LIÈGE université
Sciences Appliquées

Densys immersive week

Introduction to optimization – linear programming

Bertrand Cornélusse

March 29, 2023

Overview

1. Mathematical programming

2. Linear programming

3. Network flow model

Lecture overview

- The goal of this lecture is to introduce the concept of mathematical programming, particularly linear programming.
- We also quickly review solution methods for these problems.
- These tools can then be used for optimal power flow, real-time dispatch, operational planning, or sizing of a microgrid, for instance.

The lecture is based on slides from Pr. Quentin Louveaux at University of Liège.

Mathematical programming

Mathematical programming

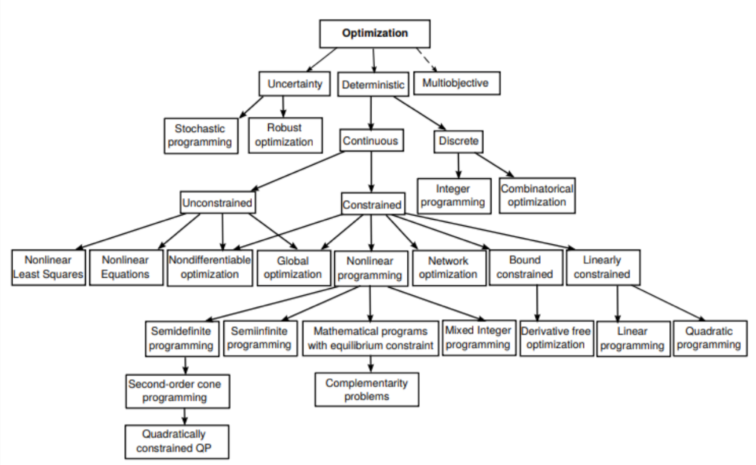
Mathematical programming is a field of applied mathematics that deals with solving optimization problems.

More precisely, it provides a framework and solution methods for computing the **decisions** of an optimization problem, given an **objective function** to minimize or maximize, and (optionally) **constraints** on the decision variables.

Mathematical programming relies on a model of the problem to solve.

There is a great variety of mathematical programming problem types, depending on the characteristics of the objective function and constraints and of the restrictions that apply to variables.

Categories of mathematical programs



https://neos-guide.org/content/optimization-taxonomy

Categories of mathematical programs

General mathematical program

A general mathematical program can be stated as follows:

$$\begin{aligned} \min\ & f(x) \\ s.t.\ & g(x) \leq 0 \\ & Ax = 0 \\ & x \in X \end{aligned}$$

It is very hard to solve, especially when

- objective and constraints are non-linear or even worse non-convex
- variables are discrete

Linear program

$$\begin{aligned} \min\ & c^T x \\ s.t.\ & Ax = b \\ & x \in \mathbb{R}_+^n \end{aligned}$$

Easy to solve even for large problems.

Mixed-Integer Linear program

$$\begin{aligned} \min\ & c^T x \\ s.t.\ & Ax = b \\ & x \in \mathbb{R}_+^{n_1} * \mathbb{Z}_+^{n_2} \end{aligned}$$

Hard problem, but feasible for moderately sized instances.

Linear programming

Linear programming I

If the objective is **linear** and the constraints are **linear**, we talk about **linear programming** (LP) or **linear optimization**.

LP in standard form

$$\begin{aligned} \min\ & c^T x \\ \text{s.t. } & Ax = b \\ & x \in \mathbb{R}^n_+ \end{aligned}$$

Linear programming II

Definition

A **polyhedron** is a set $\{x \in \mathbb{R}^n | Ax \geq b\}$

A set of the form $Ax \leq b$ is also a polyhedron.

A set $\{x \in \mathbb{R}^n | Ax = b, x \geq 0\}$ is a polyhedron in **standard form**.

