Operations Research Lab

Continuous optimization models
for Management Engineering

Lecture notes with models, fully worked numerical examples, case studies, complete Python/Gurobi implementations and sensitivity analysis. Linear, quadratic and nonlinear programming; decisions under uncertainty (newsvendor, VaR and CVaR); optimization for machine learning (SVM, robust and quantile regression).

Fabio Furini

sites.google.com/view/fabiofurini

Department of Computer, Control and Management Engineering — Sapienza University of Rome

Online version and downloadable material:
fabiofurini.github.io/operations-research-lab/
github.com/fabiofurini/operations-research-lab

Academic year 2026–2027

Contents

Part I

Tools

Chapter 1

Introduction to the laboratory

1.1 What we shall do

This laboratory teaches how to use mathematical programming as a tool for *decision making*, not merely for computation. Every chapter starts from a concrete managerial problem — how much to produce, where to locate a service, which price to set, how much risk to accept — and turns it into an optimization model that is then implemented in Python, solved with a solver and, above all, *interrogated*: how much is one extra hour of capacity worth? Does the solution hold up if the data change by 5%? Which trade-off would we recommend to a decision maker?

The right question

At the end of every laboratory session the question is not only *"what is the optimum?"*, but *"which decision do we recommend and how robust is it?"*. The value of a model lies in the quality of the recommendation it allows us to formulate.

1.2 The three classes of models and the acronyms used

Throughout these lecture notes the models belong to three classes, denoted by the standard international acronyms:

- **LP** (*Linear Programming*): the objective and the constraints are linear functions of the variables;
- **QP** (*Quadratic Programming*): quadratic objective, linear constraints;
- **NLP** (*Nonlinear Programming*): general nonlinear objective or constraints.

A problem is *convex* when every local minimum is also global (precise definition in chapter 2): for LP this is always true, for QP and NLP it depends on the functions, and we shall point it out every time. The other recurring acronyms are defined at first use in the respective chapter.

1.3 Notation

Throughout these lecture notes the same convention applies:

- **scalars and indices**: lowercase italic letters (x_{it}, b_t, λ, i, t);

- **sets of objects indexed by numbers**: the objects of a model (products, channels, securities, vehicles, districts, scenarios) are numbered and the indices run over explicitly enumerated sets, $i \in \{1, 2, \dots, n\}$ — so that the notation of the text is already that of the model being built; *counts* are positive integers ($n \in \mathbb{Z}_{\geq 1}$);
- **numerical data**: rational values, $\mathbb{Q}_{>0}$ or $\mathbb{Q}_{\geq 0}$ (the data of a real instance are always machine numbers);
- **vectors**: lowercase **boldface** letters ($\boldsymbol{x}$, $\boldsymbol{w}$, $\boldsymbol{\mu}$); all vectors are column vectors and $\boldsymbol{a}'\boldsymbol{b}$ denotes the scalar product;
- **matrices**: uppercase **boldface** letters ($\boldsymbol{Q}$, $\boldsymbol{A}$); $\boldsymbol{Q} \succeq 0$ means positive semidefinite;
- **graphs**: $G = (N, A)$ with nodes N and arcs A (the only sets that are not enumerated, by their very nature);
- **dual variables**: π_i, one per constraint; **reduced costs**: $\bar{c}_j$; **slacks**: $\bar{s}_i$;
- **solutions**: unknown variables are written plain (x_j, π_i); the *bar* denotes the values of a *feasible* solution ($\bar{x}_j$, $\bar{\pi}_i$, $\bar{s}_i$, $\bar{c}_j$), the *tilde* those of an *optimal* one ($\tilde{x}_j$, $\tilde{\pi}_i$, $\tilde{z}$).

Every coefficient and every set is *defined before being used.* In every model: (i) the variables are introduced explicitly before the formulation; (ii) the objective and each family of constraints carry an equation number; (iii) the non-negativity constraints, which define the variables, close the model; (iv) after the model, a list describes the objective and the constraints family by family, with the number of constraints in each.

1.4 A deliberate choice: continuous variables only

All the models in these lecture notes use **exclusively continuous variables**: no binary or integer variables. It is a deliberate choice, with three motivations.

1. **Teaching**: with continuous variables we can rely on duality, shadow prices and the KKT conditions — the tools that turn a numerical solution into an economic explanation. With binary variables these tools are lost or weakened.
2. **Coverage**: a surprising number of real decisions is naturally continuous: quantities, flows, portfolio weights, prices, power levels, coordinates.
3. **Progression**: the same application can move from linear programming (LP) to quadratic (QP) and nonlinear (NLP) programming, showing how the expressive power of the model grows without changing tool.

1.5 Map of the models

Ch.	Application	Class	Central concept
4	Multi-period production and inventory	LP / QP	duality, capacity, smoothing
5	Supply chain and sustainability	LP / NLP	flows, congestion, CO_2
6	Markowitz portfolio	QP	risk-return
7	Pricing and revenue management	NLP	elasticity and capacity
8	Advertising budget	convex NLP	diminishing returns
9	Continuous location	convex NLP	efficiency-equity
10	Electric vehicle charging	LP / QP	energy, peaks
11	Capacity and waiting times	convex NLP	capacity-waiting
12	Newsvendor and variants	stoch. LP	deciding before knowing
13	VaR and CVaR	LP	tail risk
14	Arbitrage and pricing	LP	the duality that prices
15	Support Vector Machine	QP	optimization for ML

1.6 Structure of each chapter

Every application chapter follows the same eight-step outline, designed to be replicated in the laboratory reports:

1. **Managerial motivation** — the problem told as a business decision;
2. **Guided construction of the model** — sets, parameters, variables, constraints and objective, explained line by line;
3. **Numerical example worked out by hand** — a minimal instance solved with all the steps, to understand the mechanism before switching on the solver;
4. **Case study** — a realistic instance with reproducible data (CSV);
5. **Python/Gurobi implementation** — the complete code, with comments;
6. **Results** — the solver output translated into managerial language;
7. **Sensitivity analysis** — shadow prices, scenarios, trade-offs;
8. **Exercises** — from verification to a small research extension.

> The numbers reported in each chapter are generated by the scripts in the `python/` folder: running `python3 run_all.py` regenerates data, tables and figures. If you modify the data, the numbers in the lecture notes must be regenerated.

1.7 Course organization and assessment

The essential path comprises four laboratory sessions (chapter 17 for the details): (1) production and inventory with duality and shadow prices; (2) Markowitz and the efficient frontier; (3) nonlinear modelling (pricing or budget); (4) a project of your own choice with a managerial presentation. The suggested assessment weights: correctness of the formulation 30%, implementation and numerical verification 25%, sensitivity analysis 25%, managerial interpretation and quality of the visualizations 20%.

Chapter 2

Background: linear, quadratic and nonlinear programming

This chapter collects the theoretical tools used throughout these lecture notes, in two subchapters: the *theory of linear programming* (duality, optimality conditions, sensitivity analysis) and its *nonlinear extension* (convexity, KKT conditions). It closes with the sensitivity protocol used in every laboratory session. It also serves as a quick reference.

2.1 Linear programming theory

2.1.1 Duality

We call $M = \{1, 2, \ldots, m\}$ the set of constraint indices and $N = \{1, 2, \ldots, n\}$ the set of variable indices. The *data* of the problem are the costs $c_j \in \mathbb{R}$ for $j \in N$, the coefficients $a_{ij} \in \mathbb{R}$ and the right-hand sides $b_i \in \mathbb{R}$ for $i \in M$. The sets $M_{\leq}$, $M_{=}$ and $M_{\geq}$ partition M according to the direction of the constraint; the sets $N_{\geq 0}$, $N_{\gtreqless 0}$ and $N_{\leq 0}$ partition N according to the sign of the variable (non-negative, free, non-positive). The *variables* are the x_j of the primal and, in the dual, one π_i for each of the m constraints of the primal.

The primal (P), general form

$$
\begin{aligned}
\text{(P)} \qquad \max \quad & \sum_{j \in N} c_j\, x_j \\
\text{subject to} \quad & \sum_{j \in N} a_{ij}\, x_j \leq b_i, && \forall i \in M_{\leq}, \\
& \sum_{j \in N} a_{ij}\, x_j = b_i, && \forall i \in M_{=}, \\
& \sum_{j \in N} a_{ij}\, x_j \geq b_i, && \forall i \in M_{\geq}, \\
& x_j \geq 0, && \forall j \in N_{\geq 0}, \\
& x_j \gtreqless 0, && \forall j \in N_{\gtreqless 0}, \\
& x_j \leq 0, && \forall j \in N_{\leq 0}.
\end{aligned}
$$

The dual (D), general form

$$
\begin{array}{lll}
\text{(D)} \quad \min & \displaystyle\sum_{i\in M} b_i\,\pi_i & \\
\text{subject to} & \displaystyle\sum_{i\in M} a_{ij}\,\pi_i \geq c_j, & \forall j \in N_{\geq 0}, \\
& \displaystyle\sum_{i\in M} a_{ij}\,\pi_i = c_j, & \forall j \in N_{\gtreqless 0}, \\
& \displaystyle\sum_{i\in M} a_{ij}\,\pi_i \leq c_j, & \forall j \in N_{\leq 0}, \\
& \pi_i \geq 0, & \forall i \in M_{\leq}, \\
& \pi_i \gtreqless 0, & \forall i \in M_{=}, \\
& \pi_i \leq 0, & \forall i \in M_{\geq}.
\end{array}
$$

The rules of duality

Every constraint of the primal generates a dual variable, every variable of the primal a dual constraint, with the correspondences:

Primal (max)		Dual (min)
constraint $\leq b_i$	$(i \in M_{\leq})$	variable $\pi_i \geq 0$
constraint $= b_i$	$(i \in M_{=})$	variable $\pi_i \gtreqless 0$
constraint $\geq b_i$	$(i \in M_{\geq})$	variable $\pi_i \leq 0$
variable $x_j \geq 0$	$(j \in N_{\geq 0})$	constraint $\geq c_j$
variable $x_j \gtreqless 0$	$(j \in N_{\gtreqless 0})$	constraint $= c_j$
variable $x_j \leq 0$	$(j \in N_{\leq 0})$	constraint $\leq c_j$

Two quick readings: the *equalities* of the primal give *free* dual variables (and vice versa); in a maximization a resource constraint $\leq$ has $\pi_i \geq 0$ (more resource, more value). For a *minimization* primal the table is read from right to left.

The pair is linked by the two duality theorems. In the first, the *bar* denotes any feasible solution; in the second, the *tilde* denotes an optimal solution.

Weak duality theorem

For every feasible solution $(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$ of the primal (P) and every feasible solution $(\bar{\pi}_1, \bar{\pi}_2, \ldots, \bar{\pi}_m)$ of the dual (D) we have

$$\sum_{j\in N} c_j\,\bar{x}_j \;\leq\; \sum_{i\in M} b_i\,\bar{\pi}_i.$$

Strong duality theorem

The primal (P) has an optimal solution $(\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_n)$ *if and only if* the dual (D) has an optimal solution $(\tilde{\pi}_1, \tilde{\pi}_2, \ldots, \tilde{\pi}_m)$, and in that case the two optimal values coincide:

$$\sum_{j\in N} c_j\,\tilde{x}_j \;=\; \sum_{i\in M} b_i\,\tilde{\pi}_i.$$

Line-by-line explanation

Weak duality says that every feasible solution of the dual is an *upper bound* for the value of the maximization primal (and every feasible solution of the primal a lower bound for the dual): if two feasible solutions have the same value, they are both optimal. Strong duality guarantees that at the optimum this always happens: the gap closes. Existence does not mean uniqueness: there may be more than one optimal solution, but the optimal values coincide. The dual variables π_i and the reduced costs that we shall shortly define are the numbers we shall use all the time.

Building the dual with the rules: the 2×2 LP

Let us apply the rules to the 2×2 LP, which is exactly in the canonical maximization form (constraints $\leq$, non-negative variables):

$$\begin{array}{rrcrcr} \max & 30\,x_1 + 50\,x_2 & & & & \\ \text{subject to} & x_1 + & 3\,x_2 & \leq & 90, \\ & 2\,x_1 + & x_2 & \leq & 80, \\ & x_1, & x_2 & \geq & 0. \end{array}$$

- **Step 1 — every constraint generates a dual variable.** Two constraints $\leq$ in a maximization $\Rightarrow \pi_1, \pi_2 \geq 0$.
- **Step 2 — every variable generates a dual constraint.** The coefficients are the *columns* of the primal, the right-hand sides are the costs: from $x_1 \geq 0$ we get $\pi_1 + 2\pi_2 \geq 30$; from $x_2 \geq 0$ we get $3\pi_1 + \pi_2 \geq 50$.
- **Step 3 — the objective swaps the roles**: the right-hand sides $(90, 80)$ become the costs of the dual, which is a *minimization* problem.

$$\begin{array}{rrcrcr} \min & 90\,\pi_1 + 80\,\pi_2 & & & & \\ \text{subject to} & \pi_1 + & 2\,\pi_2 & \geq & 30, \\ & 3\,\pi_1 + & \pi_2 & \geq & 50, \\ & \pi_1, & \pi_2 & \geq & 0. \end{array}$$

Building the dual with the rules: all the cases in a single LP

Let us now build a *minimization* LP that contains all the cases of the table — three variables with the three signs and three generic constraints with the three directions; being a minimization, the table is read from right to left:

$$
\begin{array}{lrrrcr}
\min & 5\,x_1 + 8\,x_2 - 9\,x_3 & & & & \\
\text{subject to} & x_1 + & x_2 & & \geq & 30, \\
& x_1 + & x_2 - & x_3 & = & 100, \\
& x_1 - 2\,x_2 & & & \leq & -20, \\
& x_1 & & & \geq & 0, \\
& & x_2 & & \gtreqless & 0, \\
& & & x_3 & \leq & 0.
\end{array}
$$

- **Step 1 — the dual variables**, one per constraint, with the sign given by the direction: $\pi_1 \geq 0$ (constraint $\geq$ in a minimization), $\pi_2 \gtreqless 0$ (equality), $\pi_3 \leq 0$ (constraint $\leq$ in a minimization).
- **Step 2 — the dual constraints**, one per variable, with the direction given by the sign: from $x_1 \geq 0$ we get $\pi_1 + \pi_2 + \pi_3 \leq 5$; from $x_2 \gtreqless 0$ we get $\pi_1 + \pi_2 - 2\pi_3 = 8$; from $x_3 \leq 0$ we get $-\pi_2 \geq -9$.
- **Step 3 — the objective**: the dual is a maximization, max $30\pi_1 + 100\pi_2 - 20\pi_3$.

$$
\begin{array}{lrrrcr}
\max & 30\,\pi_1 + 100\,\pi_2 - 20\,\pi_3 & & & & \\
\text{subject to} & \pi_1 + & \pi_2 + & \pi_3 & \leq & 5, \\
& \pi_1 + & \pi_2 - & 2\,\pi_3 & = & 8, \\
& - & \pi_2 & & \geq & -9, \\
& \pi_1 & & & \geq & 0, \\
& & \pi_2 & & \gtreqless & 0, \\
& & & \pi_3 & \leq & 0.
\end{array}
$$

2.1.2 The optimality conditions: complementary slackness

To every primal constraint we associate its *slack* $\bar{s}_i$, and to every variable its *reduced cost* $\bar{c}_j$, that is the slack of the corresponding *dual* constraint:

$$
\bar{s}_i = \sum_{j \in N} a_{ij}\,\bar{x}_j - b_i, \quad \forall i \in M, \qquad \bar{c}_j = c_j - \sum_{i \in M} a_{ij}\,\bar{\pi}_i, \quad \forall j \in N.
$$

The slack measures how far constraint i is from being binding; the reduced cost measures the coefficient c_j net of the value of the resources consumed by x_j, evaluated at the dual variables.

LP optimality conditions (complementary slackness)

A pair of feasible solutions, $(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$ for the primal (P) and $(\bar{\pi}_1, \bar{\pi}_2, \ldots, \bar{\pi}_m)$ for the dual (D), is optimal for both problems *if and only if*

$$
\begin{aligned}
\bar{\pi}_i \cdot \bar{s}_i &= 0, && \forall i \in M, \\
\bar{x}_j \cdot \bar{c}_j &= 0, && \forall j \in N.
\end{aligned}
$$

Line-by-line explanation

In words: **slack constraint** $\Rightarrow$ **dual at zero** ($\bar{s}_i \neq 0 \Rightarrow \bar{\pi}_i = 0$) and **variable different from zero** $\Rightarrow$ **reduced cost null** ($\bar{x}_j \neq 0 \Rightarrow \bar{c}_j = 0$). At the optimum these are *implications*, not equivalences: because of degeneracy an active constraint may have a null dual, and a variable at zero a null reduced cost. The verification examples further on check exactly these conditions.

Line-by-line explanation

Two readings to remember for the laboratory:

- **Shadow price** $\tilde{\pi}_i$: the marginal value of the right-hand side b_i,

$$\tilde{\pi}_i = \frac{\partial \tilde{z}}{\partial b_i}\,;$$

  it holds for b_i within the *validity range of the right-hand side* $[b_i^{\min}, b_i^{\max}]$: the largest interval in which, varying only b_i, the shadow price remains exactly $\tilde{\pi}_i$ (beyond it, it changes and must be recomputed). Special case: inactive constraint $\Rightarrow \tilde{\pi}_i = 0$.

- **Reduced cost** $\bar{c}_j$ (the slack of the dual constraint, defined above): for a variable at zero it says by how much c_j must improve for it to become worth activating; it holds for c_j within the *validity range of the coefficient* $[c_j^{\min}, c_j^{\max}]$: the largest interval in which, varying only c_j, the optimal solution does not change (for a variable at zero the profitability threshold is precisely one of the two endpoints). Special case: variable different from zero $\Rightarrow \bar{c}_j = 0$.

- **Beware of degeneracy**: there may exist a variable *at zero with null reduced cost*, even though it is not used. Hence only the implication holds, "$\bar{c}_j \neq 0 \Rightarrow$ variable on one of its bounds", not the converse.

The signs of shadow prices, explained properly

The shadow price is always, with a single convention,

$$\tilde{\pi}_i = \frac{\partial \tilde{z}}{\partial b_i}\,,$$

the derivative of the optimal value with respect to the right-hand side of constraint i: by how much $\tilde{z}$ changes if b_i increases by one (small) unit. The sign is always deduced by answering two questions:

- **does increasing b_i enlarge or shrink the feasible region?** With a constraint $\leq$ it enlarges it (the ceiling is raised); with a constraint $\geq$ it shrinks it (the requirement grows); with an equality it merely shifts it;

- **how does a wider region change the optimum?** It can never worsen it: in a minimization the optimal value decreases or stays the same, in a maximization it increases or stays the same.

Combining the two answers:

Direction of the constraint	minimization problem	maximization problem
$\leq$ (b_i grows $\Rightarrow$ wider region)	$\tilde{\pi}_i \leq 0$	$\tilde{\pi}_i \geq 0$
$\geq$ (b_i grows $\Rightarrow$ narrower region)	$\tilde{\pi}_i \geq 0$	$\tilde{\pi}_i \leq 0$
$=$	any sign (free dual)	

Two typical readings. In a *minimization*: a constraint $\leq$ (a capacity) has $\tilde{\pi}_i \leq 0$ — relaxing it reduces the cost, and $|\tilde{\pi}_i|$ is the saving for each extra unit of capacity; a constraint $\geq$ (a requirement to be met) has $\tilde{\pi}_i \geq 0$ — raising the requirement costs, and $\tilde{\pi}_i$ is the marginal cost. In a *maximization* the readings are reversed. In every case an inactive constraint has $\tilde{\pi}_i = 0$ by complementarity.

Duality on a 2×2 LP, worked out in full

Let us take up again the 2×2 LP and its dual, built above, and verify the optimality conditions.

- **Step 1 — solving we obtain:**

$$(\tilde{x}_1, \tilde{x}_2) = (30,\, 20), \qquad \tilde{z} = 1900, \qquad (\tilde{\pi}_1, \tilde{\pi}_2) = (14,\, 8), \qquad (\bar{c}_1, \bar{c}_2) = (0,\, 0),$$

  with validity ranges $[b_1^{\min}, b_1^{\max}] = [40, 240]$, $[b_2^{\min}, b_2^{\max}] = [30, 180]$, $[c_1^{\min}, c_1^{\max}] = [50/3,\, 100]$ and $[c_2^{\min}, c_2^{\max}] = [15,\, 90]$. Both constraints are active.

- **Step 2 — verification of the shadow prices.** One extra unit of b_1 is worth 14, one extra unit of b_2 is worth 8. Let us verify the property that characterizes them: $\tilde{x}_1$ and $\tilde{x}_2$ are different from zero, hence by complementary slackness their dual constraints must be active ($\bar{c}_1 = \bar{c}_2 = 0$):

$$\tilde{\pi}_1 + 2\tilde{\pi}_2 = 14 + 16 = 30 = c_1, \qquad 3\tilde{\pi}_1 + \tilde{\pi}_2 = 42 + 8 = 50 = c_2. \quad \checkmark$$

- **Step 3 — verification of strong duality.**

$$\sum_{i \in M} b_i\, \tilde{\pi}_i = 90 \cdot 14 + 80 \cdot 8 = 1260 + 640 = 1900 = \tilde{z}. \quad \checkmark$$

- **Step 4 — verification of the reduced costs.** With the definition $\bar{c}_j = c_j - \sum_{i \in M} a_{ij}\, \tilde{\pi}_i$:

$$\bar{c}_1 = 30 - (1 \cdot 14 + 2 \cdot 8) = 0, \qquad \bar{c}_2 = 50 - (3 \cdot 14 + 1 \cdot 8) = 0:$$

  both null, as for every positive variable at the optimum: the two variables repay exactly the resources they consume. Only a variable sitting on one of its bounds can have a reduced cost different from zero.

All the cases in a single LP: verification of the conditions

Let us take up again the "all the cases" LP and its dual (example 2.1.1) — the cases are the same ones that will come back, one at a time, in the application chapters.

- **Step 1 — solving we obtain:**

$$(\tilde{x}_1, \tilde{x}_2, \tilde{x}_3) = (60,\, 40,\, 0), \qquad \tilde{z} = 620,$$

$$(\tilde{\pi}_1, \tilde{\pi}_2, \tilde{\pi}_3) = (0,\ 6,\ -1), \qquad (\bar{c}_1, \bar{c}_2, \bar{c}_3) = (0,\ 0,\ -3),$$

with validity ranges of the right-hand sides $(-\infty, 100]$, $[30, +\infty)$ and $[-200, +\infty)$, and of the coefficients $(-\infty, 8]$, $[5, 17]$ and $(-\infty, -6]$. At the optimum the second and third constraints are active, with $\tilde{x}_3 = 0$.

- **Step 2 — verification of the shadow prices with the rules of the table.** $\tilde{\pi}_1 = 0$, consistent with complementarity: the first constraint is slack ($\tilde{x}_1 + \tilde{x}_2 = 100 > 30$). $\tilde{\pi}_2 = 6$ (free dual variable, here positive). $\tilde{\pi}_3 = -1 \leq 0$, as it must be for a constraint $\leq$ in a minimization. Let us verify that, as complementary slackness requires, the dual constraints of x_1 and x_2 (different from zero) are active:

$$\underbrace{0 + 6 - 1}_{x_1} = 5 = c_1, \qquad \underbrace{0 + 6 - 2 \cdot (-1)}_{x_2} = 8 = c_2, \quad \checkmark$$

(x_2 is free: its dual constraint is an equality). Verification by perturbation:

$$b_2 = 101 \ \Rightarrow\ \tilde{z} = 626\ (+6), \qquad b_3 = -19 \ \Rightarrow\ \tilde{z} = 619\ (-1). \quad \checkmark$$

- **Step 3 — verification of strong duality.**

$$\sum_{i \in M} b_i\, \tilde{\pi}_i = 30 \cdot 0 + 100 \cdot 6 + (-20) \cdot (-1) = 620 = \tilde{z}. \quad \checkmark$$

- **Step 4 — verification of the reduced costs, one per sign.** With the definition $\bar{c}_j = c_j - \sum_{i \in M} a_{ij}\, \tilde{\pi}_i$:

$$\bar{c}_1 = 5 - (0 + 6 - 1) = 0, \qquad \bar{c}_2 = 8 - (0 + 6 + 2) = 0:$$

consistent with $\tilde{x}_1, \tilde{x}_2 \neq 0$. x_3 equals zero, which for a variable ≤ 0 is its *upper bound*:

$$\bar{c}_3 = -9 - (-1) \cdot 6 = -3,$$

that is, forcing $x_3 = -1$ would worsen the optimum by 3; x_3 would become active only if its coefficient rose to at least -6 (precisely the endpoint $c_3^{\max}$ of the validity range). Verification:

$$x_3 = -1 \ \Rightarrow\ \tilde{z} = 623\ (+3). \quad \checkmark$$

2.1.3 Sensitivity analysis and shadow prices

Solving an LP gives an optimal solution; **sensitivity analysis** answers the next question, the one that really matters to a decision maker: *how does the optimum change if the data change?* In LPs a large part of the answer comes for free together with the solution, in the form of four numbers for each constraint and variable:

- the **shadow price** $\tilde{\pi}_i$ of a constraint: by how much the optimal value improves if the right-hand side of the constraint increases by one unit — "how much I would pay for one extra unit of that resource";
- the **validity range of the right-hand side** $[b_i^{\min}, b_i^{\max}]$: where that shadow price remains exact (outside it, the optimization must be redone);

- the **reduced cost** $\bar{c}_j$ of a variable at zero: by how much its objective coefficient must improve for it to become worth activating;
- the **validity range of the objective coefficient** $[c_j^{\min}, c_j^{\max}]$: where the coefficient can vary without the optimal solution changing. In the 2×2 LP: c_1 can vary in $[50/3 \approx 16.7,\ 100]$ and c_2 in $[15, 90]$ without moving the solution $(30, 20)$.

Shadow prices in practice, on the 2×2 LP

Let us take up again the 2×2 LP: optimum 1900 with duals $\tilde{\pi}_1 = 14$ and $\tilde{\pi}_2 = 8$.
Reading. One extra unit of b_1 makes the optimum grow by 14; if obtaining it cost 12, it would be worth it ($14 > 12$).
Verification by perturbation. Re-solving:

$$b_1 = 91 \ \Rightarrow\ 1914 = 1900 + 14, \qquad b_1 = 100 \ \Rightarrow\ 2040 = 1900 + 10 \cdot 14:$$

the shadow price is still exact because 100 lies inside the validity range $[40, 240]$.
Where it stops holding. At $b_1 = 240$ the first constraint stops being the scarce resource: beyond that, every additional unit is worth less than 14 and everything must be recomputed. This is why every shadow price is quoted *together* with its validity range.
Inactive constraints. If we had $b_1 = 300$ (constraint inactive at the optimum), its shadow price would be 0: a resource left over is worth nothing at the margin.

Reduced costs in practice: is a third variable worth it?

In the 2×2 LP both variables are positive at the optimum and their reduced costs equal 0: this is always the case for variables different from zero. To see a reduced cost at work we need a variable at zero: let us add to the model a variable $x_3 \geq 0$ with coefficient $c_3 = 20$ in the objective and consumptions $a_{13} = a_{23} = 1$ in the two constraints.

- **Step 1 — valuation with the shadow prices.** The resources absorbed by one unit of x_3 are worth, at the shadow prices,

  $$1 \cdot 14 + 1 \cdot 8 = 22 > 20 = c_3:$$

  every unit of x_3 would take resources away from better uses.

- **Step 2 — the reduced cost.** The difference is precisely the reduced cost:

  $$\bar{c}_3 = 20 - 22 = -2.$$

  Solving confirms it: the optimal solution remains $(30, 20, 0)$ with $\tilde{z} = 1900$, the variable x_3 is at zero with $\bar{c}_3 = -2$ and $c_3^{\max} = 22$: the coefficient c_3 would have to rise to at least 22 for it to be worth activating.

- **Step 3 — counter-check.** With $c_3 = 23 > 22$ the solution really does change, and not by a little: the new optimal solution is $(0, 5, 75)$ with value 1975. The reduced cost tells us *when* a variable at zero enters the solution, not *how* the solution changes: that has to be recomputed.

2.2 Nonlinear optimization theory

2.2.1 Convexity and quadratic programming

A problem $\min f(x_1, x_2, \ldots, x_n)$ over a feasible set X is **convex** if f is convex and X is convex. The property we shall use all the time:

Why convexity matters

In a convex problem every local minimum is global: optimality can be *certified.* In non-convex problems (for instance a pricing problem with a bilinear term $p \cdot q$) a local method guarantees only a local optimum; non-convex QPs can nonetheless be solved to global optimality with dedicated techniques, but the computational cost grows.

A **QP** has objective $\frac{1}{2}\sum_i \sum_j q_{ij}\, x_i x_j + \sum_j c_j x_j$ and linear constraints; it is convex if and only if the matrix $\boldsymbol{Q} = (q_{ij})$ is positive semidefinite ($\boldsymbol{Q} \succeq 0$). The example that follows shows one on a small scale; in these lecture notes QPs appear every time a cost or a risk grows more than linearly with the decision.

A 2×2 QP, worked out in full

Consider a minimization QP with a separable quadratic objective (the second variable "costs" twice as much as the first, at the margin) and a threshold constraint on the sum:

$$\begin{array}{rrcrcl} \min & x_1^2 + 2\,x_2^2 & & & & \\ \text{subject to} & x_1 + & & x_2 & \geq & 6, \\ & x_1, & & x_2 & \geq & 0. \end{array}$$

Here $\boldsymbol{Q} = \begin{pmatrix} 2 & 0 \\ 0 & 4 \end{pmatrix} \succ 0$: the QP is convex and the optimum found is global.

- **Step 1 — solving we obtain:**

$$\tilde{x}_1 = 4, \qquad \tilde{x}_2 = 2, \qquad \tilde{f} = 24, \qquad \lambda = 8.$$

- **Step 2 — verification of the shadow price.** The constraint is active ($4 + 2 = 6$: the objective grows with the variables, hence the sum does not exceed what is needed) and the partial derivatives coincide:

$$2\tilde{x}_1 = 8 = 4\tilde{x}_2 = \lambda. \quad \checkmark$$

  If they were different it would be worth shifting quantity from one variable to the other; the common derivative is precisely the multiplier.

- **Step 3 — verification by perturbation.** With right-hand side 7 we obtain $(\tilde{x}_1, \tilde{x}_2) = (14/3,\ 7/3)$ with

$$\tilde{f} = 32.67 \approx 24 + 8$$

  (on top of that there is the curvature term: here the shadow price is exact only at the margin, not for finite increments).

2.2.2 The optimality conditions: the KKT conditions

The optimality conditions of constrained nonlinear problems are known as **KKT conditions** (after Karush, Kuhn and Tucker, who formulated them). For the problem $\min f(x_1, \dots, x_n)$ subject to $g_i(x_1, \dots, x_n) \leq 0$ for every $i \in \{1, 2, \dots, m\}$ and $h_j(x_1, \dots, x_n) = 0$ for every $j \in \{1, 2, \dots, q\}$, the Lagrangian is

$$L(x_1, \dots, x_n) = f(x_1, \dots, x_n) + \sum_{i=1}^{m} \lambda_i \, g_i(x_1, \dots, x_n) + \sum_{j=1}^{q} \nu_j \, h_j(x_1, \dots, x_n), \qquad \lambda_i \geq 0.$$

KKT conditions

At a regular optimal point $(\tilde{x}_1, \dots, \tilde{x}_n)$ there exist multipliers $\lambda_i \geq 0$ and ν_j such that

$$\frac{\partial L}{\partial x_k}(\tilde{x}_1, \dots, \tilde{x}_n) = 0 \quad \forall k \in \{1, 2, \dots, n\} \quad \text{(stationarity)},$$

$$\lambda_i \, g_i(\tilde{x}_1, \dots, \tilde{x}_n) = 0 \quad \forall i \in \{1, 2, \dots, m\} \quad \text{(complementarity)}.$$

If the problem is convex, the KKT conditions are also *sufficient*: a point that satisfies them is a global optimum.

