Lecture notes: Optimization and Operations Research

Jordi Villà i Freixa
Department of Biosciences
Universitat de Vic - Universitat Central de Catalunya
jordi.villa@uvic.cat

September 5, 2024

Contents

Chapter 1

Introduction to Optimization and Operational Research

1.1 Session 1: Introduction

- Learn about the origins and applications of Operations Research
- Understand system modelling principles
- Understand algorithm efficiency and problem complexity
- Contrast between the optimality and practicality
- Learn about software for operations Research
- Introduction to the Python/Colab environment

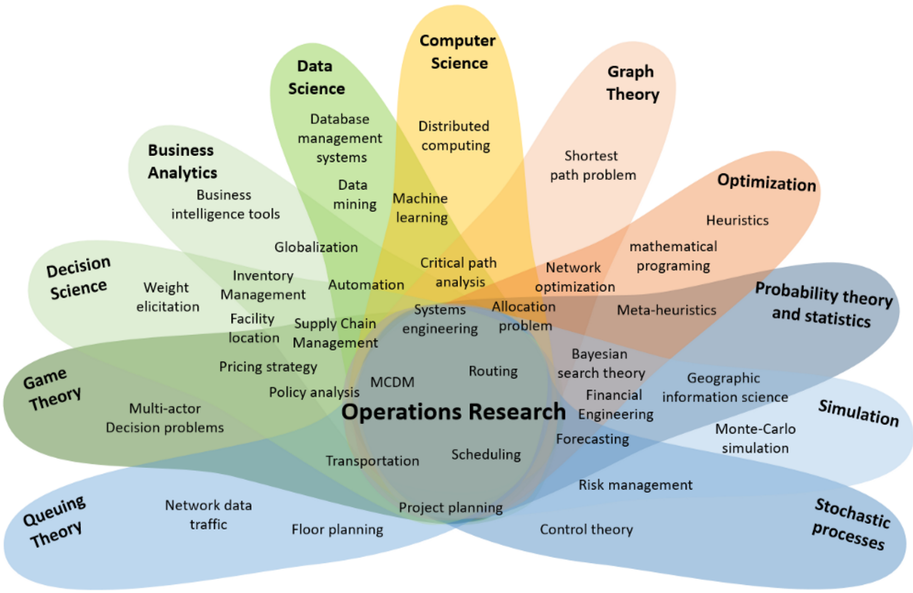
1.1.1 Operation research

The use of quantitative methods to assist analysis and decision-makers in designing, analysizing and improving the performance or operation of systems.

. . . or. . .

Applied math field, where mathematical tools and operators arent used to investigate mathematics further but rather to analyse and solve problems within the OR domain by designing innovative solution approaches.

Link to a big picture view of OR



- OR incorporates analytical tools from many different disciplines, so that they can be applied in a rational way to help decision-makers solve problems and control operations in practice

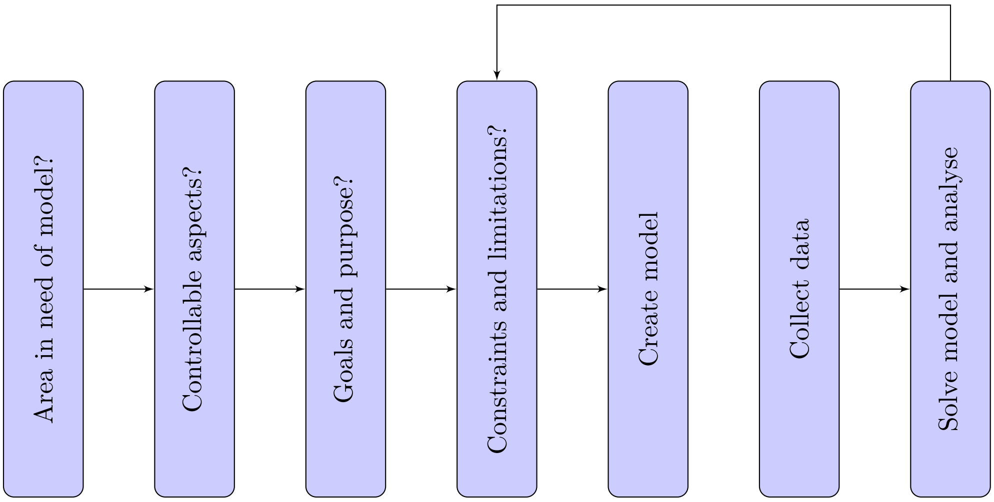
- OR has been taking shape since the industrial revolution, and most notably after WWII

The term Operations Research was first coined in 1940 by McClosky and Trefthen in a small town, Bowdsey, of the United Kingdom, in a military context (the Battle of England).

- A model is a simplified, idealized representation of a real object, a real process, or a real system

- In mathematical models, the building blocks are mathematical structures (equations, inequalities, matrices, functions and operators)

- Building a model:



- The best model of a system strikes a practical compromise between being realistic vs understandable and computationally tractable
- Detail $\neq$ Accuracy (not all details are correct nor necessary)
- A too detailed model brings complexity to analyze it and exploit it
- Models need to be realistic and simple

"Everything should be made as simple as possible, but not simpler"

Albert Einstein

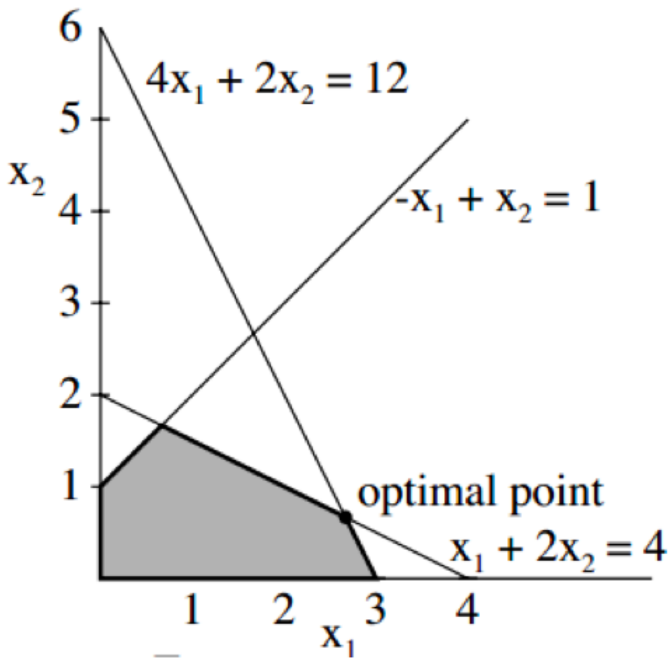
- Does the problem need to be solved?
- What is the *real* problem?
- Would the eventual solution to the problem useful for somebody?
- Would anybody try to implement the solution?
- How much of the analyst's time and cost is worth?
- Are there time and resources available to solve the problem?
- Will the solution create other serious problems for which there is no apparent remedy?

1.1.2 Problem formulation in Operational Research

Decision variables In what are we going to base our decision?

Objective function How is our function build with respect to the variables?

Constraints What are the limits in our model variables and function?



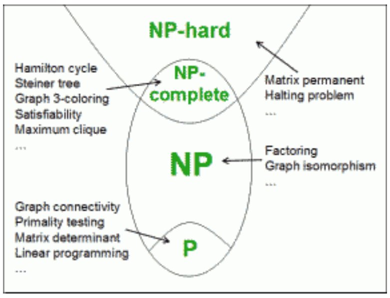
1.1.3 Algorithms

Sequence of operations that can be carried out in a finite amount of time

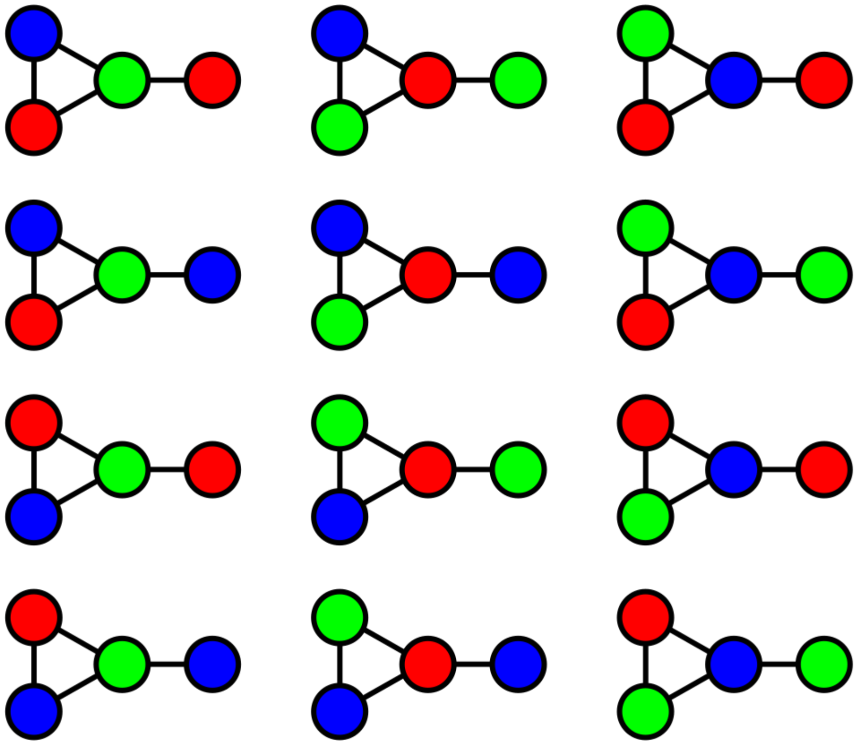
- It can be repeated or called recursively but it will eventually terminate.
- Factors influencing the execution time (unrelated to the algorthm itself):
  - the programming language,
  - the programmer's skills,
  - the hardware being used,
  - the task load on the computer system during execution.
- The performance of an algorithm is typically linked to the size of the problem.

  Class P Problems that can be solved by an algorithm within an amount of computation time proportional to some polynomial function of the problem size.

  Class NP Problems that require computation time proportional to some exponential (or larger) function of the problem size. (The subset sum problem, for example, is $O(2^n \cdot n)$)

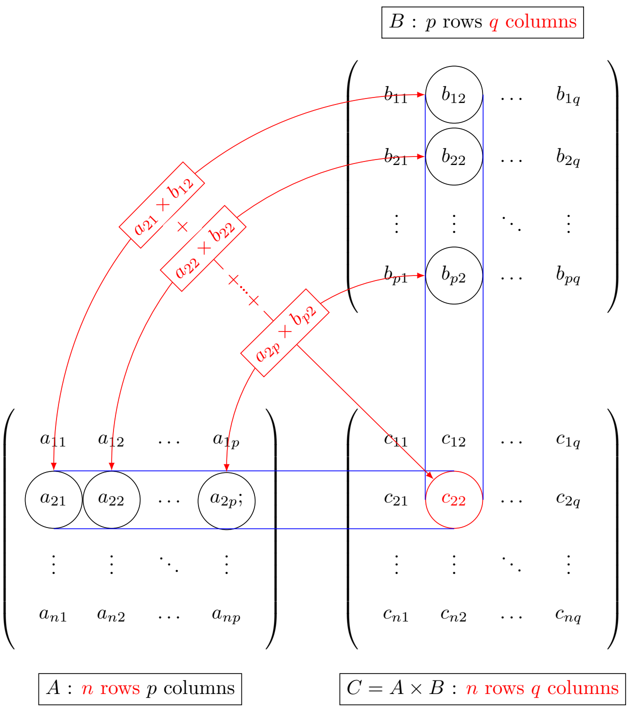


http://www.solipsys.co.uk/new/PVsNP.html



- Performance of algorithm independent of software/hardware/developer skill....? # of computational steps.

- Consider worst case scenario: the largest number of steps that may be necessary.



Multiplying two $n \times n$ matrices, in worst case, involves time proportional to $2n^3$.

1.1.4 Matrix multiplication is $O(n^3)$

```
# this code is contributed by shivanisinghss2110
# https://www.geeksforgeeks.org/strassens-matrix-multiplication/

def multiply(A, B, C):

        for i in range(N):

                for j in range( N):

                        C[i][j] = 0
                        for k in range(N):

                                C[i][j] += A[i][k]*B[k][j]
```

1.1.5 $O(n)$ notation

- We say $f(n) = O(g(n))$, and we read as "$f(n)$ is big-O of $g(n)$", if there is some C and N such that

$$|f(n)| \le Cg(n),\ \forall n \ge N.$$

  In the previous example, multiplying two $n \times n$ matrices costs $2n^3 = O(n^3)$ flops[1]. We say the algorithm is of order 3.

- n denotes the problem size and $g(n)$ is some function of problem size.

[1]FLOating Point operationS per second

- $g(n)$ is the algorithm's worst case step count, as a function of n.

- c is the constant of proportionality, and accounbts for extraneous factors affecting execution time (hardware speed, programming style, computer system load during execution)

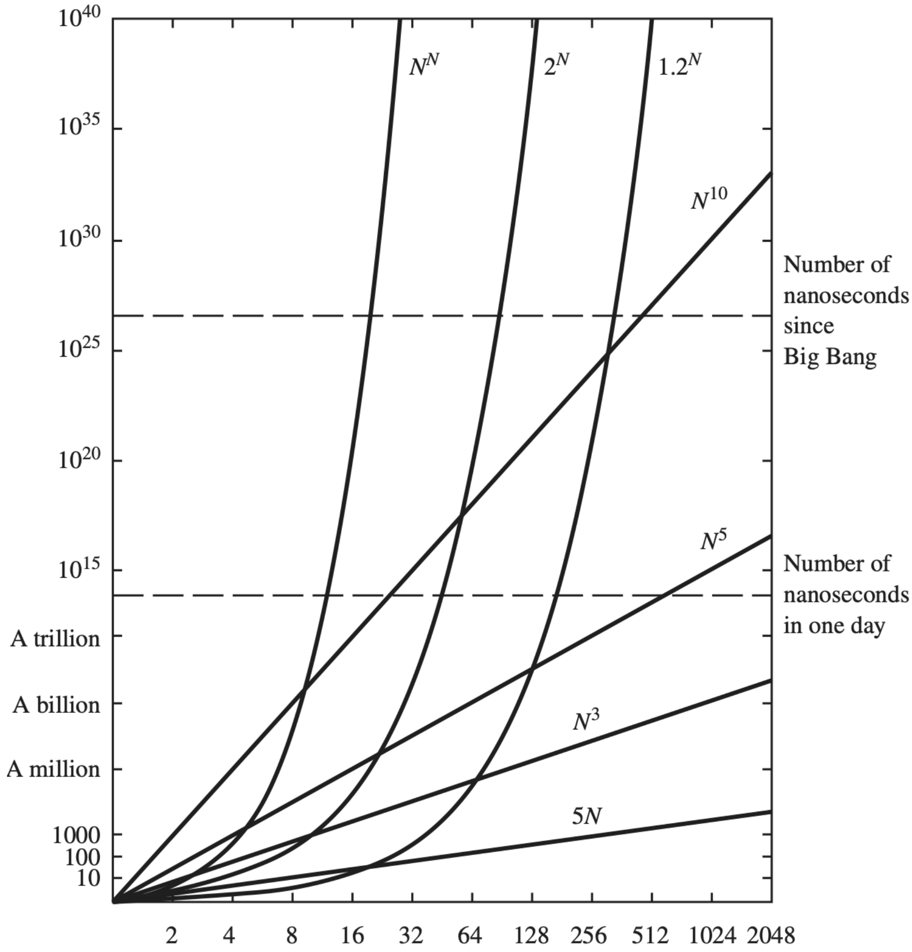
1.1.6 Different orders, different complexity

	Function \ N	20	60	100	300	1000
Polynomial	$5N$	100	300	500	1500	5000
Polynomial	$N \times \log_2 N$	86	354	665	2469	9966
Polynomial	N^2	400	3600	10,000	90,000	1 million (7 digits)
Polynomial	N^3	8000	216,000	1 million (7 digits)	27 million (8 digits)	1 billion (10 digits)
Exponential	2^N	1,048,576	a 19-digit number	a 31-digit number	a 91-digit number	a 302-digit number
Exponential	$N!$	a 19-digit number	an 82-digit number	a 161-digit number	a 623-digit number	unimaginably large
Exponential	N^N	a 27-digit number	a 107-digit number	a 201-digit number	a 744-digit number	unimaginably large

Extracted from [3]

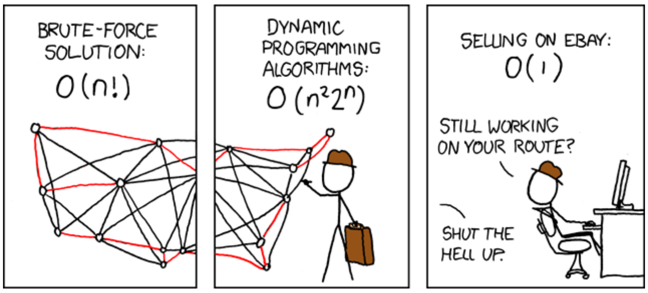
1.1.7 Growth rate of some functions

For comparison: the number of protons in the known universe has 79 digits; the number of nanoseconds since the Big Bang has 27 digits.[3]



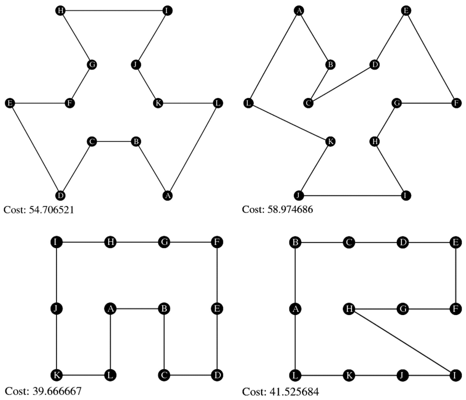
1.1.8 Search space complexity: the travelling salesman problem (TSP)

- A salesman must visit n cities
- Each city must be visited just once
- Which path should he take to minimize the total distance travelled?



https://xkcd.com/399

1.1.9 Search space complexity: the travelling salesman problem (TSP)



`https://doi.org/10.1073/pnas.0609910104`

1.1.10 TSP practical example

Let us assume there are 4 cities and these are the ”distances”[2] between them:

D_{ij}	city A	city B	city C	city D
city A	0	5	10	20
city B	10	0	35	20
city C	20	10	0	5
city D	10	5	10	0

Check code at GitHub

1.1.11 TSP practical example

- The number of combinations is $(n-1)!$. This is the search space
- Asssuming a computer takes a milisecond to evaluate a solution, How long would it take for the computer to find the optimal solution in our 4 nodes problem? and how long for problems with 10, 20, 50 cities?

