

SAPIENZA
UNIVERSITÀ DI ROMA

LP Modelling

Linear programming models

Lecture notes with models, fully worked numerical examples, case studies, complete Python/Gurobi implementations and sensitivity analysis. Linear programming models: production, electric vehicle charging, DEA; decisions under uncertainty (newsvendor, VaR and CVaR, arbitrage); robust and quantile regression.

Fabio Furini
fabiofurini.github.io
Department of Computer, Control and Management Engineering — Sapienza University of Rome

Online version and downloadable material:
fabiofurini.github.io/lp-modelling/
github.com/fabiofurini/lp-modelling

Academic year 2026–2027

Contents

I Tools **4**

1 Introduction to LP modelling **5**
1.1 What we shall do . . . 5
1.2 Linear models and acronyms used . . . 6
1.3 Notation . . . 6
1.4 A deliberate choice: continuous variables only . . . 6
1.5 Map of the models . . . 7
1.6 Structure of each chapter . . . 7
1.7 Course organization . . . 7
1.8 In summary . . . 8

2 Background: LP and sensitivity analysis **9**
2.1 Linear programming theory . . . 9
2.1.1 Duality . . . 9
2.1.2 The optimality conditions: complementary slackness . . . 13
2.1.3 Sensitivity analysis and shadow prices . . . 16
2.2 The sensitivity analysis protocol . . . 17
2.3 In summary . . . 17

3 The solver: building, solving, interpreting **19**
3.1 The general commands . . . 19
3.1.1 Installation and licence . . . 19
3.1.2 The five steps to build a model . . . 20
3.1.3 Running the model and reading the log . . . 20
3.1.4 Retrieving the solution . . . 21
3.2 The commands for linear models . . . 21
3.2.1 Shadow prices, reduced costs and ranges: the attributes . . . 21
3.2.2 Interpreting the output: the checklist . . . 21
3.2.3 Two complete readings . . . 22
3.3 In summary . . . 24

II Deterministic models **25**

4 Multi-period production and inventory **26**
4.1 Managerial motivation . . . 26
4.2 Guided construction of the model . . . 27

4.3 Worked example by hand . . . 28
4.4 Case study . . . 28
4.5 Implementation . . . 29
4.6 Results . . . 29
4.7 Sensitivity analysis . . . 30
4.8 In summary . . . 32
4.9 Exercises . . . 32

5 Smart charging of electric vehicles 33
5.1 Managerial motivation . . . 33
5.2 Guided construction of the model . . . 34
5.3 Worked example by hand . . . 35
5.4 Case study . . . 35
5.5 Results . . . 36
5.6 Sensitivity analysis . . . 37
5.7 In summary . . . 38
5.8 Exercises . . . 38

6 Data Envelopment Analysis 39
6.1 Managerial motivation . . . 39
6.2 Units, inputs, outputs and efficiency . . . 40
6.3 The input-oriented CCR model . . . 40
6.4 The envelopment form: the dual . . . 42
6.5 The BCC model and scale efficiency . . . 43
6.6 Slacks and weak and strong efficiency . . . 44
6.7 Worked example by hand . . . 45
6.8 Case study . . . 46
6.9 Implementation . . . 46
6.10 Results . . . 47
6.11 Sensitivity analysis . . . 49
6.12 In summary . . . 50
6.13 Exercises . . . 50

III Decisions under uncertainty 52

7 The Newsvendor and its variants 53
7.1 Managerial motivation . . . 53
7.2 The basic model . . . 54
7.3 Worked example by hand (discrete demand) . . . 55
7.4 Case study: the quantile rule in the continuous case . . . 55
7.5 The scenario-based linear formulation . . . 56
7.6 Service levels . . . 58
7.7 Risk aversion: mean-CVaR . . . 58
7.8 Multi-product with a shared budget . . . 59
7.9 In summary . . . 59
7.10 Exercises . . . 59

8 VaR and CVaR **61**
8.1 Managerial motivation . . . 61
8.2 Definitions and properties . . . 62
8.3 Worked example by hand . . . 63
8.4 Case study 1: mean-CVaR portfolio . . . 63
8.5 Case study 2: two-stage supply chain . . . 64
8.6 Sensitivity analysis . . . 65
8.7 In summary . . . 66
8.8 Exercises . . . 66

9 Arbitrage and pricing **68**
9.1 Managerial motivation . . . 68
9.2 Guided construction of the model . . . 69
9.3 Duality is pricing: the risk-neutral probabilities . . . 70
9.4 Worked example by hand . . . 71
9.5 Case study . . . 71
9.6 Results . . . 72
9.7 Sensitivity analysis: the interval of consistent prices . . . 72
9.8 In summary . . . 73
9.9 Exercises . . . 73

IV Optimization and machine learning **74**

10 Robust and quantile regression **75**
10.1 Managerial motivation . . . 75
10.2 The model . . . 76
10.3 Case study: the bakery history . . . 77
10.4 The dual reading: the support points . . . 78
10.5 Quantile regression and safety stock . . . 80
10.6 Feature selection with a budget . . . 81
10.7 In summary . . . 82
10.8 Exercises . . . 83

V Lab organization **84**

11 Organization of the lab **85**
11.1 Essential path in four lab sessions . . . 85
11.2 Recurring mistakes to avoid . . . 85
11.3 Course materials . . . 86

Part I

Tools

Chapter 1

Introduction to LP modelling

Prerequisites

- From the module *Linear Programming*, Chapter 4 «Linear Programming problems»: the form of a Linear Programming problem: objective function, linear constraints, domains of the variables.
- From the module *Linear Programming*, Chapter 16 «Linear Programming modeling techniques»: the basic techniques to turn a statement into a linear programming model.