The KKT conditions are complementary slackness, generalized

The two parts of this chapter mirror each other. For an LP the KKT conditions reduce *exactly* to the optimality conditions of section 2.1.2: the stationarity of the Lagrangian is equivalent to the feasibility of the dual (the multipliers λ_i are the dual variables π_i, and the multipliers of the sign constraints are the reduced costs), and the complementarity $\lambda_i \, g_i(\tilde{x}_1, \dots, \tilde{x}_n) = 0$ is the complementary slackness condition $\bar{\pi}_i \cdot \bar{s}_i = 0$. Complementary slackness is the KKT conditions in the linear case.

The KKT conditions on the QP of the example, worked out in full

Let us take up again the QP just solved: $\min \; x_1^2 + 2x_2^2$ subject to $x_1 + x_2 \geq 6$, $x_1, x_2 \geq 0$.

- **Step 1.** Let us rewrite the constraint in standard form: $g(x_1, x_2) = 6 - x_1 - x_2 \leq 0$.
- **Step 2.** Stationarity of the Lagrangian $L = x_1^2 + 2x_2^2 + \lambda\,(6 - x_1 - x_2)$:

  $$\frac{\partial L}{\partial x_1} = 2x_1 - \lambda = 0, \qquad \frac{\partial L}{\partial x_2} = 4x_2 - \lambda = 0 \;\Rightarrow\; x_1 = \frac{\lambda}{2},\; x_2 = \frac{\lambda}{4}$$

  (the constraints $x_1, x_2 \geq 0$ will turn out to be inactive: their multipliers are null by complementarity).
- **Step 3.** Complementarity: if the threshold constraint were inactive ($\lambda = 0$) we would have $x_1 = x_2 = 0$, which violates $x_1 + x_2 \geq 6$: impossible. Hence the constraint is active: $\lambda/2 + \lambda/4 = 6 \Rightarrow \lambda = 8$, $\tilde{x}_1 = 4$, $\tilde{x}_2 = 2$, $\tilde{f} = 24$ — the same numbers found by elementary means in the QP example, now obtained with the general procedure.
- **Step 4.** Reading of the multiplier: with right-hand side $6 + \varepsilon$ the optimal value grows by about $\lambda\varepsilon = 8\varepsilon$. Verification: with right-hand side d the solution is $\tilde{x}_1 = 2d/3$, $\tilde{x}_2 = d/3$, hence

  $$\tilde{f}(d) = \tfrac{2}{3}d^2 \;\Rightarrow\; \tilde{f}(6 + \varepsilon) = 24 + 8\varepsilon + \tfrac{2}{3}\varepsilon^2. \quad \checkmark$$

Line-by-line explanation

The KKT multipliers generalize the shadow prices of the LP: they measure the marginal variation of the optimal value with respect to the right-hand side of the constraint. In the laboratory, when the duals are not available (general nonlinear models), we estimate the multiplier *by perturbation*: the right-hand side is increased by ε, the problem is re-optimized and the incremental ratio is computed. This numerical verification is itself a recurring exercise.

2.3 The sensitivity analysis protocol

Sensitivity analysis is part of the model, not an appendix. In every laboratory session we follow the same six-step protocol:

1. Base scenario	solve, verify feasibility, identify the active constraints.
2. One-at-a-time	vary one key parameter over a grid; plot value/decisions.
3. Shadow prices and reduced costs	compare the marginal prediction of the dual with a re-optimization after a small perturbation; read the reduced cost $\bar{c}_j$ of every variable at zero: the threshold beyond which it is worth activating it.
4. Scenarios	build at least three consistent scenarios (pessimistic, central, optimistic).
5. Trade-off	vary a weight or a threshold to trace a frontier (cost-service, risk-return, cost-emissions).
6. Stability	flag abrupt changes in the optimal mix and fragile estimated parameters.

Exercise 2.1 — duality

In the 2×2 LP of the example it is possible to increase b_1 by 10 units by paying 12 per unit. Is it worth it? Up to which unit price is it worth it? And up to which value of b_1 does the shadow price of 14 hold? (Hint: recompute the optimal vertex when the first constraint moves to $90 + \Delta$ and find the Δ beyond which the solution changes structure.)

Exercise 2.2 — KKT

Solve with the KKT conditions: $\min\ (x-3)^2 + (y-1)^2$ subject to $x + 2y \leq 2$, $x, y \geq 0$. Say which constraints are active at the optimum and interpret the multipliers.

Exercise 2.3 — convexity

For which values of the parameter γ is the function $f(x_1, x_2) = x_1^2 + \gamma x_1 x_2 + x_2^2$ convex? (Write Q and impose $Q \succeq 0$.)

Chapter 3

The solver: building, solving, interpreting

This chapter is the toolbox of the laboratory: how to build a model with `gurobipy`, how to run it, how to retrieve the solution and — the most important part — how to interpret the output. The examples are generic and use the same LPs as chapter 2: together, the two chapters provide the basis for understanding all the models and examples of the following chapters. A stand-alone version of this guide is in the file `GUIDA_GUROBI.md` in the course folder. Three subchapters: the *general commands* (building, solving, reading the solution), then the commands specific to *linear models* (duals, reduced costs, validity ranges) and to *nonlinear models* (functional and bilinear constraints).

3.1 The general commands

3.1.1 Installation and licence

```
python3 -m pip install gurobipy
```

The package includes a demonstration licence (up to 2000 variables and 2000 constraints), sufficient for almost all the models in these lecture notes. For the full and free academic licence: register at `https://portal.gurobi.com` with your institutional email address, request a *Named-User Academic License* and run the command `grbgetkey` indicated by the portal (on the university network or VPN).

3.1.2 The five steps to build a model

Let us take up again the 2×2 LP of chapter 2: $\max 30x_1 + 50x_2$ with $x_1 + 3x_2 \leq 90$ (constraint 1), $2x_1 + x_2 \leq 80$ (constraint 2) and $x_1, x_2 \geq 0$.

```python
import gurobipy as gp
from gurobipy import GRB

# Step 1: the model container
m = gp.Model("lp_2x2")

# Step 2: the decision variables (default: continuous, >= 0)
x1 = m.addVar(name="x1")
x2 = m.addVar(name="x2")

# Step 3: the constraints (keep the object to read the dual later)
v1 = m.addConstr(x1 + 3*x2 <= 90, name="vincolo1")
v2 = m.addConstr(2*x1 + x2 <= 80, name="vincolo2")
```

```python

# Step 4: the objective function
m.setObjective(30*x1 + 50*x2, GRB.MAXIMIZE)

# Step 5 (debug): write the model in readable format
m.write("lp_2x2.lp")
```

The three golden rules of `gurobipy`

1. **Indexed variables** with `addVars`:
   `x = m.addVars(I, T, name="x")` creates the family `x[i,t]` over the index sets `I` and `T`; the sums are written `gp.quicksum(...)` or `x.sum(i, "*")`.

2. **Free variables** (which may be negative, such as the intercept b of the SVM or the threshold η of the CVaR): `m.addVar(lb=-GRB.INFINITY)`. Forgetting `lb` is the most frequent mistake in the laboratory: the default is ≥ 0!

3. **Families of constraints** with a generator: `m.addConstrs((x.sum("*", t) <= b[t] for t in T), name="cap")`.

3.1.3 Running the model and reading the log

```python
m.Params.OutputFlag = 0     # 0 = silent, 1 = log on screen
m.Params.TimeLimit  = 60    # seconds
m.Params.NonConvex  = 2     # only needed for non-convex QPs (e.g. pricing p*q)
m.optimize()
```

The solver log, when active, must be read at three points:

```
Optimize a model with 2 rows, 2 columns and 4 nonzeros   <- expected size?
Coefficient statistics: Matrix range [1e+00, 3e+00]      <- data well scaled?
Optimal objective  1.900000000e+03                        <- status and value
```

If the coefficient ranges span from 10^{-6} to 10^{9} the model is badly scaled: change the units of measurement (thousands of euros, MWh, ...) before looking for other errors.

3.1.4 Retrieving the solution

Always check the status before reading any number:

```python
if m.Status == GRB.OPTIMAL:
    print(m.ObjVal, x1.X, x2.X)
elif m.Status == GRB.INFEASIBLE:
    m.computeIIS(); m.write("conflitto.ilp")    # which constraints contradict each other
elif m.Status == GRB.UNBOUNDED:
    pass  # almost always a constraint or a bound is missing
```

Attribute	Object	Meaning
`Status`	model	`OPTIMAL`=2, `INFEASIBLE`=3, `UNBOUNDED`=5, `TIME_LIMIT`=9
`ObjVal`	model	optimal value of the objective
`X`	variable	value of the variable at the optimum
`Slack`	constraint	slack: 0 $\Rightarrow$ active constraint

The attributes specific to LPs (shadow prices, reduced costs, validity ranges) are in the subchapter that follows.

3.2 The commands for linear models

3.2.1 Shadow prices, reduced costs and ranges: the attributes

The bridge with the notation of chapter 2

The solver attributes are exactly the theoretical quantities of the background chapter:

Theory (ch. 2)	Gurobi attribute	Object
shadow price $\tilde{\pi}_i$	`Pi`	constraint
reduced cost $\bar{c}_j$	`RC`	variable
slack $\bar{s}_i$	`Slack`	constraint
range $[b_i^{\min}, b_i^{\max}]$	`SARHSLow`–`SARHSUp`	constraint
range $[c_j^{\min}, c_j^{\max}]$	`SAObjLow`–`SAObjUp`	variable

If a constraint contains a constant on the left-hand side, such as `quicksum(x[v,t] ...) + base[t] <= C`, Gurobi moves it into the right-hand side: the stored RHS is `C - base[t]`. Anyone who in a sensitivity analysis writes `constr.RHS = nuovaC` is in fact imposing `quicksum(x) <= nuovaC`, which is something else. The correct form is `constr.RHS = nuovaC - base[t]`. This mistake actually happened to us while preparing the case study on electric vehicle charging, which discusses it.

3.2.2 Interpreting the output: the checklist

1. Is the status `OPTIMAL`? If not, stop and diagnose (`computeIIS` for infeasibility, the `.lp` file for unboundedness).
2. Does the optimal value have the expected order of magnitude?
3. Which constraints are active (`Slack` $= 0$)? Are they the expected ones?
4. How large are the shadow prices? Which resource is worth expanding first? (Remember the signs: in a minimization problem, a constraint $\leq$ has `Pi` ≤ 0.)
5. Are there variables at zero? Read their reduced cost (`RC`): it says by how much the objective coefficient must improve for them to be worth activating (section 2.1.3).
6. Is the solution stable? Re-run with data perturbed by 5%.
7. Translate everything into a three-line managerial recommendation.

3.2.3 Two complete readings

Complete reading of the 2×2 example

Running the code of the previous section:

```
Status: 2 (OPTIMAL)       ObjVal: 1900.0
x1 = 30.0    RC = 0.0    SAObj = [16.7, 100.0]
x2 = 20.0    RC = 0.0    SAObj = [15.0, 90.0]
vincolo1: Slack = 0.0    Pi = 14.0     SARHS = [40.0, 240.0]
vincolo2: Slack = 0.0    Pi = 8.0      SARHS = [30.0, 180.0]
```

Reading: both constraints are active (binding); one extra unit of b_1 is worth 14, and the valuation remains valid as long as b_1 stays in $[40, 240]$; beyond that, everything must be recomputed. If only one right-hand side could be increased, better b_1 ($14 > 8$ per unit, at equal cost). The reduced costs (attribute `x1.RC`) are null because both variables are basic — careful: there may also exist a variable *basic at value zero* (degeneracy), so $\texttt{RC} = 0$ alone does not say that the variable is used. Adding the third variable of chapter 2 the solver would give $x_3 = 0$ with $\texttt{RC} = -2$: its coefficient must rise by at least 2 (that is, to 22) for it to enter the solution, with validity range $\texttt{SAObj} = (-\infty, 22]$.

In the general case: what the solver returns and with which sign

Shadow prices (attribute `Pi`, one per constraint): always the "derivative of the optimal value with respect to the right-hand side". The sign is deduced in two moves: increasing b_i enlarges the feasible region (constraint $\leq$) or shrinks it (constraint $\geq$); and a wider region can never worsen the optimum. Hence:

- in a *maximization* problem: resource constraint $\leq \Rightarrow \texttt{Pi} \geq 0$ (more resource, more profit); constraint $\geq \Rightarrow \texttt{Pi} \leq 0$;
- in a *minimization* problem: demand constraint $\geq \Rightarrow \texttt{Pi} \geq 0$ (more demand, more cost); capacity constraint $\leq \Rightarrow \texttt{Pi} \leq 0$ (more capacity, less cost);
- *equality* constraint $\Rightarrow$ any sign, as prescribed by the rules of duality. Inactive constraint $\Rightarrow \texttt{Pi} = 0$; validity within the range `SARHSLow`–`SARHSUp`.

Reduced costs (attribute `RC`, one per variable):

- *basic* variable: $\texttt{RC} = 0$ — as a rule it lies strictly between its bounds, but it may also sit on a bound with $\texttt{RC} = 0$ (*degenerate* basis);
- variable at its *lower bound* (typically at zero): $\texttt{RC} \geq 0$ in a minimization (by how much its cost must fall for it to be worth using), $\texttt{RC} \leq 0$ in a maximization (by how much its margin must rise); validity within the range `SAObjLow`–`SAObjUp`;
- variable at its *upper bound*: `RC` is the shadow price of the bound — in a minimum-cost flow model a saturated arc has $\texttt{RC} < 0$: one extra unit of capacity on that arc saves $|\texttt{RC}|$.

The complete loop, for *all* the constraints and variables (LP only):

```python
for v in m.getConstrs():
    print(v.ConstrName, v.Pi, v.SARHSLow, v.SARHSUp)
for v in m.getVars():
    print(v.VarName, v.X, v.RC, v.SAObjLow, v.SAObjUp)
```

The complete example: all the cases, from the model to the reading

Let us implement the "all the cases" LP of chapter 2 (three constraint directions, three variable signs) and read *everything* the solver returns:

```
m = gp.Model("tutti_i_casi")
x1 = m.addVar(name="x1")                                    # x1 >= 0 (default)
x2 = m.addVar(lb=-GRB.INFINITY, name="x2")                  # x2 free
x3 = m.addVar(lb=-GRB.INFINITY, ub=0, name="x3")            # x3 <= 0
v1 = m.addConstr(x1 + x2 >= 30,          name="vincolo1")   # direction >=
v2 = m.addConstr(x1 + x2 - x3 == 100,    name="vincolo2")   # direction =
v3 = m.addConstr(x1 - 2*x2 <= -20,       name="vincolo3")   # direction <=
m.setObjective(5*x1 + 8*x2 - 9*x3, GRB.MINIMIZE)
m.optimize()
for v in m.getVars():
    print(v.VarName, v.X, v.RC, v.SAObjLow, v.SAObjUp)
for c in m.getConstrs():
    print(c.ConstrName, c.Slack, c.Pi, c.SARHSLow, c.SARHSUp)
```

```
Status: 2 (OPTIMAL)    ObjVal: 620.0
x1: X =  60.0  RC =  0.0  SAObj = [-inf, 8.0]
x2: X =  40.0  RC =  0.0  SAObj = [5.0, 17.0]
x3: X =   0.0  RC = -3.0  SAObj = [-inf, -6.0]
vincolo1: Slack = -70.0  Pi =  0.0  SARHS = [-inf, 100.0]
vincolo2: Slack =   0.0  Pi =  6.0  SARHS = [30.0, inf]
vincolo3: Slack =   0.0  Pi = -1.0  SARHS = [-200.0, inf]
```

Here all the cases of the previous explanation appear: the inactive constraint with `Pi` $= 0$ (`vincolo1`, `Slack` $\neq 0$); the equality with a positive dual ($+6$); the $\leq$ in a minimization with a negative dual (-1); two basic variables with `RC` $= 0$; and the variable x_3 stuck at its upper bound (0) with `RC` $= -3$ and profitability threshold `SAObjUp` $= -6$. The verification by hand of each of these numbers, with the rules of duality and by perturbation, is in the "all the cases" example of chapter 2.

3.3 The commands for nonlinear models

A single solver for the whole laboratory: Gurobi covers LP and QP natively (including non-convex ones, with `m.Params.NonConvex = 2`) and, from version 12, also solves *globally* models with general nonlinear functions.

3.3.1 Functional constraints and bilinear terms

Nonlinear functions — logarithms, exponentials, powers — enter the model through *functional constraints*: an auxiliary variable y is introduced and $y = f(x)$ is imposed. Ratios such as $1/(\mu - \lambda)$ are instead reformulated with an auxiliary variable and a bilinear constraint ($w\,(\mu - \lambda) = 1$).

```
m.Params.FuncNonlinear = 1      # functions handled exactly (globally)
z = m.addVar(lb=-GRB.INFINITY)
```

```
m.addGenConstrLog(g, z)          # z = log(g);  also Exp, Pow, Sin, ...
m.addQConstr(w * v == 1)         # bilinear terms: NonConvex = 2 is
    required
```

3.3.2 Tolerances and scaling

Two practical devices, both used in the scripts of these lecture notes:

- **tolerances**: for accurate marginal analyses (differences between two nearby optima) tighten `MIPGap`, `FeasibilityTol` and `OptimalityTol` to 10^{-9};
- **scaling**: reformulate to avoid tiny quantities — for instance $q \leq A\,p^{-\varepsilon}$ is written $q \cdot r \leq A$ with $r = p^{\varepsilon}$, which keeps the numbers in a healthy range.

3.3.3 When the duals are not there

For models with functional or bilinear constraints the solver does not provide the duals: shadow prices and multipliers are estimated *by perturbation* (increase the right-hand side by ε, re-optimize, take the incremental ratio), as in the protocol of chapter 2.

Why a single solver

Same syntax, same interpretation checklist and — above all — a *certified global* optimum even in non-convex problems: with a local solver every result would have to be accompanied by the question "is it only a local optimum?". Convex problems with a purely quadratic or conic structure (constraints such as $d_x^2 + d_y^2 \leq d^2$) require nothing special.

Exercise 3.1

Implement the 2×2 LP, print `Pi`, `Slack`, `SARHSLow/Up` of both constraints and verify the shadow price of the first constraint by re-optimizing with $b_1 : 90 \to 91$.

Exercise 3.2

Remove the second constraint and observe the outcome. Then change the objective to minimization with no other changes: why is the result $x = 0$?

Exercise 3.3

Deliberately build an infeasible model (two incompatible constraints, for instance $x_1 + x_2 \geq 10$ and $x_1 + x_2 \leq 8$) and use `computeIIS` to identify the conflict.

Exercise 3.4

Solve with Gurobi's functional constraints the one-variable problem $\max\ 5\log(1+x) - x$ with $x \in [0, 10]$ and compare with the analytical solution ($5/(1+x) - 1 = 0 \Rightarrow x = 4$).

Part II

Deterministic continuous models

Chapter 4

Multi-period production and inventory

Class: LP / convex QP

Script: `python/lab04_production.py`

Online page

4.1 Managerial motivation

A firm decides how much to produce today and how much to keep in inventory for future demand. Producing in advance costs holding; producing at the last moment risks crashing into the capacity limit precisely in the peak months. The model balances margins, capacity, inventory and stability of the plan — and it is the ideal starting point of the laboratory, because every one of its numbers has an immediate economic interpretation.

In practice this problem arises every time a firm plans production over several periods with shared resources: master production schedules (MPS), aggregate planning, supply contracts with monthly profiles. The core of the difficulty is the *intertemporal link*: today's decision (to produce in advance or not) changes tomorrow's constraints.

In this chapter we will learn how to: (1) formulate a multi-period LP with inventory balance constraints; (2) read and check the shadow prices of capacity and their validity ranges; (3) use a quadratic term to stabilize the plan; (4) build the capacity value curve.

4.2 Guided construction of the model

The problem in words. *We decide* how many units of each product to produce in each month and how many to keep in inventory. *The objective* is to minimize the sum of the production and holding costs over the whole horizon. *The constraints* are two: the inventory accounting must add up in every month (what comes in equals what goes out, and all demand must be served) and the machine hours used in a month cannot exceed those available.

Data (input of the model)

Symbol	Type	Meaning
n	$\in \mathbb{Z}_{\geq 1}$	number of products; the products are indexed by $i \in \{1, 2, \ldots, n\}$
m	$\in \mathbb{Z}_{\geq 1}$	number of months of the horizon; the months are indexed by $t \in \{1, 2, \ldots, m\}$
d_{it}	$\in \mathbb{Q}_{\geq 0}$	demand for product i in month t (units)
c_{it}	$\in \mathbb{Q}_{>0}$	production cost of one unit of i in month t (€)
h_{it}	$\in \mathbb{Q}_{>0}$	holding cost of one unit of i for month t (€)
a_i	$\in \mathbb{Q}_{>0}$	machine hours needed to produce one unit of product i
b_t	$\in \mathbb{Q}_{\geq 0}$	machine hours available in month t
s_{i0}	$\in \mathbb{Q}_{\geq 0}$	initial inventory of product i (units)

Decision variables

We introduce the following $2\,n\,m$ non-negative variables:

$$\begin{cases} x_{it} = \text{units of product } i \text{ produced in month } t \\ s_{it} = \text{units of product } i \text{ in stock at the end of month } t \end{cases}$$

$$\forall i \in \{1, 2, \ldots, n\},\ \forall t \in \{1, 2, \ldots, m\}.$$

Using these variables, an LP model for the problem is the following:

Minimum-cost production and inventory LP model

$$\min \quad \sum_{i=1}^{n} \sum_{t=1}^{m} (c_{it}\, x_{it} + h_{it}\, s_{it}) \tag{4.1a}$$

$$\text{subject to} \quad s_{i,t-1} + x_{it} = d_{it} + s_{it}, \quad \forall i \in \{1, 2, \ldots, n\},\ \forall t \in \{1, 2, \ldots, m\}, \tag{4.1b}$$

$$\sum_{i=1}^{n} a_i\, x_{it} \leq b_t, \quad \forall t \in \{1, 2, \ldots, m\}, \tag{4.1c}$$

$$x_{it} \geq 0, \quad \forall i \in \{1, 2, \ldots, n\},\ \forall t \in \{1, 2, \ldots, m\}, \tag{4.1d}$$

$$s_{it} \geq 0, \quad \forall i \in \{1, 2, \ldots, n\},\ \forall t \in \{1, 2, \ldots, m\}. \tag{4.1e}$$

Description of the objective function and of the constraints of the LP model:

- the linear objective function (4.1a) minimizes the total cost of the plan, the sum of the production costs $c_{it}\, x_{it}$ and of the holding costs $h_{it}\, s_{it}$ over all products and months;
- the linear constraints (4.1b) are the inventory balances: for every product and every month, what comes in (previous inventory plus production) is equal to what goes out (demand served plus subsequent inventory); for $t = 1$ the initial inventory s_{i0} is used ($n\,m$ linear constraints);
- the linear constraints (4.1c) ensure that, in every month, the machine hours used by all the products do not exceed those available (m linear constraints);
- the constraints (4.1d) and (4.1e) define the variables of the model.

Line-by-line explanation

Why the balance (4.1b) is the heart of the model. It is the constraint that *links the months to one another*: without it the problem would break up into m independent problems and inventory would make no sense.

What happens if... we remove the balance? The model produces zero (minimum cost!) because nothing forces it to serve demand. If total demand exceeds the cumulated capacity, the model becomes infeasible: in that case either lost demand is allowed with a variable $u_{it} \geq 0$ and a penalty π_{it} in the objective, or capacity has to be revised.

4.3 Worked example by hand

One product, two months

Demand $d_1 = 60$, $d_2 = 100$; capacity $b_1 = b_2 = 80$ units/month ($a = 1$); production cost $c = 10$ €, holding $h = 1$ €/unit/month, zero initial inventory.

- **Step 1 — is it feasible?** Total demand $160 \leq$ total capacity 160: yes, but only by producing at full capacity in both months.
- **Step 2 — balances.** $x_1 = 60 + s_1$ and $s_1 + x_2 = 100$. With $x_2 \leq 80$ we need $s_1 \geq 20$, hence $x_1 \geq 80$: but $x_1 \leq 80$, therefore $x_1 = 80$, $s_1 = 20$, $x_2 = 80$.
- **Step 3 — cost.** $10 \cdot (80 + 80) + 1 \cdot 20 = 1620$ €. The solution is *forced*: both capacity constraints are active.
- **Step 4 — shadow price by hand.** With one more hour in month 2 ($b_2 = 81$): $x_2 = 81$, $s_1 = 19$, $x_1 = 79$: the cost decreases by 1 € (one unit no longer sits in inventory for a month). The dual of the capacity of month 2 is therefore -1 €/unit. With one more hour in month 1, instead, nothing changes (month 1 already produces what is needed): dual 0. This mechanism — *the capacity of a month is worth as much as the holding it saves* — reappears identically in the case study.

4.4 Case study

Three products ($n = 3$) over six months ($m = 6$), with increasing seasonal demand (peak in months 4–5), a single shared resource and these parameters (CSV files `produzione_domanda` and `produzione_capacita` in the folder `data/`):

Product	c_i (€/u)	h_i (€/u/month)	a_i (hours/u)	s_{i0}
1	12.00	0.80	0.9	30
2	18.00	1.20	1.4	20
3	25.00	1.60	2.1	10

month	1	2	3	4	5	6
b_t (hours)	420	420	460	460	460	420
$d_{1,t}$	110	130	150	190	210	160
$d_{2,t}$	70	80	110	140	150	100
$d_{3,t}$	50	55	60	80	95	70

4.5 Implementation

The heart of the script (full version in `lab04_production.py`):

```python
m = gp.Model("produzione_scorte")
x = m.addVars(prodotti, mesi, name="x")    # quantity produced
s = m.addVars(prodotti, mesi, name="s")    # end-of-month inventory

# balance: incoming inventory + production
#          = demand + outgoing inventory
m.addConstrs(
    ((s0[i] if t == 1 else s[i, t-1]) + x[i, t]
     == domanda[i, t] + s[i, t]
     for i in prodotti for t in mesi), name="bilancio")

# shared capacity (keep the constraints for the duals)
v_cap = m.addConstrs(
    (gp.quicksum(a[i] * x[i, t] for i in prodotti)
     <= capacita[t] for t in mesi), name="capacita")

m.setObjective(
    gp.quicksum(c[i]*x[i, t] + h[i]*s[i, t]
                for i in prodotti for t in mesi),
    GRB.MINIMIZE)
m.optimize()
```

4.6 Results

```
Optimal total cost: 32.889.33 EUR

Production plan (units/month):
month       1      2      3      4      5      6
prod.1   80.0  130.0  150.0  190.0  210.0  160.0
prod.2   50.0   80.0  110.0  140.0  150.0  100.0
prod.3   89.5   91.0   81.4   44.3   29.0   64.8

End-of-month inventory:
month       1     2      3     4    5    6
prod.1    0.0   0.0    0.0   0.0  0.0  0.0
prod.2    0.0   0.0    0.0   0.0  0.0  0.0
prod.3   49.5  85.5  106.9  71.2  5.2  0.0
```

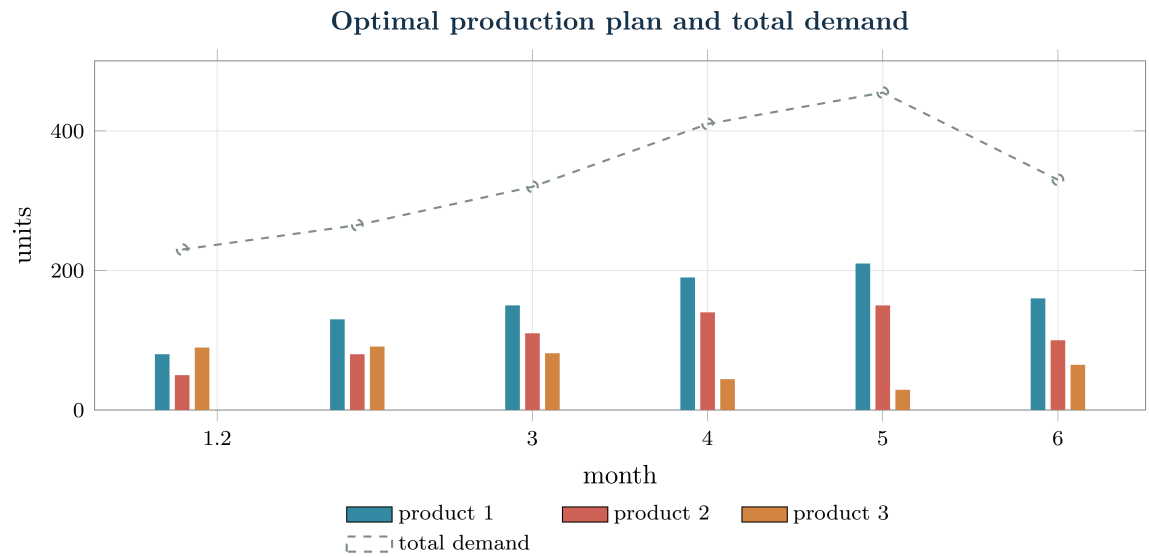


Figure 4.1: Optimal production plan of the LP: bars = units produced of the three products in each month; dashed line = total demand of the month. In the peak months (4–5) total production stays below demand: the difference is covered by the inventory of product 3 accumulated in the first months.

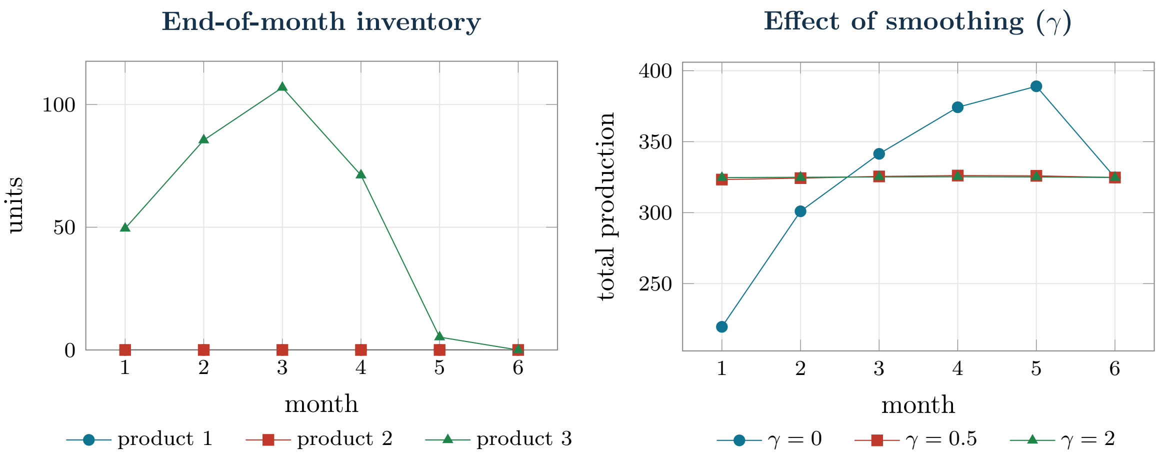


Figure 4.2: On the left: end-of-month inventory by product — only product 3 uses the warehouse (pre-build strategy), and the inventory goes to zero in the final month. On the right: total monthly production for three values of the smoothing weight γ: already with $\gamma = 0.5$ the profile becomes almost flat.