1.1.12 Optimality and Practicality

- We have been trained in mathematics to find exact/perfect solutions. This is not always possible:
  - Models are approximate representations of the real systems
  - Accumulated round-off errors in computers
  - Input data is usually approximated
  - Exponential time algorithms require suboptimal solutions
- ”good enough” is not always lowering expectations, as the real world can be extremely complex.

[2]Note that the distance is not necessarily identical in both directions

1.1.13 Software for Operations Research

Modeling environments Spreadsheets (Excel, Numbers, Google), AMPL, MPL, LINGO, OPL, AIMMS, SAS/OR, OPTMODEL,GAMS, NEOS[3], ...

Solvers Gurobi, Frontline Solver, CPLEX, ...

Software libraries Google OR-Tools, COIN-OR, IMSL, ...

We will be using Google colab through the course when possible.

1.1.14 Introduction to Operational Research

- Learn about the origins and applications of Operations Research
- Understand system modelling principles
- Understand algorithm efficiency and problem complexity
- Contrast between the optimality and practicality
- Learn about software for operations Research
- Introduction to the Python/Colab environment

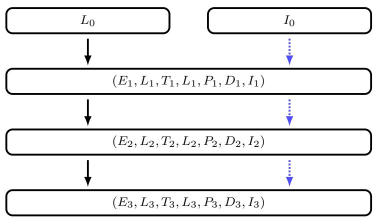
[3] check the TSP demo in NEOS/GAMS

Chapter 2

Nonlinear optimization

Example Production cost:

A company aims at minimizing the production cost during a series of production periods of time T_i, characterised by the demand $D_i i$, equipment capacities and material limitations E_i, labor force L_i (which cost depends quadratically on the difference of labor force between two periods $C_L(L_i - L_{i-1})^2$), productivity of each worker P_i, the number of units of inventory at the end of each production period I_i and the cost to bring them to the next production period C_I.



The problem, as stated, aims at minimizing the function:

$$f(\vec{L}, \vec{I}) = \sum_{i=1}^{T} C_L(L_i - L_{i-1})^2 + C_I I_i$$

subject to:

$$\begin{cases} L_i P_i \leq E_i \\ I_{i-1} + L_i P_i \geq D_i \\ I_i = I_{i-1} + L_i P_i - D_i \\ L_i, I_i \geq 0, \ \forall i = 1, \dots, T \end{cases}$$

Quadratic objective function with linear constraints.

- We want to obtain the best solution to a mathematical programming problem in which both objective function and constraints have general non-linear forms.

- Most of the problems are non-linear indeed!

  – Unconstrained Problems (often dealt with differential calculus)

- Constrained Problems (may include systems of equations to be solved)

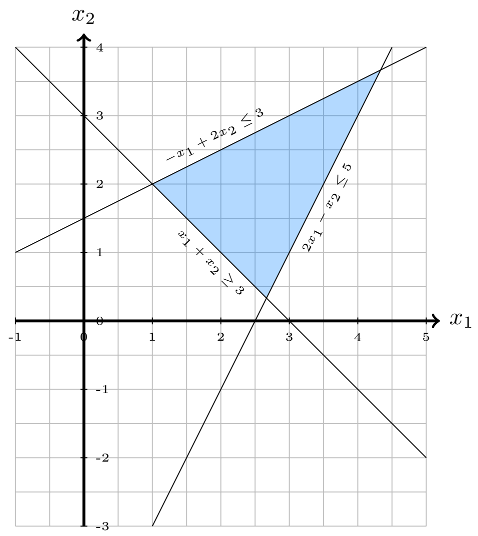
- Classical underlying mathematical theories do not necessarily provide practical methods suitable for efficient numerical computation.

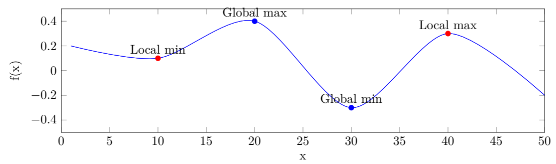
- Points of optimality can be anywhere inside the problem boundaries

- No methods applicable to all non-linear problems

2.0.1 Feasible region

Set of points satisfying all the constraints (the area between constraint boundaries).





A local maximum of the function $f(x)$ exists in x^* if there is a small positive number ϵ such that

$$f(x^*) > f(x),\ \forall x \in \mathbf{R} : \|x - x^*\| < \epsilon$$

A global maximum of $f(x)$ exists in x^* if

$$f(x^*) > f(x),\ \forall x \in \mathbf{R}$$

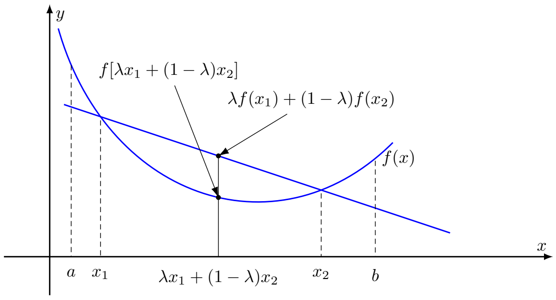
(analogous definitions for local/global minimum)

2.0.2 Concavity/convexity

For a continuous convex function, given any two points x_1 and x_2:

$$f[\lambda x_1 + (1-\lambda)x_2] \leq \lambda f(x_1) + (1-\lambda)f(x_2),\ 0 \leq \lambda \leq 1 \tag{2.1}$$

(analogous situation for a continuous concave function)



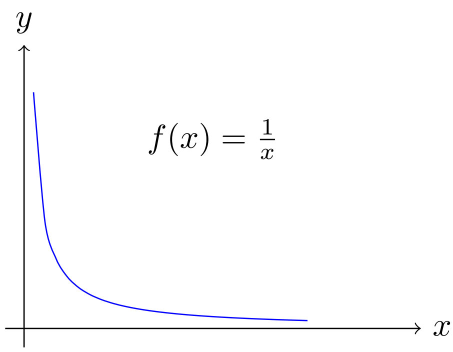
The term "convex" can be applied both to sets and functions. A set $S \in \mathbf{R}^n$ is a *convex set* if the straight line segment connecting any two points in S lies entirely inside S.

Note that $f(x)$ is a convex function if its domain S is a convex set and if for any two points x_1 and x_2 in S Eq. 2.1 holds.

Exercise 1

Can you draw a convex function that depends on two variables, $f(x, y)$? Can you generalize Eq. 2.1?

- If a convex nonlinear function is to be optimized without constraints, a global minimum may occur when $f'(x) = 0$. Not always this is the case:



- If the feasible region for a nonlinear programming problem is convex, each of the constraint functions is convex and are of the form $g_i(x) \leq b_i$

- A local minimum is guaranteed to be a global minimum for a convex objective function in a convex feasible region, and

- a local maximum is guaranteed to be a global maximum for a concave objective function in a convex feasible region.

Many functions in nonlinear programming problems are neither concave nor convex!

Exercise 2

Is the function $f(x) = x^2$ convex? Are the regions defined by $x^2 = 4$ or $x^2 \geq 9$ convex?

Exercise 3

Are the functions $f(x,y) = 3x^2 - 2xy + y^2 + 3e^{-x}$ and $g(x,y) = x^4 - 8x^3 + 24x^2 - 32x + 16$ convex?

Exercise 4

Consider these two nonlinear problems[2]:

$$\begin{array}{ll|ll} \text{minimize} & f(x,y) = x - 2xy + 2y & \text{minimize} & f(x,y) = x - 2xy + 2y \\ \text{subject to} & \begin{cases} x^2 + 3y^2 \leq 10 \\ 3x + 2y \geq 1 \\ x, y \geq 0 \end{cases} & \text{subject to} & \begin{cases} x^3 - 12x - y \geq 0 \\ x \geq 1 \end{cases} \end{array}$$

Are their feasible regions convex?

$$(\mathrm{Hess}\, f)_{ij} \equiv \frac{\partial^2 f}{\partial x_i \partial x_j} \tag{2.2}$$

$$\mathrm{Hess}\left(\frac{x^2}{y}\right) = \begin{bmatrix} \frac{2}{y} & -\frac{2x}{y^2} \\ -\frac{2x}{y^2} & \frac{2x^2}{y^3} \end{bmatrix} \tag{2.3}$$

2.1 Session 2. Unconstrained optimization

Let f be a function of n variables and let $1 \leq i \leq n$ and $1 \leq j \leq n$. If the partial derivatives f'_i and f'_j of f exist in an open set S containing $(x_1, \ldots, x_n)$ and both of these partial derivatives are differentiable at $(x_1, \ldots, x_n)$ then $(\mathrm{Hess}\, f)_{ij} = (\mathrm{Hess}\, f)_{ji}$.

- If the Hessian at a given point has all positive eigenvalues, it is said to be a positive-definite matrix. This is the multivariable equivalent of concave up (we called it ”convex” in this course).
- If all of the eigenvalues are negative, it is said to be a negative-definite matrix. This is like concave down. (we call it ”concave” in this course).
- It the eigenvalues are mixed, you have a saddle point.

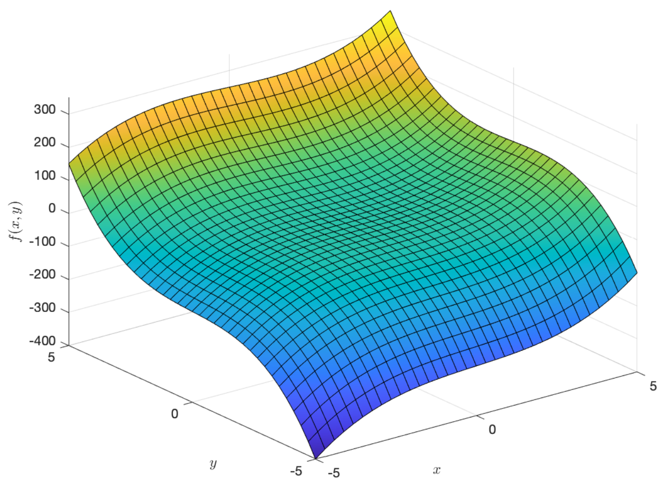
Exercise 5

Study the critical points of the function $f(x,y) = 4x + 2y - x^2 - 3y^2$

Exercise 6

Check the character of the stationary point in $(0,0)$ for the functions:

- $f(x, y) = x^2 + y^2$
- $g(x, y) = -x^2 - y^2$
- $h(x, y) = x^2 - y^2$
- $k(x, y) = x^3 + 2y^3 - xy$ (is it concave or convex in $(3, 3)$? what about in $(-3, -3)$?)



For unconstrained problems with just one variable x, if first and second derivatives exist in x^*:

Necessary conditions If $\frac{df}{dx} = 0$ at $x = x^*$

- $\frac{d^2 f}{dx^2} \geq 0$, for a local minimum at $x = x^*$
- $\frac{d^2 f}{dx^2} \leq 0$, for a local maximum at $x = x^*$

Sufficient conditions If $\frac{df}{dx} = 0$ at $x = x^*$

- $\frac{d^2 f}{dx^2} > 0$, for a local minimum at $x = x^*$
- $\frac{d^2 f}{dx^2} < 0$, for a local maximum at $x = x^*$

For unconstrained problems with several variables $\mathbf{x} = (x_1, \ldots, x_n)$:

Sufficient condition If $\frac{\partial f}{\partial x_i} = 0$ at $\mathbf{x} = \mathbf{x}^*$:

- $\boldsymbol{\nabla} f = \mathbf{0}$, for a local minimum at $x = x^*$ if the function is convex;
- $\boldsymbol{\nabla} f = \mathbf{0}$, for a local maximum at $x = x^*$ if the function is concave.

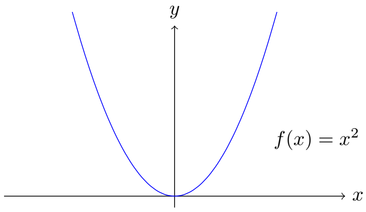
Decision conditions The function f is a convex function if H_f is positive definite or positive semidefinite for all $\mathbf{x}$; and f is concave if H_f is negative definite or negative semidefinite for all $\mathbf{x}$. For $f : \mathbf{R}^2 \to \mathbf{R}$

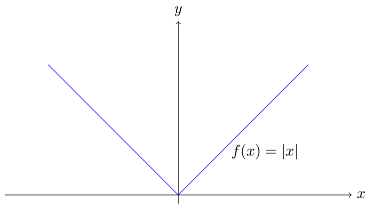
$$\mathbf{H}_f(x, y) = \begin{pmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} \end{pmatrix} = \begin{pmatrix} A & C \\ C & B \end{pmatrix}$$

Let f be a twice-differentiable function of many variables on the convex open set S and denote the Hessian of f at the point x by $H(x)$. Then

- f is concave if and only if $H(x)$ is negative semidefinite $\forall x \in S$
- if $H(x)$ is negative definite $\forall x \in S$ then f is strictly concave (see Eq. 2.1)
- f is convex if and only if $H(x)$ is positive semidefinite $\forall x \in S$
- if $H(x)$ is positive definite $\forall x \in S$ then f is strictly convex (see Eq. 2.1)

Things are not so simple if, for example, functions are not derivable in x^*:





Or, if the function is not convex (nor concave), the sufficient condition is lost and we may find local optimal points, instead of global.

For constrained optimization problems, the shape of the feasible region adds difficulty.

2.1.1 Iterative methods

Line search

Let us assume a concave function. First, establish error tolerance ϵ. Then:

1. Set up lower and upper bounds for x by finding x_l, x_u such that $\frac{df}{dx_l} \geq 0$ and $\frac{df}{dx_u} \leq 0$
2. Get new trial intermediate solution (eg, $x = \frac{x_l + x_u}{2}$)
3. If $x_u - x_l \leq \epsilon$, then terminate
4. If $\frac{df}{dx} \geq 0$, set $x_l = x$; if $\frac{df}{dx} \leq 0$, set $x_u = x$
5. Go to step 1

For an implementation, see this link For a continuous function it guarantees to find an extremum.

Golden search

Two quantities are in the golden ratio if their ratio is the same as the ratio of their sum to the larger of the two quantities:

$$\frac{a+b}{a} = \frac{a}{b} = \psi \Rightarrow \psi = \frac{1+\sqrt{5}}{2} = 1.616033988...$$

Procedure of the Golden search method, given a function to be minimized, $f(x)$, the interval to be searched as $x \in (x_1, x_4)$, and their functional values $f(x_1)$ and $f(x_2)$:

1. Calculate an interior point and its functional value $f(x_3)$. Ensure the position of x_3 follows the golden ratio between the distances $\frac{x_2-x_1}{x_3-x_1} = \frac{x_3-x_1}{x_2-x_3} = \psi$.