Objectives

At the end of this chapter you will be able to:

- describe the path of every chapter: managerial problem, model, Python and solver, interpretation;
- use the notation of the course for data, variables, duals, reduced costs and slacks;
- explain why the course uses continuous variables only and what this buys;
- find your way in the map of the models and in the structure of each chapter.

1.1 What we shall do

This course teaches how to use linear programming as a tool for *decision making*, not merely for computation. Every chapter starts from a concrete managerial problem — how much to produce, how to charge a fleet, how efficient a branch is, how much risk to accept — and turns it into a linear programming 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 Linear models and acronyms used

All the models in these lecture notes are **LP** (*Linear Programming*): the objective and the constraints are linear functions of the variables. An LP is always a convex problem: every local optimum is also global, and the duality of chapter 2 holds without exceptions. The other recurring acronyms are defined at first use in the respective chapter.

The prerequisite of the course is the module *Linear Programming*, in particular duality and sensitivity analysis. Models with a quadratic or nonlinear objective are in the module *NLP Modelling*.

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, $i \in \{1, 2, \ldots, n\}$ — 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 ($\boldsymbol{x}$, $\boldsymbol{w}$); all vectors are column vectors and $\boldsymbol{a}'\boldsymbol{b}$ denotes the scalar product;
- **matrices**: uppercase **boldface** letters ($\boldsymbol{A}$);
- **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 reduced costs — 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, power levels, weights.
3. **Progression**: the same tools — an LP, its dual, the shadow prices — serve production, finance and statistics, without changing solver.

1.5 Map of the models

Ch.	Application	Class	Central concept
4	Multi-period production and inventory	LP	duality, capacity
5	Electric vehicle charging	LP	energy, peaks
6	Data Envelopment Analysis	LP	relative efficiency
7	Newsvendor and variants	stoch. LP	deciding before knowing
8	VaR and CVaR	LP	tail risk
9	Arbitrage and pricing	LP	the duality that prices
10	Robust and quantile regression	LP	estimating with an LP

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

The essential path comprises four laboratory sessions (chapter 11 for the details): (1) production and inventory with duality and shadow prices; (2) Data Envelopment Analysis and the envelopment form; (3) scenario-based decisions (newsvendor or CVaR); (4) a project of your own choice with a managerial presentation.

1.8 In summary

In summary

- Every chapter goes from a managerial problem to a model, to Python and the solver; the model does not just return a number, it is interrogated.
- The bar marks the values of a feasible solution, the tilde those of an optimal one; π_i are duals, $\bar{c}_j$ reduced costs, $\bar{s}_i$ slacks.
- Continuous variables only, by choice: duality, shadow prices and reduced costs then apply to every model.
- Each chapter follows the same outline: motivation, model, example by hand, case study, code, results, sensitivity and exercises.

Chapter 2

Background: linear programming and sensitivity analysis

Prerequisites

- From the module *Linear Programming*, Chapter 9 «Duality theory»: the dual of a Linear Programming problem, weak and strong duality, complementary slackness.
- From the module *Linear Programming*, Chapter 14 «Sensitivity analysis»: shadow prices, reduced costs and their validity ranges.

Objectives

At the end of this chapter you will be able to:

- write the dual of an LP in general form with the conversion rules;
- check optimality with complementary slackness;
- deduce the sign of a shadow price from the direction of the constraint and the sense of the objective;
- read a shadow price and a reduced cost together with their validity ranges;
- apply the six steps of the sensitivity protocol.

This chapter collects the theoretical tools of linear programming used throughout these lecture notes: duality, optimality conditions and sensitivity analysis. It closes with the sensitivity protocol used in every laboratory session. It is a quick reference: the complete theory, with proofs and algorithms, is in the module *Linear Programming*, in the chapters "Duality theory" and "Sensitivity analysis".

2.1 Linear programming theory

2.1.1 Duality

We call $M = \{1, 2, \dots, m\}$ the set of constraint indices and $N = \{1, 2, \dots, n\}$ 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 $N_{\geq 0}$, $N_{\gtrless 0}$ 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

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

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.

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

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:

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

- **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, $\max\ 30\pi_1 + 100\pi_2 - 20\pi_3$.

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

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*

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

Line-by-line explanation

In words: **slack constraint ⇒ dual at zero** ($\bar{s}_i \neq 0 \Rightarrow \bar{\pi}_i = 0$) and **variable different from zero ⇒ 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 ⇒ $\tilde{\pi}_i = 0$.

- **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 ⇒ $\bar{c}_j = 0$.

- **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 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.

2.3 In summary

In summary

- Every constraint generates a dual variable and every variable a dual constraint; weak duality gives a bound, strong duality equal values at the optimum.
- Complementary slackness: a slack constraint has a null dual, a non-null variable has a null reduced cost.
- The shadow price $\tilde{\pi}_i = \partial \tilde{z} / \partial b_i$ holds only inside its range; a number is always quoted with its range.
- The reduced cost says by how much c_j must improve for a variable at zero to be worth using.
- The protocol: base case, one-at-a-time, shadow prices checked by perturbation, scenarios, trade-off, stability.

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.)

Chapter 3

The solver: building, solving, interpreting

Prerequisites

- From this module, Chapter 2 «Background: linear programming and sensitivity analysis»: duals, reduced costs, slacks and validity ranges, which the solver returns as attributes.

Objectives

At the end of this chapter you will be able to:

- build an LP in `gurobipy` in five steps;
- declare free variables, remembering the default lower bound 0;
- check `Status` before reading `ObjVal`, `X` and `Slack`;
- read `Pi`, `RC`, `SARHS` and `SAObj` as the quantities of the background chapter;
- recognise the typical errors, from an infeasible model to a right-hand side that contains constants.

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 `GUROBI_GUIDE.md` in the course folder. Two subchapters: the *general commands* (building, solving, reading the solution), then the commands specific to *linear models* (duals, reduced costs, validity ranges).