Managerial insight

The solver discovered the *pre-build* strategy all by itself: products 1 and 2 (high holding costs relative to the machine hours they absorb) are produced to demand, whereas product 3 — the most expensive one in machine hours — is produced in advance in months 1–3 and stockpiled (up to 107 units in inventory) in order to free up capacity in the peak months. The inventory goes back to zero in the final month: a positive terminal inventory would be wasted money, and the model knows it.

4.7 Sensitivity analysis

Shadow prices of capacity

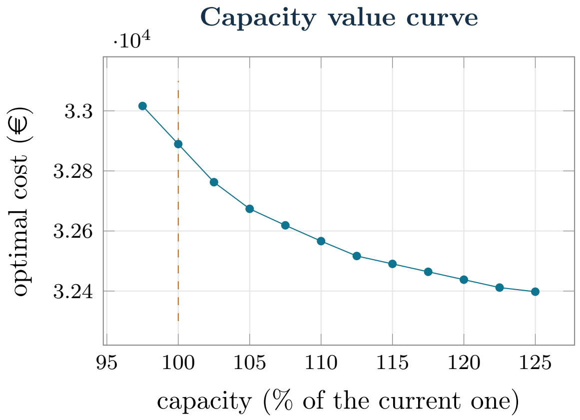
```
month 1: use 330.0/420   | Pi =  0.000 EUR/hour | valid for b_t in [330.0, +inf]
month 2: use 420.0/420   | Pi = -0.762 EUR/hour | valid for b_t in [330.0,
    524.0]
month 3: use 460.0/460   | Pi = -1.524 EUR/hour | valid for b_t in [370.0,
    564.0]
month 4: use 460.0/460   | Pi = -2.286 EUR/hour | valid for b_t in [370.0,
    564.0]
month 5: use 460.0/460   | Pi = -3.048 EUR/hour | valid for b_t in [399.0,
    564.0]
month 6: use 420.0/420   | Pi = -3.810 EUR/hour | valid for b_t in [330.0,
    431.0]

Check: +1 hour in month 6 -> cost from 32889.33 to 32885.52 (delta -3.810)
```

Where the number 0.762 comes from

The duals grow by exactly 0.762 € per month. This is no accident: $0.762 = h_3/a_3 = 1.6/2.1$. One more hour in month t allows us to produce $1/a_3$ units of product 3 *one month later*, saving one month of holding: $h_3 \cdot (1/a_3)$. The dual of month 6 is worth $5 \times 0.762 = 3.81$ because one hour there avoids five months of holding for a unit produced in month 1. When the solver and an elementary economic argument give the same number, the model is probably right.

The value of capacity and the cost of regularity



The curve is **convex and decreasing**: the first extra hours are worth a lot, then less and less. Below 97% of the current capacity the model becomes *infeasible*: demand can no longer be served in full — managerial information as precious as the optimum itself.

The QP variant adds to the objective $\gamma \sum_{t=2}^{m} (X_t - X_{t-1})^2$ with $X_t = \sum_{i=1}^{n} x_{it}$: with $\gamma = 0.5$ the production profile becomes almost flat (maximum variation down from 81 to 1 units/month) at a cost of merely $33.003 - 32.889 = 114$ € (+0.35%).

Figure 4.3: Optimal total cost as a function of the available capacity (in % of the current one; the dashed orange line marks the current capacity): a convex and decreasing curve, interrupted on the left where the problem becomes infeasible.

Managerial insight

Typical recommendation from a lab report: "It is worth buying overtime hours in months 5–6 up to 3–3.8 €/hour below the internal price; a levelled plan costs only 0.35% more and drastically reduces the changes of pace; below 97% of the current capacity full service cannot be guaranteed."

4.8 Exercises

Exercise 4.1 — check

Verify by perturbation the duals of months 3 and 4 (add one hour and reoptimize). Why is the dual of month 1 zero?

Exercise 4.2 — lost demand

Add the variable u_{it} (unserved demand) with penalty $\pi = 40$ €/unit and reduce capacity to 90%: how much demand is lost, of which product, in which months? Interpret the choice of the model.

Exercise 4.3 — maximum profit

Turn the model into a profit maximization with revenues $p = (20, 30, 42)$ € and optional service ($y_{it} \le d_{it}$). Which demand is it worth *not* serving, and why?

Exercise 4.4 — emissions

Every unit of product 3 emits 2.5 kg of CO_2, products 1 and 2 one kg. Add the term $\tau \sum_{i=1}^{n} \sum_{t=1}^{m} e_i \, x_{it}$ to the objective and draw the cost-emissions frontier for $\tau \in [0, 10]$ €/kg.

Exercise 4.5 — safety stock

Impose $s_{it} \ge 0.1 \, d_{i,t+1}$ (10% safety stock). How much does this policy cost? Which capacity constraint becomes the most critical?

Chapter 5

Supply chain with congestion and sustainability

Class: LP / convex NLP

Script: `python/lab05_supplychain.py`
Online page

5.1 Managerial motivation

Products cross a network of plants, hubs and markets. Every leg has a cost, a capacity and a CO_2 footprint; the effective cost grows rapidly when a leg approaches saturation. Three managerial questions: how should the flows be routed at minimum cost? what changes if we penalize congestion? how much must CO_2 be worth for the "clean" routes to become convenient?

This is the everyday problem of whoever manages distribution and logistics: choosing the routes, deciding how much to load on each leg, evaluating investments in capacity and, more and more often, accounting for the carbon footprint of the shipments. The mathematical structure — the network flow — is among the most reusable ones in the whole of operations research.

In this chapter we will learn how to: (1) formulate a minimum-cost flow problem with conservation constraints; (2) model congestion with convex quadratic terms; (3) draw a cost-emissions frontier with an internal CO_2 price; (4) understand why the solutions of LPs "jump" from one vertex to another.

5.2 Guided construction of the model

The problem in words. *We decide* how many units travel on each leg of the network. *The objective* is to minimize the total transport cost (in the variants: plus congestion or CO_2). *The constraints* impose that the flows add up at every node — the plants do not ship more than they produce, the hubs retain nothing, the markets receive exactly their demand — and that no leg exceeds its own capacity.

Data (input of the model)

Symbol	Type	Meaning
$G = (N, A)$	directed graph	nodes N and arcs $A \subseteq N \times N$; we write $\|N\|$ and $\|A\|$ for the number of nodes and of arcs
b_i	$\in \mathbb{Q}$	balance of node $i \in N$: supply if > 0, demand if < 0, transit if $= 0$
u_{ij}	$\in \mathbb{Q}_{>0}$	capacity of arc $(i, j) \in A$ (units)
c_{ij}	$\in \mathbb{Q}_{\geq 0}$	transport cost of one unit on arc (i, j) (€)
e_{ij}	$\in \mathbb{Q}_{\geq 0}$	emissions per unit transported on (i, j) (kg CO_2)

Decision variables

We introduce one non-negative variable for each of the $|A|$ arcs:

$$x_{ij} = \text{units of product sent on arc } (i, j), \qquad \forall (i, j) \in A.$$

Using these variables, an LP model for the problem is the following:

Minimum-cost flow LP model

$$\min \sum_{(i,j) \in A} c_{ij}\, x_{ij} \tag{5.1a}$$

$$\text{subject to} \quad \sum_{j:\,(i,j) \in A} x_{ij} - \sum_{j:\,(j,i) \in A} x_{ji} = b_i, \qquad \forall i \in N, \tag{5.1b}$$

$$x_{ij} \leq u_{ij}, \qquad \forall (i, j) \in A, \tag{5.1c}$$

$$x_{ij} \geq 0, \qquad \forall (i, j) \in A. \tag{5.1d}$$

Description of the objective function and of the constraints of the LP model:

- the linear objective function (5.1a) minimizes the total transport cost: every unit that crosses arc (i, j) pays the unit cost c_{ij};
- the linear constraints (5.1b) are the flow conservations: at every node, what goes out minus what comes in equals the balance b_i — positive at the plants, negative at the markets, zero at the hubs ($|N|$ linear constraints); the sum of the balances over all the nodes must be ≥ 0, otherwise the model is infeasible;
- the linear constraints (5.1c) ensure that no leg carries more than its own capacity ($|A|$ linear constraints);
- the constraints (5.1d) define the variables of the model.

Variants on the objective (the constraints do not change):

$$\underbrace{\textstyle\sum_{(i,j) \in A} \left(c_{ij}\, x_{ij} + \alpha\, c_{ij}\, x_{ij}^2 / u_{ij}\right)}_{\text{congestion (convex)}} \qquad \underbrace{\textstyle\sum_{(i,j) \in A} (c_{ij} + \tau\, e_{ij})\, x_{ij}}_{\text{internal } CO_2 \text{ price}}$$

$$\underbrace{\min z \ : \ x_{ij}/u_{ij} \leq z, \ \ \forall (i, j) \in A}_{\text{minimax utilization}}$$

where $\alpha \in \mathbb{Q}_{\geq 0}$ weighs congestion and $\tau \in \mathbb{Q}_{\geq 0}$ (€/kg) is the internal price of CO_2.

Line-by-line explanation

Congestion. The term $\alpha\, c_{ij}\, x_{ij}^2/u_{ij}$ is convex and grows with the utilization of the arc: it represents overtime, queues at the terminals, insurance premiums. Its effect is to *spread* the flows over several routes instead of saturating the cheapest one.
The price of CO_2. No emissions constraint is needed: it is enough to add them to the cost with the internal price τ. By varying τ one draws the whole cost-emissions frontier ("weight" approach; the "constraint" alternative imposes $\sum_{(i,j)\in A} e_{ij}\, x_{ij} \le \bar{e}$ and reads the dual).

5.3 Worked example by hand

Two competing routes

A market asks for 100 units that can be served by two routes: route 1 with $c_1 = 2$ € and $u_1 = 80$; route 2 with $c_2 = 5$ € and abundant capacity.
Pure LP. The cheap route is filled and the rest overflows onto the other one: $x_1 = 80$, $x_2 = 20$, cost $2 \cdot 80 + 5 \cdot 20 = 260$ €. Shadow price of demand: serving the 101st unit would cost 5 € (route 1 is saturated); shadow price of capacity u_1: +1 unit of capacity saves $5 - 2 = 3$ €.
With congestion ($\alpha = 1$): cost of route 1 $= 2x_1 + 2x_1^2/80$. The marginal cost of route 1, $2 + 4x_1/80$, has to be equalized with that of route 2 (otherwise it is worth shifting flow): $2 + x_1/20 = 5 \Rightarrow x_1 = 60$, $x_2 = 40$. The cheap route *is no longer saturated*: the model leaves a safety margin, exactly as a prudent planner would do.

5.4 Case study

Network with 2 plants (S1, S2), 2 hubs (H1, H2) and 4 markets (M1–M4); 12 arcs with capacities, costs and emissions that are *not* proportional to the costs: the cheap legs are "by road" (polluting), the expensive ones "by rail" (clean). Data in `data/supplychain_archi.csv`; supply $(260, 240)$, total demand 450.

```python
m = gp.Model("supply_chain")
x = m.addVars(A, name="x", ub=U)    # ub directly on the arc
m.addConstrs((x.sum(s, "*") <= offerta[s] for s in offerta))
m.addConstrs((x.sum("*", h) == x.sum(h, "*") for h in hub))
v_dom = m.addConstrs((x.sum("*", k) == domanda[k]
                      for k in domanda), name="domanda")
obj = gp.quicksum((c[a] + tau * e[a]) * x[a] for a in A)
if congestione > 0:    # convex quadratic term
    obj += gp.quicksum(congestione * c[a] * x[a] * x[a] / U[a]
                       for a in A)
m.setObjective(obj, GRB.MINIMIZE)
```

5.5 Results

```
LP minimum cost : cost 3.385.00 EUR  emissions 2.730 kg  max utilization 100%
  saturated arcs: S1->H1, S2->H2, H2->M3
  shadow prices of demand: M1 8.00  M2 9.50  M3 10.00  M4 10.50 EUR/unit
  reduced costs of arcs at zero (threshold = SAObjLow):
    S2->H1 +2.00 (threshold 5.00)   H1->M4 +0.50 (threshold 5.50)
    H2->M1 +4.50 (threshold 1.50)   H2->M2 +1.00 (threshold 3.00)  EUR/unit
Congestion a=1  : cost 3.702.81 EUR  emissions 2.530 kg  max utilization  90%
Minimax         : minimum possible maximum utilization 58.4%
```

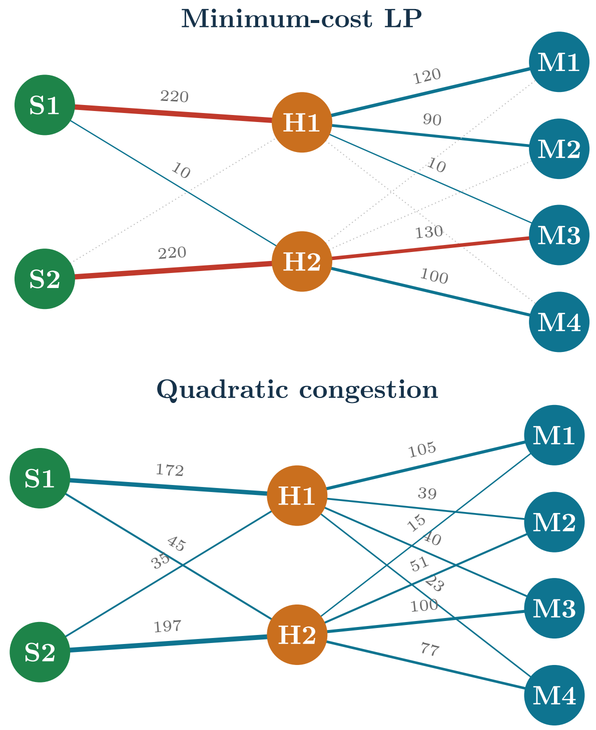


Figure 5.1: The optimal flows on the network (plants S in green, hubs H in orange, markets M in teal; the thickness of the arcs is proportional to the flow, red indicates a saturated arc, the number is the flow in units). Above: minimum-cost LP, which saturates three arcs. Below: model with quadratic congestion, which spreads the flows over several routes and saturates no arc.

Managerial insight

The shadow prices of demand (8–10.50 €/unit) are the *marginal costs of service* of the four markets: this is the number to compare with the commercial margin before accepting new orders, and the correct basis for an internal transfer price. M4 is the most expensive market to serve: +31% with respect to M1.

The reduced costs of the unused arcs, with their range

Four arcs stay at zero and their reduced cost (attribute `x[a].RC`) is their *convenience threshold*: route S2 → H1 (cost 7 €/unit, `RC` = +2) would enter the optimal solution only if its cost dropped below $7 - 2 = 5$ €/unit — the number to bring to the table in a negotiation with the carrier. As always, next to the reduced cost we quote its range of validity `SAObjLow`–`SAObjUp`: for a variable at zero it is [threshold, $+\infty$), that is, the cost of

> the arc may grow as much as it likes (it stays unused) and the threshold itself is `SAObjLow`. Route H1 → M4 is one step away from being used (`RC` = +0.50): a discount of 50 cents is enough. The positive variables all have zero reduced cost.

5.6 Sensitivity analysis: the price of CO_2

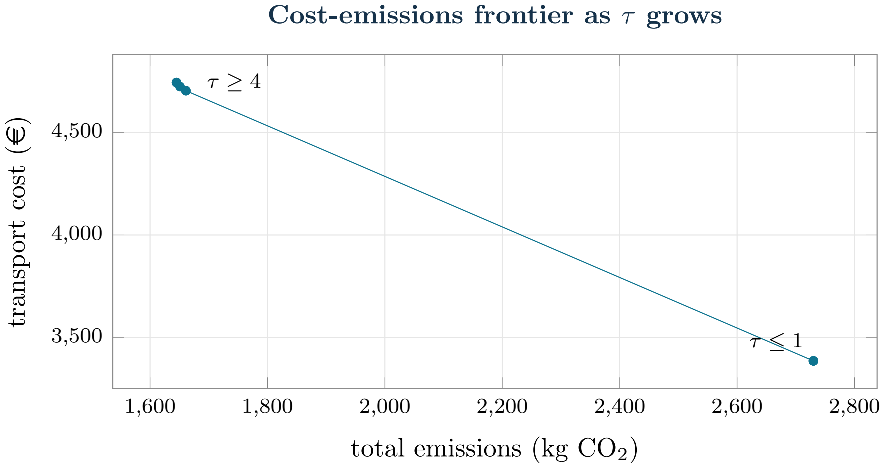


Figure 5.2: Cost-emissions frontier: every point is the optimal solution for an internal price of CO_2 (τ); at the bottom right the "by road" configuration (cheap, 2730 kg), at the top left the "by rail" configuration (expensive, 1645 kg). The frontier proceeds by jumps between vertices of the LP.

```
tau = 0.0 ... 1.0 : cost 3385  emissions 2730   (routes by road)
tau = 1.24        : cost 4025  emissions 2173   (intermediate vertex)
tau = 1.26        : cost 4265  emissions 1981
tau = 1.5         : cost 4705  emissions 1661   ("rail" configuration)
tau = 4.0 and up  : cost 4745  emissions 1645   (stable)
```

Why the frontier proceeds by jumps

The solutions of an LP lie at the *vertices* of the feasible polyhedron: as τ grows the optimum stays put on one vertex, then it *jumps* to the next one. A bisection on the model shows that up to $\tau = 1$ nothing changes, that between $\tau = 1.0$ and $\tau = 1.5$ the network crosses intermediate vertices (at $\tau = 1.24$: 2173 kg) and that the main jump happens exactly at $\tau = 1.25$ €/kg. Managerially: an internal price of CO_2 below the threshold does not change the routes; just above it, it restructures the whole network at once. The model with congestion, being strictly convex, would instead move continuously.

> The "pure" minimax ($\min z$) is degenerate: any routing with utilization $\leq z^*$ is optimal, even an extremely expensive one. In practice one always uses a combination (cost $+\rho \cdot z$) or one constrains z and minimizes the cost.

5.7 Exercises

Exercise 5.1 — check

Verify by perturbation the shadow price of the demand of M4 (raise it to 101) and that of the capacity of arc H2→M3.

Exercise 5.2 — failure of a hub

Reduce by 50% the capacities of the arcs entering and leaving H2: how much does the failure cost? which markets suffer the most? (Compare the shadow prices before/after.)

Exercise 5.3 — the rail threshold

Reproduce with a bisection on τ the threshold $\tau = 1.25$ €/kg found in the text, and list which arcs change flow when crossing it. Compare the threshold with the current ETS price of CO_2.

Exercise 5.4 — emissions constraint

Reformulate with the constraint $\sum_{(i,j)\in A} e_{ij}\, x_{ij} \leq \bar{e}$ for $\bar{e} = 2000$ kg and verify that the dual of the constraint coincides with the "weight" τ that produces the same emissions.

Exercise 5.5 — barrier

Replace the congestion term with the barrier $\alpha\, x_{ij}/(u_{ij} - x_{ij})$ and compare: which formulation keeps the flows further away from saturation?

Chapter 6

The Markowitz portfolio

Class: convex QP

Script: `python/lab06_markowitz.py`
Online page

6.1 Managerial motivation

How much should we invest in each asset in order to balance expected return and risk? The Markowitz model is the most famous QP in history: it minimizes the variance of the portfolio subject to a minimum return. The result is not a single number but an *efficient frontier*: the complete menu of risk-return trade-offs among which the decision maker chooses according to their own degree of risk aversion.

The problem was born in finance (Markowitz, 1952, Nobel prize) but its structure — allocating a budget among risky and correlated alternatives — reappears everywhere: mixes of R&D projects, customer portfolios, even the allocation of production capacity across volatile markets. It is also the natural bridge between optimization and statistics: the data of the model are *estimates*, with all the consequences that this entails.

In this chapter we will learn how to: (1) formulate and solve a convex QP; (2) build and read the efficient frontier; (3) understand the effect of mandate constraints; (4) be wary of the estimates of the expected returns.

6.2 Guided construction of the model

The problem in words. *We decide* the share of capital to invest in each asset. *The objective* is to minimize the risk of the portfolio, measured by the variance of its return. *The constraints*: invest the whole capital (the shares sum to 1), guarantee a minimum expected return and respect the minimum and maximum share of each asset.

Data (input of the model)

Symbol	Type	Meaning
n	$\in \mathbb{Z}_{\geq 1}$	number of assets; the assets are indexed by $i \in \{1, 2, \ldots, n\}$
μ_i	$\in \mathbb{Q}$	expected return of asset i (per year)
q_{ij}	$\in \mathbb{Q}$	covariance between the returns of assets i and j; the matrix $\boldsymbol{Q} = (q_{ij})$ is positive semidefinite ($\boldsymbol{Q} \succeq 0$)
$\bar{r}$	$\in \mathbb{Q}$	minimum required expected return
ℓ_i, u_i	$\in \mathbb{Q},\ 0 \leq \ell_i \leq u_i \leq 1$	minimum and maximum share that can be invested in asset i

Decision variables

We introduce the following n non-negative variables:

$$x_i = \text{share of capital invested in asset } i, \qquad \forall i \in \{1, 2, \ldots, n\}.$$

Using these variables, a QP model for the problem is the following:

Minimum-variance QP model with a minimum return

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij}\, x_i\, x_j \tag{6.1a}$$

$$\text{subject to} \quad \sum_{i=1}^{n} \mu_i\, x_i \geq \bar{r}, \tag{6.1b}$$

$$\sum_{i=1}^{n} x_i = 1, \tag{6.1c}$$

$$x_i \leq u_i, \qquad \forall i \in \{1, 2, \ldots, n\}, \tag{6.1d}$$

$$x_i \geq \ell_i, \qquad \forall i \in \{1, 2, \ldots, n\}. \tag{6.1e}$$

Description of the objective function and of the constraints of the QP model:

- the quadratic objective function (6.1a) minimizes the variance of the return of the portfolio; it contains the covariances q_{ij}, and this is where diversification comes from: combining weakly correlated assets lowers the overall risk below that of the individual components;
- the linear constraint (6.1b) imposes that the expected return of the portfolio reaches the threshold $\bar{r}$: it is the "dial" with which the efficient frontier is traced out, and its multiplier tells us how much variance each additional point of return costs (one linear constraint);
- the linear constraint (6.1c) imposes that the whole capital is invested (one linear constraint);
- the linear constraints (6.1d) are the caps per asset, typical regulatory or mandate constraints (n linear constraints);
- the constraints (6.1e) define the variables of the model (with $\ell_i = 0$: short selling is forbidden).

Equivalent formulations:

$$\max \sum_{i=1}^{n} \mu_i x_i - \lambda \sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij} x_i x_j \qquad \text{(mean-variance)},$$

and $\max \sum_{i=1}^{n} \mu_i x_i$ with variance $\le \bar{\sigma}^2$ (maximum return with bounded risk). By varying $\bar{r}$ (or λ, or $\bar{\sigma}$) the same frontier is traced out.

Line-by-line explanation

Convexity. Every covariance matrix is positive semidefinite, hence the QP is convex and the optimum that is found is a *certified global* one.

6.3 Worked example by hand

Two uncorrelated assets

Volatilities $\sigma_1 = 20\%$, $\sigma_2 = 30\%$, zero correlation. Minimize the variance with $x_1 + x_2 = 1$ (without the return constraint).

- **Step 1.** Substituting $x_2 = 1 - x_1$: $f(x_1) = 0.04\, x_1^2 + 0.09\,(1 - x_1)^2$.
- **Step 2.** $f'(x_1) = 0.08\, x_1 - 0.18\,(1 - x_1) = 0 \;\Rightarrow\; x_1 = \dfrac{0.09}{0.04 + 0.09} = \dfrac{9}{13} \approx 69.2\%$.
- **Step 3.** Variance of the portfolio: $f(9/13) = 0.04 \cdot 0.479 + 0.09 \cdot 0.095 = 0.0277$, that is a volatility of 16.6%: **less risky than either of the two assets**. This is the diversification effect in its purest form: the optimal weights are inversely proportional to the variances.

6.4 Case study

Eight sector ETFs (*Exchange Traded Fund*: listed funds that replicate a basket of securities of a given sector), 60 monthly returns simulated with a single-market-factor model (file `data/markowitz_rendimenti.csv`); $\boldsymbol{\mu}$ and $\boldsymbol{Q}$ are *estimated* from these data, as in practice. Annualized statistics:

```
ENE: mu = 19.12%  vol = 18.65%      SAN: mu =  2.80%  vol = 11.93%
FIN: mu =  8.27%  vol = 19.59%      CON: mu = 11.25%  vol = 11.67%
TEC: mu =  1.58%  vol = 29.27%      UTL: mu = 11.99%  vol =  7.53%
IND: mu =  9.97%  vol = 16.60%      MAT: mu = 20.93%  vol = 23.93%
```

```python
m = gp.Model("markowitz")
x = m.addVars(n, ub=u, name="x")              # shares, 0 <= x_i <= u
m.addConstr(x.sum() == 1, name="budget")
m.addConstr(gp.quicksum(mu[i]*x[i] for i in range(n)) >= r_min)
m.setObjective(gp.quicksum(Q[i, j]*x[i]*x[j]
                            for i in range(n) for j in range(n)),
               GRB.MINIMIZE)
m.optimize()
```

6.5 Results

```
Global minimum variance : return 11.06%, volatility  6.03%
Equally weighted (1/n)  : return 10.74%, volatility 10.13%
Min-variance composition: ENE 5.2%  IND 1.9%  SAN 11.5%  CON 22.3%  UTL 58.3%
```

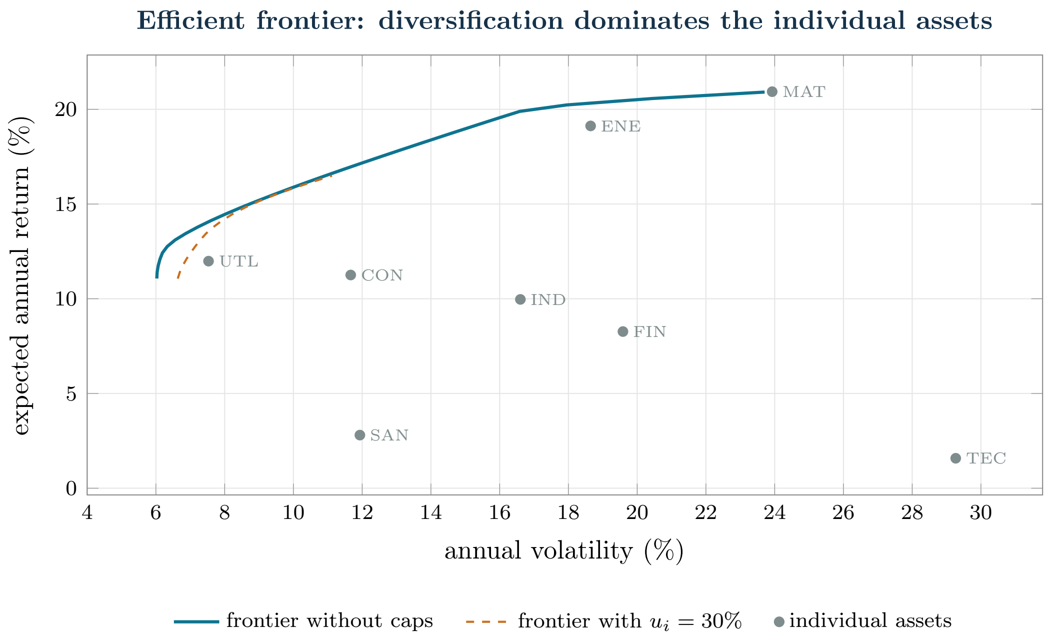


Figure 6.1: Risk-return plane: the eight assets (grey dots), the efficient frontier without caps (teal line), the frontier with a 30% cap per asset (dashed orange), the minimum-variance portfolio (red star) and the equally weighted $1/n$ portfolio (purple diamond), which turns out to be dominated.

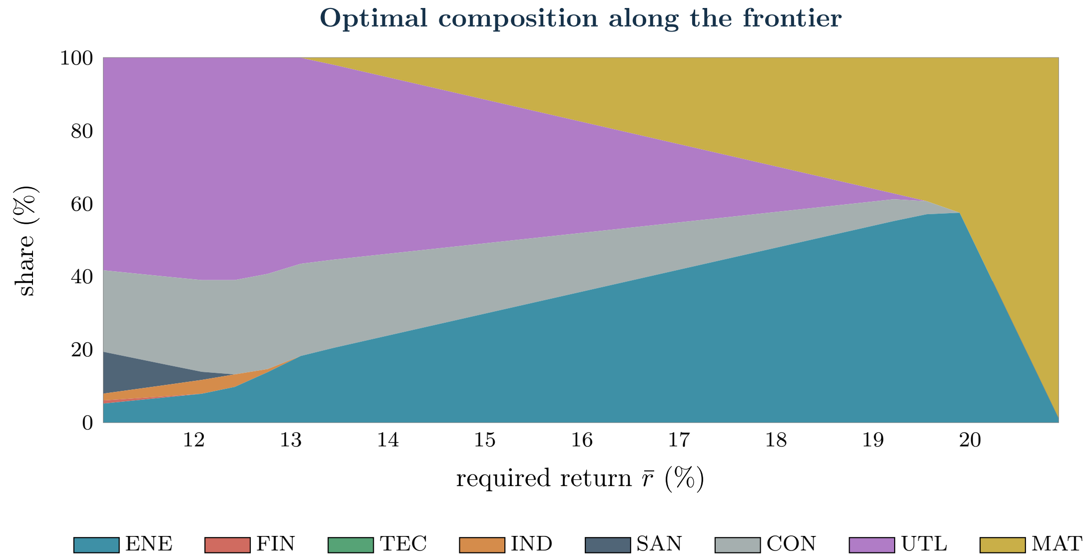


Figure 6.2: Composition of the optimal portfolio (stacked areas, in %) as the required return $\bar{r}$ grows: on the left the defensive mix dominates (UTL, CON, SAN), towards the right the portfolio concentrates progressively on the high-return assets (MAT, ENE) up to the single-asset portfolio at the extreme.

Managerial insight

Three messages from the figure. (1) The minimum-variance portfolio (volatility 6%) is much less risky than the best individual asset (UTL, 7.5%) while still returning 11%. (2) The equally weighted $1/n$ portfolio — the "I know nothing" strategy — is clearly dominated: the same region of return but almost twice the volatility. (3) The cap $u_i = 30\%$ cuts off the upper part of the frontier: the cost of the mandate constraints can be *seen* as the distance between the two curves.