2. Using the triplet, determine if convergence criteria are fulfilled. If they are, estimate the x at the minimum from that triplet and terminate/return.

3. Select x_4 within the largest interval, following the same ratios as in Step 1.

4. The three points for the next iteration will be the one where $f(x)$ is a minimum, and the two points closest to it in x.

5. Go to step 2.

Exercise 7

Use the Golden Search method to find the minimum of function $f(x) = x^2 - 6x + 15$ in the interval $x \in (0, 10)$. Set up a spreadsheet to do so.

Programming exercise 1. *E2. Build a Python code that can run one-dimensional search algorithms for user defined functions, using or not first derivatives. Implement the one-dimensional line search and the golden search algorithms and find the optimal solution for the functions:*

- $f(x) = x^4 - 16x^3 + 45x^2 - 20x + 203$ *within the range* $x \in (2.5, 14)$

- $g(x) = x^5 - 2x^4 - 23x^3 - 12x^2 + 36x$ *within the range* $x \in (2, 3)$

Try both a direct implementation as well as the use of solvers in scipy.

Exercise 8

Draw the *Rosenbruck's function*, $f(x, y) = 100(y-x^2)^2+(1-x)^2$ and discuss on its convexity, the feasibility of its minimization and the most appropriate numerical method to use.

As a reference (not the central aim of this course) here is a list of some algorithms/methods to explore to perform unconstrained optimization in multiple dimensions:

- Newton's method,

- Quasi-Newton methods (e.g., BFGS),

- Trust-region methods,

- Conjugated gradient methods.

and their many variants.

2.2 Session 4: Constrained optimization

In the previous section we have considered functions of several variables that are independent from each other. What happens if the variables are, however, not independent from each other? (see excellent material in Richard Weber's web site)

What is the maximum height we can attain constrained to the trajectory of the road?

Let us consider optimizing a function $f(x, y)$ (the height of the hill in the previous exemple) subject to the constrain $g(x, y) = c$ (the equation describing the path, the road).

One can simply substitute $g(x, y) = c$ in $f(x, y)$, but this may not involve trivial algebra.

Alternatively, let us consider the method of the Lagrange multipliers. To optimize $f(x, y)$ we require

$$df = \frac{\partial f}{\partial x}dx + \frac{\partial f}{\partial y}dy = 0$$

we saw that if dx and dy were independent, $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial y} = 0$. But they are not here: they are constrained by $g(x, y)$, which is a constant:

$$dg = \frac{\partial g}{\partial x}dx + \frac{\partial g}{\partial y}dy = 0$$

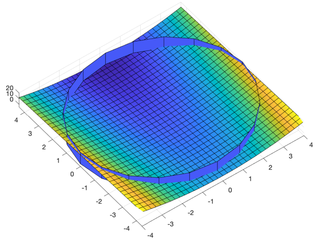
Let us consider a parameter λ (the *Lagrange multiplier*) and obtain an expression for which dx and dy are independent:

$$d(f + \lambda g) = (\frac{\partial f}{\partial x} + \lambda\frac{\partial g}{\partial x})dx + (\frac{\partial f}{\partial y} + \lambda\frac{\partial g}{\partial y})dy = 0$$

Now:

$$\begin{aligned}
\frac{\partial f}{\partial x} + \lambda\frac{\partial g}{\partial x} &= 0 \\
\frac{\partial f}{\partial y} + \lambda\frac{\partial g}{\partial y} &= 0 \\
g(x, y) &= c
\end{aligned}$$

are sufficient to find λ, x^* and y^*.



Exercise 9

Find the optimal values of the function $f(x,y) = x^2 - y$ constrained within the circumference $x^2 + y^2 = 4$. (note that, interestingly, in the optimal value x^* we will get a gradient for the constraint that is parallel to the gradient of the function: $\boldsymbol{\nabla} f(x^*) = \lambda^* \boldsymbol{\nabla} g(x^*)$).

The general problem we study takes the form

$$\begin{array}{ll} \text{maximize} & f(x) \\ \text{subject to} & \begin{cases} x \in X \\ g(x) = b \end{cases} \end{array}$$

where

$$\begin{array}{rl} x \in \mathbf{R}^n & (n \text{ decision variables}) \\ f : \mathbf{R}^n \to \mathbf{R} & (\text{objective function}) \\ X \subseteq \mathbf{R}^n & (\text{regional constraints}) \\ \left.\begin{array}{l} g : \mathbf{R}^n \to \mathbf{R}^m \\ b \subseteq \mathbf{R}^m \end{array}\right\} & (m \text{ functional constraints}) \end{array}$$

If the constraint is $g(x) \le b$ we introduce a **slack variable** z and write:

$$g(x) + z = b,\ z \ge 0$$

Regional and functional constraints define the **feasible set** for x

In the Lagrange approach, the constrained maximization (minimization) problem is rewritten as a Lagrange function

$$L(x, \lambda) = f(x) - \lambda g(x)$$

whose optimal point is a saddle point, i.e. a global maximum (minimum) over the domain of the choice variables and a global minimum (maximum) over the multipliers.

Let us take this example:

$$\begin{array}{ll} \text{maximize} & f(x) = x_1 - x_2 - 2x_3 \\ \text{subject to} & \begin{cases} x_1 + x_2 + x_3 = 5 \\ x_1^2 + x_2^2 = 4 \\ x = (x_1, x_2, x_3) \in \mathbf{R}^3 \end{cases} \end{array}$$

1. We write the Lagrangian as (note we have two constraints):

$$\begin{aligned} L(x,\lambda) &= x_1 - x_2 - 2x_3 - \lambda_1(x_1 + x_2 + x_3 - 5) - \lambda_2(x_1^2 + x_2^2 - 4) \\ &= [x_1(1-\lambda_1) - \lambda_2 x_1^2] + [x_2(-1-\lambda_1) - \lambda_2 x_2^2] + [x_3(-2-\lambda_1)] \\ &\quad +5\lambda_1 + 4\lambda_2 \end{aligned}$$

2. We want to minimize the Lagrangian subject to the regional constraints $x = (x_1, x_2, x_3) \in \mathbf{R}^3$.

3. We find the values λ such that will define a feasible set

$$Y = \left\{ \lambda : \underbrace{\min}_{x \in \mathbf{R}^3} L(x,\lambda) > -\infty \right\}$$

So, we only consider the values of λ for which we obtain a finite minimum. In the example

$$Y = \{\lambda : \lambda_1 = -2, \lambda_2 < 0\}$$

4. For $\lambda \in Y$, the minimum will be obtained at some $x(\lambda)$ (that depends on λ in general). Here, $x(\lambda) = (3/2\lambda_2, 1/2\lambda_2, x_3)^T$.

5. Finally, we find a λ value that provides a feasible value $x(\lambda)$. Here

$$x_1^2 + x_2^2 = 4 \Rightarrow \lambda_2 = -\sqrt{5/8}$$

So, x^* is optimal:

$$x^* = (-3\sqrt{2/5}, -\sqrt{2/5}, 5 + 4\sqrt{2/5})^T;\ \lambda^* = (-2, -\sqrt{5/8})^T$$

Exercise 10

Use the Lagrange multipliers to solve this problem:

$$\begin{aligned} \text{minimize} \quad & f(x,y) = x^2 y \\ \text{subject to} \quad & x^2 + y^2 = 1 \end{aligned}$$

[1] [2]

Let us consider a general optimization problem with constraints: let $f : \mathbf{R}^n \to \mathbf{R}$ be the objective function and $\mathbf{g} : \mathbf{R}^n \to \mathbf{R}^m$ a set of constraint functions ($\mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_m(\mathbf{x}))$ with $\mathbf{x} = (x_1, \ldots, x_n)$. We want to obtain

$$\text{Min } f(\mathbf{x}) \text{ such that } \mathbf{g}(\mathbf{x}) \geq 0$$

Theorem 1. *If $\boldsymbol{x}^*$ is a local minimum for the optimisation problem and the constraints $\boldsymbol{g}(\boldsymbol{x}) \geq 0$ are satisfied at $\boldsymbol{x}^*$, then $\boldsymbol{x}^*$ must satisfy:*

$$\begin{aligned} \boldsymbol{\nabla} f(\boldsymbol{x}) &= \lambda_1 \boldsymbol{\nabla} g_1(\boldsymbol{x}) + \cdots + \lambda_m \boldsymbol{\nabla} g_m(\boldsymbol{x}) \\ 0 &= \lambda_1 g_1(\boldsymbol{x}) = \cdots = \lambda_m g_m(\boldsymbol{x}) \\ 0 &\leq \boldsymbol{g}(\boldsymbol{x}) \end{aligned}$$

[1] https://people.maths.bris.ac.uk/~maxmr/opt/kt1.pdf

[2] https://www.cmi.ac.in/~madhavan/courses/dmml2018/literature/Lagrangian_Methods_for_Constrained_Optimization.pdf

Exercise 11

Consider this NLO problem[2]:

$$\begin{aligned} \text{maximize} \quad & f(x,y) = x^2y + 2y^2 \\ \text{subject to} \quad & \begin{cases} x + 3y \leq 9 \\ x + 2y \leq 8 \\ 3x + 2y \leq 18 \\ 0 \leq x \leq 5 \\ 0 \leq y \leq 2 \end{cases} \end{aligned}$$

Solve the program graphically- Begin at point $(0,0)$ and check the gradient. Check the KKT conditions. If they are not satisfied, find an improving direction in the feasible region.

To better understand these exercises, you can make use of geogebra.

Exercise 12

Is it possible to maximize $f(x,y) = xy$ subject to $100 \geq x + y$ and $x \leq 40$?

Exercise 13

What about maximizing $f(x,y) = xy$ subject to $x + y^2 \leq 2$ and $x, y \geq 0$?

Chapter 3

Linear programming

3.1 Session 6: Introduction to linear programming

- Learn how to formulate a linear programming model
- Get familiar with the terms feasible region, feasiblke solutions and optimal feasible solutions
- Managing graphical solution of the linear programming model
- Identify models with unique, multiple or no optimal feasible solutions
- Introducing the Simplex method

Linear programs have a linear objective function and linear constraints, which may include both equalities and inequalities. The feasible set is a polytope, that is, a convex, connected set with at, polygonal faces. The contours of the objective function are planar. [5]

A wide variety of applications can be modeled with linear programming.

Steps:

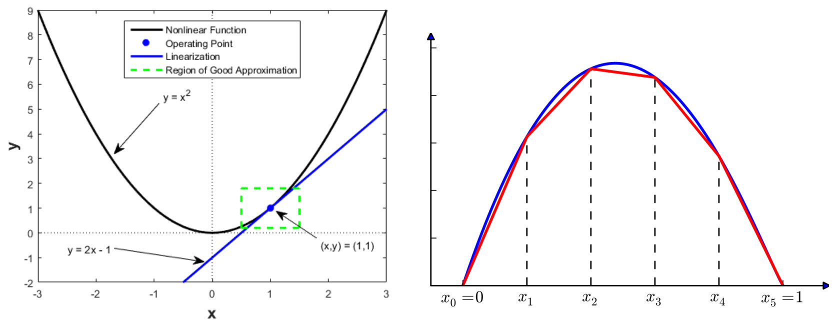
1. identify the controllable decision variables x_i, with $i = 1, \ldots, n$.
2. establish the objective criterion: to either maximize or minimize some function of the form:

$$z = c_1 x_1 + \cdots + c_n x_n = \sum_{i=1}^{n} c_i x_i$$

   where c_i represents problem dependent constants.

3. resource limitations and bounds on decision variables as linear equations or inequalities like

$$a_1 x_1 + \cdots + a_n x_n \leq b$$



Matlab example

Exercise 14

A manufacturer of computer system components assembles two models of wireless routers, model A and model B. The amounts of materials and labor required for each assembly, and the total amounts available, are shown in the following table. The profits that can be realized from the sale of each router are \$22 and \$28 for models A and B, respectively, and we assume there is a market for as many routers as can be manufactured.[2]

	Resources per Unit A	Resources B per Unit B	Resources available
Materials	8	10	3400
Labor	2	3	960

How to maximize the benefits?

$$\text{maximize} \quad z = 22x_A + 28x_B$$

$$\text{subject to} \quad \begin{aligned} 8x_A + 10x_B &\leq 3400 \\ 2x_A + 3x_B &\leq 960 \\ x_A &\geq 0 \\ x_B &\geq 0 \end{aligned}$$

Exercise 15

A company wishes to minimize its combined costs of production and inventory over a four-week time period. An item produced in a given week is available for consumption during that week, or it may be kept in inventory for use in later weeks. Initial inventory at the beginning of week 1 is 250 units. The minimum allowed inventory carried from one week to the next is 50 units. Unit production cost is \$15, and the cost of storing a unit from one week to the next is \$3. The following table shows production capacities and the demands that must be met during each week.[2]

Period	Production capacity	Demand
1	800	900
2	700	600
3	600	800
4	800	600

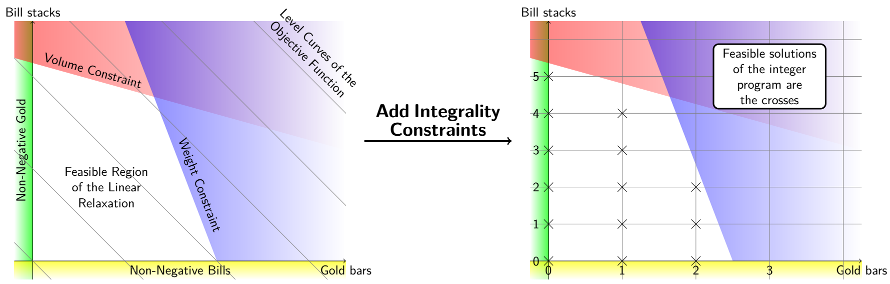
A minimum production of 500 items per week must be maintained. Inventory costs are not applied to items remaining at the end of the fourth production period, nor is the minimum inventory restriction applied after this final period.

Let x_i be the number of units produced during the i^{th} week, for $i = 1, , 4$. The formulation is somewhat more manageable if we let A_i denote the number of items remaining at the end of each week (accounting for those held over from previous weeks, those produced during the current week, and those consumed during the current week). Note that the A_i values are not decision variables, but merely serve to simplify our written formulation. Thus,

$$\begin{aligned} A_1 &= 250 + x_1 - 900 \\ A_2 &= A_1 + x_2 - 600 \\ A_3 &= A_2 + x_3 - 800 \\ A_4 &= A_3 + x_4 - 600 \end{aligned}$$

$$\text{minimize} \quad z = 15 \cdot (x_1 + x_2 + x_3 + x_4) + 3 \cdot (A_1 + A_2 + A_3)$$

$$\text{subject to} \quad \begin{aligned} 500 \le x_1 \le 700 \\ 500 \le x_2 \le 700 \\ 500 \le x_3 \le 600 \\ 500 \le x_4 \le 800 \\ x_i \ge 0,\ i = 1, 2, 3, 4 \end{aligned}$$



from Lê Nguyên Hoang's http://www.science4all.org

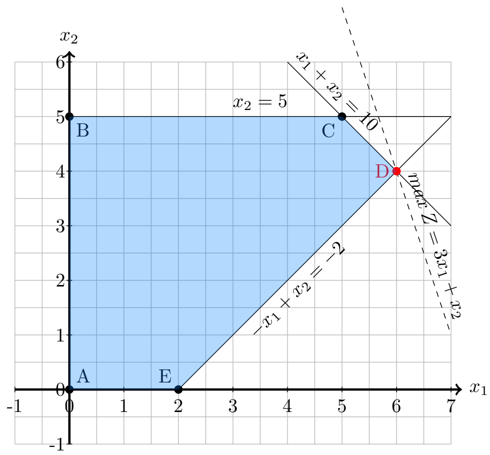
3.1.1 Graphical solution

- Remember: An optimal feasible solution is a point in the feasible space that is as effective as any other point in achieving the specified goal.
- The solution of linear programming problems with only two decision variables can be illustrated graphically.

- If an optimal feasible solution exists, it occurs at one of the extreme points of the feasible space.
- We illustrate this for problems with one, with multiple, and with none optimal feasible solutions.

