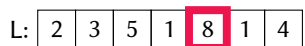
$$x_j \geq 0, \quad j = 1, \dots, n$$

Primal Problem

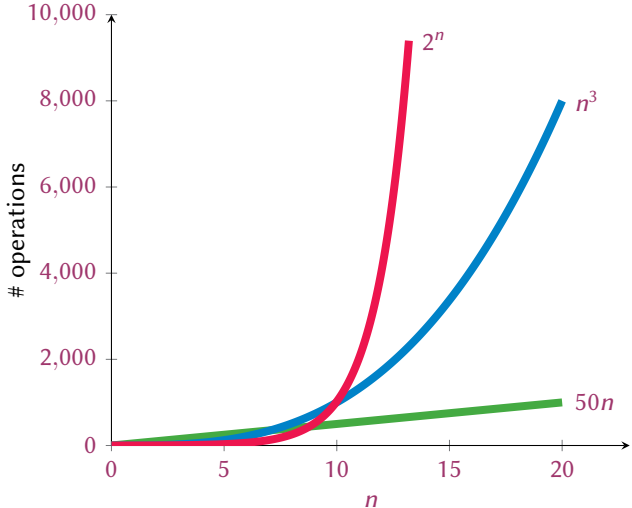
Algorithm: a finite, well-defined sequence of operations to perform a calculation

Algorithm: LargestNumber

Input: A non-empty list of numbers L
Output: The largest number in the list L
largest ← L[0]
foreach each item in the list L **do**
    if the item > largest **then**
        largest ← the item
return largest

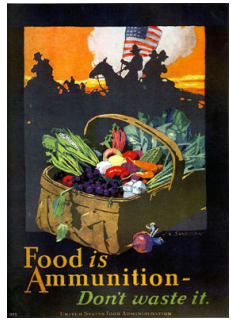


Running time: proportional to number of operations, eg $O(n)$



NP-hard problems: bad if we have to solve them, good for cryptology

- Select a set of foods that will satisfy a set of daily nutritional requirements at minimum cost.
- Motivated in the 1930s and 1940s by US army.
- Formulated as a **linear programming problem** by George Stigler
- (programming intended as planning not computer code)



min cost/weight
subject to nutrition requirements:
eat enough but not too much of Vitamin A
eat enough but not too much of Sodium
eat enough but not too much of Calories
...

Suppose there are:

- 3 foods available: corn, milk, and bread, and
- there are restrictions on the number of calories (between 2000 and 2250) and the amount of Vitamin A (between 5,000 and 50,000)

Food	Cost per serving	Vitamin A	Calories
Corn	$0.18	107	72
2% Milk	$0.23	500	121
Wheat Bread	$0.05	0	65

Parameters (given data)

F := set of foods

N := set of nutrients

a_{ij} := amount of nutrient i in food $j, \forall i \in N, \forall j \in F$

c_j := cost per serving of food $j, \forall j \in F$

$F_{min,j}$:= minimum number of required servings of food $j, \forall j \in F$

$F_{max,j}$:= maximum allowable number of servings of food $j, \forall j \in F$

$N_{min,i}$:= minimum required level of nutrient $i, \forall i \in N$

$N_{max,i}$:= maximum allowable level of nutrient $i, \forall i \in N$

Decision Variables

x_j := number of servings of food j to purchase/consume, $\forall j \in F$

Objective Function: Minimize the total cost of the food

$$\text{Minimize} \sum_{j \in F} c_j x_j$$

Constraint Set 1: For each nutrient $i \in N$, at least meet the minimum required level

$$\sum_{j \in F} a_{ij} x_j \geq N_{min,i}, \qquad \forall i \in N$$

Constraint Set 2: For each nutrient $i \in N$, do not exceed the maximum allowable level.

$$\sum_{j \in F} a_{ij} x_j \leq N_{max,i}, \qquad \forall i \in N$$

Constraint Set 3: For each food $j \in F$, select at least the minimum required number of servings

$$x_j \geq F_{min,j}, \qquad \forall j \in F$$

Constraint Set 4: For each food $j \in F$, do not exceed the maximum allowable number of servings.

$$x_j \leq F_{max,j}, \qquad \forall j \in F$$

system of equalities and inequalities

$$\min \quad \sum_{j \in F} c_j x_j$$

$$\sum_{j \in F} a_{ij} x_j \geq N_{min,i}, \qquad \forall i \in N$$

$$\sum_{j \in F} a_{ij} x_j \leq N_{max,i}, \qquad \forall i \in N$$

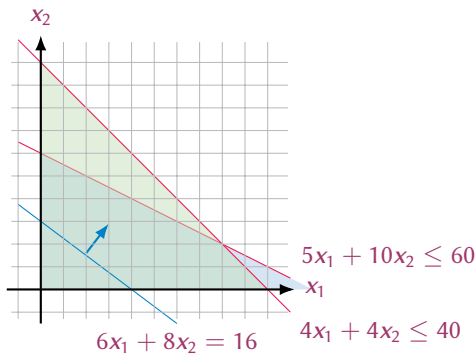
$$x_j \geq F_{min,j}, \qquad \forall j \in F$$

$$x_j \leq F_{max,j}, \qquad \forall j \in F$$

Machines/Materials A and B
Products 1 and 2

$$
\begin{aligned}
\max \quad & 6x_1 + 8x_2 & & \\
& 5x_1 + 10x_2 & \leq & 60 \\
& 4x_1 + 4x_2 & \leq & 40 \\
& x_1 & \geq & 0 \\
& x_2 & \geq & 0
\end{aligned}
$$