6.6 Sensitivity analysis

```
r_min =  6%: vol 6.03%   (constraint NOT active: same as min variance)
r_min =  8%: vol 6.03%   (same)
r_min = 10%: vol 6.03%   (same)
r_min = 12%: vol 6.10%   d(variance)/d(r_min) ~ 0.0221
```

A constraint that costs nothing

Up to $\bar{r} = 11\%$ the return constraint is *inactive*: the minimum-variance portfolio already returns 11.06%, hence asking for "at least 8%" costs nothing and the multiplier is zero. Only beyond 11.06% does the constraint bite, and every additional point of return is paid for in variance (multiplier ≈ 0.022 at $\bar{r} = 12\%$). Recognizing inactive constraints avoids "paying" for constraints that are free.

In the simulated data the asset TEC has a true α of 11% per year, but over 60 months the *estimated* return is 1.6%: the noise dominates. This is the most important practical lesson of Markowitz: **the estimates of the expected returns are far more unstable than those of the covariances**, and optimized portfolios chase the estimation errors. Remedies used in practice: the constraints u_i, shrinkage of the estimates, or optimizing risk only (minimum variance).

6.7 Exercises

Exercise 6.1 — check

Verify by hand the example with two assets and $\rho = 0.5$: the formula becomes $x_1 = (\sigma_2^2 - \rho\sigma_1\sigma_2)/(\sigma_1^2 + \sigma_2^2 - 2\rho\sigma_1\sigma_2)$. Is diversification still worthwhile?

Exercise 6.2 — frontier

Recompute the frontier with $u_i = 20\%$: how much maximum return is lost? Draw the three frontiers ($u = 1$, 0.3, 0.2) in the same plot.

Exercise 6.3 — stability of the estimates

Run the script again with a different random seed (`default_rng(1)`) and compare the optimal compositions for the same $\bar{r}$: how much do they change? Which operational implication do you draw from this?

Exercise 6.4 — tracking error

The benchmark is the equally weighted portfolio $x_i^B = 1/8$. Minimize the tracking error $\sum_{i=1}^n \sum_{j=1}^n q_{ij}(x_i - x_i^B)(x_j - x_j^B)$ with expected return $\geq 12\%$: how far must one depart from the benchmark?

Exercise 6.5 — transaction costs

Starting from the equally weighted portfolio, add the quadratic cost $\kappa \sum_{i=1}^n (x_i - 1/8)^2$ with $\kappa = 0.5$: how does the composition change with respect to the "clean" optimum?

Chapter 7

Pricing and revenue management

Class: concave / non-convex NLP

Script: `python/lab07_pricing.py`
Online page

7.1 Managerial motivation

Which price maximizes the profit of a concert, of a hotel, of a subscription, when demand decreases as the price grows? Here the price is not a datum but a *variable*: demand becomes endogenous and the profit $p \cdot q$ introduces a bilinear term. This is the chapter in which the laboratory meets non-convexity for the first time — and learns how to handle it.

Setting the price is perhaps the business decision with the highest impact on the margin: one percentage point of price is often worth more than many points of volume. Revenue management — born in the airlines and today standard in hotels, events and rentals — adds the capacity constraint: the seats left unsold tonight cannot be recovered tomorrow.

In this chapter we will learn how to: (1) model demand as a function of the price and understand when profit is concave; (2) handle a non-convex bilinear objective; (3) evaluate capacity correctly (an additional seat is *not* worth the full margin); (4) recognize corner optima, where no derivative vanishes.

7.2 Guided construction of the model

The problem in words. *We decide* the selling price and the quantity to sell. *The objective* is to maximize the profit, that is revenue minus variable costs. *The constraints*: we cannot sell more than the market asks for at that price, nor more than the available capacity.

Data (input of the model)

Symbol	Type	Meaning
$d(\cdot)$	function $\mathbb{Q}_{\geq 0} \to \mathbb{Q}_{\geq 0}$	expected demand at price p, decreasing in p
c	$\in \mathbb{Q}_{\geq 0}$	unit marginal cost (€)
k	$\in \mathbb{Q}_{>0}$	available capacity (seats, rooms, units)

Decision variables

We introduce the following 2 non-negative variables:

$$\begin{cases} p = \text{selling price decided by the firm (€)} \\ q = \text{quantity sold (units)} \end{cases}$$

Using these variables, a pricing model for the problem is the following:

Pricing model with a capacity constraint

$$\max \quad p\,q - c\,q \tag{7.1a}$$

$$\text{subject to} \quad q \le d(p), \tag{7.1b}$$

$$q \le k, \tag{7.1c}$$

$$p \ge 0, \tag{7.1d}$$

$$q \ge 0. \tag{7.1e}$$

Description of the objective function and of the constraints of the model:

- the objective function (7.1a) maximizes the profit $p\,q - c\,q = (p-c)\,q$, revenue minus variable cost; it contains the product of the two variables p and q (a *bilinear* term, non-convex), and this is where all the difficulty of the chapter comes from;
- constraint (7.1b) imposes that we do not sell more than the market asks for at price p (one constraint, linear if d is linear);
- the linear constraint (7.1c) imposes that we do not sell more than the available capacity; when it is active, it is this constraint that determines the optimal price (one linear constraint);
- constraints (7.1d) and (7.1e) define the variables of the model.

Demand functions used in the case study, with their respective data (all positive rationals): linear $d(p) = a - b\,p$; constant elasticity $d(p) = \theta\,p^{-\varepsilon}$ with $\varepsilon > 1$; logistic $d(p) = m/\big(1 + e^{-\alpha+\beta p}\big)$.

Line-by-line explanation

Why $q \le d(p)$ and not $q = d(p)$? At the optimum the inequality (7.1b) is active by itself (selling less than what is possible at that price is never worthwhile if $p > c$), but writing it as $\le$ keeps the feasible region simpler and the model feasible even when $d(p) > k$.
Where is the non-convexity? In the product $p \cdot q$ of the objective. With linear demand, substituting $q = a - b\,p$, the profit $(p-c)(a-b\,p)$ is a concave parabola: the *reduced* problem is easy. Gurobi nevertheless solves the bilinear version to global optimality (`NonConvex=2`); for general non-linear demands (constant elasticity, logistic) the function constraints of Gurobi are enough (`addGenConstrPow`, `addGenConstrExp`): same solver, still a global optimum.

7.3 Worked example by hand

Linear demand, with and without capacity

Concert: $d(p) = 1200 - 5p$ (hence $a = 1200$, $b = 5$), marginal cost $c = 20$ €, theatre with $k = 400$ seats.

- **Step 1 — without capacity.** Profit $\Pi(p) = (p - 20)(1200 - 5p)$; $\Pi'(p) = 1200 - 10p + 100 = 0$ gives $p^\circ = \dfrac{a/b + c}{2} = \dfrac{240 + 20}{2} = 130$ €, with $q^\circ = 1200 - 650 = 550$ tickets.
- **Step 2 — capacity bites.** $q^\circ = 550 > 400$: the constraint is active, hence exactly k units are sold and the price that empties the theatre is $p^* = (a - k)/b = (1200 - 400)/5 = 160$ €. Profit: $(160 - 20) \cdot 400 = 56.000$ €.
- **Step 3 — how much is an additional seat worth?** As k grows, the optimal price falls by $1/b$: $\dfrac{d\Pi}{dk} = \underbrace{(p^* - c)}_{140} + k \underbrace{\dfrac{dp^*}{dk}}_{-1/5} = 140 - 80 = 60$ €. An additional seat is *not* worth the full margin (140 €): in order to fill it the price must be lowered for everybody. A classic mistake in the evaluation of capacity investments.

7.4 Case study and implementation

The same data as in the example (this is the rare case in which the realistic instance is already small), plus the constant-elasticity variant ($\theta = 6 \cdot 10^6$, $\varepsilon = 2.2$), the logistic one ($m = 900$, $\alpha = 6$, $\beta = 0.045$) and the multi-product one.

```python
m = gp.Model("pricing_lineare")
m.Params.NonConvex = 2              # bilinear objective p*q
p = m.addVar(ub=a/b, name="p")
q = m.addVar(name="q")
m.addConstr(q <= a - b*p, name="domanda")
v_cap = m.addConstr(q <= K, name="capienza")
m.setObjective(p*q - c*q, GRB.MAXIMIZE)
m.optimize()
# no duals in non-convex QPs: sensitivity by perturbation
v_cap.RHS = K + 1; m.optimize()
```

```
Gurobi:  p* = 160.00 EUR, q* = 400, profit = 56.000.00 EUR
Marginal value of a seat: 59.80 EUR (theory: 60.00)
Constant elasticity: p* = 79.11 EUR  (optimum at the point where d(p) = k)
Logistic           : p* = 138.29 EUR, profit = 47.316.83 EUR
```

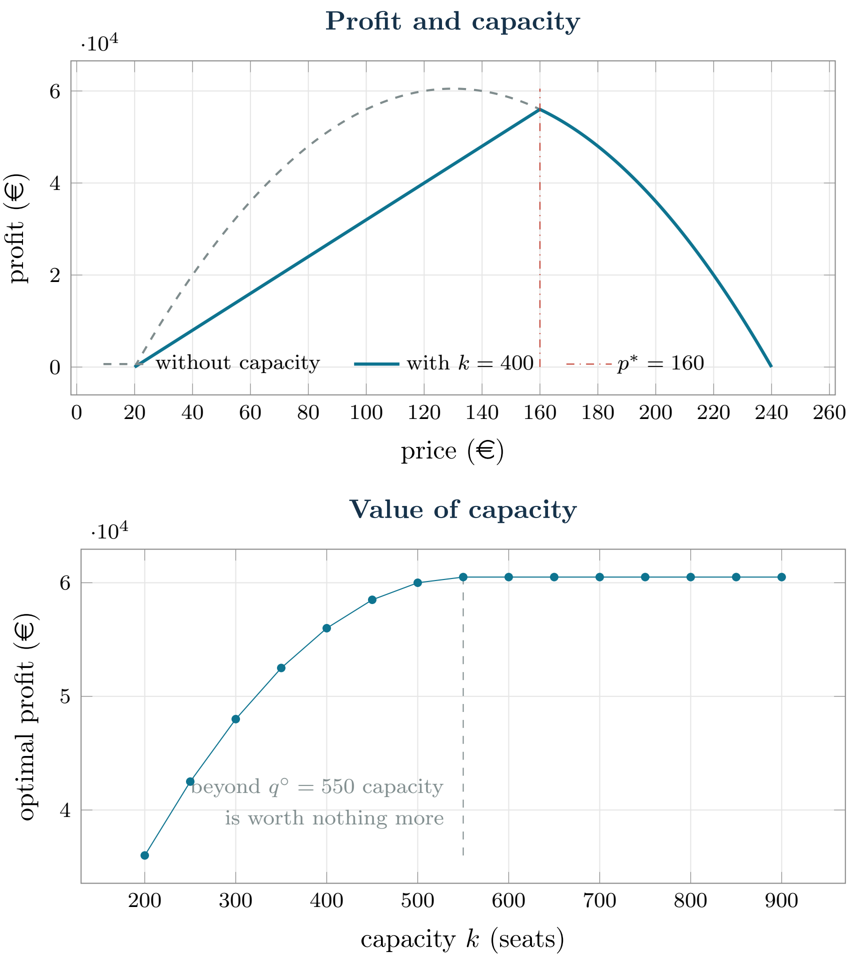


Figure 7.1: On the left: profit as a function of the price for linear demand, without the constraint (dashed grey parabola) and with capacity $k = 400$ (teal line): the constraint cuts the parabola and moves the optimum to $p^* = 160$ €. On the right: optimal profit as capacity grows; beyond $q^\circ = 550$ seats the curve becomes flat and additional capacity is worth nothing more.

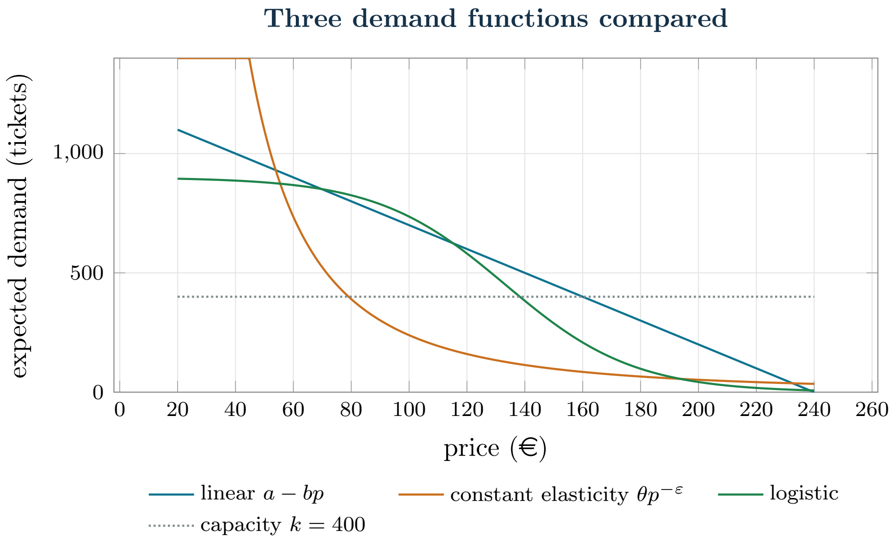


Figure 7.2: The three demand functions of the case study compared (expected tickets as a function of the price) with capacity $k = 400$ (dotted line): the linear one cuts the axis, the constant-elasticity one has a long tail, the logistic one describes the gradual transition between high and low demand.

The optimum at a corner point

With constant elasticity the *unconstrained* optimum would be $p = c\,\varepsilon/(\varepsilon - 1) = 36.67$ €, but at that price demand would exceed capacity by far. For p below 79.11 € we sell k anyway (so it is worthwhile to raise the price); above it, the profit $(p - c)\,\theta p^{-\varepsilon}$ decreases. The optimum $p^* = 79.11 = (\theta/k)^{1/\varepsilon}$ lies at the *corner point* where $d(p) = k$: it makes no derivative vanish. The moral: never look for the optimum among stationary points only when there are constraints.

7.5 Multi-product version

Two ticket categories with substitution (raising the price of the stalls pushes part of the demand towards the balcony): $d_1(p_1, p_2) = 500 - 2p_1 + 0.6\,p_2$, $d_2(p_1, p_2) = 900 + 0.8\,p_1 - 4p_2$, costs $(30, 15)$ €, capacities $(150, 300)$ seats.

```
stalls   : p1* = 234.04 EUR   q1* = 150/150 (full)
balcony  : p2* = 196.81 EUR   q2* = 300/300 (full)
total profit: 85.148.94 EUR
```

Managerial insight

Both categories fill up, but the prices are not “the ones that empty” each hall taken on its own: the model exploits substitution, keeping the stalls expensive in order to push demand towards the balcony. With substitute products prices must be decided *jointly*: optimizing them one at a time leaves money on the table.

With general demand functions the profit may fail to be concave: a local solver may stop at a local optimum. Practice of the laboratory: study the domain and the derivatives of the function *before* trusting the solver, try several starting points, and always plot the profit over the relevant interval.

7.6 Exercises

Exercise 7.1 — check

Verify with Gurobi the marginal value of a seat for $k = 300$ (from the table: 100 €) and obtain it also with the formula $p^* - c - k/b$.

Exercise 7.2 — marginal cost

How do p^*, q^* and the value of a seat change if the marginal cost rises to $c = 60$ €? Redo the three steps of the example by hand.

Exercise 7.3 — reference price

Add the reputational penalty $-\gamma\,(p-120)^2$ with $\gamma = 2$: which price balances profit and reputation? Does the problem remain concave?

Exercise 7.4 — elasticity

For the constant-elasticity demand, plot p^* as a function of $\varepsilon \in [1.3; 3]$ with $k = 400$: for which ε does the capacity constraint remain active?

Exercise 7.5 — multi-product

In the two-category model, by how much does the profit increase with 50 additional stalls seats? And with 50 additional balcony seats? Which expansion would you recommend, at equal cost?

Chapter 8

Advertising budget allocation

Class: convex NLP **Script:** python/lab08_budget.py
Online page

8.1 Managerial motivation

How should a campaign of 100.000 € be split among social, search, TV and influencers, when every channel shows diminishing marginal returns? The first euros spent on a channel yield a lot, the following ones less and less: the response is a concave function of the spend, and the optimum follows an elegant economic rule that this chapter proves and verifies numerically.

The allocation of a budget among uses with diminishing returns is a universal scheme: marketing over several channels, funds among projects, time among activities, fertilizer among fields. Theory says something strong and verifiable: at the optimum the marginal returns are equal to each other — and this is exactly what the solver will return, digit by digit.

In this chapter we will learn how to: (1) model diminishing returns with concave functions; (2) solve convex NLPs with the function constraints of Gurobi; (3) verify the KKT conditions numerically; (4) build the budget value curve and use λ to negotiate the budget.

8.2 Guided construction of the model

The problem in words. *We decide* how much to spend on each advertising channel. *The objective* is to maximize the total expected response of the campaign. *The constraints*: the overall spend cannot exceed the budget, and each channel has a maximum useful investment.

Data (input of the model)

Symbol	Type	Meaning
n	$\in \mathbb{Z}_{\geq 1}$	number of advertising channels; the channels are indexed by $i \in \{1, 2, \dots, n\}$
b	$\in \mathbb{Q}_{>0}$	total budget (thousands of €)
$r_i(\cdot)$	increasing concave function	expected response of channel i as a function of the spend; in the case study $r_i(x) = a_i \log(1 + k_i\, x)$ with given $a_i, k_i \in \mathbb{Q}_{>0}$
u_i	$\in \mathbb{Q}_{>0}$	maximum useful or allowed investment in channel i

Decision variables

We introduce the following n non-negative variables:

$$x_i = \text{spend on channel } i \text{ (thousands of €)}, \qquad \forall i \in \{1, 2, \ldots, n\}.$$

Using these variables, a convex NLP model for the problem is the following:

Allocation model with concave response

$$\max \quad \sum_{i=1}^{n} r_i(x_i) \tag{8.1a}$$

$$\text{subject to} \quad \sum_{i=1}^{n} x_i \leq b, \tag{8.1b}$$

$$x_i \leq u_i, \qquad \forall i \in \{1, 2, \ldots, n\}, \tag{8.1c}$$

$$x_i \geq 0, \qquad \forall i \in \{1, 2, \ldots, n\}. \tag{8.1d}$$

Description of the objective function and of the constraints of the model:

- the concave objective function (8.1a) maximizes the total response of the campaign; it is separable by channel and is a sum of concave functions, hence the problem has a certifiable global optimum;
- the linear constraint (8.1b) imposes that the overall spend does not exceed the budget: it is the only constraint that links the channels to each other — without it every channel would go to its own cap (one linear constraint);
- the linear constraints (8.1c) are the caps per channel: the limit beyond which the channel absorbs no further useful spend (n linear constraints);
- the constraints (8.1d) define the variables of the model.

The economic condition at the optimum

By the KKT conditions, at the optimum there exists $\lambda \geq 0$ (shadow price of the budget) such that, for every channel that is not stuck at the bounds 0 or u_i:

$$r_i'(x_i^*) = \lambda.$$

In words: **the last euro invested yields the same in all the active channels**. If this were not the case, moving one euro from the channel with the low marginal return to the one with the high marginal return would improve the response: we would not be at the optimum. The channels at the cap u_i may have a marginal return $> \lambda$ (we would like to invest more in them but we cannot); those at zero have an initial marginal return $< \lambda$ (they are not worth even the first euro).

8.3 Worked example by hand

Two logarithmic channels

$r_1(x) = 100\log(1+0.2x)$, $r_2(x) = 60\log(1+0.1x)$, budget $b = 50$ (thousands of €), without caps.

- **Step 1 — condition of equal marginal returns.**

$$\frac{100 \cdot 0.2}{1+0.2x_1} = \frac{60 \cdot 0.1}{1+0.1x_2} \quad\Longrightarrow\quad \frac{20}{1+0.2x_1} = \frac{6}{1+0.1x_2}.$$

- **Step 2 — budget.** $x_2 = 50 - x_1$. Substituting: $20\,(1+0.1(50-x_1)) = 6\,(1+0.2x_1)$, that is $20\,(6-0.1x_1) = 6+1.2x_1 \Rightarrow 120 - 2x_1 = 6 + 1.2x_1 \Rightarrow x_1 = \dfrac{114}{3.2} = 35.6$, $x_2 = 14.4$.

- **Step 3 — shadow price.** $\lambda = 20/(1+0.2 \cdot 35.6) = 2.46$: one more euro of budget would yield ≈ 2.46 units of response — the same number, whichever channel it is fed into.

8.4 Case study and implementation

Four channels, $r_i(x) = a_i \log(1+k_i x)$, budget $b = 100$ (thousands of €); data in `data/budget_canali.csv`. The problem is concave and we solve it with the function constraints of Gurobi: an auxiliary variable $z_i = \log(g_i)$ with $g_i = 1 + k_i x_i$, so that the objective becomes linear in the z_i (certified global optimum):

```python
m.Params.FuncNonlinear = 1          # log handled exactly
x = m.addVars(4, ub=u)
g = m.addVars(4, lb=1.0)            # g_i = 1 + k_i x_i
z = m.addVars(4, lb=-GRB.INFINITY)
for i in range(4):
    m.addConstr(g[i] == 1 + k[i] * x[i])
    m.addGenConstrLog(g[i], z[i])
m.addConstr(x.sum() <= B)
m.setObjective(gp.quicksum(a[i] * z[i] for i in range(4)),
                   GRB.MAXIMIZE)
```

8.5 Results and verification of the KKT conditions

```
Total response: 1.449.8 (thousands of useful contacts)
    channel |  spend |   cap | response |  marginal
     social |   24.3 |    60 |    385.3 |   8.2693
     search |   34.8 |    80 |    539.5 |   8.2693
         TV |   22.9 |   120 |    235.2 |   8.2693
 influencer |   18.0 |    35 |    289.8 |   8.2693

Check: +1000 EUR of budget -> response +8.244 ~ lambda
```

Managerial insight

The theoretical condition can be seen in the numbers: the four marginal returns coincide to the fourth digit ($\lambda = 8.2693$), and the actual increase in response obtained with one thousand euros more (8.244) confirms the reading of the multiplier. A managerial note: TV receives *less* than social despite its higher cap — what matters is not the size of the channel but the speed at which it saturates (k_i).

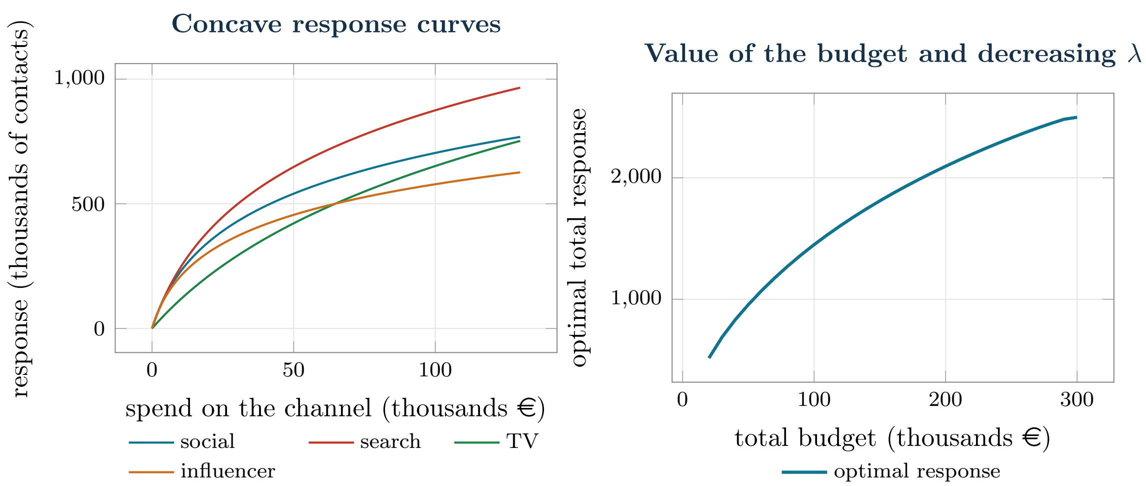


Figure 8.1: On the left: the response curves $r_i(x) = a_i \log(1 + k_i x)$ of the four channels — concave, with different saturation speeds. On the right: the optimal total response as the budget grows (value curve), concave: every tranche of budget yields less than the previous one.

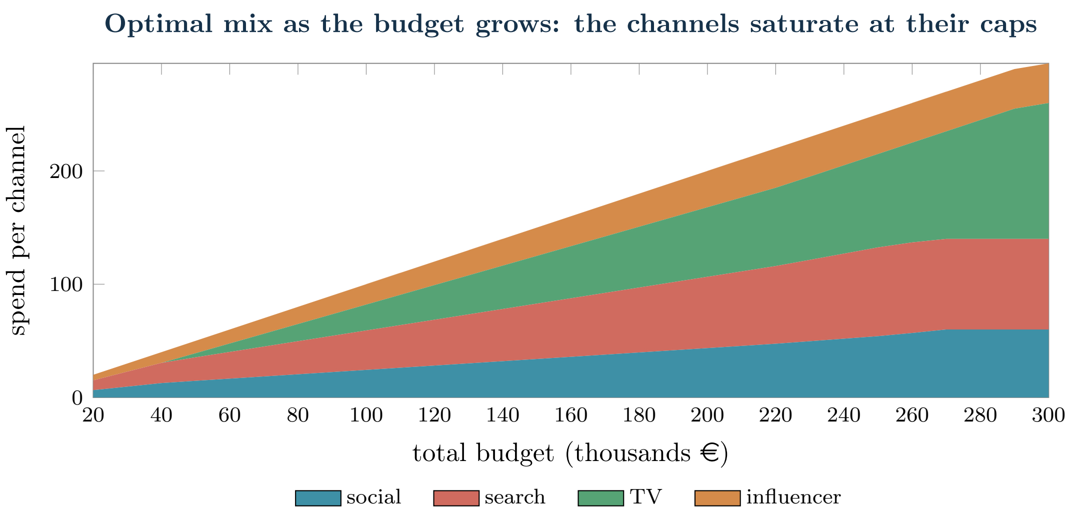


Figure 8.2: Optimal spend mix (stacked areas, thousands of €) as the total budget grows: the channels enter and grow together while keeping their marginal returns equal; the influencer channel saturates first at its cap (35), followed by the others.

8.6 Sensitivity analysis

```
b =   20: response   515.9   lambda = 18.971
b =   60: response 1.070.4   lambda = 10.909
```

```
b = 100: response 1.449.8   lambda =  8.244
b = 180: response 1.988.2   lambda =  5.538
b = 260: response 2.370.2   lambda =  4.049
b = 300: response 2.498.1   lambda =  0.000  (all channels at their caps)
```

Line-by-line explanation

The budget value curve is concave: λ falls from 19 to 4 as the budget grows. At $b = 300$ all the channels are at their caps ($60+80+120+35 = 295$): the budget stops being the scarce resource and λ collapses to zero. For a CFO: the *curve* of λ is the quantitative argument for deciding how much budget to grant to marketing — it is funded as long as λ exceeds the value of one euro invested elsewhere.

8.7 Exercises

Exercise 8.1 — check

Solve the example by hand with $b = 80$: where does the additional budget end up? How does λ change?

Exercise 8.2 — exponential response

Replace the response of TV with $r(x) = 520\,(1 - e^{-0.025x})$ and repeat the optimization: does the mix change? The function has a natural ceiling: is the bound u_i still needed?

Exercise 8.3 — excluded channel

Add a fifth channel with initial marginal return $a_5k_5 = 5 < \lambda$: verify that the optimum leaves it at zero and interpret its "reduced cost" $a_5k_5 - \lambda$.

Exercise 8.4 — fair coverage

The budget serves three customer segments with different coverage per channel. Maximize the coverage of the worst segment ($\max z$ with $z \leq \text{coverage}_g$): how much does fairness cost in terms of total response?

Chapter 9

Continuous location of a service

Class: convex NLP

Script: python/lab09_location.py
Online page

9.1 Managerial motivation

Where should a fast-charging station, a logistics micro-hub, a health-care post be placed in order to be "close" to demand? The answer depends on what close means: *mean* distance (efficiency), *maximum* distance (equity towards the most unlucky user) or *squared* distance (which leads to the centroid). Three objectives, three different points on the map: geometry makes the trade-off visible as in no other model.

Location decisions are among the most expensive ones to correct: a hub in the wrong place stays in the wrong place for years. The continuous version of the problem (free coordinates on the plane) is the ideal laboratory for a theme that reappears in all public and private decisions: the conflict between *efficiency* (serving well on average) and *equity* (leaving nobody behind).

In this chapter we will learn how to: (1) formulate three location objectives and understand when they coincide; (2) reformulate Euclidean distances with conic constraints and solve them with Gurobi from good starting points; (3) handle convex geographic constraints; (4) build an efficiency-equity frontier with the ε-constraint method.

9.2 Guided construction of the model

The problem in words. *We decide* the coordinates at which the new facility is placed. *The objective* — and this is the real managerial choice — may be to minimize the mean distance weighted by the users, or the distance of the farthest user, or the squared distance. *The constraints*, when present, delimit the feasible area (a rectangular zone, a maximum radius from an infrastructure).

Data (input of the model)

Symbol	Type	Meaning
n	$\in \mathbb{Z}_{\geq 1}$	number of demand points (districts), indexed by $i \in \{1, 2, \ldots, n\}$
(a_i, b_i)	$\in \mathbb{Q}^2$	coordinates of district i (km)
w_i	$\in \mathbb{Q}_{\geq 0}$	weight of district i: population, demand or priority
(a_0, b_0), r	$\in \mathbb{Q}^2$, $\in \mathbb{Q}_{>0}$	centre and radius of the possible geographic constraint (maximum distance from an infrastructure)

Decision variables

We introduce the following 2 free variables (they may take any real value) and, for the minimax version, a non-negative auxiliary variable:

$$\begin{cases} x = \text{east coordinate of the new facility (km)} \\ y = \text{north coordinate of the new facility (km)} \\ z = \text{maximum distance from the farthest district (minimax only)} \end{cases}$$

Using these variables, the models for the three classical objectives are the following:

Weber model (weighted mean distance)

$$\min \quad \sum_{i=1}^{n} w_i \sqrt{(x-a_i)^2 + (y-b_i)^2} \tag{9.1a}$$

$$\text{subject to} \quad x \gtreqless 0, \tag{9.1b}$$

$$y \gtreqless 0. \tag{9.1c}$$

Minimax model (maximum distance)

$$\min \quad z \tag{9.2a}$$

$$\text{subject to} \quad \sqrt{(x-a_i)^2 + (y-b_i)^2} \leq z, \qquad \forall i \in \{1, 2, \ldots, n\}, \tag{9.2b}$$

$$x \gtreqless 0, \tag{9.2c}$$

$$y \gtreqless 0, \tag{9.2d}$$

$$z \geq 0. \tag{9.2e}$$

Description of the objective functions and of the constraints of the two models:

- the convex objective function (9.1a) minimizes the total distance weighted by the districts (efficiency); constraints (9.1b) and (9.1c) define the variables, which are free;
- the objective function (9.2a) minimizes the distance of the farthest district (equity); the convex constraints (9.2b) impose that every district is at most at distance z from the facility (n constraints); constraints (9.2c)–(9.2e) define the variables;

- a third variant, **quadratic**: minimize $\sum_{i=1}^{n} w_i\big[(x-a_i)^2+(y-b_i)^2\big]$, which has a closed-form solution (the weighted centroid).

Typical convex geographic constraints, to be added to any variant: rectangular zone $x^L \le x \le x^U$, $y^L \le y \le y^U$; maximum distance from an infrastructure $(x-a_0)^2+(y-b_0)^2 \le r^2$.