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 $x_1, x_2 \geq 0$.

```
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")

# 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 regression 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

```
m.Params.OutputFlag = 0     # 0 = silent, 1 = log on screen
m.Params.TimeLimit  = 60    # seconds
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:

```
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 ⇒ 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 `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 `RC` $= -2$: its coefficient must rise by at least 2 (that is, to 22) for it to enter the solution, with validity range `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 $\leq \Rightarrow$ `Pi` ≥ 0 (more resource, more profit); constraint $\geq \Rightarrow$ `Pi` ≤ 0;
- in a *minimization* problem: demand constraint $\geq \Rightarrow$ `Pi` ≥ 0 (more demand, more cost); capacity constraint $\leq \Rightarrow$ `Pi` ≤ 0 (more capacity, less cost);
- *equality* constraint $\Rightarrow$ any sign, as prescribed by the rules of duality. Inactive constraint $\Rightarrow$ `Pi` $= 0$; validity within the range `SARHSLow`–`SARHSUp`.

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

- *basic* variable: `RC` $= 0$ — as a rule it lies strictly between its bounds, but it may also sit on a bound with `RC` $= 0$ (*degenerate* basis);
- variable at its *lower bound* (typically at zero): `RC` ≥ 0 in a minimization (by how

much its cost must fall for it to be worth using), `RC` ≤ 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 `RC` < 0: one extra unit of capacity on that arc saves |`RC`|.

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

```
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 In summary

In summary

- Five steps: `Model`, `addVar(s)`, `addConstr(s)`, `setObjective`, `write`.
- Variables are ≥ 0 by default: a free variable needs `lb=-GRB.INFINITY`.
- `Status` always comes first; only then `ObjVal`, `X` and `Slack`.
- `Pi`, `RC`, `Slack`, `SARHS` and `SAObj` are the theoretical quantities of the background chapter; the stored RHS includes the constants moved from the left-hand side.
- `RC` $= 0$ is not enough to say that a variable is used: degeneracy is always possible.

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.

Part II

Deterministic models

Chapter 4

Multi-period production and inventory

Class: LP **Script:** `python/lab04_production.py`
Online page

Prerequisites

- From this module, Chapter 2 «Background: linear programming and sensitivity analysis»: shadow prices, their validity ranges and the sensitivity protocol.
- From this module, Chapter 3 «The solver: building, solving, interpreting»: building an LP in `gurobipy` and reading `Pi` and `SARHS`.

Objectives

At the end of this chapter you will be able to:

- formulate a multi-period LP with inventory balance constraints;
- read and check the shadow prices of capacity and their validity ranges;
- build the capacity value curve;
- explain how inventory moves capacity from the light months to the peak months.

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.

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 $i \in \{1, 2, \dots, n\}$
m	$\in \mathbb{Z}_{\geq 1}$	number of months of the horizon; the months are indexed by $t \in \{1, 2, \dots, m\}$
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:

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

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

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

Minimum-cost production and inventory LP model

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

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

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

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

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

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`):

```
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
```

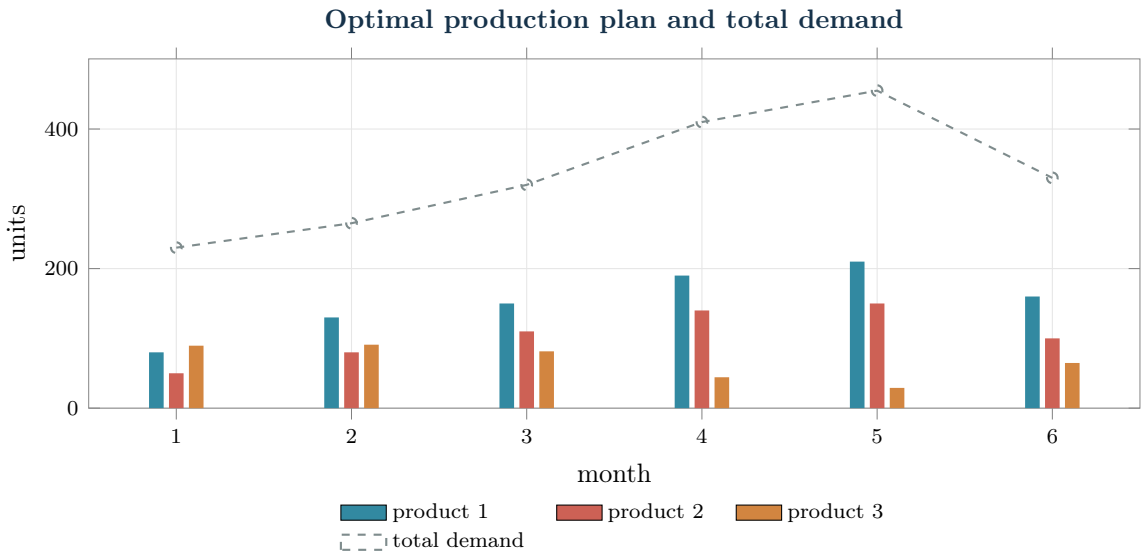


Figure 4.1: Optimal production plan of the LP: bars = units produced of the three products in each month; dashed line = total demand of the month. In the peak months (4–5) total production stays below demand: the difference is covered by the inventory of product 3 accumulated in the first months.