Graphical Representation:



$$
\begin{array}{rrcrcrcrcl}
\max & c_1 x_1 & + & c_2 x_2 & + & c_3 x_3 & + & \dots & + & c_n x_n & = & z \\
\text{s.t.} & a_{11} x_1 & + & a_{12} x_2 & + & a_{13} x_3 & + & \dots & + & a_{1n} x_n & \leq & b_1 \\
& a_{21} x_1 & + & a_{22} x_2 & + & a_{23} x_3 & + & \dots & + & a_{2n} x_n & \leq & b_2 \\
& \dots & & & & & & & & & & \\
& a_{m1} x_1 & + & a_{m2} x_2 & + & a_{m3} x_3 & + & \dots & + & a_{mn} x_n & \leq & b_m \\
& & & & & & & & & x_1, x_2, \dots, x_n & \geq & 0
\end{array}
$$

$$
\mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}, \quad
A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}, \quad
\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad
\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}
$$

$$
\begin{array}{rrcl}
\max & z & = & \mathbf{c}^T \mathbf{x} \\
& A\mathbf{x} & \leq & \mathbf{b} \\
& \mathbf{x} & \geq & 0
\end{array}
$$

Abstract mathematical model:
Parameters, Decision Variables, Objective, Constraints
(+ Domains & Quantifiers)

The Syntax of a Linear Programming Problem

objective func.	$\max / \min$	$\mathbf{c}^T\mathbf{x}$	$\mathbf{c} \in \mathbb{R}^n$
constraints	s.t.	$A\mathbf{x} \begin{matrix}\geqq \\ \leqq\end{matrix} \mathbf{b}$	$A \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{R}^m$
		$\mathbf{x} \geq \mathbf{0}$	$\mathbf{x} \in \mathbb{R}^n, \mathbf{0} \in \mathbb{R}^n$

Essential features: continuity, linearity (proportionality and additivity), certainty of parameters

- Any vector $\mathbf{x} \in \mathbb{R}^n$ satisfying all constraints is a **feasible solution**. $\mathbf{x} \in F$, F feasibility region.
- Each $\mathbf{x}^* \in \mathbb{R}^n$ that gives the best possible value for $\mathbf{c}^T\mathbf{x}$ among all feasible $\mathbf{x}$ is an **optimal solution** or **optimum**: x^* is optimum iff $\nexists\, \mathbf{x} \in F \mid \mathbf{c}^T\mathbf{x} < \mathbf{c}^T\mathbf{x}^*$
- The value $\mathbf{c}^T\mathbf{x}^*$ is the **optimum value**

- The linear programming model consisted of 9 equations in 77 variables
- In 1944, Stigler guessed a near-optimal solution using a heuristic method
- In 1947, the National Bureau of Standards used the newly developed simplex method to solve Stigler's model.
  It took 9 clerks using hand-operated desk calculators 120 man days to solve for the optimal solution
- The original instance:
  https://developers.google.cn/optimization/lp/stigler_diet

```
# diet.mod
set NUTR;
set FOOD;

param cost {FOOD} > 0;
param f_min {FOOD} >= 0;
param f_max { j in FOOD} >= f_min[j];
param n_min { NUTR } >= 0;
param n_max {i in NUTR } >= n_min[i];
param amt {NUTR,FOOD} >= 0;

var Buy { j in FOOD} >= f_min[j], <= f_max[j]

minimize total_cost: sum { j in FOOD } cost [j] * Buy[j];
subject to diet { i in NUTR }:
       n_min[i] <= sum {j in FOOD} amt[i,j] * Buy[j] <= n_max[i];
```

```
# diet.dat
data;

set NUTR := A B1 B2 C ;
set FOOD := BEEF CHK FISH HAM MCH MTL SPG TUR;

param:   cost  f_min  f_max :=
  BEEF   3.19    0      100
  CHK    2.59    0      100
  FISH   2.29    0      100
  HAM    2.89    0      100
  MCH    1.89    0      100
  MTL    1.99    0      100
  SPG    1.99    0      100
  TUR    2.49    0      100 ;

param:   n_min  n_max :=
    A      700   10000
    C      700   10000
    B1     700   10000
    B2     700   10000 ;
# %
```

```
param amt (tr):
          A    C    B1   B2 :=
   BEEF   60   20   10   15
   CHK     8    0   20   20
   FISH    8   10   15   10
   HAM    40   40   35   10
   MCH    15   35   15   15
   MTL    70   30   15   15
   SPG    25   50   25   15
   TUR    60   20   15   10 ;
```

```
# Model diet.py
m = Model("diet")

# Create decision variables for the foods to buy
buy = {}
for f in foods:
    buy[f] = m.addVar(obj=cost[f], name=f)

# Nutrition constraints
for c in categories:
    m.addConstr(
      quicksum(nutritionValues[f,c] * buy[f] for f in foods) <= maxNutrition[c], name=c+'max')
    m.addConstr(
      quicksum(nutritionValues[f,c] * buy[f] for f in foods) >= minNutrition[c], name=c+'min')

# Solve
m.optimize()
```

```
import gurobipy as gp

categories, minNutrition, maxNutrition = gp.multidict({
  'calories': [1800, 2200],
  'protein':  [91, gp.GRB.INFINITY],
  'fat':      [0, 65],
  'sodium':   [0, 1779] })

foods, cost = gp.multidict({
  'hamburger': 2.49,
  'chicken':   2.89,
  'hot dog':   1.50,
  'fries':     1.89,
  'macaroni':  2.09,
  'pizza':     1.99,
  'salad':     2.49,
  'milk':      0.89,
  'ice cream': 1.59 })
```

```
# Nutrition values for the foods
nutritionValues = {
  ('hamburger', 'calories'): 410,
  ('hamburger', 'protein'):  24,
  ('hamburger', 'fat'):      26,
  ('hamburger', 'sodium'):   730,
  ('chicken',   'calories'): 420,
  ('chicken',   'protein'):  32,
  ('chicken',   'fat'):      10,
  ('chicken',   'sodium'):   1190,
  ('hot dog',   'calories'): 560,
  ('hot dog',   'protein'):  20,
  ('hot dog',   'fat'):      32,
  ('hot dog',   'sodium'):   1800,
  ('fries',     'calories'): 380,
  ('fries',     'protein'):  4,
  ('fries',     'fat'):      19,
  ('fries',     'sodium'):   270,
  ('macaroni',  'calories'): 320,
  ('macaroni',  'protein'):  12,
  ('macaroni',  'fat'):      10,
  ('macaroni',  'sodium'):   930,
```

⇝ It is impossible to find out who knew what, when, first.
Just two “references”:

- Egyptians and Babylonians considered about 2000 B.C. the solution of special linear equations. But, of course, they described examples and did not describe the methods in "today’s style".