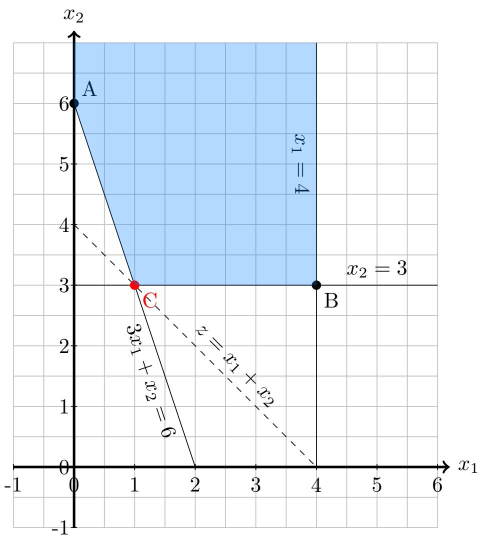
Exercise 16

$$\begin{array}{ll} \text{maximize} & z = 3x_1 + x_2 \\ \text{subject to} & \begin{array}{rcl} x_2 & \leq & 5 \\ x_1 + x_2 & \leq & 10 \\ -x_1 + x_2 & \geq & -2 \\ x_1, x_2 & \geq & 0 \end{array} \end{array}$$



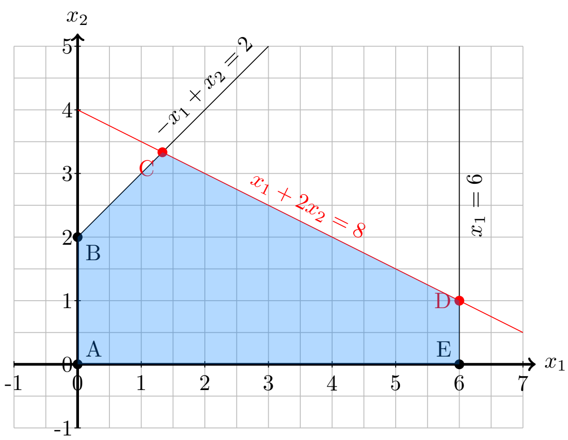
Exercise 17

$$\begin{array}{ll} \text{minimize} & z = x_1 + x_2 \\ \text{subject to} & \begin{array}{rcl} 3x_1 + x_2 & \geq & 6 \\ x_2 & \geq & 3 \\ x_1 & \leq & 4 \\ x_1, x_2 & \geq & 0 \end{array} \end{array}$$



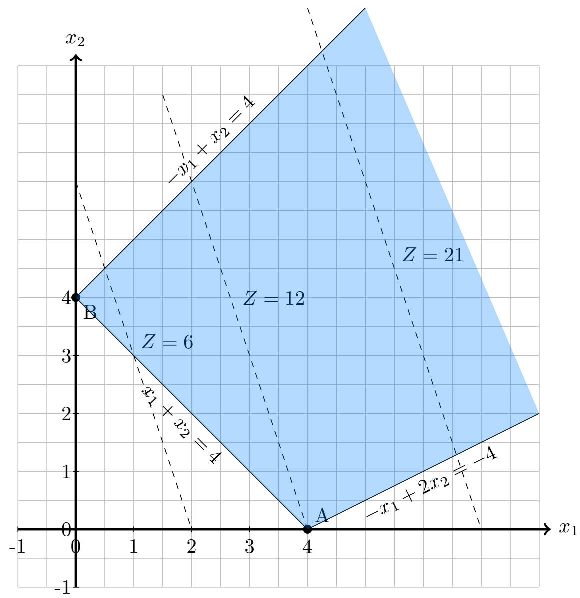
Exercise 18

$$
\begin{array}{llcl}
\text{maximize} & z = x_1 + 2x_2 & & \\
\text{subject to} & -x_1 + x_2 & \leq & 2 \\
 & x_1 + 2x_2 & \leq & 8 \\
 & x_1 & \leq & 6 \\
 & x_1, x_2 & \geq & 0
\end{array}
$$



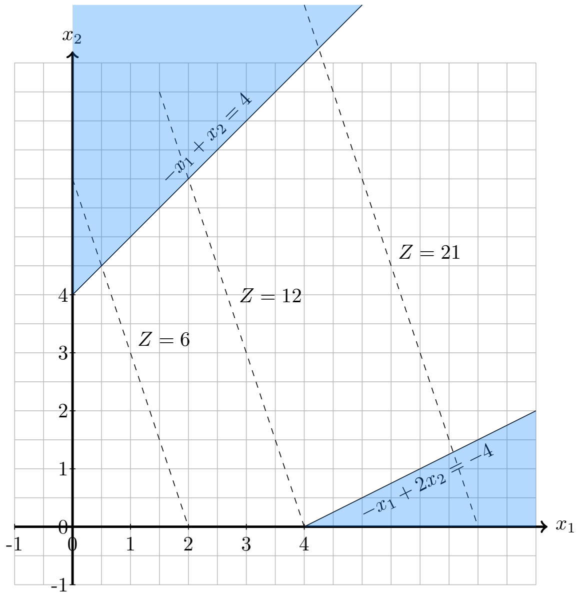
Exercise 19

$$
\begin{array}{ll}
\text{maximize} & z = 3x_1 + x_2 \\
\text{subject to} & \begin{array}{rcl} x_1 + x_2 & \geq & 4 \\ -x_1 + x_2 & \leq & 4 \\ -x_1 + 2x_2 & \geq & -4 \\ x_1, x_2 & \geq & 0 \end{array}
\end{array}
$$



Exercise 20

$$
\begin{array}{ll}
\text{maximize} & z = 3x_1 + x_2 \\
\text{subject to} & \begin{array}{rcl} -x_1 + x_2 & \geq & 4 \\ -x_1 + 2x_2 & \leq & -4 \\ x_1, x_2 & \geq & 0 \end{array}
\end{array}
$$



Programming exercise 2. *E4. Build a Python code using Google OR tools' MPSolver interface to solve an arbitrary LP problem. Test it with the different exercises in this presentation.*

3.1.2 General solution method

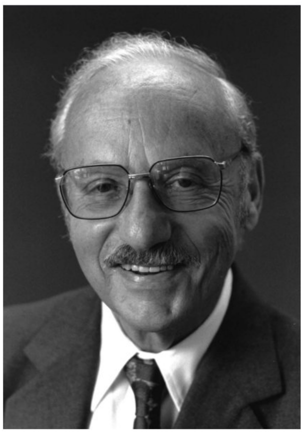
- If an optimal solution exists, it occurs at an extreme point of the feasible region.
- Only the finitely many extreme points need be examined (rather than all the points in the feasible region).
- Thus, an optimal solution may be found systematically by considering the objective function values at the extreme points.
- In fact, in actual practice, only a small subset of the extreme points need be examined.

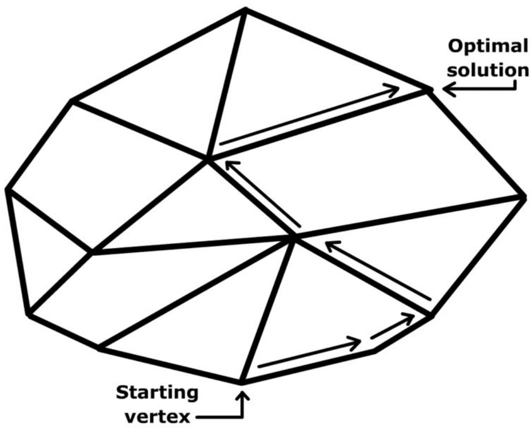
This is the foundation for a **general solution method** of a LP problem called the **Simplex method**.

3.2 Session 7: Introduction to the Simplex method for LP

- Understanding the rational behind the Simplex method for LP.
- Understanding and practicing the algorithm in 2 variables.
- Recognizing the different types of results one can achieve in LP from the Simplex Method algorithm.

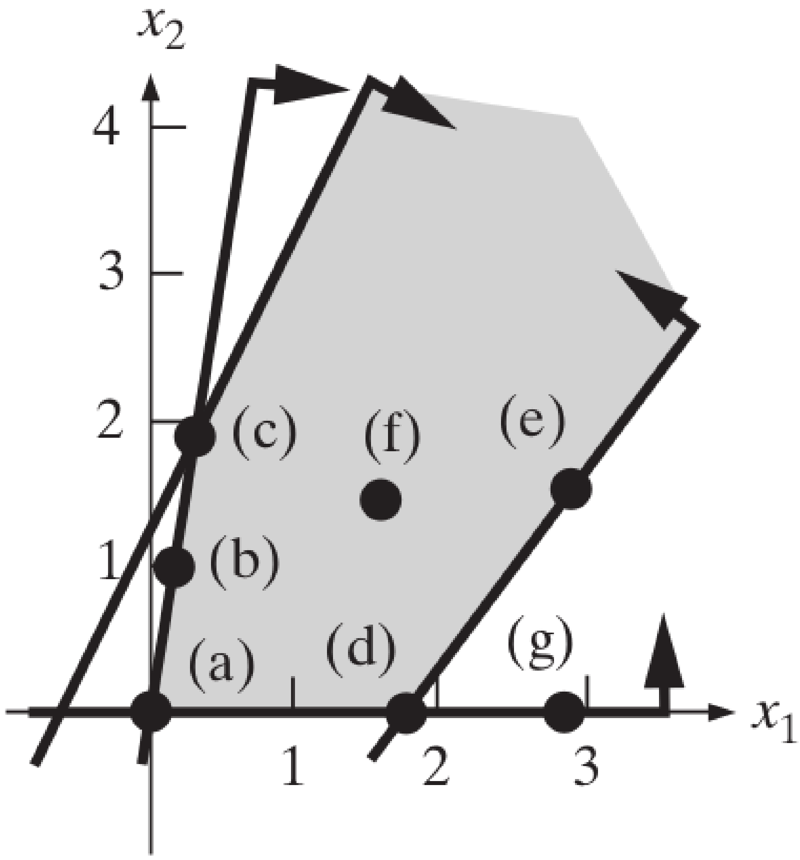
- Getting familiar with slack, surplus and artificial variables.





Left: George Dantzig (1914-2005), the American mathematical scientist who devised linear programming and the simplex algorithm. Right: The simplex algorithm moves along the edges of the polytope until it reaches the optimum solution [Image Wikimedia Commons].

3.2.1 Preparing the LP problem for applying the Simplex method



(a), (c), and (d) are both extreme points and boundary points.
(b) and (e) are boundary points that are not extreme.
(f) is interior because no constraint is active.
(g) is neither interior nor boundary (nor extreme) because it is infeasible.[6]

Standard form

For a LP with n variables and m constraints, the standard form is given by:

$$\begin{array}{ll} \text{maximize} & z = c_1x_1 + c_2x_2 + \cdots + c_nx_n \\ \text{subject to} & \begin{array}{rcl} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & = & b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & = & b_2 \\ \vdots & = & \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n & = & b_m \end{array} \end{array}$$

where $x_1, \ldots, x_n \geq 0$ and $b_1, \ldots, b_m \geq 0$

Thus:

$$\begin{array}{ll} \text{maximize} & z = cx \\ & Ax = b \\ \text{subject to} & x \geq 0 \\ & b \geq 0 \end{array}$$

Of course, not always the system is proposed in this standard form, so[2]:

- We need to leave a maximization problem. If the problem is to minimize the objective function, we simply multiply it by (-1).
- In order to change inequality constraints into equality constraints we use *slack* variables for $\leq$ inequalities:

$$3x_1 + 4x_2 \leq 7 \rightarrow 3x_1 + 4x_2 + s_1 = 7$$

  or *surplus* variables for $\geq$ inequalities:

$$x_1 + 3x_2 \geq 10 \rightarrow x_1 + 3x_2 - s_2 = 10$$

- All variables should be non-negative. If some variable x_1 needs to be considered as unrestricted in sign, we will replace it by $x_1' - x_1''$, with $x_1', x_1'' \geq 0$

3.2.2 The algebra behind the standard form

- The system of linear equations, $Ax = b$, consists of m equations and n unknowns (the original decision variables plus the rest introduced to get a standard form). If the system has any solution, then $m \leq n$.
- If $m = n$ (and if $rank(A) = m$ and A is nonsingular), then $x = A^{-1}b$. No optimization needed!
- If $m < n$, there are infinitely many solutions and $n-m$ degrees of freedom.
- A **basic solution** is obtained by setting $n - m$ of the variables to zero (**non-basic variables**), and solving for the remaining (**basic**) m variables. The number of basic solutions is $\binom{n}{n-m} = \binom{n}{m} = \frac{n!}{m!(n-m)!}$
- Basic solutions that do not satisfy all problem constraints and non-negativity constraints are **infeasible solutions**.
- An **optimal basic feasible solution** is a basic feasible solution that optimizes the objective function.

3.2.3 The Simplex algorithm

We define two extreme points of the feasible region (or two basic feasible solutions) as being adjacent if all but one of their basic variables are the same. Thus, *a transition from one basic feasible solution to an adjacent basic feasible solution can be thought of as exchanging the roles of one basic variable and one non-basic variable.* The Simplex method performs a sequence of such transitions and thereby examines a succession of adjacent extreme points. A transition to an adjacent extreme point will be made only if by doing so the objective function is improved (or stays the same). It is a property of linear programming problems that this type of search will lead us to the discovery of an optimal solution (if one exists). The Simplex method is not only successful in this sense, but it is remarkably efficient because it succeeds after examining only a fraction of the basic feasible solutions.[2]

The algorithm consists in:

1. obtaining an initial feasible solution,
2. make transitions to a better basic feasible solution,
3. recognizing an optimal solution.

From any basic feasible solution, we have the assurance that, if a better solution exists at all, then there is an adjacent solution that is better than the current one.[2]

1. Convert the system of inequalities to equations (using slack or surplus variables).
2. Set the objective function to zero.
3. Create the Simplex tableau and label active and basic variables.
4. Select the pivot column (the one with the most negative coefficient in the zeroed objective function). Linked to the **entering variable**.
5. Select the pivot row (once divided the entry in the constant column by the ceffient in that row in the pivot column, we choose the smallest ratio). Linked to the **leaving variable**.
6. The pivot is the intersection between the pivot row and pivot column.
7. Use the pivot value to make zeros in the rest of elements in the column.
8. Repeat the process from step 4, until the last row is all non-negative.

$$
\begin{array}{ll}
\text{maximize} & z = 8x_1 + 5x_2 \\
\text{subject to} & \begin{array}{rcl} x_1 & \leq & 150 \\ x_2 & \leq & 250 \\ 2x_1 + x_2 & \leq & 500 \\ x_1, x_2 & \geq & 0 \end{array}
\end{array}
$$

We build first the standard form:

$$
\begin{array}{rcl}
-8x_1 - 5x_2 - 0s - 0t - 0u + z & = & 0 \\
x_1 + s & = & 150 \\
x_2 + t & = & 250 \\
2x_1 + x_2 + u & = & 500
\end{array}
$$

We write the coefficients matrix and we identify the basic ($m = 3$) and non-basic ($n - m = 5 - 3 = 2$ degrees of freedom) variables:

$$
\begin{array}{cc|c|cccccc|c}
 & & z & x_1 & x_2 & s & t & u & b \\
\hline
 & z & 1 & -8 & -5 & 0 & 0 & 0 & 0 \\
\hline
\rightarrow & s & 0 & 1 & 0 & 1 & 0 & 0 & 150 \\
 & t & 0 & 0 & 1 & 0 & 1 & 0 & 250 \\
 & u & 0 & 2 & 1 & 0 & 0 & 1 & 500 \\
\text{basic} & & & \uparrow & & & & & \\
\end{array}
\tag{3.1}
$$

Note that the basic variables are those for which each column is a collection of 1 and zero. We will arbitrarily assign zeros to the non-basic variables. So a possible solution is $\boxed{P_A = (x_1 = 0, x_2 = 0) \Rightarrow z = 0}$.

The arrow marks the pivot column. We will take the pivot row by considering which is the lowest value among 150/1 and 500/2. So, the pivot row corresponds to the s basic variable.