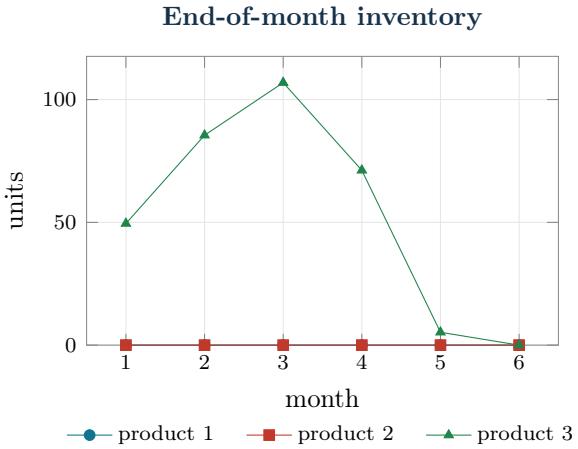


Figure 4.2: End-of-month inventory by product — only product 3 uses the warehouse (pre-build strategy), and the inventory goes to zero in the final month.

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

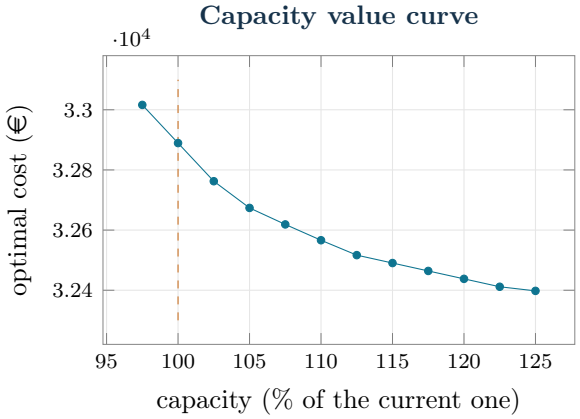
```
month 1: use 330.0/420   | Pi =   0.000 EUR/hour | valid for b_t in [330.0, +inf]
month 2: use 420.0/420   | Pi = -0.762 EUR/hour | valid for b_t in [330.0,
    524.0]
month 3: use 460.0/460   | Pi = -1.524 EUR/hour | valid for b_t in [370.0,
    564.0]
month 4: use 460.0/460   | Pi = -2.286 EUR/hour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



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.

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; below 97% of the current capacity full service cannot be guaranteed."

A *smooth* plan, which penalizes the changes of production from one month to the next, requires a quadratic term in the objective and is no longer an LP: it is the quadratic variant of this model, in the module *NLP Modelling*, page "Production with quadratic smoothing".

4.8 In summary

In summary

- The inventory balance links the months: the problem cannot be solved one month at a time.
- Inventory moves capacity from the light months to the peak months; emptying the warehouse at the end is always optimal.
- The shadow price of a capacity is checked by adding one hour and re-optimizing; a non-binding capacity has a null shadow price.
- One more hour of capacity is worth the holding cost it saves; the capacity value curve is convex and decreasing.

4.9 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

Smart charging of electric vehicles

Class: LP **Script:** `python/lab05_ev_charging.py`
Online page

Prerequisites

- From this module, Chapter 2 «Background: linear programming and sensitivity analysis»: shadow prices and the sensitivity protocol.
- From this module, Chapter 3 «The solver: building, solving, interpreting»: variable bounds in `gurobipy` and the right-hand side stored by the solver.

Objectives

At the end of this chapter you will be able to:

- model decisions over an hourly horizon with availability windows;
- read the duals as prices of the marginal hours;
- handle the cost-peak trade-off with a weighted objective;
- diagnose a real pitfall of sensitivity analysis: the right-hand side with constants.

5.1 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.

5.2 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 $v \in \{1, 2, \dots, n\}$
m	$\in \mathbb{Z}_{\geq 1}$	number of hourly intervals, indexed by $t \in \{1, 2, \dots, m\}$, 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:

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

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

Minimum-cost charging LP model

$$\min \quad \sum_{v=1}^{n} \sum_{t=1}^{m} \pi_t \, \Delta t \, x_{vt} \tag{5.1a}$$

$$\text{subject to} \quad \eta \sum_{t=1}^{m} \Delta t \, x_{vt} \geq e_v, \qquad \forall v \in \{1, 2, \dots, n\}, \tag{5.1b}$$

$$\sum_{v=1}^{n} x_{vt} + b_t \leq k, \qquad \forall t \in \{1, 2, \dots, m\}, \tag{5.1c}$$

$$x_{vt} \leq a_{vt} \, \bar{p}_v, \qquad \forall v \in \{1, 2, \dots, n\},\ \forall t \in \{1, 2, \dots, m\}, \tag{5.1d}$$

$$x_{vt} \geq 0, \qquad \forall v \in \{1, 2, \dots, n\},\ \forall t \in \{1, 2, \dots, m\}. \tag{5.1e}$$

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

- the linear objective function (5.1a) 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 (5.1b) 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 (5.1c) 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 (5.1d) 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 (5.1e) define the variables of the model.

Variants: peak shaving ($\min z$ with $\sum_{v=1}^{n} x_{vt} + b_t \le z$ for every $t \in \{1, 2, \dots, m\}$); 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.

5.3 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.

5.4 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
                                    )
```