- What we call “**Gaussian elimination**” today has been explicitly described in Chinese “Nine Books of Arithmetics” which is a compendium written in the period 2000 B.C. to A.D. 9, but the methods were probably known long before that.

- Gauss, by the way, never described “Gaussian elimination”. He just used it and stated that the linear equations he used can be solved “per eliminationem vulgarem”

George B. Dantzig, (2002) Linear Programming. Operations Research 50(1):42-47.
https://doi.org/10.1287/opre.50.1.42.17798

- Origins of LP date back to Newton, Leibnitz, Lagrange, etc.
- In 1827, Fourier described a variable elimination method for **systems of linear inequalities**, today often called Fourier-Motzkin elimination (Motzkin, 1937). It can be turned into an LP solver but inefficient.
- In 1932, Leontief (1905-1999) Input-Output model to represent interdependencies between branches of a national economy (1976 Nobel prize)
- In 1939, Kantorovich (1912-1986): Foundations of linear programming (Nobel prize in economics with Koopmans on LP, 1975) on Optimal use of scarce resources: foundation and economic interpretation of LP
- The math subfield of **Linear Programming** was created by George Dantzig, John von Neumann (Princeton), and Leonid Kantorovich in the 1940s.

- In 1947, Dantzig (1914-2005) invented the (primal) simplex algorithm working for the US Air Force at the Pentagon. (program=plan)
- In 1954, Lemke: dual simplex algorithm,
- In 1954, Dantzig and Orchard Hays: revised simplex algorithm
- In 1970, Victor Klee and George Minty created an example that showed that the classical simplex algorithm has exponential worst-case behavior.
- In 1979, L. Khachain found a new efficient algorithm for linear programming. It was terribly slow. (Ellipsoid method)
- In 1984, Karmarkar discovered yet another new efficient algorithm for linear programming. It proved to be a strong competitor for the simplex method. (Interior point method)

- In 1951, Nonlinear Programming began with the Karush-Kuhn-Tucker Conditions
- In 1952, Commercial Applications and Software began
- In 1950s, Network Flow Theory began with the work of Ford and Fulkerson.
- In 1955, Stochastic Programming began
- In 1958, Integer Programming began by R.E. Gomory.
- In 1962, Complementary Pivot Theory

Has $A\mathbf{x} \leq \mathbf{b}$ a solution? (Assumption: $A \in \mathbb{Q}^{m \times n}, \mathbf{b} \in \mathbb{Q}^n$)
Idea:

1. transform the system into another by eliminating some variables such that the two systems have the same solutions over the remaining variables.
2. reduce to a system of constant inequalities that can be easily decided

Let x_r be the variable to eliminate
Let $M = \{1 \dots m\}$ indices of the constraints
For a variable j let's partition the rows of the matrix in

$$\begin{aligned} N &= \{i \in M \mid a_{ij} < 0\} \\ Z &= \{i \in M \mid a_{ij} = 0\} \\ P &= \{i \in M \mid a_{ij} > 0\} \end{aligned}$$

$$\begin{cases} x_r \geq b'_{ir} - \sum_{k=1}^{r-1} a'_{ik} x_k, & a_{ir} < 0 \\ x_r \leq b'_{ir} - \sum_{k=1}^{r-1} a'_{ik} x_k, & a_{ir} > 0 \\ \text{all other constraints} \quad i \in Z \end{cases} \qquad \begin{cases} x_r \geq A_i(x_1, \ldots, x_{r-1}), & i \in N \\ x_r \leq B_i(x_1, \ldots, x_{r-1}), & i \in P \\ \text{all other constraints} \quad i \in Z \end{cases}$$

Hence the original system is equivalent to

$$\begin{cases} \max\{A_i(x_1, \ldots, x_{r-1}), i \in N\} \leq x_r \leq \min\{B_i(x_1, \ldots, x_{r-1}), i \in P\} \\ \text{all other constraints} \quad i \in Z \end{cases}$$

which is equivalent to

$$\begin{cases} A_i(x_1, \ldots, x_{r-1}) \leq B_j(x_1, \ldots, x_{r-1}) \qquad i \in N, j \in P \\ \text{all other constraints} \qquad i \in Z \end{cases}$$

we eliminated x_r but:

$$\begin{cases} |N| \cdot |P| \text{ inequalities} \\ |Z| \text{ inequalities} \end{cases}$$

after d iterations if $|P| = |N| = m/2$ exponential growth: $(1/4^d)(m/2)^{2^d}$

$$
\begin{array}{rcrcl}
-7x_1 & + & 6x_2 & \leq & 25 \\
x_1 & - & 5x_2 & \leq & 1 \\
x_1 & & & \leq & 7 \\
-x_1 & + & 2x_2 & \leq & 12 \\
-x_1 & - & 3x_2 & \leq & 1 \\
2x_1 & - & x_2 & \leq & 10
\end{array}
$$

x_2 variable to eliminate
$N = \{2, 5, 6\}, Z = \{3\}, P = \{1, 4\}$
$|Z \cup (N \times P)| = 7$ constraints

By adding one variable and one inequality, Fourier-Motzkin elimination can be turned into an LP solver.