Graphic representation

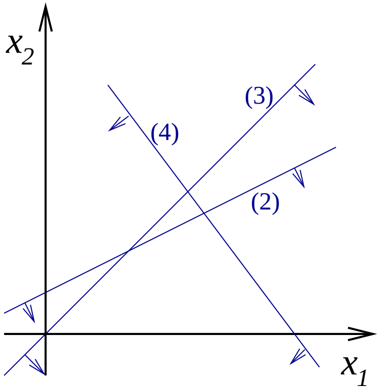
We can represent a problem in two dimensions graphically:

$$
\begin{aligned}
\max \ x_1 &+ 2x_2 && (1)\\
-x_1 &+ 2x_2 \leq 1 && (2)\\
-x_1 &+ \ \ x_2 \leq 0 && (3)\\
4x_1 &+ 3x_2 \leq 12 && (4)\\
x_1, & \quad x_2 \geq 0 && (5)
\end{aligned}
$$

Graphic representation

We can represent a problem in two dimensions graphically:

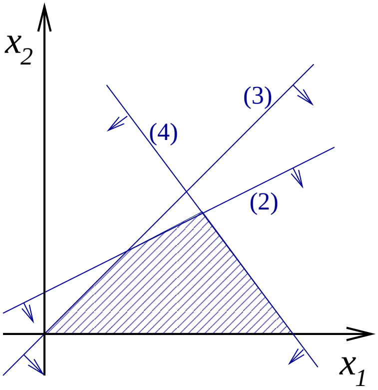
$$
\begin{aligned}
\max\ & x_1 + 2x_2 && (1)\\
& -x_1 + 2x_2 \leq 1 && (2)\\
& -x_1 + x_2 \leq 0 && (3)\\
& 4x_1 + 3x_2 \leq 12 && (4)\\
& x_1,\ \ x_2 \geq 0 && (5)
\end{aligned}
$$



Graphic representation

We can represent a problem in two dimensions graphically:

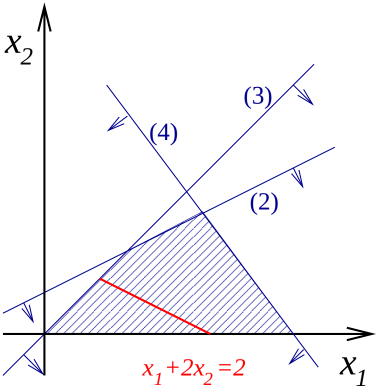
$$
\begin{aligned}
\max\ & x_1 + 2x_2 && (1)\\
-x_1 + 2x_2 &\le 1 && (2)\\
-x_1 + x_2 &\le 0 && (3)\\
4x_1 + 3x_2 &\le 12 && (4)\\
x_1,\quad x_2 &\ge 0 && (5)
\end{aligned}
$$



Graphic representation

We can represent a problem in two dimensions graphically:

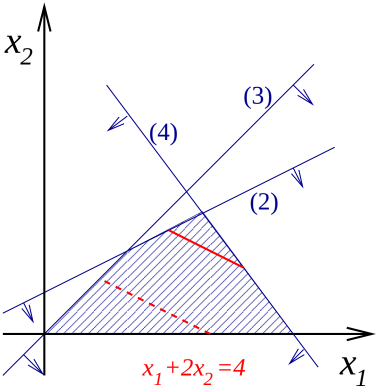
$$
\begin{array}{rlr}
\max & x_1 + 2x_2 & (1) \\
 & -x_1 + 2x_2 \le 1 & (2) \\
 & -x_1 + x_2 \le 0 & (3) \\
 & 4x_1 + 3x_2 \le 12 & (4) \\
 & x_1, \quad x_2 \ge 0 & (5)
\end{array}
$$



Graphic representation

We can represent a problem in two dimensions graphically:

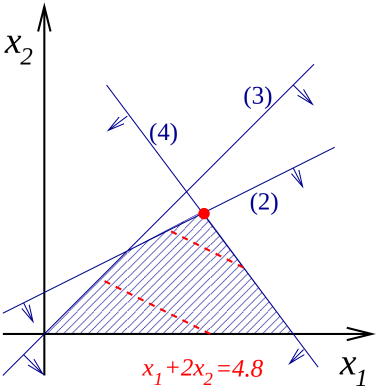
$$
\begin{aligned}
\max\ x_1 + 2x_2 & & (1)\\
-x_1 + 2x_2 &\leq 1 & (2)\\
-x_1 + x_2 &\leq 0 & (3)\\
4x_1 + 3x_2 &\leq 12 & (4)\\
x_1,\quad x_2 &\geq 0 & (5)
\end{aligned}
$$



Graphic representation

We can represent a problem in two dimensions graphically:

$$
\begin{aligned}
\max\ x_1 + 2x_2 & & (1)\\
-x_1 + 2x_2 \le 1 & & (2)\\
-x_1 + x_2 \le 0 & & (3)\\
4x_1 + 3x_2 \le 12 & & (4)\\
x_1,\quad x_2 \ge 0 & & (5)
\end{aligned}
$$



Extreme points and vertices

Definition

Let P be a polyhedron. A point $x \in P$ is an **extreme point** of P if there do not exist two points $y, z \in P$ such that x is a convex combination of y and z.

Definition

Let P be a polyhedron. A point $x \in P$ is a **vertex** of P if there exists $c \in \mathbb{R}^n$ such that $c^T x < c^T y$ for all $y \in P$ and $y \neq x$.

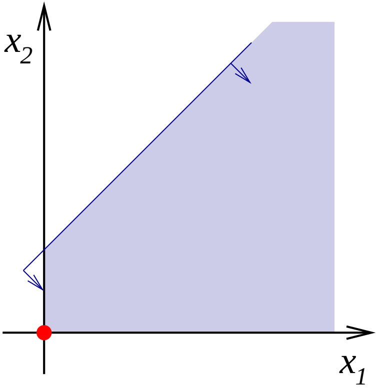
Degenerate cases

In the example we had a **unique solution** at a **vertex** of the **polyhedron**.
Some degenerate cases can lead to different solutions.

Degenerate cases

In the example we had a **unique solution** at a **vertex** of the **polyhedron**.
Some degenerate cases can lead to different solutions.

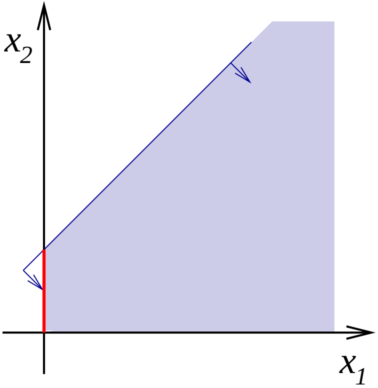
$$\begin{aligned} \min\ & x_1 + x_2 \\ \text{s.t.}\ & -x_1 + x_2 \leq 1 \\ & x_1, x_2 \geq 0 \end{aligned}$$



Degenerate cases

In the example we had a **unique solution** at a **vertex** of the **polyhedron**.
Some degenerate cases can lead to different solutions.

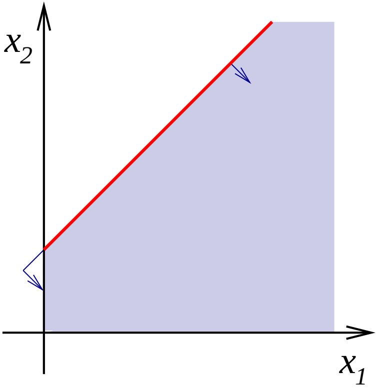
$$\min x_1$$
$$\text{s.t.} \; -x_1 + x_2 \leq 1$$
$$x_1, x_2 \geq 0$$



Degenerate cases

In the example we had a **unique solution** at a **vertex** of the **polyhedron**.
Some degenerate cases can lead to different solutions.