Line-by-line explanation

Quadratic ⇒ centroid. Setting the gradient to zero: $x^* = \sum_{i=1}^{n} w_i\, a_i \big/ \sum_{i=1}^{n} w_i$ (and analogously for y): the only case with a closed formula, useful as a starting point for the others.
Weber. It has no closed form; the function is not differentiable at the points (a_i, b_i). At the optimum the weighted unit "pulls" of the districts cancel out: $\sum_{i=1}^{n} w_i\, \boldsymbol{u}_i = \mathbf{0}$, where $\boldsymbol{u}_i$ is the unit vector from the facility towards district i.
Minimax. The optimum is the centre of the *smallest enclosing circle* of all the points: it depends only on the extreme districts and ignores the weights — this is why it protects the outskirts.

9.3 Worked example by hand

Three customers on a road

Customers at km 0, 4 and 10 with weights 1, 1 and 3.
Weber in 1D = weighted median. The cost is $f(x) = |x| + |x-4| + 3|x-10|$. Moving by δ to the left from $x = 10$, the cost of the first two customers decreases by 2δ but that of the third increases by 3δ: it gets worse. From the right, symmetrically, it gets worse again. Optimum: $x^* = 10$, the point that leaves *at least half of the total weight* ($5/2 = 2.5$) on both sides — the weighted median.
Centroid. $x = (1\cdot 0 + 1\cdot 4 + 3\cdot 10)/5 = 6.8$: sensitive to *how far* the points are, not only to which side they are on.
Minimax. Midpoint of the extremes: $(0+10)/2 = 5$, weights ignored.
Three reasonable objectives, three different answers: 10, 6.8, 5. The choice of the objective IS the managerial decision.

9.4 Case study

Twelve districts with weights between 5 and 25 thousand inhabitants; the data are in the file `localizzazione_quartieri.csv` of the folder `data/`. Technical constraint: the station must lie within $r = 2$ km of the substation at $(7, 6)$.

The modelling trick: a variable $d_k \ge$ the Euclidean distance from district k, imposed with the (convex) conic constraint $d_x^2 + d_y^2 \le d_k^2$; Weber minimizes $\sum_k w_k d_k$, the minimax a variable z with $d_k \le z$.

```
d = m.addVars(n)                          # d_k >= distance from district k
for k in range(n):
    m.addConstr(dx[k] == px - coord[k, 0])
    m.addConstr(dy[k] == py - coord[k, 1])
    m.addQConstr(dx[k]**2 + dy[k]**2 <= d[k]*d[k])    # cone
# Weber:   min sum w_k d_k
```

```
# minimax: min z  with  d_k <= z
# geographic constraint: (px-7)^2 + (py-6)^2 <= R^2
```

9.5 Results

```
Weighted centroid : (4.897, 5.397)
Weber             : (4.886, 5.310)  mean dist. 3.344 km  max 6.314 km
Minimax           : (5.496, 5.029)  mean dist. 3.391 km  max 5.680 km
Constrained Weber : (5.083, 5.429)  cost +0.2% (constraint active, optimum on
    boundary)
```

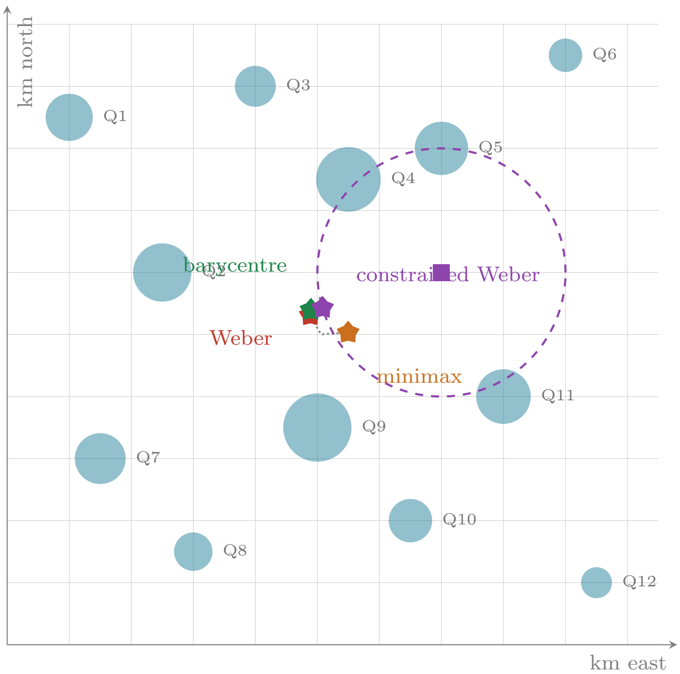


Figure 9.1: Map of the city: the teal circles are the 12 districts (area proportional to the population); the stars indicate the four optimal locations (Weber in red, minimax in orange, centroid in green, constrained Weber in purple); the dashed purple circle is the constraint of 2 km from the electrical substation; the dotted line is the trajectory of the compromise solutions between minimax and Weber.

> **Managerial insight**
>
> Weber and the centroid almost coincide here (the weights are evenly distributed), but the minimax moves more than half a kilometre towards the south-east in order to protect the peripheral districts. The constraint of the substation costs very little (+0.2% of total distance): discovering that a feared constraint is almost free is a managerial result as important as the optimum itself.

9.6 Sensitivity analysis: the efficiency-equity frontier

We minimize the mean distance while imposing a cap D on the maximum distance (the ε-constraint method), and we vary D between the two extremes:

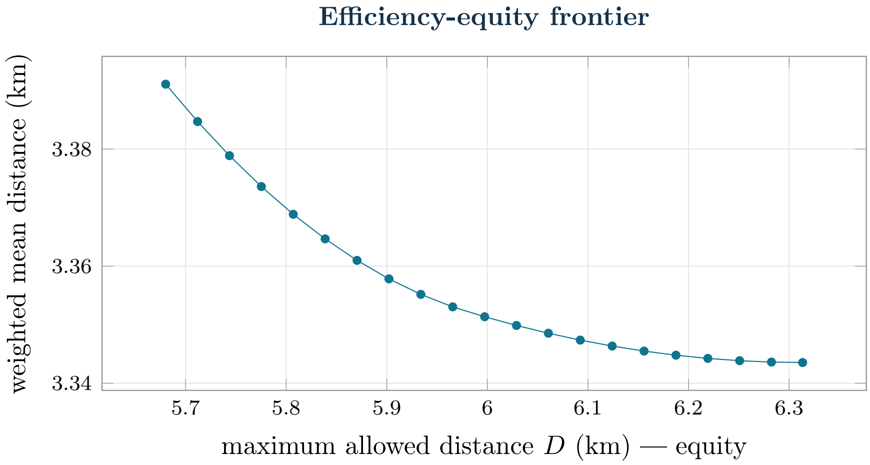


Figure 9.2: Efficiency-equity frontier (ε-constraint): minimum weighted mean distance attainable for every cap D imposed on the maximum distance. The extremes are the minimax solution (on the left) and the Weber point (on the right); the almost flat curve indicates that equity costs very little in terms of efficiency.

```
cap D = 5.680 km: mean 3.391   (minimax solution)
cap D = 5.997 km: mean 3.351
cap D = 6.314 km: mean 3.344   (Weber solution)
```

Line-by-line explanation

The frontier is almost flat: guaranteeing 0.63 km less to the farthest district costs only 47 metres of mean distance. When the frontier is this flat, the equitable solution is "almost free" — a very strong argument in a public discussion. The opposite case (a steep frontier) would signal a genuine efficiency-equity conflict.

9.7 Exercises

Exercise 9.1 — check

Compute by hand the weighted centroid of the 12 districts from the CSV data and compare it with the output of the script.

Exercise 9.2 — a weight that changes everything

Raise the weight of Q12 (the district at the bottom right) from 5 to 40: how far do Weber and minimax move? Which of the two is more sensitive to the weights, and why?

Exercise 9.3 — two facilities

With *two* stations and the assignment of every district to the nearest one, does the problem remain convex? Try an alternating approach (assign–optimize) and discuss why it only finds local optima.

Exercise 9.4 — Manhattan distance

Replace the Euclidean distance with $|x - a_i| + |y - b_i|$ and show that the Weber problem becomes an LP (introduce auxiliary variables for the absolute values). Solve it with Gurobi and compare the optimum.

Exercise 9.5 — multiplier of the radius

Estimate by perturbation the multiplier of the constraint $r = 2$ km (solve with $r = 2.1$): how much is every additional 100 m of radius worth, in km of total distance?

Chapter 10

Smart charging of electric vehicles

Class: LP / convex QP **Script:** `python/lab10_ev_charging.py`
Online page

10.1 Managerial motivation

A company fleet must be fully charged by the morning by exploiting the cheap night hours — but if all the vehicles charge together during the cheap hours, the peak draw explodes and with it the power costs and the risk of exceeding the capacity of the meter. The model decides *how much power* to give to each vehicle in each hour: a naturally continuous decision, with no need at all for on/off variables.

Smart charging is one of the new operational problems created by the energy transition: electric fleets, hourly energy prices, power limits of the meters. It is also a perfect example of a problem that *seems* to require on/off variables and instead is a pure LP, because the real decision — how much power — is continuous.

In this chapter we will learn how to: (1) model decisions over an hourly horizon with availability windows; (2) read the duals as "prices of the marginal hours"; (3) handle the cost-peak trade-off with a multi-objective approach; (4) diagnose a real pitfall of sensitivity analysis (the right-hand side with constants).

10.2 Guided construction of the model

The problem in words. *We decide* how much power to deliver to each vehicle in each hour. *The objective* is to minimize the energy expenditure of the night. *The constraints*: every vehicle must receive all the energy it needs before departure, it can charge only while it is plugged in and never above the power of its charger, and the overall draw (charging plus building) can never exceed the power of the meter.

Data (input of the model)

Symbol	Type	Meaning
n	$\in \mathbb{Z}_{\geq 1}$	number of vehicles, indexed by $v \in \{1, 2, \ldots, n\}$
m	$\in \mathbb{Z}_{\geq 1}$	number of hourly intervals, indexed by $t \in \{1, 2, \ldots, m\}$, of duration Δt (here $\Delta t = 1$ h)
π_t	$\in \mathbb{Q}_{\geq 0}$	energy price in hour t (€/kWh)
a_{vt}	$\in \{0, 1\}$	observed *datum*: 1 if vehicle v is plugged in during hour t, 0 otherwise
e_v	$\in \mathbb{Q}_{>0}$	energy required by vehicle v before departure (kWh)
$\bar{p}_v$	$\in \mathbb{Q}_{>0}$	maximum charging power of vehicle v (kW)
b_t	$\in \mathbb{Q}_{\geq 0}$	base load of the building in hour t (kW)
k	$\in \mathbb{Q}_{>0}$	maximum power that can be drawn from the meter (kW)
η	$\in \mathbb{Q},\ 0 < \eta \leq 1$	charging efficiency

Decision variables

We introduce the following $n\,m$ non-negative variables:

$$x_{vt} = \text{power assigned to vehicle } v \text{ in hour } t \text{ (kW)}, \qquad \forall v \in \{1, 2, \ldots, n\},\ \forall t \in \{1, 2, \ldots, m\}.$$

Using these variables, an LP model for the problem is the following:

Minimum-cost charging LP model

$$\min \sum_{v=1}^{n} \sum_{t=1}^{m} \pi_t \,\Delta t\, x_{vt} \tag{10.1a}$$

$$\text{subject to} \quad \eta \sum_{t=1}^{m} \Delta t\, x_{vt} \geq e_v, \qquad \forall v \in \{1, 2, \ldots, n\}, \tag{10.1b}$$

$$\sum_{v=1}^{n} x_{vt} + b_t \leq k, \qquad \forall t \in \{1, 2, \ldots, m\}, \tag{10.1c}$$

$$x_{vt} \leq a_{vt}\, \bar{p}_v, \qquad \forall v \in \{1, 2, \ldots, n\},\ \forall t \in \{1, 2, \ldots, m\}, \tag{10.1d}$$

$$x_{vt} \geq 0, \qquad \forall v \in \{1, 2, \ldots, n\},\ \forall t \in \{1, 2, \ldots, m\}. \tag{10.1e}$$

Description of the objective function and of the constraints of the LP model:

- the linear objective function (10.1a) minimizes the energy expenditure of the night: the power x_{vt} delivered for the duration Δt costs $\pi_t\,\Delta t\, x_{vt}$ euros;
- the linear constraints (10.1b) ensure that every vehicle receives all the energy it requires before departure; the efficiency η stands on the left: of the energy that is drawn, only the fraction η ends up in the battery (n linear constraints);
- the linear constraints (10.1c) impose that the overall draw (charging plus base load) never exceeds the power of the meter: these are the constraints that couple the vehicles to each other (m linear constraints);

- the linear constraints (10.1d) limit the power of every vehicle to that of its charger and set it to zero in the hours in which the vehicle is absent ($a_{vt} = 0$): availability is a *datum*, not a decision, and this is why no binary variables are needed ($n\,m$ linear constraints);
- the constraints (10.1e) define the variables of the model.

Variants: peak shaving ($\min z$ with $\sum_{v=1}^{n} x_{vt} + b_t \le z$ for every $t \in \{1, 2, \dots, m\}$); smooth profile (QP with $\gamma \sum_{t=2}^{m} (z_t - z_{t-1})^2$); multi-objective cost $+\,\rho\, z$.

Line-by-line explanation

The energy constraint written as $\ge$. Charging more than necessary is allowed but never worthwhile (the price is positive), hence at the optimum it holds with equality.

10.3 Worked example by hand

One vehicle, two hours

We need $e = 10$ kWh ($\eta = 1$); prices $\pi_1 = 0.10$ and $\pi_2 = 0.20$ €/kWh; maximum power $\bar{p} = 8$ kW.
Case $\bar{p} = 8$. In the cheap hour we charge at the maximum (8 kWh); the remaining 2 kWh go into the expensive hour. Cost: $8 \cdot 0.10 + 2 \cdot 0.20 = 1.20$ €. *Shadow price of the energy requirement*: the eleventh kWh would end up in the expensive hour $\Rightarrow$ 0.20 €/kWh. *Shadow price of the power limit* in hour 1: one more kW would allow us to move one kWh from the expensive hour to the cheap one $\Rightarrow$ 0.10 €/kW.
Case $\bar{p} = 12$. Everything in the cheap hour: cost 1.00 €, dual of the requirement 0.10, dual of the power 0 (constraint not active). The pattern "the dual is the price of the marginal hour" is the key for reading the whole chapter.

10.4 Case study

Six vans with different night charging windows (data in `data/ev_flotta.csv`), stylized hourly prices with an evening peak, base load of the building, a meter of $k = 120$ kW, $\eta = 0.95$.

```
x = m.addVars(veicoli, ore, name="x")
for v in veicoli:
    for t in ore:
        x[v, t].UB = P[v] * disp[v, t]     # 0 if not plugged in
v_ene  = m.addConstrs((eta * gp.quicksum(x[v, t] for t in ore)
                       >= E[v] for v in veicoli), name="energia")
v_rete = m.addConstrs((gp.quicksum(x[v, t] for v in veicoli)
                       + base[t] <= C for t in ore), name="rete")
m.setObjective(gp.quicksum(prezzo[t] * x[v, t]
                           for v in veicoli for t in ore), GRB.MINIMIZE
                               )
```

10.5 Results

```
Minimum cost    : 20.36 EUR/night   peak draw 103.4 kW (limit 120)
Peak shaving    : minimum possible peak 68.0 kW
Shadow prices of the energy requirement (EUR/kWh):
  V1 0.0947   V2 0.0842   V3 0.0842   V4 0.0947   V5 0.0947   V6 0.0842
```

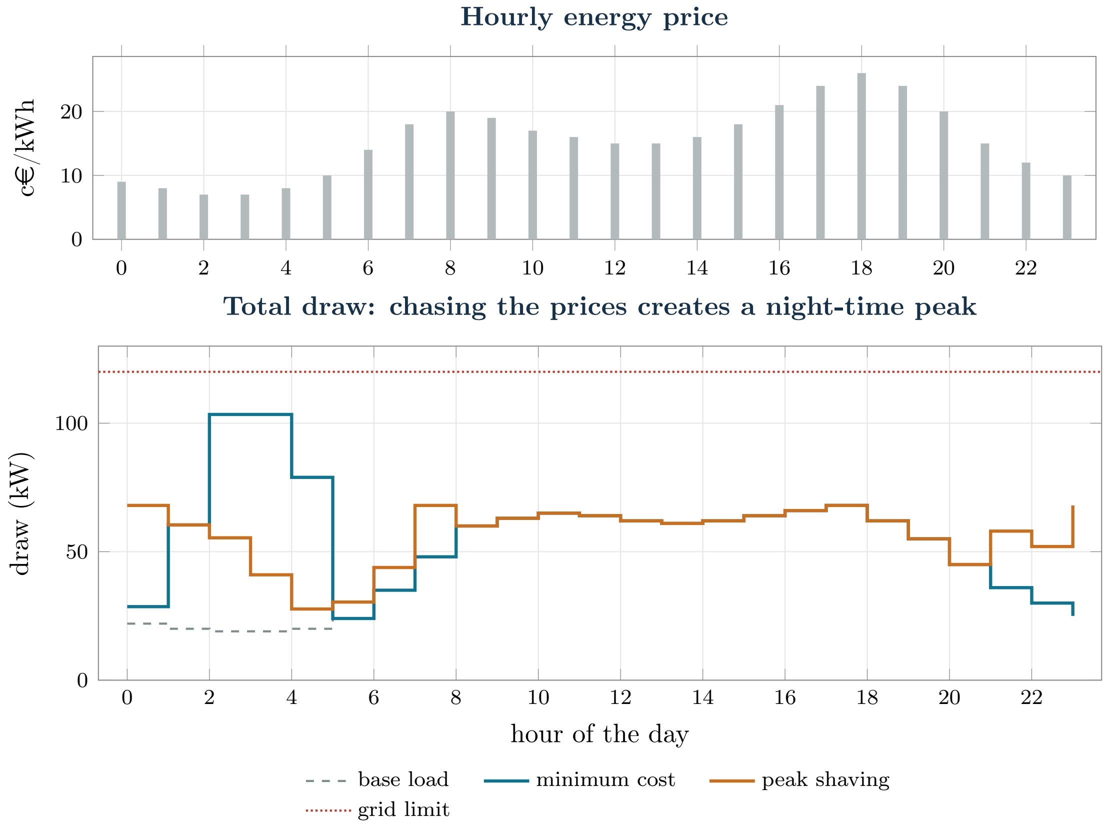


Figure 10.1: Above: hourly energy price (grey bars), low at night and with an evening peak. Below: total draw (base + charging) hour by hour under the two strategies: the minimum cost one (teal) concentrates the charging in the cheap hours and creates a night-time peak of 103 kW; peak shaving (orange) spreads the charging out and keeps the draw at 68 kW, below the grid limit (red line).

The duals tell the story of the marginal hours

The shadow prices of the energy requirement are equal to $0.0842 = 0.08/\eta$ or $0.0947 = 0.09/\eta$: they are the prices of the *marginal hours* of each vehicle (the cheapest hour still free within its window), corrected for the efficiency. V2, V3 and V6 have windows that cover the hours at 0.08; V1, V4 and V5 arrive late or leave early and their marginal hour costs 0.09. One additional kWh of requirement therefore costs between 8.4 and 9.5 cents depending on the vehicle.

10.6 Sensitivity analysis

```
Trade-off cost + rho*peak:
  rho = 0.00: cost 20.36   peak 103.4     rho = 0.20: cost 22.15   peak 68.0
```

```
  rho = 0.05: cost 20.69  peak  86.5     rho = 0.50: cost 22.15  peak 68.0
  rho = 0.10: cost 21.39  peak  74.9
Meter capacity:
  k = 60, 65 kW: INFEASIBLE     k = 80: 21.09     k = 120: 20.36
  k = 70 kW    : 21.93          k = 90: 20.63
```

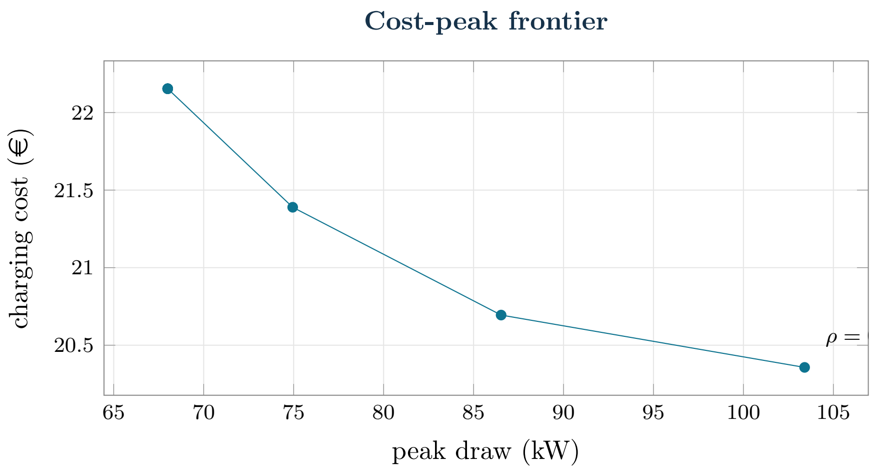


Figure 10.2: Cost-peak frontier obtained by varying the weight ρ of the peak in the objective: every point is an optimal charging strategy. Reducing the peak from 103 to 75 kW costs about one euro per night; the point $\rho \geq 0.2$ reaches the minimum possible peak (68 kW) at the lowest cost that allows it.

Managerial insight

Cutting the peak from 103 to 75 kW (−28%) costs only one euro per night; getting down to 68 kW costs less than two. If the power tariff or the upgrade of the meter cost more than that, peak shaving pays for itself. Note: the pure minimax (peak 68) without the cost term would spend 27.79 € — the trade-off with $\rho = 0.2$ achieves *the same peak* at 22.15 €: never optimize a single objective when there are two of them.

In the constraint `quicksum(x) + base[t] <= C`, Gurobi moves the constant `base[t]` into the right-hand side: the stored RHS is `C - base[t]`. The first version of our sensitivity analysis wrote `v.RHS = nuovaC` and obtained identical costs for every capacity — it was in fact relaxing the constraint by `base[t]`. The correct form is `v.RHS = nuovaC - base[t]`. When a sensitivity analysis changes nothing, suspect your own code before suspecting the model.

10.7 Exercises

Exercise 10.1 — check

Verify by perturbation the shadow price of the energy requirement of V1 (raise e_1 from 46 to 47 kWh) and explain why it is equal to 0.09/0.95.

Exercise 10.2 — late arrival

V3 arrives at 23 instead of 20: how do cost and peak change? Which shadow price increases?

Exercise 10.3 — smooth profile

Implement the QP variant with $\gamma \sum_{t=2}^{m}(z_t - z_{t-1})^2$ where $z_t = b_t + \sum_{v \in V} x_{vt}$, for $\gamma \in \{0.01; 0.1; 1\}$: compare cost, peak and "jumpiness" of the profile.

Exercise 10.4 — emissions

With an hourly carbon intensity g_t (high in the evening, low at night), draw the cost-emissions frontier. Are the cheap hours also the cleanest ones in this case?

Exercise 10.5 — minimum capacity

Find by bisection the minimum capacity k that keeps the problem feasible, and explain which combination of windows and $\bar{p}_v$ determines it.

Chapter 11

Service capacity and waiting times

Class: convex NLP (M/M/1 queue) **Script:** `python/lab11_queues.py`
Online page

11.1 Managerial motivation

How much capacity should be assigned to a customer service, to a service desk, to a cloud service? More capacity costs money; too little capacity makes the waiting times explode. Queueing theory provides the quantitative relation between capacity and waiting, and optimization turns it into a decision. The central message is counter-intuitive for anyone who reasons "by efficiency": **the optimal utilization is not 100%**.

Every service with random arrivals — call centres, emergency rooms, help desks, the API of a cloud service — lives with the same dilemma: capacity costs, and so does waiting. Queueing theory gives the exact relation between the two; optimization chooses the right point. The chapter is short but its message (the optimal utilization is not 100%) is by itself worth a discussion in a board meeting.

In this chapter we will learn how to: (1) use the M/M/1 formulas inside an optimization problem; (2) derive the optimum analytically and confirm it numerically; (3) put a price on a service promise; (4) protect the sizing against uncertainty about demand.

11.2 Guided construction of the model

The problem in words. *We decide* how much service capacity to buy. *The objective* is to minimize the total cost: the capacity that is bought plus the value of the time the customers spend in the system. *The constraint* of a structural nature is only one, the stability of the queue ($\mu > \lambda$); in the variants the service promise $w \leq w_{\max}$ is added.

We use the simplest queueing model, denoted by the standard notation **M/M/1**: *Markovian* arrivals (a Poisson process), *Markovian* service times (exponential) and *one* server. For this system the closed formulas reported in the table hold; what is new here is treating the capacity μ as an optimization variable.

Data (input of the model)

Symbol	Type	Meaning
λ	$\in \mathbb{Q}_{>0}$	mean arrival rate of the customers (requests/hour)
c	$\in \mathbb{Q}_{>0}$	cost of one unit of capacity (€ per unit of μ per hour)
h	$\in \mathbb{Q}_{>0}$	value of one hour spent by a customer in the system (€)

Decision variable and derived quantities

We introduce one continuous variable:

$$\mu = \text{service capacity (service rate, requests/hour)},$$

with the stability requirement $\mu > \lambda$. From M/M/1 queueing theory we obtain the mean time in the system $w(\mu) = 1/(\mu - \lambda)$ (in the literature: W) and the mean number of customers in the system $l(\mu) = \lambda/(\mu - \lambda)$ (Little's law; in the literature: L).

Using this variable, the model for the problem is the following:

Total cost model: capacity + waiting

$$\min \quad c\,\mu + h\,\frac{\lambda}{\mu - \lambda} \tag{11.1a}$$

$$\text{subject to} \quad \mu > \lambda. \tag{11.1b}$$

Description of the objective function and of the constraint of the model:

- the objective function (11.1a) is the sum of two terms: the first, $c\,\mu$, buys the capacity; the second, $h \cdot l(\mu)$, monetizes the total time the customers spend in the system; it is convex on $(\lambda, +\infty)$;
- the constraint (11.1b) defines the variable and guarantees the stability of the queue: with $\mu \le \lambda$ the line grows without bound.

Setting the derivative of the objective to zero gives the closed-form solution:

$$C'(\mu) = c - \frac{h\,\lambda}{(\mu - \lambda)^2} = 0 \qquad \Longrightarrow \qquad \boxed{\mu^* = \lambda + \sqrt{h\,\lambda/c}}$$

Line-by-line explanation

The formula has the classical structure "capacity = demand + safety stock": the mean demand λ is covered and a margin $\sqrt{h\lambda/c}$ is added, which grows with the value of the customers' time and decreases with the cost of capacity. The margin grows like $\sqrt{\lambda}$: large systems enjoy economies of scale in the "buffer" (pooling).

11.3 Worked example by hand

A service desk with 8 customers per hour

$\lambda = 8/\text{h}$, $c = 2$ € per unit of capacity, $h = 4$ €/h.

- **Step 1.** $\mu^* = 8 + \sqrt{4 \cdot 8/2} = 8 + 4 = 12$ customers/hour.
- **Step 2.** Utilization $\rho = 8/12 = 66.7\%$; mean time $w = 1/(12-8) = 15$ minutes; cost $C = 2 \cdot 12 + 4 \cdot 8/4 = 32$ €/h.
- **Step 3 — why not cut the capacity?** With $\mu = 9$ ($\rho = 89\%$, "more efficient"): $C = 18 + 32/1 = 50$ €/h, much worse. The apparent waste (33% of unused capacity) is exactly what keeps the queues short.

11.4 Case study and results

Customer service: $\lambda = 42$ requests/hour, $c = 3$ €, $h = 1.5$ €.

```
Analytical: mu* = 42 + sqrt(1.5*42/3) = 46.583   cost 153.495 EUR/h
Gurobi    : mu* = 46.582 (bilinear w(mu-lam) = 1, NonConvex=2: matches)
At optimum: rho = 90.2%    w = 13.1 minutes
```

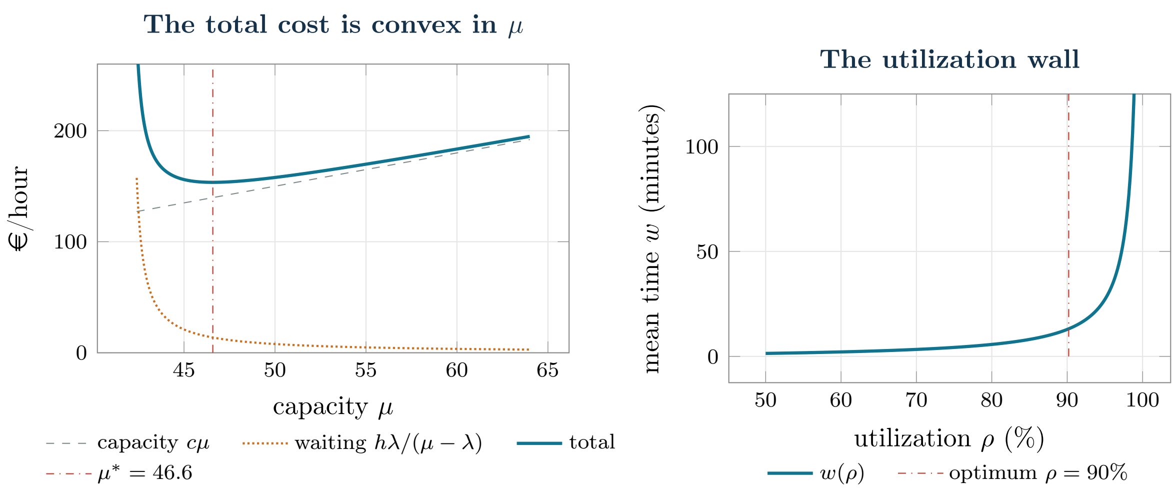


Figure 11.1: On the left: the two components of the cost (capacity, increasing; waiting, decreasing) and the total cost, convex, with its minimum at $\mu^* = 46.6$. On the right: the "utilization wall" — the mean time in the system w as a function of the utilization ρ: beyond 90% the waiting time explodes hyperbolically.

Managerial insight

The plot on the right is perhaps the most important one in these notes to show to a manager: between $\rho = 90\%$ and $\rho = 99\%$ the waiting time is multiplied by ten. "Saturating the resources" and "giving good service" are objectives in mathematical, not organizational, conflict: no reorganization removes the hyperbola $w = 1/(\mu - \lambda)$.