- $[a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$ closed interval
  $(a, b) = \{x \in \mathbb{R} \mid a < x < b\}$ open interval

- column vectors and matrices
  scalar product: $\mathbf{y}^T\mathbf{x} = \sum_{i=1}^{n} y_i x_i$

- $A\mathbf{x}$ column vector combination of the columns of A;
  $\mathbf{u}^T A$ row vector combination of the rows of A

- **linear combination**

$$\begin{matrix} \mathbf{v}_1, \mathbf{v}_2 \ldots, \mathbf{v}_k \in \mathbb{R}^n \\ \boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_k]^T \in \mathbb{R}^k \end{matrix} \qquad \mathbf{x} = \lambda_1\mathbf{v}_1 + \cdots + \lambda_k\mathbf{v}_k = \sum_{i=1}^{k} \lambda_i \mathbf{v}_i$$

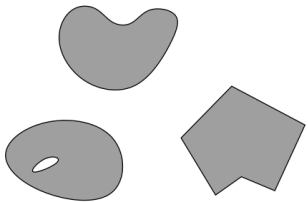
moreover:

$\boldsymbol{\lambda} \geq \mathbf{0}$		**conic combination**
	$\boldsymbol{\lambda}^T\mathbf{1} = 1$	**affine combination**
$\boldsymbol{\lambda} \geq \mathbf{0}$	and $\boldsymbol{\lambda}^T\mathbf{1} = 1$	**convex combination**

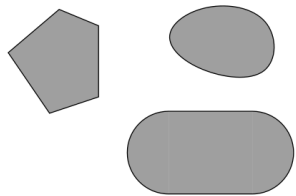
$$\left(\sum_{i=1}^{k} \lambda_i = 1 \right)$$

- set S is **linear (affine) independent** if no element of it can be expressed as linear combination of the others
  Eg: $S \subseteq \mathbb{R}^n \implies$ max n lin. indep. (max $n+1$ aff. indep.)

- **convex set**: if $\mathbf{x}, \mathbf{y} \in S$ and $0 \leq \lambda \leq 1$ then $\lambda\mathbf{x} + (1-\lambda)\mathbf{y} \in S$



nonconvex

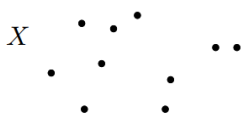


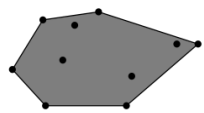
convex

- $f : X \to \mathbb{R}$ is a **convex function** if its epigraph $\{(x, y) \in \mathbb{R}^2 : y \geq f(x)\}$ is a convex set or if $\forall x, y \in X, \lambda \in [0, 1]$ it holds that $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$

- For a set of points $S \subseteq \mathbb{R}^n$

  $\mathrm{lin}(S)$ linear hull (span)
  $\mathrm{cone}(S)$ conic hull
  $\mathrm{aff}(S)$ affine hull
  $\mathrm{conv}(S)$ convex hull





the convex hull of X

$$\mathbf{conv}(X) = \{\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2 + \ldots + \lambda_n \mathbf{x}_n \mid \mathbf{x}_i \in X,\ \lambda_1, \ldots, \lambda_n \geq 0 \text{ and } \textstyle\sum_i \lambda_i = 1\}$$

- **rank** of a matrix for columns (= for rows)
  if (m, n)-matrix has rank $= \min\{m, n\}$ then the matrix is full rank
  if (n, n)-matrix is full rank then it is regular and admits an inverse

- $G \subseteq \mathbb{R}^n$ is an **hyperplane** if $\exists\, \mathbf{a} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ and $\alpha \in \mathbb{R}$:

$$G = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{a}^T\mathbf{x} = \alpha\}$$

- $H \subseteq \mathbb{R}^n$ is an **halfspace** if $\exists\, \mathbf{a} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ and $\alpha \in \mathbb{R}$:

$$H = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{a}^T\mathbf{x} \leq \alpha\}$$

  ($\mathbf{a}^T\mathbf{x} = \alpha$ is a supporting hyperplane of H)

- a set $P \subset \mathbb{R}^n$ is a **polyhedron** if $\exists\, m \in \mathbb{Z}^+, A \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{R}^m$:

$$P = \{\mathbf{x} \in \mathbb{R}^n \mid A\mathbf{x} \leq \mathbf{b}\} = \bigcap_{i=1}^{m} \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{a}_{i,\cdot}^T \mathbf{x} \leq b_i\}$$

  i.e., a polyhedron $P \neq \mathbb{R}^n$ is determined by finitely many halfspaces

- a polyhedron P is a **polytope** if it is bounded: $\exists\, B \in \mathbb{R}, B > 0$:

$$P \subseteq \{\mathbf{x} \in \mathbb{R}^n \mid \|\mathbf{x}\| \leq B\}$$

  ($\|\mathbf{x}\| = \sqrt{\sum_{i=1}^{n} x_i^2}$ is the Euclidean norm of the vector $\mathbf{x} \in \mathbb{R}$)

- a point $\mathbf{x}$ of a polyhedron P is said to be an **extreme point** or a **vertex** of P if it cannot be expressed as a strict convex combination of other two points of the polyhedron, i.e., if there exist no $\mathbf{y}, \mathbf{z} \in P$, $\mathbf{y} \neq \mathbf{z}$ and $\lambda \in (0, 1)$ such that $\mathbf{x} = \lambda\mathbf{y} + (1 - \lambda)\mathbf{z}$

- every point of a **polytope** can be obtained as the convex combination of its vertices. (Minkowski-Weyl Theorem)

- If A and $\mathbf{b}$ are made of rational numbers, $P = \{\mathbf{x} \in \mathbb{R}^n \mid A\mathbf{x} \leq \mathbf{b}\}$ is a **rational polyhedron**

- General optimization problem: $\max\{\varphi(\mathbf{x}) \mid \mathbf{x} \in F\}$, F is feasible region for $\mathbf{x}$

- Note: if F is open, eg, $x < 5$ then: $\sup\{x \mid x < 5\}$
  supremum: least element of $\mathbb{R}$ greater or equal than any element in F

- $\arg\min\{f(i) \mid i \in I\}$ argument $i^* \in I$ such that $f(i^*) = \min\{f(i) \mid i \in I\}$

- The inequality denoted by $(\mathbf{a}, \alpha)$ is called a **valid inequality for P** if $\mathbf{ax} \leq \alpha, \forall \mathbf{x} \in P$.
  Note that $(\mathbf{a}, \alpha)$ is a valid inequality if and only if P lies in the half-space $\{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{ax} \leq \alpha\}$.