5.5 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
```

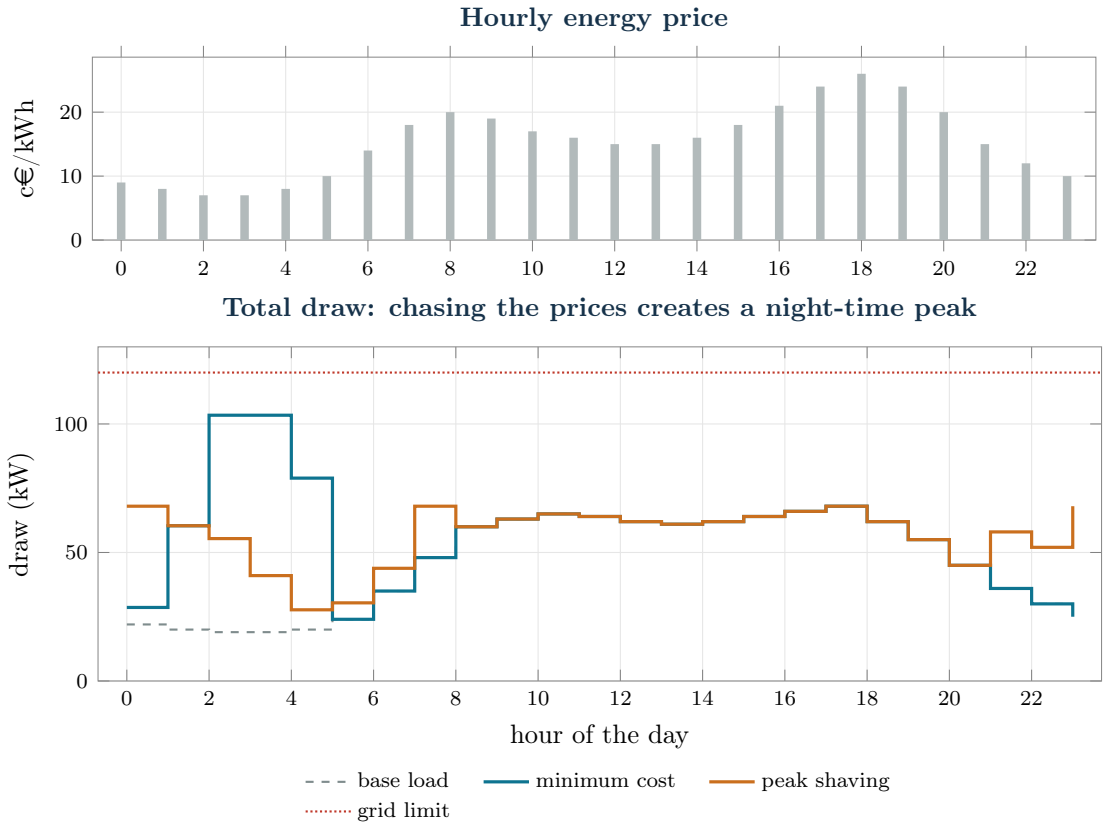


Figure 5.1: Above: hourly energy price (grey bars), low at night and with an evening peak. Below: total draw (base + charging) hour by hour under the two strategies: the minimum cost one (teal) concentrates the charging in the cheap hours and creates a night-time peak of 103 kW; peak shaving (orange) 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.

5.6 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
```

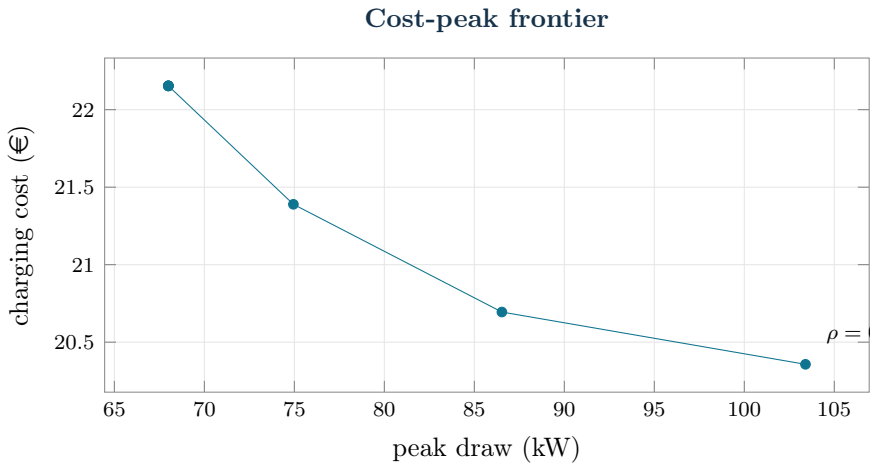


Figure 5.2: Cost-peak frontier obtained by varying the weight ρ of the peak in the objective: every point is an optimal charging strategy. Reducing the peak from 103 to 75 kW costs about one euro per night; the point $\rho \geq 0.2$ reaches the minimum possible peak (68 kW) at the lowest cost that allows 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.

A *smooth* load profile, which penalizes the changes of the load from one hour to the next, requires a quadratic term in the objective and is no longer an LP: it is the quadratic variant of this model, in the module *NLP Modelling*, page "Electric vehicle charging with a smooth profile".

5.7 In summary

In summary

- Smart charging is a pure LP: the decision is how much power, and availability is a datum, not a binary variable.
- The cheap hours are filled first; the shadow price of the energy requirement is the price of the marginal hour, divided by η.
- Cost $+\,\rho$ peak traces the cost-peak frontier: the peak drops a lot for a small increase in cost.
- The constants of the constraints end up in the right-hand side: if a sensitivity analysis changes nothing, check the code first.

5.8 Exercises

Exercise 5.1 — 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 5.2 — late arrival

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

Exercise 5.3 — 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 5.4 — 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 6

Data Envelopment Analysis

Class: LP **Script:** `python/lab06_dea.py`
Online page

Prerequisites

- From the module *Linear Programming*, Chapter 9 «Duality theory»: the dual of an LP and strong duality, which give the envelopment form.
- From this module, Chapter 2 «Background: linear programming and sensitivity analysis»: shadow prices and their checking by perturbation.

Objectives

At the end of this chapter you will be able to:

- define relative efficiency as a weighted ratio of outputs to inputs;
- turn the fractional model into an LP and write its dual;
- read in the dual the reference units and the improvement targets;
- separate technical inefficiency from scale inefficiency.

6.1 Managerial motivation

A bank has twelve branches. Each one uses staff and operating costs and produces loans and deposits. Management wants to know which branches perform well and which do not, and by how much the weaker ones should reduce their resources.

The comparison is not trivial: the outputs have no common price, and a large branch is not inefficient just because it spends more. We need an index that compares each branch with the *best observed practices*, without fixing in advance how much a million of deposits is worth compared with a million of loans.