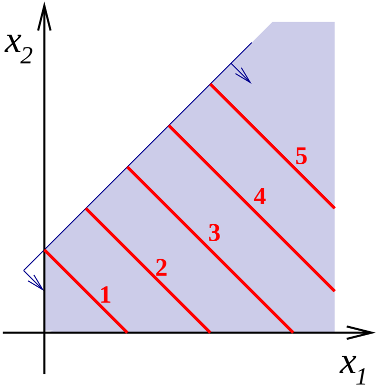
$$\begin{aligned} \max \quad & -x_1 + x_2 \\ \text{s.t.} \quad & -x_1 + x_2 \leq 1 \\ & x_1, x_2 \geq 0 \end{aligned}$$



Degenerate cases

In the example we had a **unique solution** at a **vertex** of the **polyhedron**.
Some degenerate cases can lead to different solutions.

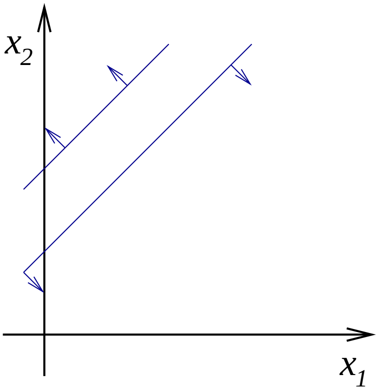
$$\max x_1 + x_2$$
$$\text{s.t.} \quad -x_1 + x_2 \leq 1$$
$$x_1, x_2 \geq 0$$



Degenerate cases

In the example we had a **unique solution** at a **vertex** of the **polyhedron**.
Some degenerate cases can lead to different solutions.

$$\begin{aligned} \max\ & x_1 + 2x_2 \\ \text{s.t.}\ & -x_1 + x_2 \leq 1 \\ & -x_1 + x_2 \geq 2 \\ & x_1, x_2 \geq 0 \end{aligned}$$



Bases of a polyhedron I

We subdivide the equalities and inequalities into three categories:

$$
\begin{aligned}
a_i^T x &\geq b_i \qquad & i &\in M_{\geq} \\
a_i^T x &\leq b_i \qquad & i &\in M_{\leq} \\
a_i^T x &= b_i \qquad & i &\in M_{=}
\end{aligned}
$$

Definition

Let $\bar{x}$ be a point satisfying $a_i^T \bar{x} = b_i$ for some $i \in M_{\geq}, M_{\leq}$ or $M_{=}$. The constraint i is said to be **active** or **tight**.

Bases of a polyhedron II

Definition

Let P be a polyhedron and let $\bar{x} \in \mathbb{R}^n$.

(a) $\bar{x}$ is a **basic solution** if

- all equalities ($i \in M_=$) are **active**
- among the active constraints, there are n **linearly independent**

(b) if $\bar{x}$ is a basic solution **that satisfies all constraints**, then $\bar{x}$ is a **feasible basic solution**.

Bases of a polyhedron III

Theorem

Let P be a polyhedron and let $\bar{x} \in P$. The three following statements are equivalent.

(i) $\bar{x}$ is a **vertex**

(ii) $\bar{x}$ is an **extreme point**

(iii) $\bar{x}$ is a **basic feasible solution**

Linear programming algorithms I

There are two main types of algorithms used in practice.

Simplex methods

- moves from one vertex (extreme point) of the feasible domain to another until the objective stops decreasing
- very efficient in practice but can be exponential on some special problems
- can keep information of one solution to quickly compute a solution to a perturbed problem (useful in a B&B setting), dual simplex, ...

Linear programming algorithms II

Interior point methods

- iteratively penalizes the objective with a function of constraints to force successive points to lie within the feasible domain
- polynomial time, very efficient, especially for large sparse systems
- but no extremal solution hence crossover required in a B&B setting

More advanced topics

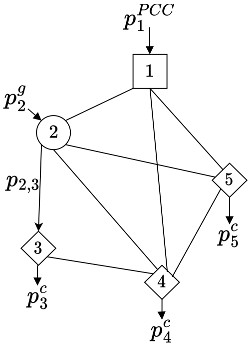
- Duality
- Shadow prices
- Complementary slackness
- Sensitivity analyses
- ...