- A **face** of P is $F = \{\mathbf{x} \in P \mid \mathbf{ax} = \alpha\}$ where $(\mathbf{a}, \alpha)$ is a valid inequality for P. Hence, it is the intersection of P with the hyperplane of a valid inequality. It is said to be **proper** if $F \neq \emptyset$ and $F \neq P$.

- If $F \neq \emptyset$ we say that the corresponding hyperplane **supports** P.
  If $\mathbf{c}$ is a non zero vector for which $\delta = \max\{\mathbf{c}^T\mathbf{x} \mid \mathbf{x} \in P\}$ is finite,
  then the set $\{\mathbf{x} \mid \mathbf{c}^T\mathbf{x} = \delta\}$ is called **supporting hyperplane**.

- A point $\mathbf{x}$ for which $\{\mathbf{x}\}$ is a face is called a **vertex** of P and also a **basic solution** of $A\mathbf{x} \leq \mathbf{b}$
  (0 dim face)

- A **facet** is a maximal face distinct from P
  $\mathbf{cx} \leq d$ is facet defining if $\mathbf{cx} = d$ is a supporting hyperplane of P of $n - 1$ dim

Input: a matrix $A \in \mathbb{R}^{m \times n}$ and column vectors $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^n$

Task:

1. decide that $\{\mathbf{x} \in \mathbb{R}^n; A\mathbf{x} \leq \mathbf{b}\}$ is empty (prob. infeasible), or
2. find a column vector $\mathbf{x} \in \mathbb{R}^n$ such that $A\mathbf{x} \leq \mathbf{b}$ and $\mathbf{c}^T\mathbf{x}$ is max, or
3. decide that for all $\alpha \in \mathbb{R}$ there is an $\mathbf{x} \in \mathbb{R}^n$ with $A\mathbf{x} \leq \mathbf{b}$ and $\mathbf{c}^T\mathbf{x} > \alpha$ (prob. unbounded)

1. $F = \emptyset$
2. $F \neq \emptyset$ and $\exists$ solution
   1. one solution
   2. infinite solutions
3. $F \neq \emptyset$ and $\nexists$ solution

Theorem (Fundamental Theorem of Linear Programming)

Given:

$$\min\{\mathbf{c}^T\mathbf{x} \mid \mathbf{x} \in P\} \quad \textit{where } P = \{\mathbf{x} \in \mathbb{R}^n \mid A\mathbf{x} \leq \mathbf{b}\}$$

If P is a bounded polyhedron and not empty and $\mathbf{x}^*$ is an optimal solution to the problem, then:

- $\mathbf{x}^*$ *is an extreme point (vertex) of P, or*
- $\mathbf{x}^*$ *lies on a face $F \subset P$ of optimal solutions*

Proof idea:

- assume $\mathbf{x}^*$ not on the boundary of P then $\exists$ a ball around it still in P. Show that a point in the ball has better cost
- if $\mathbf{x}^*$ is not unique then any convex combination of other optimal points are also optimal.

Implications:

- the optimal solution is at the intersection of supporting hyperplanes.
- hence finitely many possibilities
- solution method: write all inequalities as equalities and solve all $\binom{m}{n}$ systems of linear equalities (n # variables, m # equality constraints)
- for each point we then need to check if feasible and if best in cost.
- each system is solved by Gaussian elimination
- Stirling approximation:

$$\binom{2m}{m} \approx \frac{4^m}{\sqrt{\pi m}} \text{ as } m \to \infty$$

1. find a solution that is at the intersection of some n hyperplanes
2. try systematically to produce the other points by exchanging one hyperplane with another
3. check optimality, proof provided by duality theory

1. Forward elimination
   reduces the system to row echelon form by elementary row operations
   - multiply a row by a non-zero constant
   - interchange two rows
   - add a multiple of one row to another

   (or LU decomposition)

2. Back substitution (or reduced row echelon form - RREF)

Example

$$
\begin{aligned}
2x + y - z &= 8 && (R1)\\
-3x - y + 2z &= -11 && (R2)\\
-2x + y + 2z &= -3 && (R3)
\end{aligned}
$$

R1	2	1	-1	8
R2	-3	-1	2	-11
R3	-2	1	2	-3

$$
\begin{aligned}
2x + y - z &= 8 && (R1)\\
+ \tfrac{1}{2}y + \tfrac{1}{2}z &= 1 && (R2)\\
+ 2y + 1z &= 5 && (R3)
\end{aligned}
$$

R1'=1/2 R1	1	1/2	-1/2	4
R2'=R2+3/2xR1	0	1/2	1/2	1
R3'=R3+R1	0	2	1	5

$$
\begin{aligned}
2x + y - z &= 8 && (R1)\\
+ \tfrac{1}{2}y + \tfrac{1}{2}z &= 1 && (R2)\\
- z &= 1 && (R3)
\end{aligned}
$$

R1'=R1	1	1/2	-1/2	4
R2'=2 R2	0	1	1	2
R3'=R3-4xR2	0	0	-1	1

$$
\begin{aligned}
1x + 1/2y &= 7/2 && (R1)\\
1y &= 3 && (R2)\\
z &= -1 && (R3)
\end{aligned}
$$

R1'=R1-1/2xR3	1	1/2	0	7/2
R2'=R2+R3	0	1	0	3
R3'=-R3	0	0	1	-1

$$
\begin{aligned}
x &= 2 && (R1)\\
y &= 3 && (R2)\\
z &= -1 && (R3)
\end{aligned}
$$