Data Envelopment Analysis (DEA) builds this index by solving one LP per branch. It is used to compare hospitals, schools, public offices, plants, departments: wherever there are similar units with several inputs and several outputs.

6.2 Units, inputs, outputs and efficiency

The units being compared are called DMUs (*Decision Making Units*). Each one turns the same types of inputs into the same types of outputs. The data are the following.

Data (input of the model)

Symbol	Type	Meaning
n	$\in \mathbb{Z}_{\geq 1}$	number of DMUs; the DMUs are indexed by $j \in \{1, 2, \dots, n\}$
m	$\in \mathbb{Z}_{\geq 1}$	number of inputs; the inputs are indexed by $i \in \{1, 2, \dots, m\}$
p	$\in \mathbb{Z}_{\geq 1}$	number of outputs; the outputs are indexed by $r \in \{1, 2, \dots, p\}$
x_{ij}	$\in \mathbb{Q}_{>0}$	amount of input i used by DMU j
y_{rj}	$\in \mathbb{Q}_{>0}$	amount of output r produced by DMU j
o	$\in \{1, 2, \dots, n\}$	the DMU being evaluated

Here x and y are *data*, not variables: this is the standard DEA notation, declared once and for all. Inputs and outputs are added up with non-negative weights: v_i for input i and u_r for output r.

Definition 6.1 — Relative efficiency

Given weights $u_r \geq 0$ and $v_i \geq 0$, not all zero, the *efficiency ratio* of DMU j is

$$E_j(u, v) = \frac{\sum_{r=1}^{p} u_r \, y_{rj}}{\sum_{i=1}^{m} v_i \, x_{ij}}.$$

The *relative efficiency* of DMU o is the maximum of $E_o(u, v)$ over the weights for which no DMU has a ratio greater than 1. DMU o is *efficient* if its relative efficiency equals 1.

Each DMU therefore chooses the weights that show it in the best light. If even with the most favourable weights it stays below 1, there is at least one other DMU (or a combination of DMUs) that does better with those weights: the inefficiency does not depend on an arbitrary choice of prices.

6.3 The input-oriented CCR model

The model is named after Charnes, Cooper and Rhodes (CCR), who proposed it. It is input-oriented: it measures how much the inputs can be reduced while leaving the outputs unchanged.

The problem in words. *We decide* the weights of the inputs and of the outputs. *The objective* is to maximise the efficiency ratio of DMU o. *The constraints* require that, with those same weights, no DMU has a ratio greater than 1.

Decision variables

We introduce the following $p + m$ non-negative variables:

$$\begin{cases} u_r = \text{weight of output } r, & \forall r \in \{1, 2, \dots, p\}, \\ v_i = \text{weight of input } i, & \forall i \in \{1, 2, \dots, m\}. \end{cases}$$

Using these variables, the fractional model for DMU o is the following:

Fractional CCR model for DMU o

$$\max \quad \frac{\sum_{r=1}^{p} u_r\, y_{ro}}{\sum_{i=1}^{m} v_i\, x_{io}} \tag{6.1a}$$

$$\text{subject to} \quad \frac{\sum_{r=1}^{p} u_r\, y_{rj}}{\sum_{i=1}^{m} v_i\, x_{ij}} \le 1, \qquad \forall j \in \{1, 2, \ldots, n\}, \tag{6.1b}$$

$$u_r \ge 0, \qquad \forall r \in \{1, 2, \ldots, p\}, \tag{6.1c}$$

$$v_i \ge 0, \qquad \forall i \in \{1, 2, \ldots, m\}. \tag{6.1d}$$

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

- the objective function (6.1a) maximises the efficiency ratio of DMU o;
- constraints (6.1b) require that with the same weights no DMU, including o, exceeds the ratio 1 (n fractional constraints);
- constraints (6.1c) and (6.1d) define the variables of the model.

The Charnes–Cooper transformation

Model (6.1) is not linear, but it has a decisive property: if all the weights are multiplied by the same $t > 0$, no ratio changes. We can therefore choose t so that the denominator of the objective equals 1.

With $\sum_{i=1}^{m} v_i\, x_{io} = 1$ the objective becomes the numerator alone, and each constraint (6.1b), multiplied by its positive denominator, becomes linear. We obtain the following LP, called the *multiplier form*.

CCR model, multiplier form, for DMU o

$$\max \quad \sum_{r=1}^{p} u_r\, y_{ro} \tag{6.2a}$$

$$\text{subject to} \quad \sum_{i=1}^{m} v_i\, x_{io} = 1, \tag{6.2b}$$

$$\sum_{r=1}^{p} u_r\, y_{rj} - \sum_{i=1}^{m} v_i\, x_{ij} \le 0, \qquad \forall j \in \{1, 2, \ldots, n\}, \tag{6.2c}$$

$$u_r \ge 0, \qquad \forall r \in \{1, 2, \ldots, p\}, \tag{6.2d}$$

$$v_i \ge 0, \qquad \forall i \in \{1, 2, \ldots, m\}. \tag{6.2e}$$

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

- the linear objective function (6.2a) maximises the weighted output of DMU o, which is its efficiency ratio because the weighted input equals 1;
- the linear constraint (6.2b) normalises the weighted input of DMU o to 1 (1 linear constraint);

- the linear constraints (6.2c) require that for every DMU the weighted output does not exceed the weighted input, that is, that the ratio does not exceed 1 (n linear constraints);
- constraints (6.2d) and (6.2e) define the variables of the model.

The optimal value $\tilde{h}_o$ of LP (6.2) is the relative efficiency of DMU o, and $0 < \tilde{h}_o \leq 1$. To evaluate n DMUs we solve n LPs, one per DMU: only the objective and the normalisation constraint change.