11.5 Sensitivity analysis: the price of a promise

If marketing promises a "mean time below $w_{\max}$", we need $\mu \geq \lambda + 1/w_{\max}$: the constraint becomes active when it is tighter than the economic optimum (13.1 minutes).

```
w_max = 12 min: mu = 47.00  cost 153.60   (almost free)
w_max =  9 min: mu = 48.67  cost 155.45
w_max =  6 min: mu = 52.00  cost 162.30
w_max =  4 min: mu = 57.00  cost 175.20
w_max =  3 min: mu = 62.00  cost 189.15
w_max =  2 min: mu = 72.00  cost 218.10   (+42% on the base cost)
```

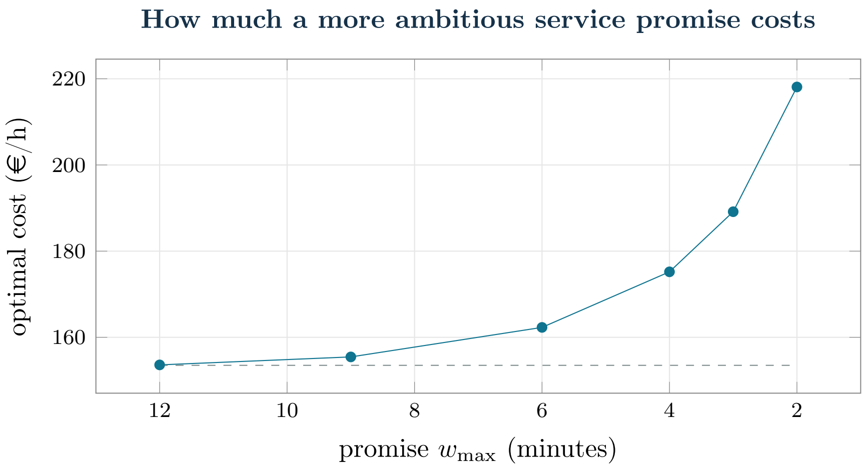


Figure 11.2: Optimal cost as a function of the service promise $w_{\max}$ (reversed axis: towards the right the promise becomes more ambitious). The dashed line is the cost without any promise: up to 12 minutes the promise is almost free, below 4 minutes every additional minute cut costs explosively.

The marginal cost of the promise

The shadow price of the constraint grows explosively: going from 12 to 9 minutes costs 1.85 €/h, from 3 to 2 minutes costs 28.95 €/h. When the constraint is active, $\mu = \lambda + 1/w_{\max}$ and $C = c\lambda + c/w_{\max} + h\lambda\, w_{\max}$, from which $dC/dw_{\max} = -c/w_{\max}^2 + h\lambda$ (with $w_{\max}$ in hours): at $w_{\max} = 6$ minutes it is equal to $-300 + 63 = -237$ €/h per hour of promise — exactly the finite-difference estimate of the script. This is the number to give to marketing *before* it signs the SLA (*Service Level Agreement*, the contractual commitment on the service level).

Robustness

If λ is uncertain in $[36, 48]$, sizing on the nominal optimum $\mu^* = 46.58$ is **unstable**: with $\lambda = 48 > \mu^*$ the queue diverges (infinite cost). Minimizing the cost of the worst case gives $\mu_{\text{rob}} = 52.90$ with a worst-case cost of 173.39 €/h: an insurance premium of $\approx 13\%$ is paid in order to be protected from a disaster. Capacity decisions must always be checked against the *maximum* plausible demand, not only against the mean.

11.6 Exercises

Exercise 11.1 — check

Derive on your own the formula $dC/dw_{\max} = -c/w_{\max}^2 + h\lambda$ of the previous explanation and verify it numerically at $w_{\max} = 4$ and 9 minutes by finite differences.

Exercise 11.2 — quadratic penalty

Replace the linear waiting cost with $h\,[1/(\mu - \lambda)]^2$: derive μ^* analytically and compare the optimal utilization with the linear case.

Exercise 11.3 — two queues vs one

Two separate service desks each receive $\lambda = 21$; alternatively, a single system receives $\lambda = 42$ with aggregate capacity. At equal capacity cost, which configuration gives shorter waiting times? (This is the quantitative argument for pooling.)

Exercise 11.4 — capacity-service frontier

Plot the frontier (cost, w) by varying $w_{\max}$ continuously and identify the point beyond which every additional promised minute cut costs more than 10 €/h.

Part III

Decisions under uncertainty

Chapter 12

The Newsvendor and its variants

Class: 1D convex / scenario-based stochastic LP **Script:** `python/lab12_newsvendor.py`
Online page

12.1 Managerial motivation

The Newsvendor ("newspaper seller") describes a very common decision: choosing a quantity *before* observing demand. Ordering too much generates unsold items; too little, lost sales. It applies to fashion and seasonal goods, fresh and perishable products, drugs, hotel capacity, spare parts. It is the gateway to stochastic optimization, and its didactic strength is a clear-cut result: **the optimal quantity is not the average demand**.

The model is more than a century old and remains the foundation of inventory management for single-cycle products. Its didactic importance goes further: it is the first encounter with the idea that *we decide first and find out later*, and with the tools needed to do it well (scenarios, value of the stochastic solution, risk measures).

In this chapter we will learn to: (1) derive and apply the quantile rule; (2) turn a stochastic problem into a scenario-based LP with the positive-part trick; (3) quantify the value of modelling uncertainty (VSS); (4) incorporate service levels and risk aversion.

12.2 The basic model

The problem in words. *We decide* a single quantity q, before knowing demand. *The objective* is to minimize the total expected cost of the error: the cost of having ordered too much (unsold items) plus the cost of having ordered too little (lost sales). *Constraints* are absent in the basic model ($q \geq 0$); they appear in the variants: shared budget, service levels, capacity limits.

Data (input of the model)

Symbol	Type	Meaning
D	random variable ≥ 0	demand, with cumulative distribution function F; an exception to the convention on capital letters, to follow the literature: the random variable in upper case, its realizations in lower case (d_s)
c_u	$\in \mathbb{Q}_{>0}$	unit *underage* cost (unmet demand): lost margin plus any penalty (€)
c_o	$\in \mathbb{Q}_{>0}$	unit *overage* cost (unsold unit): cost minus salvage value (€)

With selling price p, purchase cost c and salvage value $v \leq c$ (all in $\mathbb{Q}_{>0}$): $c_u = p - c$ and $c_o = c - v$.

Decision variable

We introduce a non-negative variable:

$$q = \text{quantity ordered } \textit{before} \text{ observing demand } D.$$

Using this variable, the model for the problem is the following:

Newsvendor model in cost form

$$\min \quad c_o\,\mathbb{E}\big[(q-D)^+\big] + c_u\,\mathbb{E}\big[(D-q)^+\big] \tag{12.1a}$$

$$\text{subject to} \quad q \geq 0. \tag{12.1b}$$

Description of the objective function and of the constraint of the model:

- the convex objective function (12.1a) minimizes the expected cost of the error: overage $(q-D)^+$ charged at c_o and underage $(D-q)^+$ charged at c_u (the symbol $(x)^+ = \max\{x, 0\}$ denotes the positive part);
- constraint (12.1b) defines the variable of the model.

If F is continuous, the optimum satisfies the **quantile rule**:

$$F(q^*) = \alpha^* = \frac{c_u}{c_u + c_o} \qquad \Longrightarrow \qquad q^* = F^{-1}(\alpha^*).$$

Where the quantile rule comes from

The argument works on the marginal unit: ordering the q-th extra unit yields c_u if demand absorbs it (probability $1 - F(q)$) and costs c_o if it remains unsold (probability $F(q)$). It is worth doing as long as $c_u\,(1 - F(q)) \geq c_o\,F(q)$, that is, as long as $F(q) \leq c_u/(c_u + c_o)$. If underage costs far more than overage, α^* is high and one orders a high quantile; if unsold items are expensive (perishables), one orders little.

12.3 Worked example by hand (discrete demand)

Three equally likely scenarios

$D \in \{80, 100, 120\}$ with probability $1/3$; $c_u = 9$, $c_o = 4$ (hence $\alpha^* = 9/13 = 0.6923$).

- **Step 1 — expected cost for the three natural candidates.**

$$C(80) = \tfrac{1}{3}\left[0 + 9\cdot 20 + 9\cdot 40\right] = 180$$
$$C(100) = \tfrac{1}{3}\left[4\cdot 20 + 0 + 9\cdot 20\right] = \tfrac{260}{3} = 86.7$$
$$C(120) = \tfrac{1}{3}\left[4\cdot 40 + 4\cdot 20 + 0\right] = 80$$

- **Step 2 — discrete quantile rule.** We need the smallest q with $F(q) \geq 0.6923$: $F(80) = 1/3$, $F(100) = 2/3 = 0.667 < 0.6923$, $F(120) = 1$. Hence $q^* = 120$: consistent with step 1.

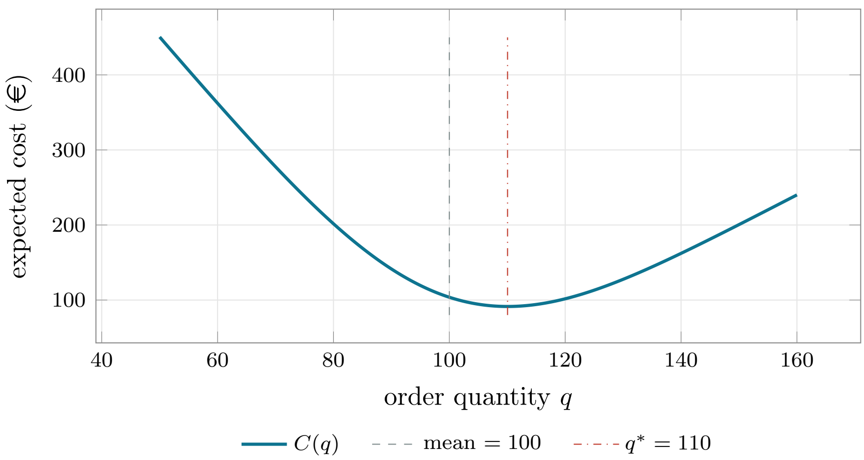
- **Step 3 — interpretation.** Ordering the historical *maximum* even though the mean is 100: with $c_u = 9 \gg c_o = 4$, running short costs more than twice as much as running long. Whoever orders "the mean" pays 86.7 against 80: uncertainty is managed, not ignored.

12.4 Case study: the quantile rule in the continuous case

Bakery: $p = 15$, $c = 6$, $v = 2$ € $\Rightarrow c_u = 9$, $c_o = 4$; normally distributed demand with $\mu = 100$, $\sigma = 20$.

```
alpha* = 9/13 = 0.6923
q* = F^-1(0.6923) = 110.05 units   (mean = 100)
expected cost at q*: 91.43 EUR    at q = mean: 103.72 EUR
```

Minimum at the 69th percentile



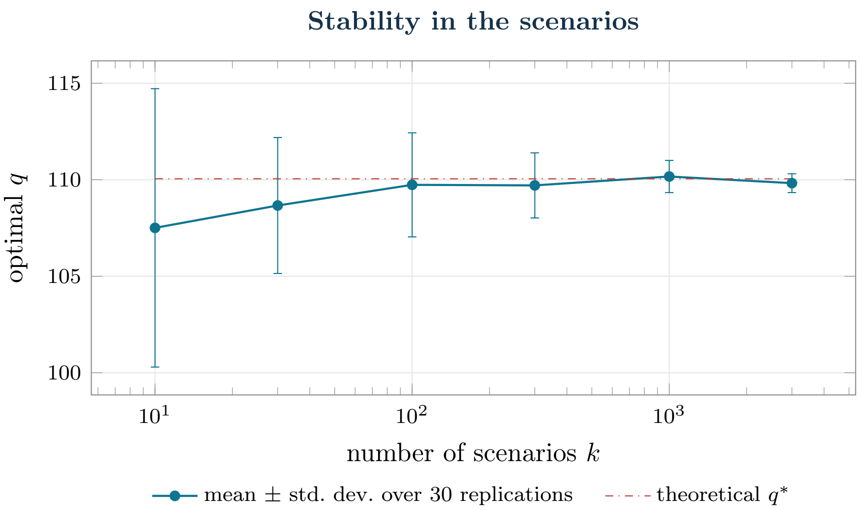


Figure 12.1: Left: the expected cost $C(q)$ is convex and the minimum falls at $q^* = 110$ (69th percentile of demand), not at the mean 100. Right: the optimal q estimated from k sampled scenarios, mean and standard deviation over 30 replications — with few scenarios the decision is unstable.

12.5 The scenario-based linear formulation

When F is not known in closed form — the normal situation in a company — one uses scenarios: historical data, samples from an estimated distribution, or weighted scenarios. New model data: the number of scenarios $k \in \mathbb{Z}_{\geq 1}$, indexed by $s \in \{1, 2, \dots, k\}$; for each scenario, demand $d_s \in \mathbb{Q}_{\geq 0}$ and probability $\pi_s \in \mathbb{Q}_{\geq 0}$, with $\sum_{s=1}^{k} \pi_s = 1$.

Besides q, we introduce the following $2\,k$ non-negative variables:

$$\begin{cases} o_s = \text{overage (unsold units) in scenario } s \\ u_s = \text{underage (unmet demand) in scenario } s \end{cases} \qquad \forall s \in \{1, 2, \dots, k\}.$$

Using these variables, an LP model for the problem is the following:

Scenario-based Newsvendor LP model

$$\min \quad \sum_{s=1}^{k} \pi_s \big(c_o\, o_s + c_u\, u_s\big) \tag{12.2a}$$

$$\text{subject to} \quad o_s \geq q - d_s, \qquad \forall s \in \{1, 2, \dots, k\}, \tag{12.2b}$$

$$u_s \geq d_s - q, \qquad \forall s \in \{1, 2, \dots, k\}, \tag{12.2c}$$

$$q \geq 0, \tag{12.2d}$$

$$o_s,\ u_s \geq 0, \qquad \forall s \in \{1, 2, \dots, k\}. \tag{12.2e}$$

Description of the objective function and of the constraints of the LP model:

- the linear objective function (12.2a) is the weighted average, over the scenarios, of the overage and underage costs: the “scenario-based” version of the expected cost of the basic model;

- the linear constraints (12.2b) and (12.2c) define the overage and the underage of each scenario starting from the only genuine decision q; together with the non-negativity conditions they realize the positive parts, as explained below ($2\,k$ linear constraints);
- constraints (12.2d) and (12.2e) define the variables of the model.

Why it is linear (the positive-part trick)

$(q - d_s)^+$ is not linear, but at the *optimum* the variable o_s, which appears in the objective with a positive coefficient, is pushed down onto the maximum between $q - d_s$ and 0: constraint (12.2b) together with $o_s \geq 0$ does exactly the job of $\max\{q - d_s, 0\}$ without binary variables. The same mechanism applies to u_s. This trick — linearizing the positive parts that the objective pushes downwards — returns unchanged in the CVaR and in the SVM.

```python
q = m.addVar(name="q")
o = m.addVars(S, name="o")      # overage per scenario
u = m.addVars(S, name="u")      # underage per scenario
m.addConstrs((o[s] >= q - dom[s] for s in range(S)))
m.addConstrs((u[s] >= dom[s] - q for s in range(S)))
m.setObjective(gp.quicksum((Co*o[s] + Cu*u[s]) / S
                           for s in range(S)), GRB.MINIMIZE)
```

```
LP with k = 600 scenarios: q = 108.61
empirical quantile at 69.23%: 108.60   (they coincide: it is a theorem)
VSS: decision q = E[D] = 100 costs 99.50; stochastic q costs 88.33
     value of the stochastic solution = 11.17 EUR per cycle
```

Managerial insight

The *value of the stochastic solution* (VSS) answers the boss's question: "why complicate our lives with scenarios?". Here it is worth 11.17 € per cycle, 11% of the cost: it is the gain from *modelling* uncertainty instead of replacing it with the mean. The figure on stability gives the other half of the message: below 100 scenarios the optimal q swings by ± 3 units from one estimate to the next — scenarios are themselves a sample, and the decision inherits their variance.

12.6 Service levels

```
cycle service level 90%: q = 125.63  (cost +23.41 EUR over the economic optimum
   )
cycle service level 95%: q = 132.90  (cost +45.59)
cycle service level 99%: q = 146.53  (cost +95.56)
fill rate at q*: 96.06%   probability of no stockout at q*: 69.23%
```

At $q^* = 110$ the probability of avoiding a stock-out is only 69%, but the *fill rate* (share of demand served on average) is 96%: different measures that contracts often confuse. Promising "95% probability of full coverage" costs +45.59 € per cycle compared with the economic optimum; promising "95% fill rate" is almost free. Read the SLA carefully before signing it.

12.7 Risk aversion: mean-CVaR

Expected cost does not distinguish between many small losses and rare disasters. With the linear formulation of the CVaR (here only anticipated) we minimize $(1-\lambda)\,\mathbb{E}[\text{cost}] + \lambda\,\text{CVaR}_{0.90}(\text{cost})$:

```
lambda = 0.00: q = 108.61   mean cost 88.33   CVaR90 245.93
lambda = 0.25: q = 112.45   mean cost 90.37   CVaR90 229.29
lambda = 0.50: q = 114.48   mean cost 92.52   CVaR90 225.49
lambda = 1.00: q = 116.17   mean cost 94.87   CVaR90 224.56
```

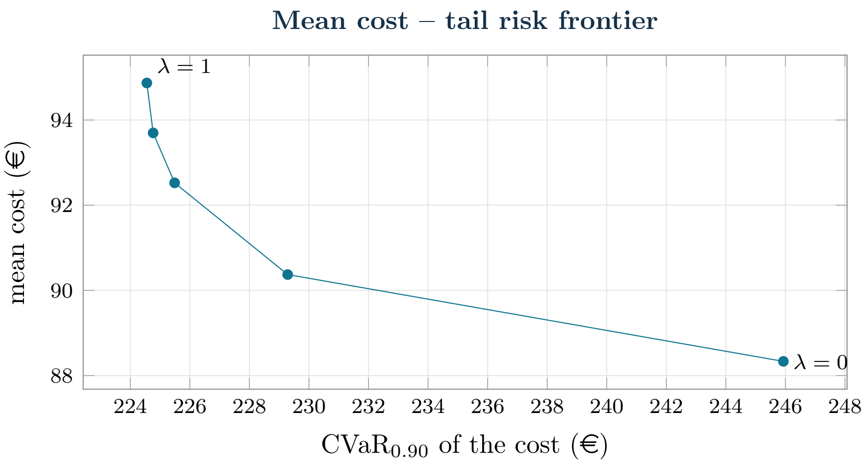


Figure 12.2: Frontier between mean cost and $\text{CVaR}_{0.90}$ of the cost, obtained by varying the weight λ of risk aversion: from $\lambda = 0$ (optimal on average, heavy tail) to $\lambda = 1$ (maximum protection). The first stretch is almost vertical: a lot of protection for very little mean cost.

The risk-averse decision maker orders more (from 108.6 to 116.2 units): pays 6.5 € more on average in order to cut by 21.4 € the mean cost of the worst 10% of the scenarios. Most of the protection is already obtained with $\lambda = 0.25$: the frontier is steep at the beginning and flat afterwards.

12.8 Multi-product with a shared budget

Three cakes with *correlated* demands ($\rho = 0.7$: the holidays go well or badly for everyone together) and a production budget of 1200 €:

```
  product    | q without budget | q with budget
 panettone   |            111.1 |          99.3
 pandoro     |             91.5 |          75.4
 torrone     |             65.9 |          56.8
Spend: 1200/1200   shadow price of the budget: -0.553
```

Line-by-line explanation

The binding budget compresses the three quantities below their respective optimal quantiles, but not proportionally: the cut depends on the ratio c_u/c_o and on the unit cost of each product. The dual says that one more euro of budget would reduce the expected cost by 0.55 €: a 55% return that would justify almost any financing — information that nobody would have quantified without the model. Moreover, the high correlation removes the diversification benefit: the three products fail together, and treating them as independent would underestimate the risk.

12.9 Exercises

Exercise 12.1 — check

Recompute by hand α^* and q^* if the municipality imposes a penalty $b = 3$ € per unserved customer ($c_u = p - c + b$). Check with the script.

Exercise 12.2 — historical data

Use as scenarios the 24 demand observations of the last two years (generate them with the script) instead of the distribution: how much does q differ? Which approach would you recommend to a company that has only 24 observations?

Exercise 12.3 — perishables

The salvage value becomes negative ($v = -1$ €: disposal costs money): how do α^* and q^* change? Plot q^* as a function of $v \in [-2, 4]$.

Exercise 12.4 — EVPI

Compute the EVPI (*Expected Value of Perfect Information*): mean cost with q chosen *after* having seen demand (that is, $q = d_s$, cost 0) against the cost of the stochastic solution. How much is a perfect demand forecast worth at most?

Exercise 12.5 — constrained fill rate

Add to the LP the fill rate constraint $\sum_{s=1}^{k} \pi_s u_s \leq (1 - 0.98)\,\mathbb{E}[D]$ and compare quantity and cost with the cycle service level at 98%.

Exercise 12.6 — multi-product

Rerun the multi-product case with independent demands ($\rho = 0$), everything else being equal: does the optimal expected cost improve or worsen? Why?

Chapter 13

VaR and CVaR: measuring and optimizing risk

Class: scenario-based LP

Script: `python/lab13_var_cvar.py`
Online page

13.1 Managerial motivation

Many models optimize an average value. A simple choice, but one that can hide rare and very large losses. VaR and CVaR describe the *tail* of the loss distribution and turn risk aversion into quantitative decisions. VaR answers "what is a high loss threshold?"; CVaR also answers "how much do we lose *on average* when that threshold is exceeded?".

After the financial crises, "how much do we lose in the worst scenarios?" has become a regulatory question as well as a managerial one. VaR and CVaR are the standard language with which banks, insurers and, increasingly, supply chain managers speak about tail risk. The didactic surprise is that CVaR — an apparently sophisticated concept — is optimized with a simple LP.

In this chapter we will learn to: (1) define and compute VaR and CVaR; (2) use the Rockafellar–Uryasev linear formulation; (3) build portfolios and logistics plans protected against the worst scenarios; (4) recognize the statistical limits of tail estimates.

13.2 Definitions and properties

The problem in words. In the models of this chapter *we decide* as before (portfolio weights, quantities, booked capacity), but *the objective* changes: not to minimize the *average* loss, but rather a measure of its *tail* — how much is lost in the worst scenarios. *The constraints* remain those of the underlying problem, plus the auxiliary linear constraints that define the CVaR.

Let L be the random loss (high values = worse outcomes) and $\alpha \in (0.1)$ a confidence level (typically 0.90–0.99). As for the demand of the newsvendor, the random variable is in upper case and its realizations in lower case (ℓ_s): this is the exception, to follow the literature, to our convention that reserves capital letters for sets and matrices.

VaR and CVaR

$$\mathrm{VaR}_\alpha(L) = \inf\{\eta \in \mathbb{R} : \mathbb{P}(L \le \eta) \ge \alpha\},$$
$$\mathrm{CVaR}_\alpha(L) = \text{average loss in the worst tail of mass } 1-\alpha.$$

Convexity	CVaR is convex in the decisions (if L is); VaR in general is not
Subadditivity	CVaR rewards diversification; VaR may violate it
Tail	VaR does not distinguish two tails with the same quantile but different severity
Optimization	CVaR admits a scenario-based LP formulation; VaR does not

Data of the scenario-based model: the number of scenarios $k \in \mathbb{Z}_{\geq 1}$, indexed by $s \in \{1, 2, \ldots, k\}$; for each scenario, the probability $\pi_s \in \mathbb{Q}_{\geq 0}$ (with $\sum_{s=1}^{k} \pi_s = 1$) and the loss $\ell_s(\boldsymbol{x})$, a linear function of the decision vector $\boldsymbol{x}$, constrained to a polyhedron X; the confidence level $\alpha \in (0, 1)$.

Besides the decisions $\boldsymbol{x}$, we introduce one free variable and k non-negative variables:

$$\begin{cases} \eta = \text{candidate loss threshold (at the optimum: a VaR)} \\ \xi_s = \text{loss in excess of } \eta \text{ in scenario } s \end{cases} \qquad \forall s \in \{1, 2, \ldots, k\}.$$

Using these variables, the Rockafellar–Uryasev LP model is the following:

Rockafellar–Uryasev LP model for the CVaR

$$\min \quad \eta + \frac{1}{1-\alpha} \sum_{s=1}^{k} \pi_s \, \xi_s \tag{13.1a}$$

$$\text{subject to} \quad \xi_s \geq \ell_s(\boldsymbol{x}) - \eta, \qquad \forall s \in \{1, 2, \ldots, k\}, \tag{13.1b}$$

$$\boldsymbol{x} \in X, \tag{13.1c}$$

$$\eta \gtreqless 0, \tag{13.1d}$$

$$\xi_s \geq 0, \qquad \forall s \in \{1, 2, \ldots, k\}. \tag{13.1e}$$

Description of the objective function and of the constraints of the LP model:

- the linear objective function (13.1a) is the threshold η plus the weighted average of the tail excesses, amplified by $1/(1-\alpha)$; *at the optimum η^* is a VaR and the objective value is the CVaR*: a single LP provides both measures;
- the linear constraints (13.1b) define the excesses: only the scenarios with a loss beyond the threshold contribute (k linear constraints);
- constraint (13.1c) collects the constraints of the underlying decision problem (polyhedral);
- constraints (13.1d) and (13.1e) define the variables: η is *free* (the symbol $\gtreqless$), ξ_s non-negative.

Line-by-line explanation

The variable $\xi_s = (\ell_s - \eta)^+$ is the loss in excess of the candidate threshold η (the same positive-part trick as in the Newsvendor). Minimizing over η balances two forces: raising η reduces the excesses but pushes up the threshold; the equilibrium point is exactly the α quantile. If X is a polyhedron and ℓ_s is linear, the whole model is an LP.

13.3 Worked example by hand

Six scenarios, $\alpha = 0.80$

Equally likely losses $\{2, 4, 5, 7, 12, 20\}$.
VaR. The cumulative distribution reaches 0.80 at the fifth ordered loss: $\mathrm{VaR}_{0.80} = 12$.
CVaR with the formula. The worst tail has mass 0.20; the scenario with loss 20 weighs $1/6 = 0.167$, and the remaining mass 0.033 sits *on the point* 12:

$$\mathrm{CVaR}_{0.80} = 12 + \frac{1}{0.20} \cdot \frac{1}{6}(20 - 12) = 12 + 6.67 = 18.67.$$

Check with the LP (script): $\eta^* = 12$, $\xi = (0.0, 0, 0, 0, 8)$, value 18.6667. The formula correctly handles the mass "straddling" the quantile without selecting scenarios with binary variables.
Comparison. Average loss 8.33; VaR 12; CVaR 18.67: three numbers, three stories. The mean reassures, the VaR gives a threshold, the CVaR says how much it hurts when things go badly.

13.4 Case study 1: mean-CVaR portfolio against Markowitz

The same eight securities as in Chapter 6, but $k = 220$ monthly scenarios generated with *fat tails* (Student t market factor, 4 degrees of freedom): the world in which the CVaR shows its value. Scenario loss: $\ell_s(\boldsymbol{x}) = -\sum_{i=1}^{n} r_{is}\, x_i$, where r_{is} is the return of security i in scenario s. We minimize $\mathrm{CVaR}_{0.90}$ with expected return $\geq 8\%$ per year.

```python
eta = m.addVar(lb=-GRB.INFINITY, name="eta")   # threshold: free!
xi  = m.addVars(S, name="xi")
m.addConstr(x.sum() == 1)
m.addConstr(gp.quicksum(mu[i]*x[i] for i in range(n)) >= r_min)
m.addConstrs((xi[s] >= -gp.quicksum(R[s, i]*x[i] for i in range(n))
                     - eta for s in range(S)))
m.setObjective(eta + gp.quicksum(xi[s] for s in range(S))
                     / (S * (1 - alpha)), GRB.MINIMIZE)
```

```
              | mean loss     | VaR90  | CVaR90   (monthly losses)
  mean-CVaR |       -0.0067 | 0.0327 | 0.0489
  Markowitz |       -0.0067 | 0.0329 | 0.0531
Compositions: mean-CVaR: ENE 9.9 FIN 1.9 TEC 10.5 SAN 8.6 UTL 46.5 MAT 22.5
              Markowitz: ENE 18.4 FIN 6.4 TEC 6.5 SAN 13.6 UTL 40.4 MAT 14.6
```

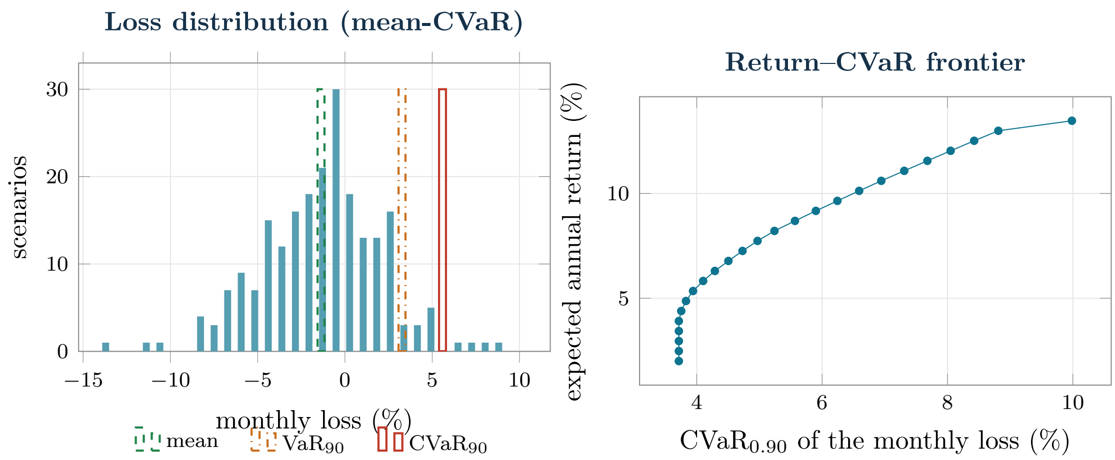


Figure 13.1: Left: histogram of the monthly losses of the mean-CVaR portfolio over the 220 scenarios, with the mean (green), VaR_{90} (orange) and CVaR_{90} (red): the CVaR is the average of the tail to the right of the VaR. Right: return–CVaR frontier obtained by varying the minimum required return.

Managerial insight

For the same required return and the same average loss, the mean-CVaR portfolio cuts the tail: CVaR_{90} of 4.89% against 5.31% for Markowitz (−8%). It does so by reducing the high-beta securities (ENE from 18% to 10%) in favour of UTL and — a surprise — of a share of TEC which, although mediocre on average, moves against the trend in the worst scenarios of this sample. Variance penalizes symmetrically above and below the mean; the CVaR looks only where it hurts.

13.5 Case study 2: two-stage supply chain

Before observing the scenario we *book* capacity from the suppliers; then, having seen the scenario, we decide the recourse. F1 is cheap but fragile (in 12% of the scenarios its availability collapses to 30%); F2 is expensive but reliable. Shortage penalty: 40 €/unit; $k = 400$ scenarios.

Data: two suppliers indexed by $a \in \{1, 2\}$ (1 cheap but fragile, 2 expensive but reliable) with booking costs $c_a^{\text{pre}} \in \mathbb{Q}_{>0}$ and purchase costs $c_a^{\text{uso}} \in \mathbb{Q}_{>0}$; k scenarios $s \in \{1, 2, \ldots, k\}$ with demand $d_s \in \mathbb{Q}_{\geq 0}$ and availability $\delta_{as} \in (0, 1]$ (the datum that models the failure); shortage penalty $g \in \mathbb{Q}_{>0}$; risk weight $\lambda \in [0, 1]$ and level $\alpha \in (0.1)$.

We introduce the non-negative variables x_a (capacity booked from supplier a, *first-stage*: the same in all scenarios), f_{as} (purchase from supplier a in scenario s) and u_s (unmet demand in scenario s), the latter two being *recourse* variables.