R1'=R1-1/2xR2	1	0	0	2 => x=2
R2'=R2	0	1	0	3 => y=3
R3'=R3	0	0	1	-1 => z=-1

$$\begin{bmatrix} 2 & 1 & -1 \\ -3 & -1 & 2 \\ -2 & 1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 8 \\ -11 \\ -3 \end{bmatrix}$$

$$A\mathbf{x} = \mathbf{b}$$

$$\mathbf{x} = A^{-1}\mathbf{b}$$

$$\begin{bmatrix} 2 & 1 & -1 \\ -3 & -1 & 2 \\ -2 & 1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}$$

$$A = PLU$$

$$\mathbf{x} = A^{-1}\mathbf{b} = U^{-1}L^{-1}P^T\mathbf{b}$$

$$\mathbf{z}_1 = P^T\mathbf{b}, \quad \mathbf{z}_2 = L^{-1}\mathbf{z}_1, \quad \mathbf{x} = U^{-1}\mathbf{z}_2$$

```
In [1]: import scipy as sc

In [2]: A = sc.array([[2,1,-1],[-3,-1,2],[-2,1,2]])

In [3]: from scipy import linalg as sl

In [4]: P,L,U = sl.lu(A)

In [5]: print(P,L,U)
        [[0. 0. 1.]
         [1. 0. 0.]
         [0. 1. 0.]]
        [[ 1.          0.          0.        ]
         [ 0.66666667  1.          0.        ]
         [-0.66666667  0.2         1.        ]]
        [[-3.         -1.          2.        ]
         [ 0.          1.66666667  0.66666667]
         [ 0.          0.          0.2       ]]
```

Polynomial time $O(n^2 m)$ but needs to guarantee that all the numbers during the run can be represented by polynomially bounded bits

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} c_j x_j \\ & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \ldots, m \\ & x_j \geq 0, \quad j = 1, \ldots, n \end{aligned}$$

$$\begin{aligned} \max \quad & 6x_1 + 8x_2 \\ & 5x_1 + 10x_2 \leq 60 \\ & 4x_1 + 4x_2 \leq 40 \\ & x_1, x_2 \geq 0 \end{aligned}$$

$$\begin{aligned} \max \quad & \mathbf{c}^T \mathbf{x} \\ & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

$$\mathbf{x} \in \mathbb{R}^n, \mathbf{c} \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{R}^m$$

$$\max \quad \begin{bmatrix} 6 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\begin{bmatrix} 5 & 10 \\ 4 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 60 \\ 40 \end{bmatrix}$$

$$x_1, x_2 \geq 0$$

Every LP problem can be converted in the **standard form**:

$$\begin{aligned} \max \quad & \mathbf{c}^T\mathbf{x} \\ & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \in \mathbb{R}^n \end{aligned}$$

$\mathbf{c} \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{R}^m$

- if equations, then put two constraints, $\mathbf{a}x \leq b$ and $\mathbf{a}x \geq b$
- if $\mathbf{a}x \geq b$ then $-\mathbf{a}x \leq -b$
- if $\min \mathbf{c}^T\mathbf{x}$ then $-\max(-\mathbf{c}^T\mathbf{x})$

and then be put in **equational standard form**:

$$\begin{aligned} \max \quad & \mathbf{c}^T\mathbf{x} \\ & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

$\mathbf{x} \in \mathbb{R}^n, \mathbf{c} \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{R}^m$

1. "$=$" constraints
2. $\mathbf{x} \geq 0$ nonnegativity constraints
3. $(\mathbf{b} \geq 0)$
4. max

Every LP problem can be transformed in eq. std. form

1. introduce slack variables (or surplus)

$$\begin{aligned} 5x_1 &+ 10x_2 &+ x_3 &= 60 \\ 4x_1 &+ 4x_2 &+ x_4 &= 40 \end{aligned}$$

2. if $x_1 \gtreqless 0$ then $x_1 = x_1' - x_1''$, $x_1' \geq 0$, $x_1'' \geq 0$

3. $(b \geq 0)$

4. $\min c^T x \equiv -\max(-c^T x)$

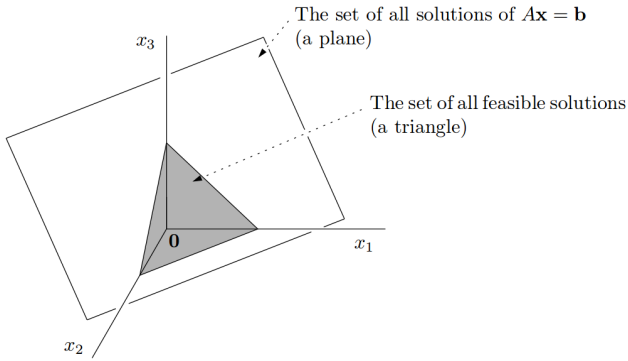
LP in $m \times n$ converted into LP with at most $(m + 2n)$ variables and m equations (n # original variables, m # constraints)

$\max\{\mathbf{c}^T\mathbf{x} \mid A\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$

From linear algebra:

- the set of solutions of $A\mathbf{x} = \mathbf{b}$ is an affine space (hyperplane not passing through the origin).
- $\mathbf{x} \geq \mathbf{0}$ nonegative orthant (octant in $\mathbb{R}^3$)

In $\mathbb{R}^3$:

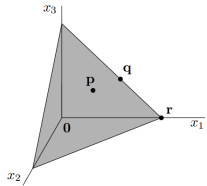


- $A\mathbf{x} = \mathbf{b}$ is a system of equations that we can solve by Gaussian elimination
- Elementary row operations of $\left[A \mid \mathbf{b}\right]$ do not affect set of feasible solutions
  - multiplying all entries in some row of $\left[A \mid \mathbf{b}\right]$ by a nonzero real number λ
  - replacing the ith row of $\left[A \mid b\right]$ by the sum of the ith row and jth row for some $i \neq j$
- Let n' be the number of vars in eq. std. form.

  $$\text{we assume } n' \geq m \text{ and } \operatorname{rank}(\left[A \mid \mathbf{b}\right]) = \operatorname{rank}(A) = m$$

  ie, rows of A are linearly independent
  otherwise, remove linear dependent rows
  If $A \in \mathbb{R}^{m \times n'}$ has rank m, then $A\mathbf{x} = \mathbf{b}$ is an $(n' - m)$-dimensional affine plane.