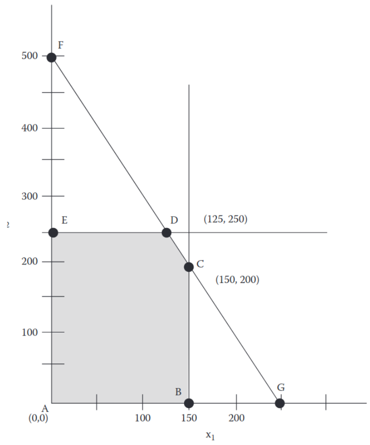
$$
\begin{array}{cc|c|cccccc|c}
 & & z & x_1 & x_2 & s & t & u & b \\
\hline
R_1 + 8R_2 & z & 1 & 0 & -5 & 8 & 0 & 0 & 1200 \\
\hline
 & x_1 & 0 & 1 & 0 & 1 & 0 & 0 & 150 \\
 & t & 0 & 0 & 1 & 0 & 1 & 0 & 250 \\
\rightarrow R_4 - 2R_2 & u & 0 & 0 & 1 & -2 & 0 & 1 & 200 \\
 & \text{basic} & & & \uparrow & & & & \\
\end{array}
$$

$P_B = (150, 0)$ and $z(P_B) = 1200$ with $t = 250$, $u = 200$, $s = 0$.

$$
\begin{array}{cc|c|cccccc|c}
 & & z & x_1 & x_2 & s & t & u & b \\
\hline
R_1 + 5R_4 & z & 1 & 0 & 0 & -2 & 0 & 5 & 2200 \\
\hline
 & x_1 & 0 & 1 & 0 & 1 & 0 & -1 & 150 \\
\rightarrow R_3 - R_4 & t & 0 & 0 & 0 & 2 & 0 & 1 & 50 \\
 & x_2 & 0 & 0 & 1 & -2 & 0 & 1 & 200 \\
 & \text{basic} & & & & \uparrow & & & \\
\end{array}
$$

$$
\begin{array}{cc|c|cccccc|c}
 & & z & x_1 & x_2 & s & t & u & b \\
\hline
R_1 + 5R_4 & z & 1 & 0 & 0 & 0 & 1 & 4 & 2250 \\
\hline
 & x_1 & 0 & 1 & 0 & 0 & -1/2 & 1/2 & 125 \\
\rightarrow R_3 - R_4 & s & 0 & 0 & 0 & 1 & 1/2 & -1/2 & 25 \\
 & x_2 & 0 & 1 & 0 & 1 & 0 & 0 & 250 \\
 & \text{basic} & & & & \uparrow & & & \\
\end{array}
$$

$\boxed{P_D = (125, 250)}$ and $\boxed{z(P_D) = 2250}$ with $t, u = 0$ (binding constraints), $s = 25$ (non-binding constraint).



Simplex steps for the given example (taken from [2]).

Solve, using the Simplex method, the Exercises 3 through 6 of the previous session.

3.2.4 Initial solutions for General Constraints

When adding slack variables, we end up finding an initial feasible set of basic variables. But what happens when not all constraints are of the form "$\leq$"?[2]

1. Make all right hand sides b_i of the constraint (in)equalities positive. If they are not, multiply the expression by (-1).

2. If we end up just with slack variables, they can be directly used as initial basic variables. If the expressions are of the type $\geq$ the selection of initial basic variables is not trivial.

3. We then create **artificial variables**, as a trick to create a starting basic solution

$$\text{maximize} \quad z = x_1 + 3x_2$$

$$\text{subject to} \quad \begin{aligned} 2x_1 - x_2 &\leq -1 \\ x_1 + x_2 &= 3 \\ x_1, x_2 &\geq 0 \end{aligned}$$

By leaving the right hand side of the constraints positive and adding a surplus variable, we end up having

$$\text{subject to} \quad \begin{aligned} -2x_1 + x_2 - s_1 &= 1 \\ x_1 + x_2 &= 3 \end{aligned}$$

and now we introduce the artificial variables to create an initial solution:

$$\text{subject to} \quad \begin{aligned} -2x_1 + x_2 - s_1 + R_1 &= 1 \\ x_1 + x_2 + R_2 &= 3 \end{aligned}$$

with all variables non-negative. Recall that, in fact, $R_1 = R_2 = 0$, so they will be just used temporarily.

There are two main methods to solve the problem: the two-phase method, explained next, and the big-M method, that we are not explaining here.

1. First we will use the Simplex method to minimize the values of R_1 and R_2.

2. If they reach the value of zero, there is a solution for the original problem. If not, there is no feasible solution to the original problem.

In the above example, we want to minimize the function $z_R = R_1 + R_2$ or, what is the same, we want to maximize the function $z_R = -R_1 - R_2$

	x_1	x_2	s_1	R_1	R_2	*Solution*
z_R	0	0	0	1	1	0
R_1	-2	1	-1	1	0	1
R_2	1	1	0	0	1	3

Next, we transform the problem in order to get basic variables (columns of one 1 and the rest zeros). For example, by using the second and third row to make appear 0 in the first row for the artificial variables:

	x_1	x_2	s_1	R_1	R_2	*Solution*
z_R	1	-2	1	0	0	-4
R_1	-2	1	-1	1	0	1
R_2	1	1	0	0	1	3

We can solve this problem with the Simplex method, obtaining a tableau:

	x_1	x_2	s_1	R_1	R_2	*Solution*
z_R	0	0	0	1	1	0
x_2	0	1	$-1/3$	$1/3$	$2/3$	$7/3$
x_1	1	0	$1/3$	$-1/3$	$1/3$	$2/3$

which provides the optimal solution for Phase 1. This means that the original problem has solution (as $R_1 = R_2 = 0$). In Phase 2, we will replace the first row by the original maximization problem and leave out the artificial variables, solving the now standard Simplex problem:

	x_1	x_2	s_1	*Solution*
z	-1	-3	0	0
x_2	0	1	$-1/3$	$7/3$
x_1	1	0	$1/3$	$2/3$

that leads to

	x_1	x_2	s_1	*Solution*
z	2	0	0	9
x_2	1	1	0	3
s_1	3	0	1	2

Draw the graphical solution of this problem and interpret the results.

Some special cases:

Multiple optimal solutions If a zero appears in the objective function row corresponding to a non-basic variable, then that non-basic variable can enter the basis without changing the value of the objective function. So, two adjacent points have the same objective function value.

Unbounded solution If in any tableau the constraint coefficients corresponding to a non-basic variable are all either negative or zero, then that non-basic variable can be increased arbitrarily without violating any constraint. Thus, the feasible region is unbounded in the direction of that variable.

Degenerate solutions A solution to a linear programming problem is said to be degenerate if one or more of the basic variables has a value of zero. This shows redundancy in the constraints.

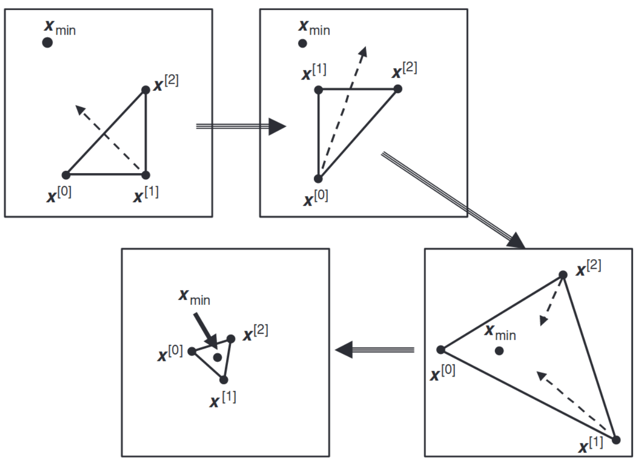
3.2.5 Application of SIMPLEX to non-linear problems

Algorithm (see example in the next page):

1. In $\mathbf{R}^2$, we form a triangle with vertices at $x^{[0]}$ and the two points $x^{[1]} = x^{[0]} + l_1 e^{[1]}$ and $x^{[2]} = x^{[0]} + l_2 e^{[2]}$. In $\mathbf{R}^N$ the triangle becomes a simplex with $N+1$ vertices.

2. We aim at moving $x^{[0]}, x^{[1]}, x^{[2]}, ...$ so that the triangle comes to enclose a local minimum x_{min} while shrinking in size. First, the vertex of highest cost function value is moved in the direction of the **triangles center** towards a region where we expect the cost function values to be lower. After a sequence of such moves, the triangle eventually contains in its interior a local minimum.

3. The size of the triangle is progressively reduced until it tightly bounds the local minimum, as the vertex moves are most likely to succeed when the triangle is small enough that $F(x)$ varies linearly over the triangle region.



The simplex method moves the vertices of the simplex (in two dimensions, a triangle) until it contains the local minimum, and shrinks the simplex to find the local minimum (taken from [1]).

[Introduction]Unit 2. Linear programming. Duality

3.3 Session 10: Duality

- Understanding Dual and Primal problems in LP
- Economic interpretation
- Conditions of optimality
- Resolution of the dual by the primal and penalty method

[1]

Linear programming problems come in pairs!

$$\begin{array}{ll} \text{maximize} & 4x_1 + x_2 + 5x_3 + 3x_4 \\ \text{subject to} & \left\{ \begin{array}{rcl} x_1 - x_2 - x_3 + 3x_4 & \leq & 1 \\ 5x_1 + x_2 + 3x_3 + 8x_4 & \leq & 55 \\ -x_1 + x_2 + 3x_3 - 5x_4 & \leq & 3 \\ x_1, x_2, x_3 & \geq & 0 \end{array} \right. \end{array}$$

note that

$$\begin{array}{rcl} y_1(x_1 - x_2 - x_3 + 3x_4) & + & \\ y_2(5x_1 + x_2 + 3x_3 + 8x_4) & + & \\ y_3(-x_1 + x_2 + 3x_3 - 5x_4) & \leq & y_1 + 55y_2 + 3y_3 \end{array}$$

We see that maximizing the **Primal objective function** $4x_1+x_2+5x_3+3x_4$ is equivalent to minimize the **Dual objective function** $y_1 + 55y_2 + 3y_3$:

$$\begin{array}{ll} \text{minimize} & y_1 + 55y_2 + 3y_3 \\ \text{subject to} & \left\{ \begin{array}{rcl} y_1 + 5y_2 - y_3 & \geq & 4 \\ -y_1 + y_2 + y_3 & \geq & 1 \\ -y_1 + 3y_2 + 3y_3 & \geq & 5 \\ 3y_1 + 8y_2 - 5y_3 & \geq & 3 \\ y_1, y_2, y_3, y_4 & \geq & 0 \end{array} \right. \end{array}$$

In general, if we can write the LP problem in its **normal formulation**:

$$\left. \begin{array}{ll} \min & \mathbf{c}^t\mathbf{x} \\ \text{subject to} & A\mathbf{x} \geq \mathbf{b}, \forall x_i \geq 0 \end{array} \right\} \text{PRIMAL}$$

$$\left. \begin{array}{ll} \max & \mathbf{b}^t\mathbf{y} \\ \text{subject to} & A^t\mathbf{y} \leq \mathbf{c}, \forall y_i \geq 0 \end{array} \right\} \text{DUAL}$$

The dual problem is a transposition of the primal problem. Since any LP can be written in the standard form above, any LP has a dual.

[1] `https://www.youtube.com/watch?v=yU8updOR87c`

1. if you are asked to minimize $f(x)$, this is equivalent to maximize $-f(x)$;

2. add slack/surplus variables if you want to transform an inequality into an equality contraint;

3. $a \cdot x = b$ is equivalent to having, simultaneously, $a \cdot x \leq b$ and $a \cdot x \geq b$;

4. replace an unconstrained variable x_i by $u_i - v_i$ and impose that $u_i, v_i \geq 0$

Trick: Leave equality constraints to use the simplex algorithm. Use the inequality constraints for the duality theorem to be used.

Exercise 21

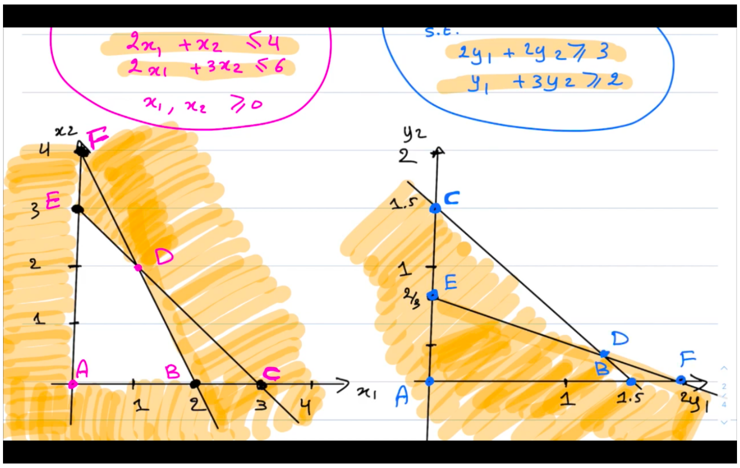
Find the dual problem of

$$\begin{array}{ll} \text{maximize} & 3x_1 + 2x_2 \\ \text{subject to} & \left\{ \begin{array}{rcl} 2x_1 + x_2 & \leq & 4 \\ 2x_1 + 3x_2 & \leq & 6 \\ x_1, x_2 & \geq & 0 \end{array} \right. \end{array}$$

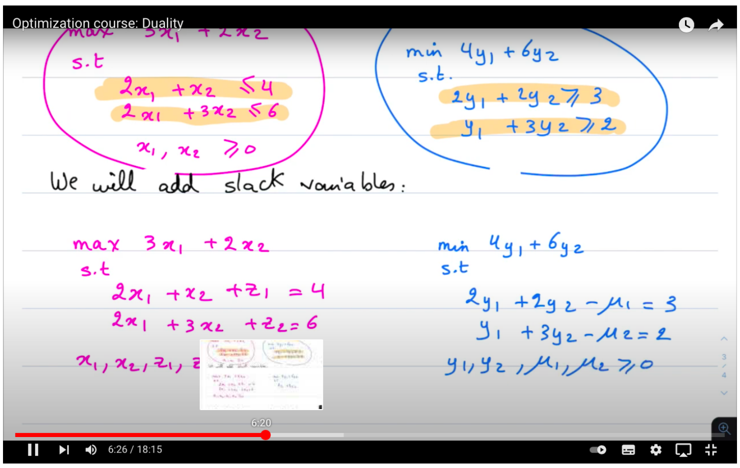
Solve it graphically and using Simplex.

2 3 4 5

2


2x1 + x2 ≤ 4
2x1 + 3x2 ≤ 6
x1, x2 ≥ 0
s.t.
2y1 + 2y2 ≥ 3
y1 + 3y2 ≥ 2
x2
y2
F
E
D
A
B
C
x1
y1

3


Optimization course: Duality
s.t
2x1 + x2 ≤ 4
2x1 + 3x2 ≤ 6
x1, x2 ≥ 0
min 4y1 + 6y2
s.t.
2y1 + 2y2 ≥ 3
y1 + 3y2 ≥ 2
We will add slack variables:
max 3x1 + 2x2
s.t
2x1 + x2 + z1 = 4
2x1 + 3x2 + z2 = 6
min 4y1 + 6y2
s.t
2y1 + 2y2 − μ1 = 3
y1 + 3y2 − μ2 = 2
y1, y2, μ1, μ2 ≥ 0
6:26 / 18:15

4

Exercise 22

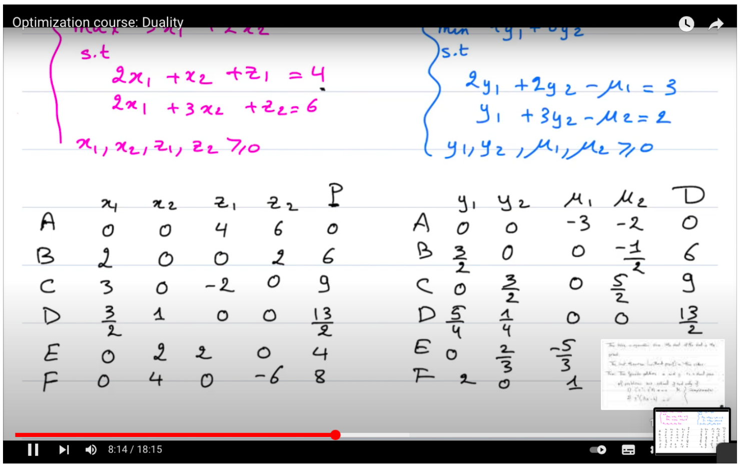
Find the dual problem of

$$\text{minimize} \quad 4x_1 + 4x_2 + x_3$$

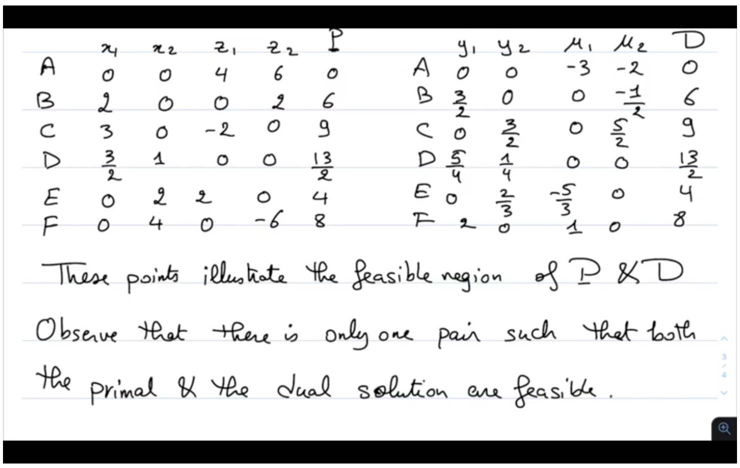
$$\text{subject to} \quad \begin{cases} x_1 + x_2 + x_3 & \leq & 2 \\ 2x_1 + x_2 & = & 3 \\ 2x_1 + x_2 + 3x_3 & \geq & 3 \\ x_1, x_2, x_3 & \geq & 0 \end{cases}$$

Trick: normalize the problem first: eg $2x_1 + x_2 = 3 \rightarrow \begin{cases} 2x_1 + x_2 \geq 3 \\ 2x_1 + x_2 \leq 3 \end{cases}$.