6.4 The envelopment form: the dual

LP (6.2) is a maximisation problem with one equality constraint and n $\leq$ constraints. Its dual has a free variable for the equality and a non-negative variable for each $\leq$ constraint (Chapter 2; the full theory is in the chapter Duality theory of the Linear Programming course).

To follow the DEA literature the dual variables are not called π: we introduce the following $1 + n$ dual variables,

$$\begin{cases} \theta = \text{dual variable of constraint (6.2b)}, \\ \lambda_j = \text{dual variable of constraint (6.2c) of DMU } j, \quad \forall j \in \{1, 2, \ldots, n\}. \end{cases}$$

The dual is written with the conversion rules and is called the *envelopment form*.

CCR model, envelopment form, for DMU o

$$\min \quad \theta \tag{6.3a}$$

$$\text{subject to} \quad \theta\, x_{io} - \sum_{j=1}^{n} \lambda_j\, x_{ij} \geq 0, \qquad \forall i \in \{1, 2, \ldots, m\}, \tag{6.3b}$$

$$\sum_{j=1}^{n} \lambda_j\, y_{rj} \geq y_{ro}, \qquad \forall r \in \{1, 2, \ldots, p\}, \tag{6.3c}$$

$$\theta \gtreqless 0, \tag{6.3d}$$

$$\lambda_j \geq 0, \qquad \forall j \in \{1, 2, \ldots, n\}. \tag{6.3e}$$

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

- the linear objective function (6.3a) minimises the factor θ by which all the inputs of DMU o are contracted;
- the linear constraints (6.3b) require that the combination of DMUs weighted by the λ_j uses, for every input, at most θ times the input of DMU o (m linear constraints; they come from the variables v_i);
- the linear constraints (6.3c) require that the same combination produces, for every output, at least as much as DMU o (p linear constraints; they come from the variables u_r);
- constraints (6.3d) and (6.3e) define the variables of the model.

What the envelopment form says

The combination $\sum_{j=1}^{n} \lambda_j \, (x_{\cdot j}, y_{\cdot j})$ is a *virtual* DMU built from the observed DMUs. The model looks for the virtual DMU that produces at least the outputs of o with the smallest fraction θ of its inputs.
With $\lambda_o = 1$ and the other $\lambda_j = 0$ we get $\theta = 1$: hence $\tilde{\theta} \leq 1$. By strong duality $\tilde{\theta} = \tilde{h}_o$: the optimal radial contraction of the inputs *is* the relative efficiency of the definition.

The optimal solution of the envelopment form gives three pieces of managerial information:

- the *reference set* (the *peers*) of DMU o: $\{j \in \{1, 2, \ldots, n\} : \tilde{\lambda}_j > 0\}$, the efficient DMUs that o should learn from;
- the *targets* of the inputs and of the outputs, that is, the projection of o onto the efficient frontier, which are written after introducing the slacks;
- the shadow prices of constraints (6.3b) and (6.3c), which are the optimal weights $\tilde{v}_i$ and $\tilde{u}_r$ of the multiplier form (one of the optimal vectors, if it is not unique).

6.5 The BCC model and scale efficiency

The CCR model assumes constant returns to scale: if a DMU is efficient, so is the DMU that uses twice the inputs and produces twice the outputs. For branches of very different sizes the assumption is strong.

The model of Banker, Charnes and Cooper (BCC) assumes variable returns to scale: it adds to the envelopment form the constraint that the λ_j sum to 1. The virtual DMU becomes a *convex* combination of the observed DMUs, that is, a DMU of comparable size.

BCC model, envelopment form, for DMU o

$$\min \quad \theta \tag{6.4a}$$

$$\text{subject to} \quad \theta \, x_{io} - \sum_{j=1}^{n} \lambda_j \, x_{ij} \geq 0, \qquad \forall i \in \{1, 2, \ldots, m\}, \tag{6.4b}$$

$$\sum_{j=1}^{n} \lambda_j \, y_{rj} \geq y_{ro}, \qquad \forall r \in \{1, 2, \ldots, p\}, \tag{6.4c}$$

$$\sum_{j=1}^{n} \lambda_j = 1, \tag{6.4d}$$

$$\theta \gtreqless 0, \tag{6.4e}$$

$$\lambda_j \geq 0, \qquad \forall j \in \{1, 2, \ldots, n\}. \tag{6.4f}$$

Description of the constraints that change with respect to the CCR model:

- the objective function (6.4a) and constraints (6.4b) and (6.4c) are those of the CCR model ($m + p$ linear constraints);
- the linear constraint (6.4d) requires the virtual DMU to be a convex combination of the observed DMUs (1 linear constraint);

- constraints (6.4e) and (6.4f) define the variables of the model.

Constraint (6.4d) shrinks the feasible region, and in a minimisation problem a smaller region never improves the optimum: $\tilde{\theta}^{\text{BCC}} \geq \tilde{\theta}^{\text{CCR}}$. The difference between the two measures is due to size.

Definition 6.2 — Scale efficiency

The *scale efficiency* of DMU o is

$$\text{SE}_o = \frac{\tilde{\theta}_o^{\text{CCR}}}{\tilde{\theta}_o^{\text{BCC}}} \in (0, 1].$$

It equals 1 when the DMU operates at the most productive scale; it is smaller than 1 when part of the inefficiency depends only on the size of the DMU.

The direction of the scale inefficiency is read in the CCR envelopment form: if $\sum_{j=1}^{n} \tilde{\lambda}_j < 1$ the DMU is too small (increasing returns), if $\sum_{j=1}^{n} \tilde{\lambda}_j > 1$ it is too large (decreasing returns). The reading holds when the optimal solution is unique.

6.6 Slacks and weak and strong efficiency

The radial contraction reduces all the inputs by the same fraction. After the contraction a margin may remain on a single input (or an output produced in excess by the virtual DMU): these are the *slacks*. They are computed with a second phase, with $\tilde{\theta}$ fixed.