Network flow model

Hands on session

It is now time to practice these concepts. We will first consider a very naive representation of a microgrid and its distribution network. We assume the distribution network is a graph containing

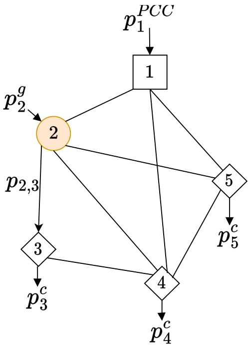
- source nodes that can inject power
  - one source node models the connection to the public grid; we call it the PCC (point of common coupling)
  - other source nodes are distributed generators
- sink nodes that always consume power
- edges that can transmit power from one node to another.



A first basic generator model

A generator is attached to a node u and can output a power $p_u^g \geq 0$ $[MW]$ limited by a maximum power $\bar{P}_u^g$ $[MW]$. The associated production cost $[EUR/h]$ is

$$c_{u,0}^g + c_{u,1}^g p_u^g.$$

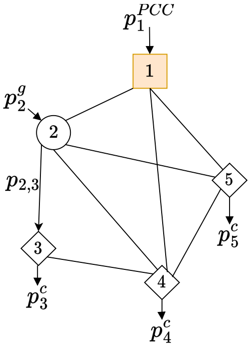


Point of common coupling I

The point of common coupling can be seen as a special generator that can either inject or consume power. Let

$$p^{PCC} = p^{PCC,+} - p^{PCC,-} \quad [MW]$$

be the power injected by the PCC in the microgrid, with $p^{PCC,+} \geq 0$ and $p^{PCC,-} >= 0$. When it consumes power ($p^{PCC} \leq 0$), it means that the power generated in the microgrid exceeds the consumption and is thus pushed into the public grid.



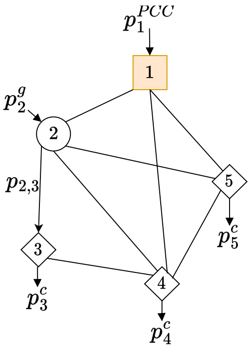
Point of common coupling II

The power exchanged with the public grid is limited, either physically or contractually, to $\bar{P}^{PCC}$:

$$-\bar{P}^{PCC} \leq p^{PCC} \leq \bar{P}^{PCC} \quad [MW].$$

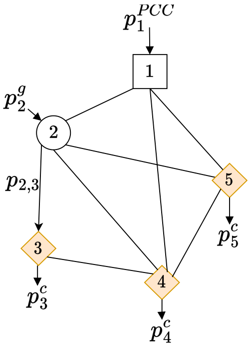
There is a cost $c^{PCC,+}$ $[EUR/MWh]$ associated with energy bought from the public grid and a revenue $c^{PCC,-}$ $[EUR/MWh]$ associated with energy injected into the public grid. We have

$$c^{PCC,+} > c^{PCC,-}.$$



Consumption nodes

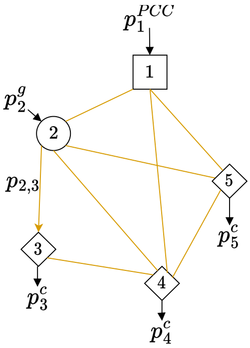
A load is attached to a consumption node u and consumes a power $p_u^c \geq 0$ that cannot be modified.



Edges

An edge (u, v) allows sending power $p_{u,v}$ from node u to node v. However, it has a maximum capacity $\bar{P}_{u,v}$ $[MW]$:

$$-\bar{P}_{u,v} \leq p_{u,v} \leq \bar{P}_{u,v}.$$



Objective

We aim to minimize the total cost of satisfying the demand while satisfying the constraints of the generators, PCC, edges, and the power balance at each node.

Formulate this problem as a linear program and solve it!

- A **template Google Colab is available here**
- It uses
  - Python as a general programming language
  - Pyomo as mathematical programming modeling library for Python
  - Ipopt as a solver, which receives the problem from Pyomo and returns a solution, if any.
- More instruction in the Colab template.