6



5



[6] https://www.youtube.com/watch?v=wVnr1HhUCTO

3.3.1 Some theory

Theorem 2. *In the case of linear programming, the dual of the dual is the primal.*

Proof.

$$\left.\begin{array}{ll} \max & \mathbf{b}^t\mathbf{y} \\ \text{subject to} & A^t\mathbf{y} \leq \mathbf{c}, \forall y_i \geq 0 \end{array}\right\} \text{DUAL}$$

is equivalent to

$$\left.\begin{array}{ll} \min & -\mathbf{b}^t\mathbf{y} \\ \text{subject to} & -A^t\mathbf{y} \geq -\mathbf{c}, \forall y_i \geq 0 \end{array}\right\} \text{DUAL}$$

where, by taking again the Dual, leads to the original Primal. □

Let us retake the example we saw in the last session:

$$\begin{array}{lrcl} \text{maximize} & z = 8x_1 + 5x_2 & & \\ & x_1 & \leq & 150 \\ \text{subject to} & x_2 & \leq & 250 \\ & 2x_1 + x_2 & \leq & 500 \\ & x_1, x_2 & \geq & 0 \end{array}$$

We found out that, from the initial tableau:

		z	x_1	x_2	s	t	u	b
	z	1	-8	-5	0	0	0	0
$\rightarrow$	s	0	1	0	1	0	0	150
	t	0	0	1	0	1	0	250
	u	0	2	1	0	0	1	500
basic			$\uparrow$					

and after applying the different steps of the Simplex method we ended up with the final tableau:

		z	x_1	x_2	s	t	u	b
$R_1 + 5R_4$	z	1	0	0	0	1	4	2250
	x_1	0	1	0	0	$-1/2$	$1/2$	125
$\rightarrow R_3 - R_4$	s	0	0	0	1	$1/2$	$-1/2$	25
	x_2	0	1	0	1	0	0	250
	basic				$\uparrow$			

Now, if we multiply the original availability of each resource (shown in the original tableau) by its marginal worth (taken from the final tableau) and get the sum, we obtain the optimal objective function value:

$$z* = 2250 = 0(150) + 1(250) + 4(500)$$

Primal / Dual	not feasible	unbounded	has a solution
not feasible	possible	possible	no
unbounded	possible	no	no
has a solution	no	no	same values

Theorem 3 (Weak Duality). *In a max LP, the value of primal objective function for any feasible solution is bounded from above by any feasible solution to its dual:*

$$\bar{z} \leq \bar{w}$$

The statement is analogous to a minimization problem.

Theorem 4 (Unboundness property). *If primal (dual) problem has an unbounded solution, then the dual (primal) is unfeasible.*

$$\bar{z} \leq \infty$$

Theorem 5 (Strong Duality). *If the primal problem has an optimal solution,*

$$x^* = (x_1^*, \ldots, x_n^*)$$

then the dual also has an optimal solution,

$$y^* = (y_1^*, \ldots, y_m^*)$$

and

$$z^* := \sum_j c_j x_j^* = \sum_i b_i y_i^* := w^*$$

Thus: if feasible objective function values are found for a primal and dual pair of problems, and if these values are equal to each other, then both of the solutions are optimal solutions.

The Shadow prices that appear at the top of the optimal tableau of the primal problem are precisely the optimal values of the dual variables!

For any two given LP problems: The simplex algorithm solves the primal and dual problems simultaneously.

Exactly one of the following mutually exclusive cases always occurs:

- Both primal and dual problems are feasible, and both have optimal (and equal) solutions.
- Both primal and dual problems are infeasible (have no feasible solution).
- The primal problem is feasible but unbounded, and the dual problem is infeasible.
- The dual problem is feasible but unbounded, and the primal problem is infeasible.

3.3.2 Complementary slackness

Because each decision variable in a primal problem is associated with a constraint in the dual problem, each such variable is also associated with a slack or surplus variable in the dual.

Variables in one problem are complementary to constraints in the other.

In any solution, if the primal variable is basic (with value ≥ 0, hence having slack, hence being non-binding), then the associated dual variable is non-basic ($= 0$, hence having no slack, hence being binding), and viceversa.

Theorem 6 (Complementary slackness)**.** *If in an optimal solution to a LP problem an inequality constraint is not binding, then the dual variable corresponding to that constraint has a value of zero in any optimal solution to the dual problem.*

Let us see how this can be better understood. Let us take a general primal/-dual case

$$\left.\begin{array}{ll} \max & \mathbf{c}^t\mathbf{x} \\ \text{subject to} & A\mathbf{x} \leq \mathbf{b}, \forall x_i \geq 0 \end{array}\right\} \text{PRIMAL}$$

$$\left.\begin{array}{ll} \min & \mathbf{b}^t\mathbf{y} \\ \text{subject to} & A^t\mathbf{y} \geq \mathbf{c}, \forall y_i \geq 0 \end{array}\right\} \text{DUAL}$$

Let vectors x_0, y_0 be feasible solutions for the (max) and the (min) problems, respectively. The weak duality theorem 3 states that (we remove the transpose where it is obvious, for simplicity)

$$\max \quad c \cdot x_0 \leq \min \quad b \cdot y_0$$

Then we can say that (we will not demonstrate it here)

$$b \cdot y_0 - c \cdot x_0 = \underbrace{(b - Ax_0)}_{u} \cdot y_0 + \underbrace{(A^t y_0 - c)}_{v} \cdot x_0$$

where u is the vector of slack variables in the primal problem and v is the vector of the slack variables in the dual.

Accordin to Theorem 5, when $x_0, y_0 = x^*, y^*$ are optimal solutions,

$$b \cdot y^* - c \cdot x^* = 0$$

This means that in such optimal solutions

$$\underbrace{(b - Ax^*)}_{u \geq 0} \cdot y^* + \underbrace{(A^t y^* - c)}_{v \geq 0} \cdot x^* = 0$$

The fact that all terms are nonnegative implies that

$$(b - Ax^*) \cdot y^* = (A^t y^* - c) \cdot x^* = 0 \tag{3.2}$$

Now let us haver a look at one of these terms:

$$(b - Ax^*) \cdot y^* \equiv u \cdot y^* = 0$$

As the components of the two vectors are nonnegative aswell: $u_1, \dots, u_m \geq 0$ and $y_1^*, \dots, y_m^* \geq 0$,

$$\underbrace{u_1 y_1^*}_{\geq 0} + \cdots \underbrace{u_m y_m^*}_{\geq 0} = 0$$

So, each term should be zero!. Now, for each $i = 1, \dots, m$ wemust have at least one of $u_i, y_i^* = 0$. So, if $u_i \neq 0 \Rightarrow y_i^* = 0$ and the other way around. The same occurs to the other term in Eq.3.5.

Thus, Theorem 6 can be reformulated as:

Theorem 7 (Complementary slackness (rewritten))**.** *Suppose we have a primal and dual problems with variables x_0, y_0 for the (max) and (min), respectively. Then, $x_0 \equiv x^*$ and $y_0 \equiv y^*$ if and only if*

$$(b - Ax^*) \cdot y^* = 0 \tag{3.3}$$

and

$$(A^t y^* - c) \cdot x^* = 0 \tag{3.4}$$

In other words:

- if a variable is positive, then the associated dual constraint must be binding; or
- if a constraint fails to bind, than the associated variable must be zero; but recall that
- it is possible for a primal constraint to be binding while the associated dual variable is equal to zero (no slack in both primal and dual).

There cannot be slack in both a constraint and the associated dual variable.

[7]

Let us show an example. Imagine that we have the primal problem

$$\begin{array}{lrcl} \min & 12x_1 + 5x_2 + 10x_3 & & \\ & x_1 - x_2 + 2x_3 & \geq & 10 \\ & -3x_1 + x_2 + 4x_3 & \geq & -9 \\ \text{subject to} & -x_1 + 2x_2 + 3x_3 & \geq & 1 \\ & 2x_1 - 3x_2 & \geq & -2 \\ & 7x_1 - x_2 - 5x_3 & \geq & 34 \\ & x_1, x_2, x_3 & \geq & 0 \end{array}$$

and we are said that the optimal solution is $x^* = (7, 0, 3)$. Find the optimal solution to the dual y^*.

[7] Veure exemple pràctic a `https://youtu.be/o1pznRt_-y0?t=1603`

We build first the dual problem:

$$
\begin{array}{lrcl}
\max & 10y_1 + 9y_2 + y_3 - 2y_4 + 34y_5 & & \\
 & y_1 - 3y_2 - y_3 + 2y_4 + 7y_5 & \leq & 12 \\
\text{subject to} & -y_1 + y_2 + 2y_3 - 3y_4 - y_5 & \leq & 5 \\
 & 2y_1 + 4y_2 + 3y_3 - 5y_5 & \leq & 10 \\
 & y_1, y_2, y_3, y_4, y_5 & \geq & 0
\end{array}
\tag{3.5}
$$

1. Find the slack of (min) by replacing the values in the primal problem constraints:

$$
\begin{array}{rcl}
(7 - 0 + 2 \cdot 3) - 10 & = & 3 \\
(-3 \cdot 7 + 0 + 4 \cdot 3) + 9 & = & 0 \\
(-7 + 2 \cdot 0 + 3 \cdot 3) - 1 & = & 1 \\
(2 \cdot 7 - 3 \cdot 0) + 2 & = & 16 \\
(7 \cdot 7 - 0 - 5 \cdot 3) - 34 & = & 0
\end{array}
$$

2. From Eq.3.3, $\begin{pmatrix} 3 \\ 0 \\ 1 \\ 16 \\ 0 \end{pmatrix} \cdot y^* = 0$, which gives $y_1^* = y_3^* = y_4^* = 0$ and

3. from Eq.3.4, since $x^* = (7, 0, 3)$ I know that the slack of the dual should be zero for its first and third constraints. So, the constraints in the dual (EQ.3.5) end up being:

$$
\begin{array}{rcl}
-3y_2^* + 7y_5^* & = & 12 \\
4y_2^* - 5y_5^* & = & 10
\end{array}
$$

leading to $y^* = (0, 10, 0, 0, 6)$.

Follow up: Verify that $c \cdot x^* = b \cdot y^*$.
[8]

Exercise 23

You are feeding animals in a farm.Your animals have some basic requirements for nutrients, and you want to minimize the cost fulfilling those requirements. You need to know:

1. What foods do I have avaliable and which is each ones cost?

2. What nutrients are needed and which is the nutrients content per food?

Formulate the problem in an abstract way.

[8] veure un exemple ben maco addicional de l'ús de CST a text

Exercise 24

Come back to the previous problem, but now, there is a seller that tells you she has the pills you need to ensure the nutrients, forgetting about specific food. Reformulate the problem in terms of the maximum revenue the seller company wants. What is the price per pill to maximize her benefits, subject to the constraint that the pills are cheaper than the actual food.

Exercise 25

Rationalize the complementarity slackness theorem within the context of the previous exercise. What does it mean to buy a positive quantity of the first food (primal LP problem) in terms of the constraint related to the first pill (dual problem)?
[9]

Exercise 26

Just by checking the feasibility of the primal and dual problem for

$$
\begin{array}{ll}
\text{maximize} & z = x_1 - x_2 \\
\text{subject to} & \begin{array}{rcl} -2x_1 + x_2 & \leq & 2 \\ x_1 - 2x_2 & \leq & 2 \\ x_1 + x_2 & \leq & 5 \\ x_1, x_2 & \geq & 0 \end{array}
\end{array}
$$

find what is the solution.
[10]

[9] Problema formulat a les notes I que tinc a material

[10] extret les notes duality. Comença per trobar els punts de creuament de les restriccions del p‘rimal. Aleshores aplica CS per avaluar què passa al problema dual

Chapter 4

Sensitivity Analysis

4.1 Session 12. Sensitivity Analysis

- Understanding the concept of postoptimality analysis
- Understanding and solving LP problems requiring sensitvity analysis
- Understanding the matrix representation of the Simplex solution
- After an optimal solution is found, the analyst needs to review the problem parameters and the solution.
- This process is called **postoptimality analysis**:
  - confirming or updating problem parameters (cost and availabilities of activities and resources)
  - if changes need to be introduced in the original parameters, assessing their impact on the optimality of the solution.
- If changes are small, re-optimization may not be needed.
- **Sensitivity analysis** is the study of the effect that types, ranges, and magnitude of changes in problem parameters have in the value of the objective function, *without the need to solve again the new linear problem.*

In a LP problem,

$$\begin{aligned} \max \quad & \mathbf{c}^t\mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} \leq \mathbf{b}, \forall x_i \geq 0 \end{aligned}$$

one can have two situations: one may have interest in knowing what happens with modifications in the parameters c or modifications in the parameters b.

In addition, one can explore what occurs when adding an extra constraint or adding a new variable.

Exercise 27

A farmer wants to minimize the cost of the food given to her lifestock. Two different types of nutrients A and B are needed by the animals, and she needs a minimum nutrition to be achieved.

	A	B	price
Feed 1	10	3	16
Feed 2	4	5	14
requirements	124	60	

Find how resilient is she to the changes in the price? What happens with respect to basic vs non-basic variables in a general case?
[1]

In such cases we will explore the coeficients of the objective function in the dual problem, instead.

Exercise 28

Given the LP problem

$$\begin{array}{ll} \text{minimize} & z = 16x_1 + 14x_2 \\ & \begin{array}{rcl} 10x_1 + 4x_2 & \geq & 124 \\ 3x_1 + 5x_2 & \geq & 60 \\ x_1, x_2 & \geq & 0 \end{array} \end{array}$$

subject to

Explore the sensitivity of the minimum value of z with respect to the parameters in the RHS of the contraints. Hint: consider the dual problem.
[2]

Solving the dual problem gives a lot of insight into the actual situation.

Exercise 29

* In a company that manufactures two types of bikes, A and B, the owners want to maximize the benefits, taking into account that the production depends on a restricted amount of titanium, the time the machines can devote to the work and the manpower, which are all given below:

	A	B	limit
Titanium	50	30	2000
Machine time	6	5	300
Labor	3	5	200
price	50	60	

Find *here* a code with the solution using ORtools.

The Sensitivity analysis tool in Excel is nicely explained *here*.

Notice that in a non-linear scenario, what we have identified here as *reduced costs* are in fact *reduced gradient* (or actual gradient of the optimal solution) while *shadow prices* (margin for profit to be obtained by changing 1 unit in each constraint RHS term) correspond to the *Lagrange multipliers*.

[1] veure `https://youtu.be/o1pznRt_-y0?t=3378` minut 50 i següent capítol

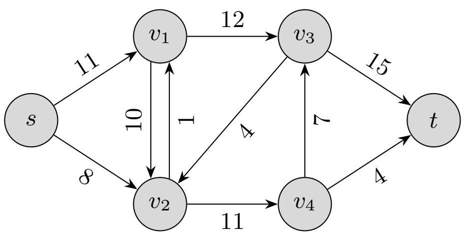
[2] venim de `https://youtu.be/o1pznRt_-y0?t=1603` i `https://www.youtube.com/watch?v=yU8updOR87c`

Chapter 5

Network analysis

5.1 Session 13: Network Analysis

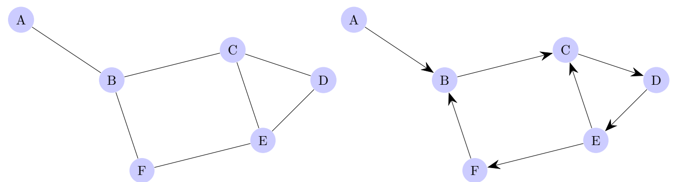
- Getting familiar with the use of network analysis in OR
- Understanding network flow in graphs
- Understanding minimum cost network flow problems
- Applying network connectivity to LP problems
- Solving shortest path problems
- Understanding and applying dynamic programming
- Some of these problems correspond to a physical or geographical network of elements within a system, while others correspond more abstractly to a graphical approach to planning or grouping or arranging the elements of a system.