Two-stage model with CVaR

$$\min \quad (1-\lambda)\Big[\sum_{a=1}^{2} c_a^{\text{pre}} x_a + \sum_{s=1}^{k} \pi_s\, \phi_s\Big] + \lambda\, \mathrm{CVaR}_\alpha \tag{13.2a}$$

$$\text{subject to} \quad f_{as} \leq \delta_{as}\, x_a, \quad \forall a \in \{1.2\},\ \forall s \in \{1, 2, \ldots, k\}, \tag{13.2b}$$

$$f_{1s} + f_{2s} + u_s \geq d_s, \quad \forall s \in \{1, 2, \ldots, k\}, \tag{13.2c}$$

$$x_a,\ f_{as},\ u_s \geq 0, \quad \forall a \in \{1.2\},\ \forall s \in \{1, 2, \ldots, k\}, \tag{13.2d}$$

where $\phi_s = \sum_{a=1}^{2} c_a^{\text{uso}} f_{as} + g\, u_s$ is the recourse cost of scenario s and CVaR_α is the CVaR of the total cost, linearized with the variables η, ξ_s of (13.1).

Description of the objective function and of the constraints of the model:

- the objective function (13.2a) combines, with weight λ, the average cost (bookings plus the expected value of purchases and penalties) and the CVaR of the total cost;
- the linear constraints (13.2b) impose that in scenario s supplier a can be used only up to the booked capacity x_a, reduced by the availability δ_{as} ($2\,k$ linear constraints);

- the linear constraints (13.2c) impose that demand be covered by the flows or, failing that, be counted as a shortage u_s, which in the objective pays the penalty: the model is never infeasible, it "buys" infeasibility at the price of the penalty (k linear constraints);
- constraints (13.2d) define the variables of the model.

```
lambda = 0.0: book suppl.1 161.5  suppl.2 134.2
              mean cost 1.724   CVaR90 2.902   service 98.7%
lambda = 0.5: book suppl.1  92.7  suppl.2 212.7
              mean cost 1.803   CVaR90 2.217   service 99.9%
lambda = 1.0: book suppl.1  68.5  suppl.2 236.9
              mean cost 1.906   CVaR90 2.139   service 99.0%
```

Managerial insight

The risk-neutral decision maker buys from the cheap supplier and accepts that in 12% of the scenarios the shortage makes costs explode (CVaR 2.902). As λ grows, capacity migrates towards the reliable supplier: +79 € of average cost buy −685 € of CVaR. This is the *cost of resilience*, and it is exactly the number needed in a discussion about reshoring and dual sourcing.

13.6 Sensitivity analysis

The parameters to vary, with the expected effect:

α	higher $\Rightarrow$ a more extreme tail estimated from fewer scenarios: more scenarios are needed for stable estimates
λ	controls the trade-off between average performance and protection
π_s	reweighted scenarios or stress tests (adding extreme scenarios)
$\lvert S\rvert$	statistical stability: repeat with different samples, as in Chapter 12

With $\alpha = 0.99$ and $k = 220$ scenarios the tail contains only 2–3 scenarios: the estimated CVaR is almost noise. Rule of thumb: at least a few dozen scenarios are needed *beyond* the quantile. Moreover, with discrete distributions the VaR may not be unique (η^* is *a* VaR): always compare η^* with the empirical quantile.

13.7 Exercises

Exercise 13.1 — check

Recompute by hand the VaR and CVaR of the 6-scenario example with $\alpha = 0.90$ and check with the LP. Careful: the tail has mass 0.10 and the scenario with loss 20 weighs $0.167 > 0.10$.

Exercise 13.2 — subadditivity

Build a scenario-based example with two activities in which $\mathrm{VaR}(L_1 + L_2) > \mathrm{VaR}(L_1) + \mathrm{VaR}(L_2)$ and check that the CVaR does not violate subadditivity.

Exercise 13.3 — frontier in α

For the portfolio, plot the optimal CVaR for $\alpha \in \{0.80; 0.90; 0.95; 0.99\}$ at equal return: how does the composition change? When do the numerical instabilities begin?

Exercise 13.4 — CVaR-constrained

Reformulate the portfolio as $\max \sum_{i=1}^{n} \mu_i x_i$ subject to $\mathrm{CVaR}_{0.90} \leq \bar{c}$: check that by varying $\bar{c}$ one traces the same frontier and read the dual of the constraint.

Exercise 13.5 — stress test

Add to the supply chain an extreme scenario (demand +50% and F1 at 10%) with probability 1%: how much does the optimal booking change for $\lambda = 0.5$? And the CVaR?

Chapter 14

Arbitrage and arbitrage-free pricing

Class: LP

Script: `python/lab14_arbitrage.py`
Online page

14.1 Managerial motivation

An *arbitrage* is a trading strategy that creates money out of nothing: it collects cash today with no risk of a loss tomorrow, or it costs nothing today and can only make money tomorrow. In a well-functioning market such opportunities cannot last: anyone would exploit them on an unlimited scale, and prices would adjust immediately. "Absence of arbitrage" is therefore the working assumption of the whole of pricing finance: of the trading desks that look for inconsistencies among quoted instruments, of the risk managers who validate the prices in a price list, of whoever has to price a new security consistently with the existing ones.

The didactic surprise is that all this is *linear programming*: detecting an arbitrage means solving an LP, and the duality of Chapter 2 — shadow prices, complementary slackness, strong duality — becomes the theory of pricing. It is the cleanest application of the primal-dual pair in the whole set of lecture notes.

In this chapter we will learn to: (1) formulate the search for an arbitrage as an LP; (2) read the *risk-neutral probabilities* in the duals of the constraints; (3) compute the interval of consistent prices of a security; (4) price a new security (an option) in a complete or incomplete market.

14.2 Guided construction of the model

The problem in words. The market has two instants: today ($t = 0$), when we buy or sell securities at the current prices, and tomorrow ($t = 1$), when one out of m *states of the world* occurs and each security pays a known amount that depends on the state. *We decide* the quantities of each security in the portfolio (positive if bought, negative if sold short). *The objective* is to minimize the cost of the portfolio today. *The constraints* impose that tomorrow, in *every* state of the world, the portfolio does not lose: if under these rules one manages to spend less than zero (that is, to collect cash), that is an arbitrage.

Data (input of the model)

Symbol	Type	Meaning
n	$\in \mathbb{Z}_{\geq 1}$	number of risky securities; the securities are numbered $0, 1, \ldots, n$ and security 0 is risk-free
m	$\in \mathbb{Z}_{\geq 1}$	number of states of the world at $t = 1$
r	$\in \mathbb{Q}_{\geq 0}$	risk-free rate; $R = 1 + r$ is the gross return of security 0, the same in every state
s_i^0	$\in \mathbb{Q}_{>0}$	price of security i today ($s_0^0 = 1$: security 0 costs 1 and pays R)
s_{ij}^1	$\in \mathbb{Q}_{\geq 0}$	payoff of security i in state j ($s_{0j}^1 = R$ for every state j)

Decision variables

We introduce one free variable for each of the $n + 1$ securities:

$$x_i = \text{position in security } i \quad (> 0 \text{ long}, < 0 \text{ short}), \qquad \forall i \in \{0, 1, \ldots, n\}.$$

Using these variables, the LP model for the search of an arbitrage is the following:

LP model for the search of an arbitrage

$$\min \sum_{i=0}^{n} s_i^0 \, x_i \tag{14.1a}$$

$$\text{subject to} \quad \sum_{i=0}^{n} s_{ij}^1 \, x_i \geq 0, \qquad \forall j \in \{1, 2, \ldots, m\}, \tag{14.1b}$$

$$x_i \gtreqless 0, \qquad \forall i \in \{0, 1, \ldots, n\}. \tag{14.1c}$$

Description of the objective function and of the constraints of the model:

- the linear objective function (14.1a) minimizes the cost of the portfolio today: a negative value is an immediate cash inflow;
- the linear constraints (14.1b) impose that tomorrow the portfolio does not lose in any of the m states of the world (m linear constraints);
- constraints (14.1c) define the variables, which are free: one can buy and sell short ($n + 1$ variables).

Reading the outcome: type A, type B, and the normalization

The null portfolio is always feasible, hence the optimum is always ≤ 0. Three possible outcomes:

- **optimum** < 0 (*type A arbitrage*): one collects cash today with no risk tomorrow. The constraints are homogeneous: the strategy can be scaled at will and the model is *unbounded*. To obtain a finite certificate one adds the *normalization* $\sum_{i=0}^{n} s_i^0 \, x_i \geq -1$ ("collect at most 1 today"): the optimum becomes -1 and the solution is a concrete strategy;
- **optimum** $= 0$ **with some state constraint slack in every optimal solution**

(*type B arbitrage*): today it costs nothing, tomorrow one never loses and in at least one state one gains;

- **optimum** $= 0$ **and all constraints active at the optimum**: no arbitrage, the prices are *consistent*.

14.3 Duality is pricing: the risk-neutral probabilities

We apply to model (14.1) the duality rules of Chapter 2 (minimization primal: table read from right to left). Each state constraint (14.1b), of type $\geq$, generates a dual variable $p_j \geq 0$; each free variable x_i generates a dual equality constraint; the right-hand sides (all zero) become the objective of the dual:

The dual: pricing measures

$$\max \sum_{j=1}^{m} 0 \cdot p_j \tag{14.2a}$$

$$\text{subject to} \quad \sum_{j=1}^{m} s_{ij}^1 \, p_j = s_i^0, \qquad \forall i \in \{0, 1, \ldots, n\}, \tag{14.2b}$$

$$p_j \geq 0, \qquad \forall j \in \{1, 2, \ldots, m\}. \tag{14.2c}$$

- objective (14.2a) is identically zero: of the dual only *feasibility* matters (every feasible solution is optimal);
- constraints (14.2b) say that today's price of every security is the sum of its payoffs weighted with the p_j: the duals *price* the securities ($n + 1$ linear constraints);
- constraints (14.2c) define the variables (m non-negative variables).

The dual constraint of security 0 gives $\sum_{j=1}^{m} R \, p_j = 1$, that is, $\sum_{j=1}^{m} p_j = 1/R$: the numbers $q_j = R \, p_j$ are non-negative and sum to 1 — they are *probabilities*, called **risk-neutral**. With them constraints (14.2b) read:

$$s_i^0 = \frac{1}{R} \sum_{j=1}^{m} q_j \, s_{ij}^1, \qquad \forall i \in \{0, 1, \ldots, n\} :$$

today's price is the discounted expected value of tomorrow's payoffs, under the probabilities q.

Fundamental theorem of asset pricing

There is no arbitrage (neither of type A nor of type B) *if and only if* there exists a risk-neutral probability measure with $q_j > 0$ for every $j \in \{1, 2, \ldots, m\}$.

Why it is true: strong duality and complementary slackness

It is Chapter 2 at work. If there is no type A arbitrage, the optimum of the primal is 0: by *strong duality* the dual is feasible, that is, a pricing measure $p \geq 0$ exists. If there is no type B arbitrage either, all the state constraints are active at the optimum, and by *complementary* *slackness* there exists a dual solution with $p_j > 0$ for every state: multiplying by R one obtains the $q_j > 0$. Conversely, if there exists $q > 0$, every strategy with non-negative payoffs has cost $\sum_i s_i^0 x_i = \frac{1}{R}\sum_j q_j\big(\sum_i s_{ij}^1 x_i\big) \geq 0$: no arbitrage.

14.4 Worked example by hand

Three states, an obvious arbitrage

Market with $m = 3$ states, $r = 4\%$ ($R = 1.04$) and two risky securities with payoffs $s_{1\cdot}^1 = (10, 15, 13)$ and $s_{2\cdot}^1 = (30, 15, 25)$; today's prices $s_1^0 = 6$ and $s_2^0 = 20$.

- **Step 1 — a candidate strategy.** We buy one unit of each risky security ($x_1 = x_2 = 1$, outlay 26) while selling short the risk-free security for 27 ($x_0 = -27$). Cost today:
$$1 \cdot (-27) + 6 \cdot 1 + 20 \cdot 1 = -1 :$$
we collect 1.
- **Step 2 — no risk tomorrow.** In the three states the portfolio pays
$$-27 \cdot 1.04 + 10 + 30 = 11.92, \quad -27 \cdot 1.04 + 15 + 15 = 1.92, \quad -27 \cdot 1.04 + 13 + 25 = 9.92,$$
all ≥ 0: it is a type A arbitrage.
- **Step 3 — why the model is unbounded.** By doubling the strategy one collects 2 today with payoffs still ≥ 0: without the normalization the cost can go down as far as one wishes.

14.5 Case study

The same market as in the example ($R = 1.04$, payoffs in `data/arbitraggio_payoff.csv`), analysed with the script in two price configurations: the one with arbitrage ($s_1^0 = 6$, $s_2^0 = 20$) and a consistent one ($s_1^0 = 13$, $s_2^0 = 18.6923$).

```
x = m.addVars(n1, lb=-GRB.INFINITY, name="x")    # free positions
v_stato = m.addConstrs(
    (gp.quicksum(s1[i, j] * x[i] for i in range(n1)) >= 0
     for j in range(n_stati)), name="stato")
if normalizza:   # without it: with arbitrage the model is unbounded
    m.addConstr(gp.quicksum(s0[i] * x[i] for i in range(n1)) >= -1)
m.setObjective(gp.quicksum(s0[i] * x[i] for i in range(n1)),
               GRB.MINIMIZE)
```

14.6 Results

```
Prices (6, 20):      without normalization: UNBOUNDED (type A)
  normalized: optimum -1  ->  collect 1 today, payoff (0; 0; 0.20) >= 0
Prices (13, 18.69):  optimum 0, no arbitrage
  duals of the state constraints p* = (0.2846; 0.6769; 0)
```

```
  q = R p*              = (0.296;  0.704;  0)   sum 1
  pricing check: s0 = (1; 13; 18.6923) reproduced exactly
```

The duals do the pricing

With consistent prices the LP returns optimum 0 and, in the duals of the state constraints, the pricing measure: $q = (0.296; 0.704; 0)$. Every security — and every *new* security — is priced as the discounted expected value of the payoffs under q. The value $q_3 = 0$ signals an informational degeneracy: state 3 is not needed to price these securities (its payoff can be replicated with the other two states), and it is the reason why here the "if and only if" of the theorem requires $q > 0$: with $q_3 = 0$ a latent type B arbitrage on state 3 remains.

14.7 Sensitivity analysis: the interval of consistent prices

If only security 0 (price 1) and security 1 (price 13) are quoted, which prices of security 2 do *not* create arbitrage? Two LPs on the pricing measures — minimizing and maximizing $\sum_j s^1_{2j}\, p_j$ over the $p \geq 0$ consistent with the quoted securities — give the interval

$$s^0_2 \in [18.69,\ 21.54].$$

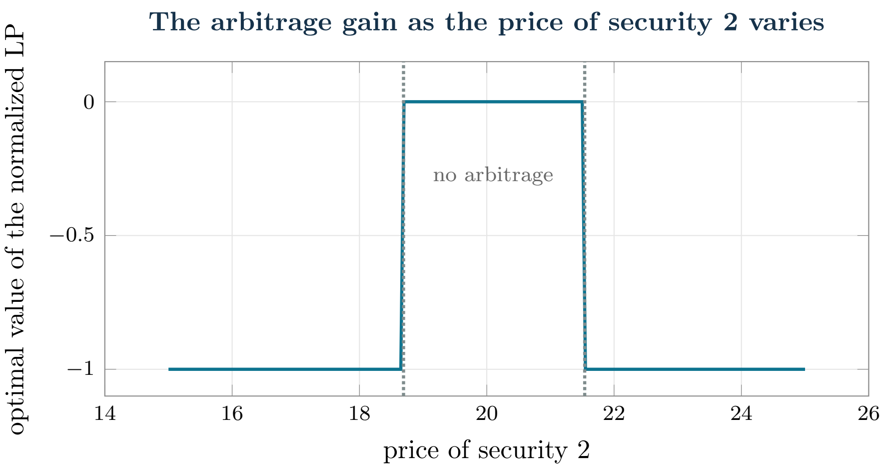


Figure 14.1: Optimal value of the normalized arbitrage LP as the price of security 2 varies (securities 0 and 1 quoted at 1 and 13). Inside the interval [18.69, 21.54] (dashed lines) the value is 0: consistent prices; outside it, the value drops to −1: an arbitrage exists, in one direction or the other.

Pricing a new security: complete vs incomplete

The same pair of LPs prices any new security. A *call* on security 1 with strike 12 pays $\max(s^1_{1j} - 12, 0) = (0,\ 3,\ 1)$ in the three states.

- **Complete market** (securities 0, 1, 2 quoted): the pricing measure is unique and so is the price: 2.0308.
- **Incomplete market** (only securities 0 and 1): the consistent measures are infinitely many and the price is an *interval*, [1.4615, 2.0308]: below the minimum or above the maximum, anyone could build an arbitrage by combining the call and the quoted securities.

14.8 Exercises

Exercise 14.1 — scale of the arbitrage

With prices $(6, 20)$, find a strategy that collects 5 today with no risk tomorrow. (Hint: the solutions of the homogeneous LP can be scaled.)

Exercise 14.2 — endpoint of the interval

Set the price of security 2 at the upper endpoint 21.5385 and solve the arbitrage LP: check that the optimum is 0 and compute the associated pricing measure. What happens just above, at 21.60?

Exercise 14.3 — pricing a call

In the complete market (prices 1, 13, 18.6923), price a call on security 2 with strike 20 (payoff $\max(s^1_{2j} - 20, 0)$) and check the price with the risk-neutral probabilities of the case study.

Part IV

Optimization and machine learning

Chapter 15

Support Vector Machine: optimization for machine learning

Class: convex QP

Script: `python/lab15_svm.py`
Online page

15.1 Managerial motivation

The Support Vector Machine connects convex optimization directly to machine learning: the classifier is obtained by solving a QP, with a clear geometric reading. The managerial applications are immediate: credit risk, churn, predictive maintenance, quality control, fraud. In this chapter we do *not* use ML libraries: every model is a QP written and solved with Gurobi, because the goal is to understand *what* a classifier optimizes.

A modelling note: the labels $y_i \in \{-1, +1\}$ are *observed data*, not variables — the variables (coefficients, intercept, slacks, multipliers) are all continuous.

This chapter closes the circle: the same tools used for production, logistics and finance — convex QPs, duality, multipliers — are the engine of one of the most elegant classifiers of machine learning. Understanding the SVM as an optimization problem, and not as a black box from a library, gives full control over what the model is really doing.

In this chapter we will learn to: (1) formulate hard and soft margin as QPs; (2) derive and solve the dual, identifying the support vectors; (3) obtain non-linear boundaries with kernels while staying in the convex world; (4) handle unbalanced classes with asymmetric costs and evaluate with the right metrics.

15.2 Hard margin: the geometry

The problem in words. *We decide* the coefficients of a hyperplane that separates two classes of observations. *The objective* is to maximize the margin, that is, the distance between the hyperplane and the closest points of the two classes (a classifier with a wide margin generalizes better). *The constraints* require every observation to lie on the right side of the hyperplane — in the soft margin, up to a slack paid in the objective.

Data (input of the model)

Symbol	Type	Meaning
n	$\in \mathbb{Z}_{\geq 1}$	number of observations of the training set, indexed by $i \in \{1, 2, \ldots, n\}$
p	$\in \mathbb{Z}_{\geq 1}$	number of features, indexed by $j \in \{1, 2, \ldots, p\}$
x_{ij}	$\in \mathbb{Q}$	value of feature j for observation i; $\boldsymbol{x}_i \in \mathbb{Q}^p$ is the vector of observation i
y_i	$\in \{-1, +1\}$	*observed* class label of i (a datum, not a variable)
C	$\in \mathbb{Q}_{>0}$	weight of the margin violations (soft margin only); upper case to follow the machine learning literature, an exception to the convention that reserves capital letters for sets and matrices

Decision variables

We introduce the following $p+1$ free variables:

$$\begin{cases} w_j = \text{coefficient } j \text{ of the separating hyperplane} \\ b = \text{intercept of the hyperplane} \end{cases} \qquad \forall j \in \{1, 2, \ldots, p\},$$

which define the hyperplane $\sum_{j=1}^{p} w_j \, x_j + b = 0$.

Using these variables, the hard margin QP model is the following:

Hard margin QP model

$$\min \quad \frac{1}{2} \sum_{j=1}^{p} w_j^2 \tag{15.1a}$$

$$\text{subject to} \quad y_i \Big(\sum_{j=1}^{p} w_j \, x_{ij} + b \Big) \geq 1, \qquad \forall i \in \{1, 2, \ldots, n\}, \tag{15.1b}$$

$$w_j \gtreqless 0, \qquad \forall j \in \{1, 2, \ldots, p\}, \tag{15.1c}$$

$$b \gtreqless 0. \tag{15.1d}$$

Description of the objective function and of the constraints of the QP model:

- the convex quadratic objective function (15.1a) minimizes $\frac{1}{2}\|\boldsymbol{w}\|^2$; since the geometric margin between the supporting hyperplanes is $2/\|\boldsymbol{w}\|$, minimizing it is equivalent to *maximizing the margin*;
- the linear constraints (15.1b) impose that every observation lies on the right side of the hyperplane, at a distance of at least one unit in the scale of $\boldsymbol{w}$ (n linear constraints);
- constraints (15.1c) and (15.1d) define the variables of the model, all free.

Two points, everything by hand

$\boldsymbol{x}_1 = (0.0)$ with $y_1 = -1$ and $\boldsymbol{x}_2 = (2.2)$ with $y_2 = +1$.

- **Step 1 — constraints.** $-(b) \geq 1$ and $2w_1 + 2w_2 + b \geq 1$, that is, $b \leq -1$ and $2(w_1 + w_2) \geq 1 - b \geq 2$.
- **Step 2 — symmetry and optimum.** To minimize $w_1^2 + w_2^2$ with $w_1 + w_2 \geq 1$ it is best to take $w_1 = w_2 = t$ and the inequalities active: $4t + b = 1$, $b = -1 \Rightarrow t = 1/2$. Hence $\boldsymbol{w} = (0.5;\, 0.5)$, $b = -1$: the hyperplane $0.5x_1 + 0.5x_2 = 1$ is the perpendicular bisector of the segment between the two points.
- **Step 3 — margin.** $2/\|\boldsymbol{w}\| = 2/\sqrt{0.5} = 2\sqrt{2} = 2.83$: exactly the distance between the two points. Both constraints are active: both points are *support vectors*. With real overlapping data, however, the hard margin is infeasible: the soft margin is needed.

15.3 Soft margin

With real overlapping data the hard margin is infeasible: one adds n non-negative slack variables,

$$\xi_i = \text{margin violation of observation } i, \qquad \forall i \in \{1, 2, \ldots, n\},$$

where $\xi_i > 1$ corresponds to a misclassification.

Soft margin QP model (primal)

$$\min \quad \frac{1}{2}\sum_{j=1}^{p} w_j^2 + C\sum_{i=1}^{n} \xi_i \tag{15.2a}$$

$$\text{subject to} \quad y_i\Big(\sum_{j=1}^{p} w_j\, x_{ij} + b\Big) \geq 1 - \xi_i, \qquad \forall i \in \{1, 2, \ldots, n\}, \tag{15.2b}$$

$$w_j \gtreqless 0, \qquad \forall j \in \{1, 2, \ldots, p\}, \tag{15.2c}$$

$$b \gtreqless 0, \tag{15.2d}$$

$$\xi_i \geq 0, \qquad \forall i \in \{1, 2, \ldots, n\}. \tag{15.2e}$$

Description of the objective function and of the constraints of the QP model:

- the convex quadratic objective function (15.2a) balances two terms: the simplicity of the model ($\frac{1}{2}\|\boldsymbol{w}\|^2$, wide margin) and the cost of the violations on the training set ($C\sum_{i=1}^{n}\xi_i$); C is the price assigned to the violations;
- the linear constraints (15.2b) require correct classification *up to the slack* ξ_i, paid in the objective (n linear constraints);
- constraints (15.2c), (15.2d) and (15.2e) define the variables of the model: w_j and b free, ξ_i non-negative.

Case study: credit risk

90 customers, two standardized indicators (capital soundness, punctuality of payments), 55 creditworthy and 35 defaulting, partially overlapping (`data/svm_clienti.csv`).

```
C =  0.05: w = (0.678, 0.474)  b = -0.148 | margin 2.418 | errors 2 | in the
    margin 22
C =  1.00: w = (1.433, 0.611)  b = -0.236 | margin 1.284 | errors 2 | in the
    margin  7
C = 20.00: w = (5.058, 2.338)  b = -2.936 | margin 0.359 | errors 0 | in the
    margin  3
```

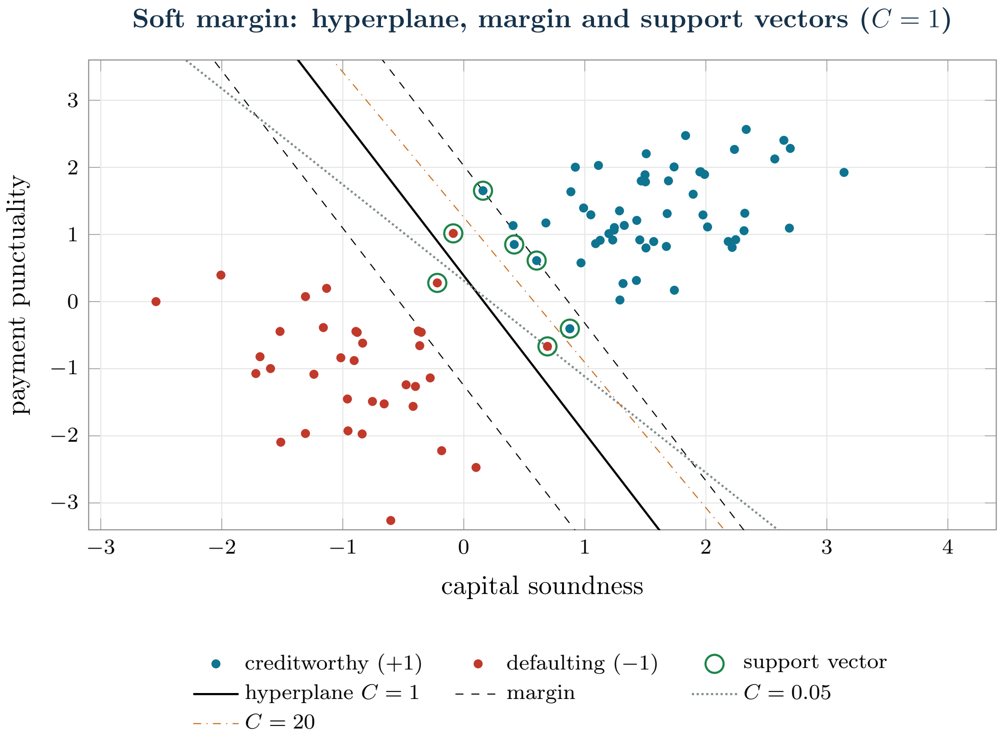


Figure 15.1: The 90 customers in the plane of the two indicators: creditworthy (teal), defaulting (red), support vectors circled in green. The black line is the optimal hyperplane for $C = 1$ with its margin lines (dashed); the grey and orange lines are the hyperplanes for $C = 0.05$ (wide margin) and $C = 20$ (narrow margin).

Managerial insight

A small C buys a wide margin while accepting 22 points inside the margin; a large C chases every point of the training set (zero errors, a very narrow margin): this is the **regularization–fit** trade-off, and it must be chosen with validation data, never by looking at the test set. Note that the three lines are similar but not equal: the slope changes because the "paid for" points change.

15.4 The dual formulation and the support vectors

By associating a multiplier α_i with each classification constraint (15.2b), one obtains n new variables:

$$\alpha_i = \text{multiplier of the classification constraint of observation } i, \qquad \forall i \in \{1, 2, \dots, n\}.$$

Dual QP model of the soft margin

$$\max \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \, \alpha_j \, y_i \, y_j \, \boldsymbol{x}_i' \boldsymbol{x}_j \tag{15.3a}$$

$$\text{subject to} \quad \sum_{i=1}^{n} \alpha_i \, y_i = 0, \tag{15.3b}$$

$$\alpha_i \leq C, \qquad \forall i \in \{1, 2, \ldots, n\}, \tag{15.3c}$$

$$\alpha_i \geq 0, \qquad \forall i \in \{1, 2, \ldots, n\}. \tag{15.3d}$$

Description of the objective function and of the constraints of the dual model:

- the concave quadratic objective function (15.3a) depends on the data *only* through the inner products $\boldsymbol{x}_i' \boldsymbol{x}_j$: this is the observation that opens the door to the kernel;
- the linear constraint (15.3b) follows from stationarity with respect to the intercept b (one linear constraint);
- constraints (15.3c) bound each multiplier by the price C of the violations (n linear constraints);
- constraints (15.3d) define the variables of the model.

Primal-dual relation: $\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i \, y_i \, \boldsymbol{x}_i$ and $f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}' \boldsymbol{x} + b)$.

```
w from the dual = (1.433, 0.611)   b = -0.236     (identical to the primal)
dual value 4.7868 = primal value (strong duality)
alpha = 0          : 83 points  -> do NOT affect the boundary
0 < alpha < C      :  2 points  -> support vectors exactly on the margin
alpha = C          :  5 points  -> inside the margin or misclassified
```

Why they are called support vectors

The solution depends *only* on the 7 points with $\alpha_i > 0$: it is they that "support" the hyperplane; the other 83 could disappear without changing anything. The KKT complementarity organizes everything: $\alpha_i = 0 \Rightarrow$ point beyond the margin; $0 < \alpha_i < C \Rightarrow$ exactly on the margin (from these one recovers b); $\alpha_i = C \Rightarrow$ the constraint "pays" a slack $\xi_i > 0$. Moreover, the dual depends on the data only through the inner products $\boldsymbol{x}_i' \boldsymbol{x}_j$: this observation opens the door to the kernel.

15.5 Kernels: non-linear boundaries from a convex QP

By replacing $\boldsymbol{x}_i' \boldsymbol{x}_j$ with a kernel $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ — for example the Gaussian RBF (*Radial Basis Function*) kernel $K(\boldsymbol{x}, \boldsymbol{z}) = e^{-\gamma \|\boldsymbol{x} - \boldsymbol{z}\|^2}$ — one classifies in a transformed space without ever building it: $f(\boldsymbol{x}) = \text{sign}\big(\sum_{i=1}^{n} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b\big)$. The boundary in the original space can be arbitrarily curved, but the training problem *remains a convex QP* if the kernel is valid (positive semidefinite matrix).

Case study: "ring-shaped" anomalies

90 process observations: normal operation in the centre, the anomalies arranged in a ring — no straight line can separate them.

```
gamma = 0.1: training errors  1   support vectors 20
gamma = 0.7: training errors  0   support vectors 15
gamma = 5.0: training errors  0   support vectors 61  <- "memorizes" the points
```

RBF kernel: the level-0 boundary for three values of γ

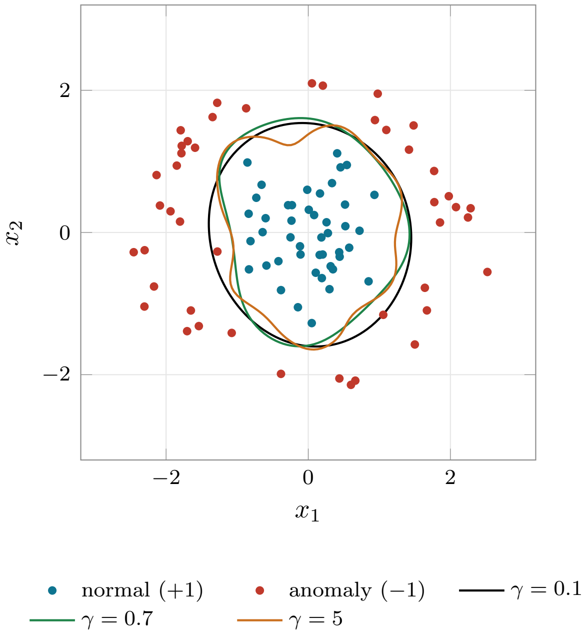


Figure 15.2: The "ring-shaped" dataset (normal operation in the centre, anomalies in a ring) and the decision boundaries of the RBF kernel for three values of γ: with $\gamma = 0.1$ the boundary is a regular circle, with $\gamma = 5$ it breaks up around the individual points (overfitting).