Basic feasible solutions are the vertices of the feasible region:



More formally:
Let $B = \{1 \dots m\}$, $N = \{m+1 \dots n+m = n'\}$ be subsets partitioning the columns of A: A_B be made of columns of A indexed by B:

Definition

$\mathbf{x} \in \mathbb{R}^n$ is a **basic feasible solution** of the linear program $\max\{\mathbf{c}^T\mathbf{x} \mid A\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$ for an index set B if:

- $x_j = 0 \ \forall j \notin B$
- the square matrix A_B is nonsingular, ie, all columns indexed by B are lin. indep.
- $\mathbf{x}_B = A_B^{-1}\mathbf{b}$ is nonnegative, ie, $\mathbf{x}_B \geq 0$ (feasibility)

We call x_j for $j \in B$ **basic variables** and remaining variables **nonbasic variables**.

Theorem

A **basic feasible solution** *is uniquely determined by the set* B.

Proof:

$$
\begin{aligned}
A\mathbf{x} = & A_B\mathbf{x}_B + A_N\mathbf{x}_N = b \\
& \mathbf{x}_B + A_B^{-1}A_N\mathbf{x}_N = A_B^{-1}b \\
& \mathbf{x}_B = A_B^{-1}b \qquad\qquad A_B \text{ is nonsingular hence one solution}
\end{aligned}
$$

Note: we call B a **(feasible) basis**

Definition

A **basic feasible solution** of a linear program with n variables is a feasible solution for which some n linearly independent constraints hold with equality.

Extreme points and basic feasible solutions are geometric and algebraic manifestations of the same concept:

Theorem

Let P be a (convex) polyhedron made of all feasible solutions of an LP in std. form. For a point $v \in P$ the following are equivalent:

(i) *v is an extreme point (vertex) of P*

(ii) *v is a basic feasible solution of LP*

Proof: see text book [MG] sec. 4.4.

Theorem

Let $LP = \max\{\mathbf{c}^T\mathbf{x} \mid A\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$ *be feasible and bounded, then there is an optimal solution that is a basic feasible solution.*

Proof. consequence of previous theorem and fundamental theorem of linear programming

A similar theorem is valid for arbitrary linear programs (not in eq. form)

However, an optimal solution does not need to be basic:

$$\max\ x_1 + x_2 \text{ subject to } x_1 + x_2 \leq 1$$

(bounded case with infinite solutions)

- Idea for solution method:
- examine all basic solutions.
- There are finitely many: $\binom{m+n}{m}$.
- However, if $n = m$ then $\binom{2m}{m} \approx 4^m$.

$$\max \quad z = \begin{bmatrix} 6 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\begin{bmatrix} 5 & 10 & 1 & 0 \\ 4 & 4 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 60 \\ 40 \end{bmatrix}$$

$$x_1, x_2, x_3, x_4 \geq 0$$

Canonical eq. std. form: one decision variable is isolated in each constraint with coefficient 1 and does not appear in the other constraints nor in the obj. func. and b terms are positive

It gives immediately a basic feasible solution:

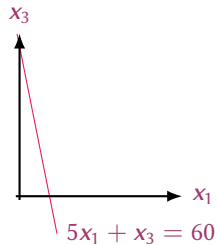
$$x_1 = 0,\ x_2 = 0,\ x_3 = 60,\ x_4 = 40$$

Is it optimal? Look at signs in $z \rightsquigarrow$ if positive then an increase would improve.

Let's try to increase a promising variable, ie, x_1, one with positive coefficient in z

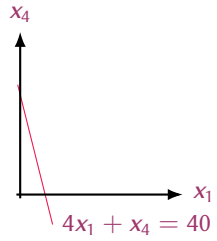
$$5x_1 + x_3 = 60$$
$$x_1 = \frac{60}{5} - \frac{x_3}{5}$$
$$x_3 = 60 - 5x_1 \geq 0$$

If $x_1 > 12$ then $x_3 < 0$



$$4x_1 + x_4 = 40$$
$$x_1 = \frac{40}{4} - \frac{x_4}{4}$$
$$x_4 = 40 - 4x_1 \geq 0$$

If $x_1 > 10$ then $x_4 < 0$



we can take the minimum of the two $\rightsquigarrow$ x_1 increased to 10
x_1 **enters** the basis and x_4 **leaves** it.

Hence, we increase x_1 to 10 and consequently set $x_4 = 0$
That is, x_1 **enters** the basis and x_4 **leaves** it.

$$
\begin{array}{rrrrrrrrl}
\max & 6x_1 & + & 8x_2 & & & & & \\
 & 5x_1 & + & 10x_2 & + & x_3 & & = & 60 \\
 & 4x_1 & + & 4x_2 & & & + & x_4 = & 40 \\
 & & & x_1, x_2 & \geq & 0 & & &
\end{array}
$$

$$
\begin{array}{rrrrrrr}
x_3 & = & 60 & - & 5x_1 & - & 10x_2 \\
x_4 & = & 40 & - & 4x_1 & - & 4x_2 \\
\hline
z & = & & + & 6x_1 & + & 8x_2
\end{array}
$$

$$
\begin{array}{rrrrrrr}
x_3 & = & 10 & - & 5x_2 & + & 5/4x_4 \\
x_1 & = & 10 & - & x_2 & - & 1/4x_4 \\
\hline
z & = & 60 & + & 2x_2 & - & 6/4x_4
\end{array}
$$

First simplex tableau:

	x_1	x_2	x_3	x_4	$-z$	b
x_3	5	10	1	0	0	60
x_4	4	4	0	1	0	40
	6	8	0	0	1	0

we want to reach this new tableau

	x_1	x_2	x_3	x_4	$-z$	b
x_3	0	?	1	?	0	?
x_1	1	?	0	?	0	?
	0	?	0	?	1	?