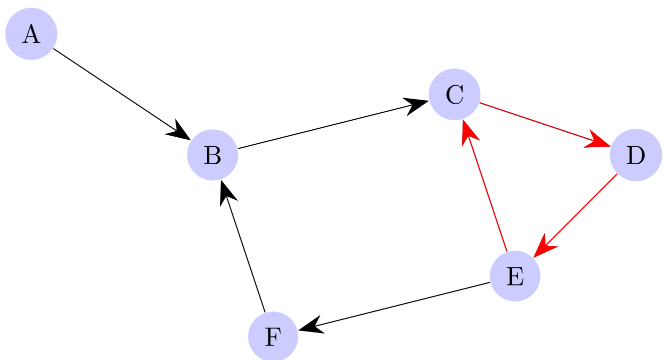
- Systems of highways, railroads, shipping lanes, or aviation patterns, where some supply of a commodity is transported or distributed to satisfy a demand;
- pipeline systems or utility grids can be viewed as fluid flow or power flow networks;
- computer communication networks represent the flow of information;

- an economic system may represent the flow of wealth;
- routing a vehicle or a commodity between certain specified points in the network;
- assigning jobs to machines, or matching workers with jobs for maximum efficiency;
- project planning and project management, where various activities must be scheduled in order to minimize the duration of a project or to meet specified completion dates, subject to the availability of resources;
- and many, many others.

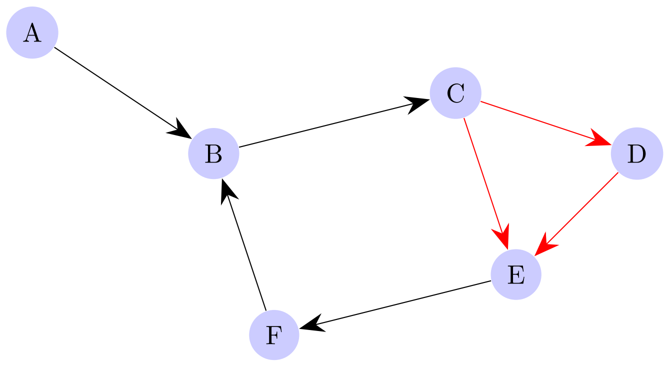
5.1.1 Definitions



- A graph consists of a set of nodes V (vertices, points, or junctions) and a set of connections called arcs A (edges, links, or branches).
- Each connection is associated with a pair of nodes and is usually drawn as a line joining two points. The graph can be defined as *directed* or *undirected*.
- The **degree of a node** is the number of arcs attached to it. An isolated node is of degree zero.
- In a directed graph, or digraph, the arc is often designated by the ordered pair (A, B). In digraphs, the direction of the flow matters.

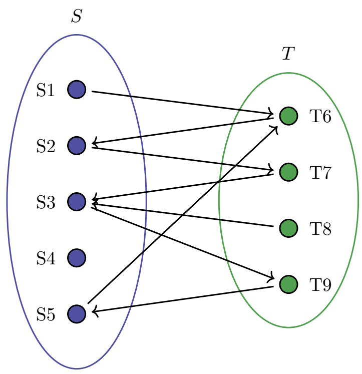


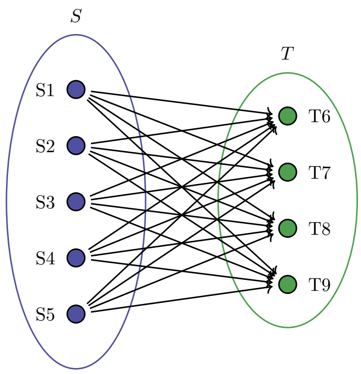
$$A, (A, B), B, (B, C), \underbrace{C, (C, D), D, (D, E), E, (E, C), C}_{\text{cyclic path and chain}}$$



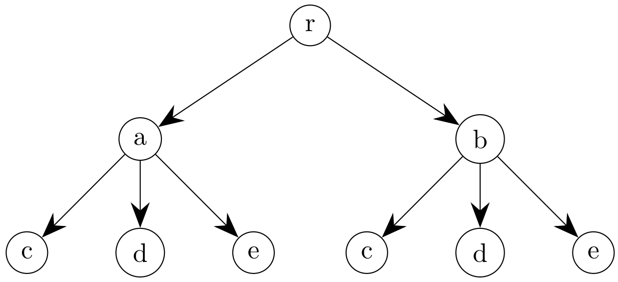
$$A, (A, B), B, (B, C), \underbrace{C, (C, D), D, (D, E), E, (C, E), C}_{\text{cyclic path, non cyclic chain}}$$

- If all the arcs in a path are forward arcs, the path is called **directed chain** or simply **chain**.
- **path** and **chain** are synonimous if the graph is undirected.
- In the second example above we saw a cyclic path but not a cyclic chain, as it included the backward arc (C, E).
- A **connected graph** has at least one path connecting every pair of nodes.
- In a **bipartite graph** the nodes can be partitioned into two subsets S and T, such that each node is in exactly one of the subsets, and every arc in the graph connects a node in set S with a node in set T.
- Such a graph is **complete bipartite** if each node in S is connected to every node in T.





- A **tree** is a directed connected graph in which each node has at most on predecessor, and one node (the root node) has none. In an undirected graph, we have a tree if the graph is connected and contains no cycles.

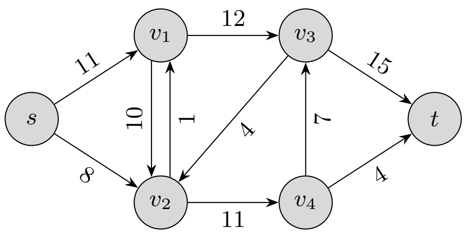


- A **network** is a directed connected graph that is used to model/represent a system/process. The arcs are typically assigned weights representing cost, value or capacity corresponding to each link.
- Nodes in networks can be designated as **sources**, **sinks** or **transshipments**. A **cut set** is any set of arcs which, if removed from the network, would disconnet the sources(s) from the sink(s).
- **Flow** can be thought of as the total amount of an entity that originates at the source, makes it through the different nodes and reaches the sink.

5.1.2 Maximum flow

Determine the maximum possible flow that can be routed through the various network links, from source (s) to sink (t), without violating the capacity constraints.

Important! the commodity is only generated at the source and consumed at the sink.



The **maximum flow problem** can be stated as a LP formulation.

$$\begin{array}{lrll} \text{maximize} & z = f & & \\ & \sum_{i=2}^{n} x_{1i} = f & & \\ \text{subject to} & \sum_{i=1}^{n-1} x_{in} = f & & \\ & \sum_{i=1}^{n} x_{ij} = \sum_{k=1}^{n} x_{jk}, & \text{for} & j = 2, 3, \ldots, n-1 \\ & x_{ij} \leq u_{ij}, & \text{for all} & i, j = 1, 2, \ldots, n \end{array}$$

[1]

[1] Constraints (1) and (2) state that all the flow is generated at the source and consumed at the sink. Constraint (1) ensures that a flow of f leaves the source, and because of conservation of flow, that flow stops only at the sink. Constraints (3) are the flow conservation equations for all the intermediate nodes; nothing is generated or consumed at these nodes. Constraints (4) enforce arc capacity restrictions. All flow amounts xij must be non-negative. Actually, constraint (2) is redundant[2]

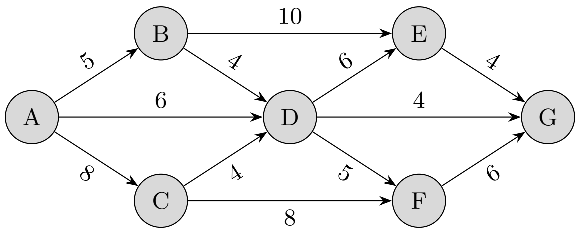
All network problems here can be solved using the Simplex method, but the network structure can help us solving it more efficiently. In the **Ford-Fulkerson labelling algorithm**:

1. Use a labelling procedure to look for a flow augmenting path. If none can be found, stop; the current flow is optimal;

2. Increase the current flow as much as possible in the flow augmenting path, until reaching capacity of some arc. Come back to step 1.

Exercise 30

Find the maximum flow in this network using the Ford-Fulkerson labelling algorithm:



[2]

- In any network, there is always a bottleneck that in some sense impedes the flow through the network.

- The total capacity of the bottleneck is an upper bound on the total flow in the network.

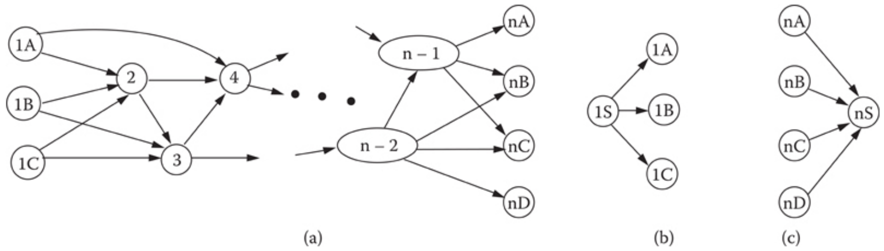
- Cut sets are, by definition, essential in order for there to be a flow from source to sink, since removal of the cut set links would render the sink unreachable from the source.

- The capacities on the links in any cut set potentially limit the total flow.

- The minimum cut (i.e., the cut set with minimum total capacity) is in fact the bottleneck that precisely determines the maximum possible flow in the network (Max-Flow Min-Cut Theorem): the capacity of the cut is precisely equal to the current flow and this flow is optimal. In other words, a saturated cut defines the maximum flow.

[2]Let us assume initially that the flow in all arcs is zero, x ij = 0 and f = 0. In the first iteration, we label nodes A, B, C, D, and G, and discover the flow augmenting path (A, D) and (D, G), across which we can increase the flow by 4. So now, x AD = 4, xDG = 4, and f = 4. In the second iteration, we label nodes A, B, C, then nodes E, D, and F, and finally node G. A flow augmenting path consists of links (A, B), (B, D), (D, E), and (E, G) and flow on this path can be increased by 4. Now x AB = 4, xBD = 4, xDE = 4, x EG = 4, and f = 8. In the third iteration, we see that there remains some unused capacity on link (A, B), so we can label nodes A, B, and E, but not G. It appears we cannot use the full capacity of link (A, B). However, we can also label nodes C, D, F, and G, and augment the flow along the links (A, D), (D, F), and (F, G) by 2, the amount of remaining capacity in (A, D). Now x AD = 6, xDF = 2, x FG = 2, and f = 10. In the fourth iteration, we can label nodes A, B, C, D, F, and G. Along the path from A, C, D, F, to G, we can add a flow of 4, the remaining capacity in (F, G). So x AC = 4, xCF = 4, x FG = 6, and f = 14. In the fifth iteration, we can label all nodes except G. Therefore, there is no flow aug- menting path, and the current flow of 14 is optimal[2]

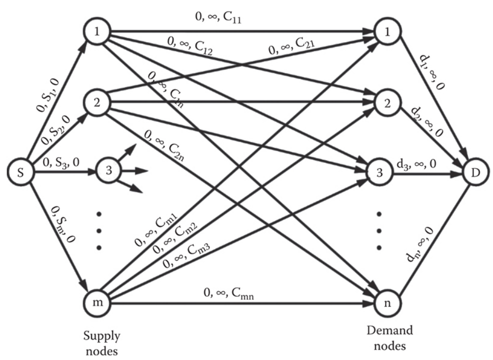
We can generate a supersource or a supersink node with unlimited capacity and repeat the process of optimization as above:[2]



5.1.3 Minimum Cost Network Flow

- Useful when there are costs associated with the flow, given a link capacity.
- Let us assume that every node is a source (supply) and a sink (demand). Imagine a distributor with several warehouses and a group of costumers. Serving each customer from a given warehouse has an associated cost.
- For m supply nodes, each providing s_i suplly, and n demand nodes, each demanding d_j. Assuming that the total demand eaquals the total supply: $\sum_{i=1}^{m} s_i = \sum_{j=1}^{n} d_j$ we aim at satisfying the demand using the available supply minimizing cost routes.

$$
\begin{array}{ll}
\text{minimize} & z = \displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \\
& \sum_{j=1}^{n} x_{ij} = s_i \quad \text{for} \quad i = 1, \ldots, m \\
\text{subject to} & \sum_{i=1}^{m} x_{ij} = d_j \quad \text{for} \quad j = 1, \ldots, n \\
& x_{ij} \geq 0 \quad \text{for all} \quad i, j
\end{array}
$$



Exercise 31

Find the minimum cost in this trasportation problem:[2]

Sources (Warehouses)	Sinks (Customers) 1	2	3	4	5	Supply
1	[28]	[7]	[16]	[2]	[30]	20
2	[18]	[8]	[14]	[4]	[20]	20
3	[10]	[12]	[13]	[5]	[28]	25
Demand	12	14	12	18	9	65

NOTE: The Simplex method says that we should first find any basic feasible solution and then look for a simple pivot to improve the solution. repeat until the optimal solution is found.

1. Finding initial solution.
   - Northwest corner rule
   - Minimum cost method
   - Minimum ”row” cost method
   - Vogel’s method (opportunity cost)
2. Transportation simplex

Once we have any feasible solution, we aim at finding the optimal one. Consider:

Minimum Row Cost Final Solution

Sources (Warehouses)	Sinks (Customers) 1	2	3	4	5	Supply
1	[28]	2 [7]	[16]	18 [2]	[30]	20
2	8 [18]	12 [8]	[14]	[4]	[20]	20
3	4 [10]	[12]	12 [13]	[5]	9 [28]	25
Demand	12	14	12	18	9	65

We can reduce the total cost by reducing the individual costs in every row i by u_i and in every column j by v_j:

$$c'_{ij} = c_{ij} - u_i - v_j$$

Check that, now:

- $\sum_i \sum_j x_{ij} c'_{ij} = 0$
- Check how some costs are now negative in non-basic cells.
- If we increase the number of units in those non-basic cells from 0 to some value, reducing at the same time the number of units in the basic cells, we can reduce the overall cost $\sum_i \sum_j x_{ij} c_{ij}$

In practice:

1. We find first an initial feasible solution as explained above
2. Calculate the u_i and v_j, taking into account that $c_{ij} = u_i + v_j$, for all basic variables (used squares in the table). We start by assigning $u_1 = 0$.
3. We calculate the *improvement index* by $I_{ij} = c_{ij} - u_i - v_j$ for all non-used squares in the table.
4. If all I_{ij} are positive, the solution is already optimal and we are done.
5. If some $I_{ij} < 0$, then build a loop with such value in the corner and alternative $\pm$ signs in all vertex.
6. Use the above $\pm$ to increase the number of units in the position that had $I_{ij} < 0$
7. We return to step 2.

[3]

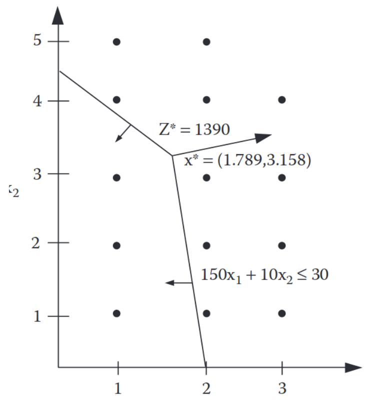
[3]dynamic programming: https://www.youtube.com/watch?v=jqyfhFziDjw https://www.techiedelight.com/word-break-problem/

Chapter 6

Integer Programming

- Getting familiar with the use of integer programming
- Solving integer programming problems
- Problems in which the feasible set is composed of only integer values.
- The feasible set is neither continuous nor feasible.
- NP-hard problems in general.
- Integer problems that have a network structure are easy to solve using the Simplex method (assignment and matching problems, transportation and transshipment problems, and network flow problems always produce integer results, provided that the problem bounds are integers).
- Rounding can be effective in some problems and clearly not in others:
  - not the same tires than aircrafts!
  - values 0/1 for variable: zero-one or binary integer programming (produce or not produce cars in this factory)
  - mixed integer programming problems

$$
\begin{array}{ll}
\text{maximize} & \displaystyle\sum_{j=1}^{n} c_j x_j \\
\text{subject to} & \begin{array}{rll} \sum_{j=1}^{n} a_{ij} x_j = b_i & i = 1, 2, \ldots, m & \\ x_j > 0 & j = 1, 2, \ldots, n & \\ x_j \quad \text{integer} & \text{for some or all} & j = 1, 2, \ldots, n \end{array}
\end{array}
$$



Graphical representation of a typical 2 dimensional integer programming[2].