> With $\gamma = 5$ the boundary breaks up into islands around the individual points and the support vectors rise to 61 out of 90: the model is memorizing the training set (overfitting), while still remaining a convex QP — convexity of *training* and the ability to *generalize* are different properties. C and γ must be chosen together by cross-validation.

15.6 Unbalanced classes and asymmetric costs

In managerial applications the important class is rare and an error on the rare cases costs more. It is enough to weight the slacks: $C_- \sum_{i:\, y_i=-1} \xi_i + C_+ \sum_{i:\, y_i=+1} \xi_i$.

```
equal costs (C=1)          : precision 1.00   recall defaulters 0.94
10x cost on the defaulters : precision 0.97   recall defaulters 1.00
```

Managerial insight

With the 10× cost the boundary moves back towards the "healthy" class: no defaulter escapes (recall 100%) at the price of a few more false alarms (precision from 100% to 97%). The weights must reflect the *decision costs* (how much a loan granted to someone who will not repay costs vs. a good customer turned down), not the frequencies of the classes.

15.7 Support Vector Regression

SVR estimates a continuous function while ignoring the errors within a tube of width ε: $\min \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$ with $|y_i - (\boldsymbol{w}'\boldsymbol{x}_i + b)| \leq \varepsilon + \xi_i^{(*)}$ for every i.

```
SVR line: demand = -10.86 * price + 220.32    (true: -11p + 220)
Tube eps = 8: 14 points out of 40 outside the tube -> only they determine the
   line
```

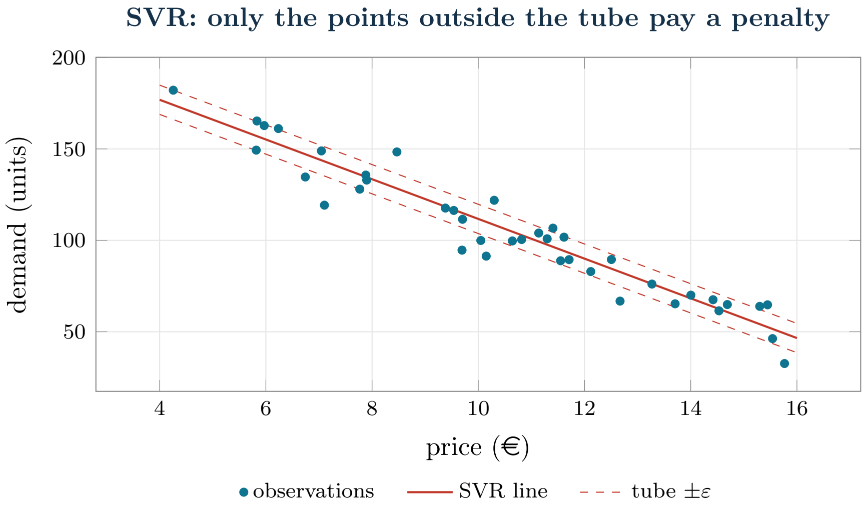


Figure 15.3: Support Vector Regression on the price–demand relation: the 40 observations, the estimated line and the tolerance tube $\pm\varepsilon = 8$: only the 14 points outside the tube pay a penalty and determine the line.

15.8 Exercises

Exercise 15.1 — check

Solve by hand the hard margin with $\boldsymbol{x}_1 = (0.0)$, $y_1 = -1$ and $\boldsymbol{x}_2 = (4.0)$, $y_2 = +1$, and check with Gurobi ($\boldsymbol{w} = (0.5; 0)$, $b = -1$, margin 4).

Exercise 15.2 — complementarity

For $C = 1$, list from the script the 7 points with $\alpha_i > 0$ and check numerically the three complementarity conditions ($\alpha_i = 0 \Rightarrow \xi_i = 0$ and margin ≥ 1; and so on).

Exercise 15.3 — validation curve

Split the 90 customers into training (60) and validation (30); plot the validation error for $C \in \{0.01; 0.1; 1; 10; 100\}$ and choose C. Is the training error monotone in C? And the validation error?

Exercise 15.4 — decision threshold

For the same model ($C = 1$), move the threshold: classify as defaulting if $\boldsymbol{w}'\boldsymbol{x} + b < \theta$ with $\theta \in [-1, 1]$. Plot precision and recall as a function of θ: where is the point that maximizes profit if an undetected defaulter costs 10 and a false alarm 1?

Exercise 15.5 — one-class

Implement the One-Class SVM on the ring-shaped dataset using only the normal points: $\min \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{1}{\nu n}\sum_{i=1}^{n} \xi_i - \rho$ subject to $\boldsymbol{w}'\phi(\boldsymbol{x}_i) \geq \rho - \xi_i$ (RBF kernel, $\gamma = 0.7$). Which fraction of anomalies does it detect for $\nu \in \{0.05; 0.1; 0.2\}$?

Chapter 16

Robust and quantile regression: estimating parameters with an LP

Class: LP (compared with a QP)

Script: `python/lab16_regression.py`
Online page

16.1 Managerial motivation

Every model seen so far starts from known parameters: the demand curve of chapter 7, the costs of chapter 4, the scenarios of chapter 12. Somebody, however, has to estimate those numbers — and the estimation is itself an optimization problem, solved with the same solver as the rest of the course.

Chapter 15 used optimization to *classify*. Here we use it to *predict a quantity*, and the choice of the objective function becomes a managerial decision: minimising the sum of the *absolute* deviations keeps the problem an LP, the estimate is not dragged around by anomalous days and — by weighting deviations above and below asymmetrically — one can aim at a *quantile* of demand instead of its centre. That is exactly the number the newsvendor of chapter 12 needs.

In this chapter we will learn to: (1) formulate regression as an LP; (2) read its dual, recognising the points that support the estimate and the quantile property; (3) estimate the quantity to produce conditional on the conditions of the day; (4) select the relevant features with a budget on the coefficients, reading its shadow price.

16.2 The model

The problem in words. *We decide* the coefficients of a linear function that, from the observed features of a day, predicts demand. *The objective* is to make the deviations between observed and predicted demand small, weighting those above and those below asymmetrically. *The constraints* define, observation by observation, the deviation of the prediction from the datum.

Data (model input)

Symbol	Type	Meaning
n	$\in \mathbb{Z}_{\geq 1}$	number of historical observations, indexed by $i \in \{1, 2, \dots, n\}$
p	$\in \mathbb{Z}_{\geq 0}$	number of observed features, indexed by $j \in \{1, 2, \dots, p\}$
x_{ij}	$\in \mathbb{Q}$	value of feature j in observation i
y_i	$\in \mathbb{Q}$	observed value to be explained in observation i (demand)
τ	$\in \mathbb{Q}$, $0 < \tau < 1$	quantile aimed at: $\tau = 1/2$ gives the median, a large τ a prudent prediction on the high side

Decision variables

We introduce $p + 1$ free variables for the function to be estimated,

$$\begin{cases} w_j = \text{coefficient of feature } j \\ b = \text{intercept of the function} \end{cases} \qquad \forall j \in \{1, 2, \dots, p\},$$

and, for each observation, two non-negative variables measuring the deviation of the prediction from the datum:

$$\begin{cases} u_i = \text{deviation above of observation } i \\ v_i = \text{deviation below of observation } i \end{cases} \qquad \forall i \in \{1, 2, \dots, n\}.$$

Using these variables, the LP model of quantile regression is the following:

LP model of quantile regression

$$\min \quad \sum_{i=1}^{n} \left[\tau \, u_i + (1 - \tau) \, v_i \right] \tag{16.1a}$$

$$\text{subject to} \quad \sum_{j=1}^{p} w_j \, x_{ij} + b + u_i - v_i = y_i, \qquad \forall i \in \{1, 2, \dots, n\}, \tag{16.1b}$$

$$w_j \gtreqless 0, \qquad \forall j \in \{1, 2, \dots, p\}, \tag{16.1c}$$

$$b \gtreqless 0, \tag{16.1d}$$

$$u_i, \; v_i \geq 0, \qquad \forall i \in \{1, 2, \dots, n\}. \tag{16.1e}$$

Description of the objective function and of the constraints of the LP model:

- the linear objective function (16.1a) pays τ for every unit of deviation above and $1 - \tau$ for every unit of deviation below: with $\tau = 1/2$ it is half the sum of the absolute deviations, with a large τ falling short of the datum costs more than overshooting it;
- the linear constraints (16.1b) define the deviation of every observation as the difference of its two parts, $u_i - v_i = y_i -$ prediction (n linear constraints);
- constraints (16.1c), (16.1d) and (16.1e) define the variables of the model: free coefficients and intercept, non-negative deviations.

Why two variables for a single deviation

The deviation $y_i - \left(\sum_j w_j x_{ij} + b\right)$ has an arbitrary sign, but the objective must pay its *absolute value*: splitting it into the positive part u_i and the negative part v_i, both non-negative, keeps the objective linear. At the optimum at least one of the two is zero: if both were positive, reducing them by the same amount would leave constraint (16.1b) satisfied and would decrease the objective, because $\tau > 0$ and $1 - \tau > 0$. It is the same positive-part trick already used for the scenario newsvendor in chapter 12.

With no features: the empirical quantile comes back

With $p = 0$ only the intercept is left and the model chooses a single number b. Seven days of observed sales: 88, 96, 104, 112, 120, 132, 148 pieces. The bakery of chapter 12 has $c_u = 9$ and $c_o = 4$, hence $\tau = \alpha^* = 9/13 = 0.6923$.

- **Step 1 — what it costs to move b.** Moving b up by one unit costs τ more on every observation lying *above* b and saves $1 - \tau$ on every observation lying *below*. It pays to go up as long as the observations above are numerous enough to compensate.
- **Step 2 — the optimum.** The optimal value is $\tilde{b} = 120$, the fifth observation in increasing order: 4 lie below it and 2 above, and $4/7 = 0.5714 \le \tau \le 5/7 = 0.7143$. The optimal cost is

$$\tilde{z} = \tfrac{4}{13}(32 + 24 + 16 + 8) + \tfrac{9}{13}(12 + 28) = \tfrac{320+360}{13} = \tfrac{680}{13} = 52.3077.$$

- **Step 3 — the duals.** The duals of the seven constraints are $\tilde{\boldsymbol{\pi}} = \left(-\frac{4}{13}, -\frac{4}{13}, -\frac{4}{13}, -\frac{4}{13}, -\frac{2}{13}, \frac{9}{13}, \frac{9}{13}\right)$: $\tau = 9/13$ on the observations above the estimate, $-(1-\tau) = -4/13$ on those below, an interior value on the interpolated one. Their sum is zero and the dual value $\sum_i \tilde{\pi}_i\, y_i = 680/13$ matches the primal one.

With no features the model returns the **empirical quantile**: it is the quantile rule of chapter 12, written as an LP. With features it becomes a quantile *conditional* on the conditions of the day.

16.3 Case study: the bakery history

Sixty days of sales of the bakery of chapter 12: for each day the demand in pieces, the maximum temperature, the price charged and a weekend indicator (`data/panetteria_storico.csv`). Five of those days fall during two transport strikes: the town centre is deserted and sales collapse to about twenty pieces.

We first estimate the relation with temperature alone ($p = 1$), which can be drawn in a chart, comparing two different objectives on the same data: the sum of the absolute deviations (LP, $\tau = 1/2$) and the sum of the squared deviations (the least-squares QP, $\min \sum_i r_i^2$ with r_i free and $\sum_j w_j x_{ij} + b + r_i = y_i$).

```
LP  (median)        : demand = 209.77 - 3.63 * temperature
QP  (least squares): demand = 202.34 - 3.61 * temperature
Mean absolute deviation over the 55 normal days: LP 13.08 pieces, QP 14.81
    pieces
QP without the 5 strike days: demand = 213.58 - 3.72 * temperature
Distance from the "clean" intercept 213.58: LP 3.81 pieces, QP 11.24 pieces
```

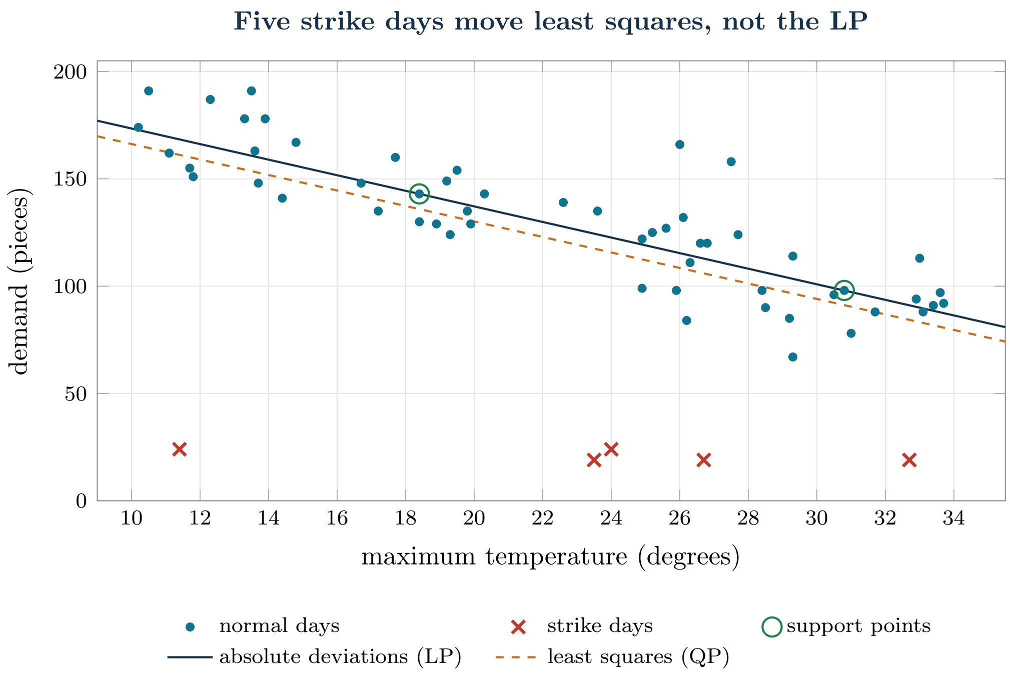


Figure 16.1: The 60 days of the bakery: the LP line (blue) ignores the five strike days, while the least-squares line (dashed orange) is pulled down by more than 7 pieces on the intercept. The two points circled in green are the support points, the only ones the LP line interpolates.

Managerial insight

A single anomalous datum out of ten moves least squares because their objective pays the deviation *squared*: one deviation of 100 weighs as much as ten thousand deviations of one. The LP pays the deviation *once*, and the line stays where the normal data are: it reaches on its own the same answer the QP gives only after somebody has cleaned the history by hand. Changing the objective, here, is not a technical refinement: it is the difference between a model that needs supervision and one that can be left running every week.

16.4 The dual reading: the support points

Associating a dual π_i with each deviation constraint (16.1b), the optimality condition on the (non-negative) variables u_i and v_i confines every dual to a window that depends on τ alone:

$$-(1-\tau) \;\le\; \tilde{\pi}_i \;\le\; \tau, \qquad \forall i \in \{1,2,\dots,n\}, \tag{16.2}$$

while optimality on the free variables w_j and b gives the two conditions

$$\sum_{i=1}^{n} \tilde{\pi}_i\, x_{ij} = 0 \quad \forall j \in \{1,2,\dots,p\}, \qquad\qquad \sum_{i=1}^{n} \tilde{\pi}_i = 0. \tag{16.3}$$

Complementary slackness says where each dual sits:

Position of the observation	Positive variable	Dual
above the estimate (y_i larger)	$\tilde{u}_i > 0$	$\tilde{\pi}_i = \tau$
below the estimate (y_i smaller)	$\tilde{v}_i > 0$	$\tilde{\pi}_i = -(1-\tau)$
on the estimate (interpolated)	$\tilde{u}_i = \tilde{v}_i = 0$	$-(1-\tau) < \tilde{\pi}_i < \tau$

The interpolated observations are the **support points**: they are the only ones whose dual is not stuck at an endpoint, and they are what determines the estimate. In general there are $p+1$ of them, as many as the free variables of the model: moving any of the other observations, as long as it stays on its side of the estimate, changes nothing.

```
Observations above the line: 29, below: 29, interpolated: 2  <- p + 1 = 2
pi = +0.5 (point above the line): 29 points
pi = -0.5 (point below the line): 29 points
sum of the duals                     = 0.000000
sum of the duals times temperature   = 0.000000
dual value 616.9960 = primal value 616.9960   (strong duality)
```

The dual of the intercept is the quantile property

The condition $\sum_i \tilde{\pi}_i = 0$ is the dual of the intercept, and it is the quantile property written algebraically: substituting the values of the table, $\tau\, n_+ - (1-\tau)\, n_- + \sum_{i \text{ interpolated}} \tilde{\pi}_i = 0$, where n_+ and n_- count the observations above and below the estimate. Up to the $p+1$ support points, the estimate therefore leaves a fraction τ of the observations below it: with $\tau = 1/2$, half above and half below (29 and 29 in the example). The condition $\sum_i \tilde{\pi}_i\, x_{ij} = 0$ says the same thing feature by feature: the sign of the deviations must retain no link with the data.

Managerial insight

The support points are, for regression, what the support vectors of chapter 15 are for classification: very few observations sustain the solution and the others could disappear without changing it. The difference is that here the problem is an LP and not a QP, but the reading is identical — and it is the same complementarity of chapter 2.

The deviation itself is a diagnostic tool: estimating the model with all three features on the 60 days, the normal days have a deviation of at most 23.2 pieces and the five strike days of at least 67.5. A threshold at 40 pieces isolates them exactly, knowing nothing about the strikes.

```
Three-feature model over the 60 days:
  demand = 276.46 - 3.97*temperature - 23.12*price + 28.51*weekend
Absolute deviation: 23.2 pieces at most on normal days, 67.5 at least on
    anomalous ones
Days with a deviation above 40 pieces: [12, 23, 30, 42, 47] -> they are the
    strike days
```

16.5 Quantile regression and safety stock

On the 55 normal days we estimate the three-feature model for several values of τ. The last column checks the quantile property: the fraction of observations falling below the estimate.

τ	intercept	temperature	price	weekend	share below the estimate
0.10	269.29	−4.04	−24.71	+37.05	0.073
0.50	272.01	−3.85	−22.24	+28.74	0.455
$9/13 = 0.6923$	281.19	−3.96	−23.21	+31.47	0.673
0.90	287.49	−3.57	−25.69	+27.98	0.873

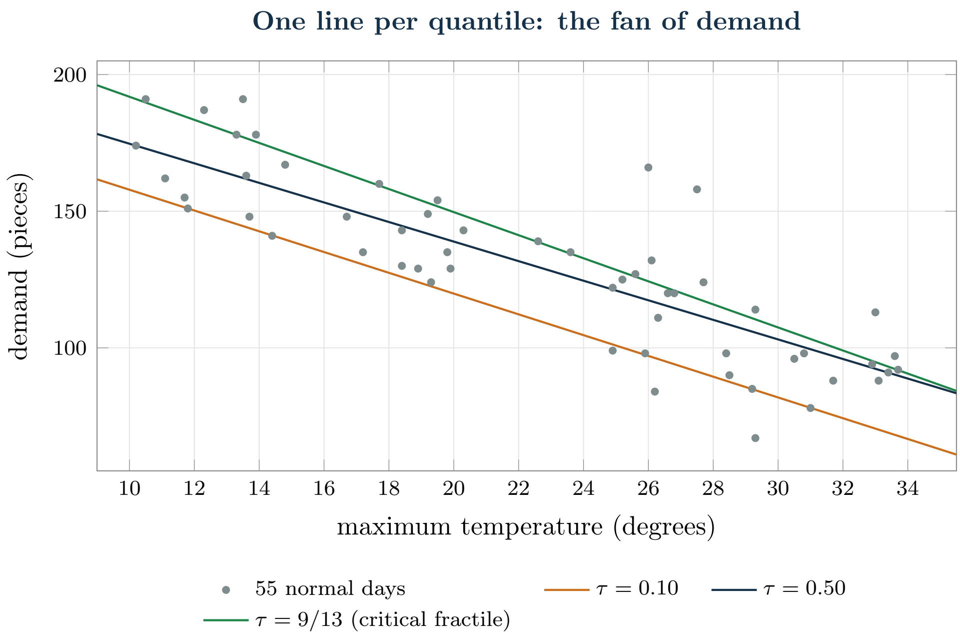


Figure 16.2: The lines estimated with temperature alone for three values of τ: each leaves the corresponding fraction of observations below it. The green line, with τ equal to the critical fractile of the bakery, is the quantity to bake day by day.

For a hot weekend day at full price (28 degrees, price 3.00) the model gives two different numbers depending on τ:

```
median predicted demand          126.3 pieces
predicted 0.6923 quantile        132.1 pieces    <- quantity to bake
safety stock                       5.8 pieces
0.6923 quantile with no features 148.0 pieces
```

Managerial insight

The quantile rule of chapter 12 still holds, but the number it applies to changes: the quantile computed over the whole history says to bake 148 pieces, the one *conditional* on the conditions of the day asks for 132. Sixteen unsold pieces a day fewer, without touching the costs or the decision rule: just by using information already in hand. The gap between the median estimate and the one at the critical fractile, 5.8 pieces, is the safety stock — and the model computes it directly, with no assumption about the distribution.

16.6 Feature selection with a budget

Adding features always improves the fit on the historical data, even when they are irrelevant: given enough freedom the model ends up chasing noise. One way to govern this is to put a *budget* on the coefficients. We need p new non-negative variables,

$$z_j = \text{coefficient budget assigned to feature } j, \qquad \forall j \in \{1, 2, \ldots, p\},$$

and the datum $t \in \mathbb{Q}_{\geq 0}$, the total budget. Model (16.1) gains the constraints

$$-z_j \leq w_j \leq z_j, \qquad \forall j \in \{1, 2, \ldots, p\}, \tag{16.4a}$$

$$\sum_{j=1}^{p} z_j \leq t, \tag{16.4b}$$

$$z_j \geq 0, \qquad \forall j \in \{1, 2, \ldots, p\}. \tag{16.4c}$$

Constraints (16.4a) force z_j to be at least the absolute value of w_j (at the optimum it equals it exactly, because z_j consumes budget); constraint (16.4b) shares the budget out among the features. The model remains an LP.

The budget adds up coefficients of features measured in different units: degrees, euros, 0/1 indicators. Without standardising, constraint (16.4b) would compare incommensurable quantities and the selection would depend on the units of measurement. Here every feature is brought to zero mean and unit standard deviation, so each coefficient reads as *pieces per standard deviation* of the feature.

To the three real features we add four with no link whatsoever to demand (social followers, rainfall in a neighbouring region, a stock index, the day of the month). As the budget grows we observe which ones enter.

```
budget  temperature   price  weekend  followers    rain  index  day_of_month
    10       -10.00    0.00     0.00       0.00    0.00   0.00          0.00
    20       -18.16   -1.51     0.00       0.00   -0.33   0.00          0.00
    30       -22.43   -4.45     2.35       0.00   -0.77   0.00          0.00
    45       -26.98   -8.22     9.09       0.00   -0.63   0.00         -0.09

budget  5: mean deviation 22.37 pieces, shadow price -19.952 pieces per unit
budget 10: mean deviation 18.83 pieces, shadow price -18.665 pieces per unit
budget 20: mean deviation 13.84 pieces, shadow price -10.658 pieces per unit
budget 30: mean deviation 10.63 pieces, shadow price  -7.271 pieces per unit
budget 45: mean deviation  7.19 pieces, shadow price  -4.197 pieces per unit
```

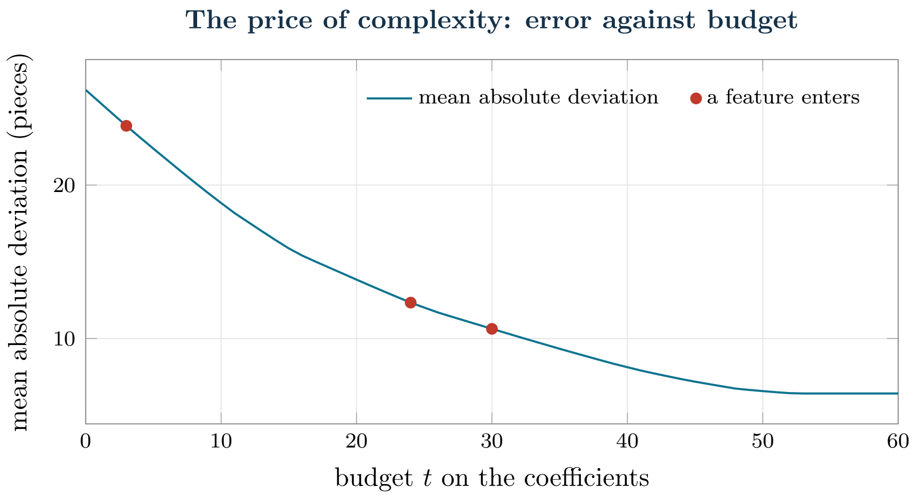


Figure 16.3: The mean absolute deviation as a function of the budget on the coefficients: a convex decreasing curve, like the optimal value of an LP as the right-hand side varies. The red dots mark the budget at which temperature ($t = 3$), price ($t = 24$) and weekend ($t = 30$) enter, in this order.

Managerial insight

The shadow price of constraint (16.4b) is the **price of complexity**: how many pieces of error are saved for every extra unit of budget. It is -19.95 pieces per unit when the budget is scarce and -4.20 when it is plentiful: the first units buy temperature, which explains almost everything; the last ones buy noise. The three real features enter in the order of their importance — temperature, price, weekend — while none of the four irrelevant ones exceeds 2 pieces per standard deviation until the budget becomes plainly excessive. The budget to choose is not the one minimising the historical error, but the one beyond which the shadow price becomes negligible.

16.7 Exercises

Exercise 16.1 — quantile by hand

With the seven days of the example (88, 96, 104, 112, 120, 132, 148) and $\tau = 1/2$, compute by hand the median, the optimal value of the objective and the seven duals; check with the script that the duals sum to zero.

Exercise 16.2 — support points

Estimate the model with temperature alone and $\tau = 1/2$, then move an observation lying above the line up by $+30$ pieces and re-solve: does the estimate change? And if one of the two support points is moved down by -30 pieces? Explain through complementary slackness.

Exercise 16.3 — asymmetric cost of error

The bakery considers selling the leftovers off at the end of the day, taking c_o from 4 to 2 €. Recompute α^*, re-run the quantile regression and say by how many pieces the production of the typical day changes.

Exercise 16.4 — budget and convexity

Plot the optimal value of the LP with a budget for $t \in \{0, 1, \ldots, 60\}$ and check that it is convex and piecewise linear; locate the breakpoints and show that they coincide with the budgets at which the shadow price changes value.

Exercise 16.5 — validation

Split the 55 normal days into estimation (40) and validation (15). For $t \in \{5, 10, \ldots, 55\}$ plot the mean absolute deviation on the two sets: is the estimation one monotone in t? And the validation one? Which budget do you choose?

Part V

Lab organization

Chapter 17

Organization of the lab

17.1 Essential path in four lab sessions

	Content	Learning objectives
Lab 1	Production and inventory (Ch. 4)	formulate a multi-period LP; read duals, slacks and ranges; verify a shadow price by perturbation
Lab 2	Markowitz (Ch. 6)	build a convex QP; plot a frontier; discuss the fragility of the estimates
Lab 3	Pricing *or* budget (Chs. 7–8)	model non-linear functions; study concavity; check the KKT conditions numerically
Lab 4	Project of your choice	supply chain, EV charging, facility location, queues, Newsvendor, CVaR or SVM; managerial presentation of the results

Chapters 12–15 can replace lab sessions 3–4 in an edition oriented towards uncertainty and machine learning, or become in-depth material for individual projects.

17.2 Structure of the deliverable

Each group hands in a notebook or a short report (max 8 pages) with this mandatory structure, which follows the outline of the chapters:

1. **Problem and assumptions** — the context in one page, the simplifications stated explicitly;
2. **Model** — sets, parameters, variables, constraints, objective, with the explanation of every constraint (not just the formula);
3. **Data** — origin, units of measurement, generation if any;
4. **Results** — optimal value, main decisions, active constraints;
5. **Sensitivity** — the protocol of Chapter 2, Section 2.3: at least verified shadow prices, a one-at-a-time analysis and a trade-off frontier;
6. **Managerial recommendation** — at most ten lines, without formulas: what to do, how much it is worth, how robust it is.

17.3 Assessment criteria

Dimension	Weight	What is assessed
Correctness of the formulation	30%	correct and justified constraints, consistent units, convexity discussed
Implementation and verification	25%	reproducible code; solver status checked; at least one independent check (example by hand, perturbation, limiting case)
Sensitivity analysis	25%	complete protocol; duals interpreted with their validity ranges; trade-offs plotted
Interpretation and communication	20%	clear recommendation; readable figures; honesty about the limits of the model

17.4 Typical discussion questions

The questions with which to defend the deliverable — and which it is worth asking oneself *beforehand*:

- Which resource is it best to increase first, and how much can be paid for it?
- What is the cost of a more ambitious service promise?
- Does the solution remain credible if the data change by 5%? Which parameter is the most fragile?
- Which trade-off would you recommend to a decision maker, and why exactly that point of the frontier?
- What does the model NOT say? (assumptions, horizon, excluded variables)

17.5 Recurring mistakes to avoid

1. Reading `.X` or `.Pi` without checking `m.Status`.
2. Forgetting `lb=-GRB.INFINITY` on the free variables (b of the SVM, η of the CVaR).
3. Interpreting a shadow price outside its validity range (`SARHSLow/Up`).
4. Confusing the sign of the duals in minimization problems (constraint $\leq$: `Pi` ≤ 0).
5. Updating the RHS of a constraint that contains constants on the left-hand side (Chapter 10).
6. Optimizing a single objective when the problem has two (pure minimax with no cost).
7. Choosing the hyperparameters (C, γ, λ) by looking at the test set or at the scenarios used to decide.
8. Reporting six decimal digits from estimates that wobble at the second.

17.6 Course materials

`notes/`	these lecture notes (`main.tex`; the figures read the CSV files in `figure/dat/`)
`python/`	one script per chapter; `run_all.py` regenerates data and figures
`data/`	all the case study data in CSV format
`GUIDA_GUROBI.md`	standalone guide to the solver
`solutions/`	solutions to the exercises (for instructors)
`slides/`	slides of the four lab sessions (Italian and English)

Reproducibility: `python3 -m pip install gurobipy matplotlib pandas scipy` (scipy is needed only for the statistical functions, not for the optimization), then `python3 python/run_all.py` and `latexmk -pdf notes/main.tex`.