Pivot operation:

1. Choose pivot:

   column: one s with positive coefficient in obj. func.

   row: ratio between coefficient b and pivot column: choose the one with smallest ratio:

$$\theta = \min_i \left\{ \frac{b_i}{a_{is}} : a_{is} > 0 \right\}, \qquad \theta \text{ increase value of entering var.}$$

2. elementary row operations to update the tableau

- x_4 leaves the basis, x_1 enters the basis
  - Divide pivot row by pivot
  - Send to zero the coefficient in the pivot column of the first row
  - Send to zero the coefficient of the pivot column in the third (cost) row

	x1	x2	x3	x4	-z	b
I'=I-5II'	0	5	1	-5/4	0	10
II'=II/4	1	1	0	1/4	0	10
III'=III-6II'	0	2	0	-6/4	1	-60

From the last row we read: $2x_2 - 3/2x_4 - z = -60$, that is: $z = 60 + 2x_2 - 3/2x_4$.
Since x_2 and x_4 are nonbasic we have $z = 60$ and $x_1 = 10, x_2 = 0, x_3 = 10, x_4 = 0$.

- Done? No! Let x_2 enter the basis

	x1	x2	x3	x4	-z	b
I'=I/5	0	1	1/5	-1/4	0	2
II'=II-I'	1	0	-1/5	1/2	0	8
III'=III-2I'	0	0	-2/5	-1	1	-64

Definition (Reduced costs)

We call **reduced costs** the coefficients in the objective function of the nonbasic variables, $\bar{c}_N$

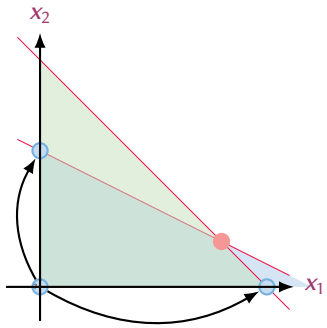
Proposition (Optimality Condition)

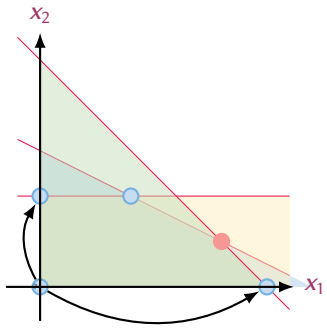
The basic feasible solution is **optimal** *when the* **reduced costs** *in the corresponding simplex tableau are nonpositive, ie, such that:*

$$\bar{c}_N \leq 0$$

Proof: Let z_0 be the obj value when $\bar{\mathbf{c}}_N \leq 0$.
For any other feasible solution $\tilde{\mathbf{x}} = [\tilde{\mathbf{x}}_B, \tilde{\mathbf{x}}_N]$, $\tilde{\mathbf{x}}_N \geq 0$ reachable from the tableau we have:

$$z = \mathbf{c}^T\tilde{\mathbf{x}} \quad \text{and from the last line of the tableau} \quad \mathbf{c}^T\tilde{\mathbf{x}} = z = z_0 + \bar{\mathbf{c}}_B^T\tilde{\mathbf{x}}_B + \bar{\mathbf{c}}_N^T\tilde{\mathbf{x}}_N \leq z_0$$


x_2
x_1


x_2
x_1

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} c_j x_j \\ & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \ldots, m \\ & x_j \geq 0, \quad j = 1, \ldots, n \end{aligned}$$

$$x_{n+i} = b_i - \sum_{j=1}^{n} a_{ij} x_j, \quad i = 1, \ldots, m$$

$$z = \sum_{j=1}^{n} c_j x_j$$

Tableau

$$\left[\begin{array}{c|c|c|c} I & \bar{A}_N & 0 & \bar{b} \\ \hline 0 & \bar{c}_N & 1 & -\bar{d} \end{array}\right]$$

Dictionary

$$x_r = \bar{b}_r - \sum_{s \notin B} \bar{a}_{rs} x_s, \quad r \in B$$

$$z = \bar{d} + \sum_{s \notin B} \bar{c}_s x_s$$

pivot operations in dictionary form:
choose col s with r.c. > 0
choose row with $\min\{-\bar{b}_i/\bar{a}_{is} \mid a_{is} < 0, i = 1, \ldots, m\}$
update: express entering variable and substitute in other rows

$$
\begin{aligned}
\max\ & 6x_1 + 8x_2 \\
& 5x_1 + 10x_2 \leq 60 \\
& 4x_1 + 4x_2 \leq 40 \\
& x_1, x_2 \geq 0
\end{aligned}
$$

	x_1	x_2	x_3	x_4	$-z$	b
x_3	5	10	1	0	0	60
x_4	4	4	0	1	0	40
	6	8	0	0	1	0

$$
\begin{aligned}
x_3 &= 60 - 5x_1 - 10x_2 \\
x_4 &= 40 - 4x_1 - 4x_2 \\
\hline
z &= + 6x_1 + 8x_2
\end{aligned}
$$

After 2 iterations:

	x_1	x_2	x_3	x_4	$-z$	b
x_2	0	1	$1/5$	$-1/4$	0	2
x_1	1	0	$-1/5$	$1/2$	0	8
	0	0	$-2/5$	-1	1	-64

$$
\begin{aligned}
x_2 &= 2 - 1/5x_3 + 1/4x_4 \\
x_1 &= 8 + 1/5x_3 - 1/2x_4 \\
\hline
z &= 64 - 2/5x_3 - 1x_4
\end{aligned}
$$