- Capital budgeting: deciding between a collection of investiments
- Warehouse location: in modelling distribution systems, we should decide about tradeoffs between transportation costs and costs for operating distributions centers
- Scheduling: students-faculty-classrooms allocations, vehicle dispatching, etc (see next slide)

Airline crew scheduling problem: The airlines first design a flight schedule composed of a large number of *flight legs* (specific flight on a specific piece of equipment, such as a 747 from New York to Chicago departing at 6:27 a.m.). A flight crew is a complete set of people, including pilots, navigator, and flight attendants who are trained for a specific airplane. A work schedule or rotation is a collection of flight legs that are feasible for a flight crew, and that normally terminate at the point of origin. Variables x_{ij} have value 1 if flight leg i is assigned to crew j. All flight legs should be covered at minimum total cost.

Also called a *set-partitioning problem*

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{j=1}^{n} c_j x_j \\ \text{subject to} & \begin{array}{rl} \sum_{j=1}^{n} a_{ij} x_j = 1 & i = 1, 2, \ldots, m \\ x_j = 0, 1 & j = 1, 2, \ldots, n \end{array} \end{array}$$

Exercise 32

Consider the travelling salesman problem. Starting from his home, a salesman wishes to visit each of $(n-1)$ other cities and return home at minimal cost. He must visit each city exactly once and the cost to travel from city i to j is c_{ij}. Let x_{ij} be 1 or 0 depending on the fact that he goes or not from city i to city j

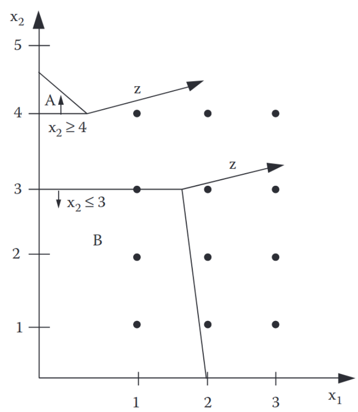
- formulate the optimization problem

- how to avoid disjoint tours?

Simplex is not useful to solve, in general, integer problems. Instead, many other techniques have been proposed:

- Enumeration techniques, including the branch-and-bound procedure;
- cutting plane techniques; and
- group-theoretic techniques,

as well as several composite techniques



Separation into two subproblems in the Branch-and-Bound method[2].

Exercise 33

Using the graphical representation and the branch-and-bound procedure, solve this integer program:

$$
\begin{aligned}
\text{maximize} \quad & z = x_1 + 5x_2 \\
\text{subject to} \quad & -4x_1 + 3x_2 \leq 6 \\
& 3x_1 + 2x_2 \leq 18 \\
& x_1, x_2 \geq 0 \quad \text{and integer}
\end{aligned}
$$

Chapter 7

Stochastic processes

7.0.1 Markov chains

Exercise 34

Here is a simple game:

- a player bets on coin tosses, a dollar each time,
- the game ends either when the player has no money left or is up to five dollars.

If the player starts with three dollars, what is the chance that the game takes at least five flips? Twenty-five flips?

At any point, this player has either \$0, or \$1, ..., or \$5. We say that the player is in the state s_0, s_1, ..., or s_5. A game consists of moving from state to state. For instance, a player now in state s_3 has on the next flip a .5 chance of moving to state s_2 and a .5 chance of moving to s_4. The boundary states are a bit different; once in state s_0 or state s_5, the player never leaves.

Let $p_i(n)$ be the probability that the player is in state s_i after n flips. Then, for instance, we have that the probability of being in state s_0 after flip $n+1$ is $p_0(n+1) = p_0(n) + 0.5 \cdot p_1(n)$. This matrix equation sumarizes.

$$\begin{pmatrix} 1 & .5 & 0 & 0 & 0 & 0 \\ 0 & 0 & .5 & 0 & 0 & 0 \\ 0 & .5 & 0 & .5 & 0 & 0 \\ 0 & 0 & .5 & 0 & .5 & 0 \\ 0 & 0 & 0 & .5 & 0 & 0 \\ 0 & 0 & 0 & 0 & .5 & 1 \end{pmatrix} \begin{pmatrix} p_0(n) \\ p_1(n) \\ p_2(n) \\ p_3(n) \\ p_4(n) \\ p_5(n) \end{pmatrix} = \begin{pmatrix} p_0(n+1) \\ p_1(n+1) \\ p_2(n+1) \\ p_3(n+1) \\ p_4(n+1) \\ p_5(n+1) \end{pmatrix}$$

With the initial condition that the player starts with three dollars, calculation gives this.

$$
\begin{array}{c|cccccc}
n=0 & n=1 & n=2 & n=3 & n=4 & \cdots & n=24 \\
\hline
\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} &
\begin{pmatrix} 0 \\ 0 \\ .5 \\ 0 \\ .5 \\ 0 \end{pmatrix} &
\begin{pmatrix} 0 \\ .25 \\ 0 \\ .5 \\ 0 \\ .25 \end{pmatrix} &
\begin{pmatrix} .125 \\ 0 \\ .375 \\ 0 \\ .25 \\ .25 \end{pmatrix} &
\begin{pmatrix} .125 \\ .1875 \\ 0 \\ .3125 \\ 0 \\ .375 \end{pmatrix} &
\cdots &
\begin{pmatrix} .39600 \\ .00276 \\ 0 \\ .00447 \\ 0 \\ .59676 \end{pmatrix}
\end{array}
$$

- The player quickly ends in either state s_0 or state s_5. For instance, after the fourth flip there is a probability of 0.50 that the game is already over: The boundary states are said to be **absorbtive**.

- *The Markov chain model is historyless*: with a fixed transition matrix, the next state depends only on the current state, not on any prior states. More formally:

$$
\begin{aligned}
P(x_{t+1} = s_{t+1} | x_t = s_t, x_{t-1} = s_{t-1}, ..., x_1 = s_1, x_0 = s_0) = \\
P(x_{t+1} = s_{t+1} | x_t = s_t)
\end{aligned}
$$

- In addition, it is clear that the probability of a transition from state i to state j is independent of time (**stationarity property**):

$$
P(x_{t+1} = j | x_t = i) = p_{ij}
$$

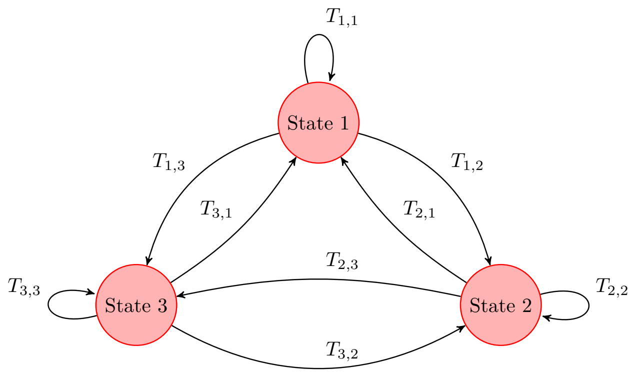
- Each column vector is a probability vector (and it sums up 1) and the matrix is a transition matrix.

Exercise 35

A study ([4], p. 202) divided occupations in the United Kingdom into upper level (executives and professionals), middle level (supervisors and skilled manual workers), and lower level (unskilled). To determine the mobility across these levels in a generation, about two thousand men were asked, “At which level are you, and at which level was your father when you were fourteen years old?”

This equation and transition diagram summarizes the results.

$$
\begin{pmatrix} T_{1,1} & T_{1,2} & T_{1,3} \\ T_{2,1} & T_{2,2} & T_{2,3} \\ T_{3,1} & T_{3,2} & T_{3,3} \end{pmatrix}^t \rightarrow \begin{pmatrix} .60 & .29 & .16 \\ .26 & .37 & .27 \\ .14 & .34 & .57 \end{pmatrix} \begin{pmatrix} p_U(n) \\ p_M(n) \\ p_L(n) \end{pmatrix} = \begin{pmatrix} p_U(n+1) \\ p_M(n+1) \\ p_L(n+1) \end{pmatrix}
$$



For instance, a child of a lower class worker has a .27 probability of growing up to be middle class. Notice that the Markov model assumption about history seems reasonable– we expect that while a parent's occupation has a direct influence on the occupation of the child, the grandparent's occupation has no such direct influence. With the initial distribution of the respondents's fathers given below, this table lists the distributions for the next five generations.

$n=0$	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$
$\begin{pmatrix} .12 \\ .32 \\ .56 \end{pmatrix}$	$\begin{pmatrix} .23 \\ .34 \\ .42 \end{pmatrix}$	$\begin{pmatrix} .29 \\ .34 \\ .37 \end{pmatrix}$	$\begin{pmatrix} .31 \\ .34 \\ .35 \end{pmatrix}$	$\begin{pmatrix} .32 \\ .33 \\ .34 \end{pmatrix}$	$\begin{pmatrix} .33 \\ .33 \\ .34 \end{pmatrix}$

Exercise 36

The World Series of American baseball is played between the team winning the American League and the team winning the National League. The series is won by the first team to win four games. That means that a series is in one of twenty-four states: 0-0 (no games won yet by either team), 1-0 (one game won for the American League team and no games for the National League team), etc. If we assume that there is a probability p that the American League team wins each game then we have the following transition matrix.

$$\begin{pmatrix} 0 & 0 & 0 & 0 & \ldots \\ p & 0 & 0 & 0 & \ldots \\ 1-p & 0 & 0 & 0 & \ldots \\ 0 & p & 0 & 0 & \ldots \\ 0 & 1-p & p & 0 & \ldots \\ 0 & 0 & 1-p & 0 & \ldots \\ \vdots & \vdots & \vdots & \vdots & \end{pmatrix} \begin{pmatrix} p_{\text{0-0}}(n) \\ p_{\text{1-0}}(n) \\ p_{\text{0-1}}(n) \\ p_{\text{2-0}}(n) \\ p_{\text{1-1}}(n) \\ p_{\text{0-2}}(n) \\ \vdots \end{pmatrix} = \begin{pmatrix} p_{\text{0-0}}(n+1) \\ p_{\text{1-0}}(n+1) \\ p_{\text{0-1}}(n+1) \\ p_{\text{2-0}}(n+1) \\ p_{\text{1-1}}(n+1) \\ p_{\text{0-2}}(n+1) \\ \vdots \end{pmatrix}$$

An especially interesting special case is $p = 0.50$; this table lists the resulting components of the $n = 0$ through $n = 7$ vectors. Check what happens with other values of p.

	$n=0$	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$
$0-0$	1	0	0	0	0	0	0	0
$1-0$	0	0.5	0	0	0	0	0	0
$0-1$	0	0.5	0	0	0	0	0	0
$2-0$	0	0	0.25	0	0	0	0	0
$1-1$	0	0	0.5	0	0	0	0	0
$0-2$	0	0	0.25	0	0	0	0	0
$3-0$	0	0	0	0.125	0	0	0	0
$2-1$	0	0	0	0.375	0	0	0	0
$1-2$	0	0	0	0.375	0	0	0	0
$0-3$	0	0	0	0.125	0	0	0	0
$4-0$	0	0	0	0	0.0625	0.0625	0.0625	0.062
$3-1$	0	0	0	0	0.25	0	0	0
$2-2$	0	0	0	0	0.375	0	0	0
$1-3$	0	0	0	0	0.25	0	0	0
$0-4$	0	0	0	0	0.0625	0.0625	0.0625	0.062
$4-1$	0	0	0	0	0	0.125	0.125	0.125
$3-2$	0	0	0	0	0	0.3125	0	0
$2-3$	0	0	0	0	0	0.3125	0	0
$1-4$	0	0	0	0	0	0.125	0.125	0.125
$4-2$	0	0	0	0	0	0	0.15625	0.156
$3-3$	0	0	0	0	0	0	0.3125	0
$2-4$	0	0	0	0	0	0	0.15625	0.156
$4-3$	0	0	0	0	0	0	0	0.156
$3-4$	0	0	0	0	0	0	0	0.156

Evenly-matched teams are likely to have a long series– there is a probability of 0.625 that the series goes at least six games.

In summary, a system can be modeled as a Markov process if it has the following four properties:

- Property 1: A finite number of states can be used to describe the dynamic behavior of the system.

- Property 2: Initial probabilities are specified for the system.

- Property 3: Markov propertyWe assume that a transition to a new state depends only on the current state and not on past conditions.

- Property 4: Stationarity propertyThe probability of a transition between any two states does not vary in time.

If Markov analysis is appropriate for the system being studiedone can try answering questions like:

- How many transitions (steps) will it likely take for the system to move from some specified state to another specified state?

- What is the probability that it will take some given number of steps to go from one specified state to another?

- In the long run, which state is occupied by the system most frequently?

- Over a long period of time, what fraction of the time does the system occupy each of the possible states?

- Will the system continue indefinitely to move among all N states, or will it eventually settle into a certain few states?

[1]

7.0.2 Hidden Markov Models

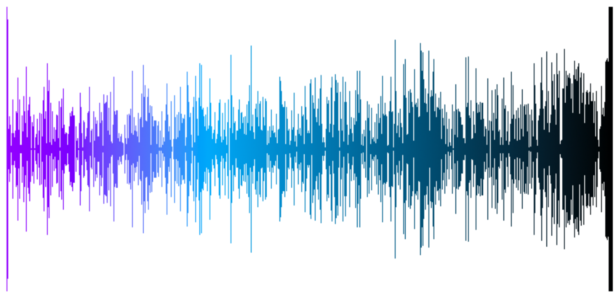
HMMs are probably the most popular directed graphical model out there. They are used in many sequential and temporal domains:

- speech recognition,
- handwriting recognition,
- visual target tracking and localization,
- machine translation,
- robot localization,
- gene prediction,
- time-series analysis,
- natural language processing and part-of-speech recognition,
- stochastic control,
- protein folding, ...

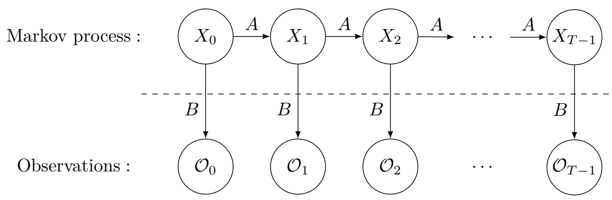
In HMMs, the random variables are divided into hidden states (phonemes, letters, target location) and observations (audio signal, pen strokes, target image). The goal is to predict the states from observations.

What is this?

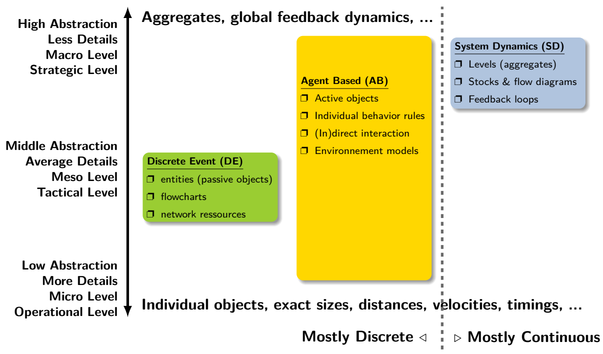
[1] dynamic programming: https://www.youtube.com/watch?v=jqyfhFziDjw https://www.techiedelight.com/word-break-problem/



We can start wth the graphical representation of the HMM



7.0.3 Discrete Evenet Simulations (DES)



Taken from `https://texample.net/tikz/examples/simulation-abstraction/`

Bibliography

[1] Kenneth J. Beers. *Numerical methods for chemical engineering: applications in Matlab*. Cambridge University Press, Cambridge, UK ; New York, 2007. OCLC: ocm71238870.

[2] Michael Carter, Camille C Price, and Ghaith Rabadi. *Operations Research. A Practical Introduction. Second Edition*. CRC Press, 2019.

[3] David Harel and Yishai A. Feldman. *Algorithmics: The Spirit of Computing*. Pearson Education, 2004. Google-Books-ID: txxLovFWkCUC.

[4] Kenneth Macdonald and John Ridge. Social Mobility. In A. H. Halsey, editor, *British Social Trends since 1900: A Guide to the Changing Social Structure of Britain*, pages 202–225. Palgrave Macmillan UK, London, 1988.

[5] Jorge Nocedal and Stephen J. Wright. *Numerical optimization*. Springer series in operations research. Springer, New York, 2nd ed edition, 2006. OCLC: ocm68629100.

[6] Ronald L. Rardin. *Optimization in operations research*. Pearson, Boston, second edition edition, 2017.