We introduce the following $n + m + p$ non-negative variables:

$$\begin{cases} \lambda_j = \text{weight of DMU } j \text{ in the virtual DMU}, & \forall j \in \{1, 2, \dots, n\}, \\ s_i^- = \text{slack of input } i, & \forall i \in \{1, 2, \dots, m\}, \\ s_r^+ = \text{slack of output } r, & \forall r \in \{1, 2, \dots, p\}. \end{cases}$$

Second phase: maximum slacks for DMU o

$$\max \quad \sum_{i=1}^{m} s_i^- + \sum_{r=1}^{p} s_r^+ \tag{6.5a}$$

$$\text{subject to} \quad \sum_{j=1}^{n} \lambda_j \, x_{ij} + s_i^- = \tilde{\theta} \, x_{io}, \qquad \forall i \in \{1, 2, \dots, m\}, \tag{6.5b}$$

$$\sum_{j=1}^{n} \lambda_j \, y_{rj} - s_r^+ = y_{ro}, \qquad \forall r \in \{1, 2, \dots, p\}, \tag{6.5c}$$

$$\lambda_j \geq 0, \qquad \forall j \in \{1, 2, \dots, n\}, \tag{6.5d}$$

$$s_i^- \geq 0, \qquad \forall i \in \{1, 2, \dots, m\}, \tag{6.5e}$$

$$s_r^+ \geq 0, \qquad \forall r \in \{1, 2, \dots, p\}. \tag{6.5f}$$

Description of the objective function and of the constraints:

- the linear objective function (6.5a) maximises the sum of the slacks;
- the linear constraints (6.5b) write the input constraints of the envelopment form as equalities, with $\theta = \tilde{\theta}$ (m linear constraints);
- the linear constraints (6.5c) do the same for the outputs (p linear constraints);
- constraints (6.5d), (6.5e) and (6.5f) define the variables of the model.

With the solution of the second phase the targets of DMU o are

$$\hat{x}_{io} = \tilde{\theta}\, x_{io} - \tilde{s}_i^-, \quad \forall i \in \{1, 2, \dots, m\}, \qquad \hat{y}_{ro} = y_{ro} + \tilde{s}_r^+, \quad \forall r \in \{1, 2, \dots, p\}.$$

For the BCC model the constraint $\sum_{j=1}^{n} \lambda_j = 1$ is added and the $\tilde{\theta}$ of the BCC model is used.

Definition 6.3 — Weak and strong efficiency

DMU o is *strongly efficient* if $\tilde{\theta} = 1$ and all the optimal slacks of the second phase are zero. It is *weakly efficient* if $\tilde{\theta} = 1$ but at least one slack is positive.

A weakly efficient DMU cannot reduce all its inputs in the same proportion, but it can reduce at least one of them without losing output: another DMU does the same with less.

6.7 Worked example by hand

Five units, one input and one output

Five DMUs use a single input x and produce a single output y:

DMU	1	2	3	4	5
x_{1j}	2	4	6	5	2
y_{1j}	2	6	7	3	1

The multiplier form (6.2) for DMU 1, with detached coefficients, is:

$$\begin{aligned} \max \quad & 2\,u_1 & & & & \\ \text{subject to} \quad & & 2\,v_1 & = & 1 \\ & 2\,u_1 & - 2\,v_1 & \le & 0 \\ & 6\,u_1 & - 4\,v_1 & \le & 0 \\ & 7\,u_1 & - 6\,v_1 & \le & 0 \\ & 3\,u_1 & - 5\,v_1 & \le & 0 \\ & u_1 & - 2\,v_1 & \le & 0 \\ & u_1, & v_1 & \ge & 0 \end{aligned}$$

- **Step 1 — the solver values.** Solving the five CCR LPs and the five BCC LPs gives:

$$\tilde{\theta}^{\text{CCR}} = (0.6667,\ 1,\ 0.7778,\ 0.4,\ 0.3333), \qquad \tilde{\theta}^{\text{BCC}} = (1,\ 1,\ 1,\ 0.5,\ 1).$$

For DMU 1 the optimal weights are $\tilde{u}_1 = 0.3333$ and $\tilde{v}_1 = 0.5$, and in the envelopment form the only peer is DMU 2 with $\tilde{\lambda}_2 = 0.3333$. For DMU 5 the BCC model gives an output slack $\tilde{s}_1^+ = 1$.

- **Step 2 — check of the CCR.** With a single input and a single output the weights cancel in the ratio, and the efficiency is the ratio y/x divided by the best one, $6/4 = 1.5$ of DMU 2:

$$\tilde{h}_1 = \frac{2/2}{1.5} = 0.6667, \qquad \tilde{h}_3 = \frac{7/6}{1.5} = 0.7778, \qquad \tilde{h}_4 = \frac{3/5}{1.5} = 0.4.$$

- **Step 3 — check of duality for DMU 1.** The weights satisfy the normalisation and give the efficiency; the constraint of DMU 2 is active:

$$2 \cdot 0.5 = 1, \qquad 2 \cdot 0.3333 = 0.6667, \qquad 6 \cdot 0.3333 - 4 \cdot 0.5 = 0.$$

  In the envelopment form the virtual DMU $\tilde{\lambda}_2 \cdot (4, 6)$ uses exactly $\tilde{\theta} \cdot 2$ of input and produces exactly the output of DMU 1:

$$0.3333 \cdot 4 = 1.3333 = 0.6667 \cdot 2, \qquad 0.3333 \cdot 6 = 2.$$

- **Step 4 — check of the BCC for DMU 4.** The solver combines DMUs 1 and 2 with $\tilde{\lambda}_1 = 0.75$ and $\tilde{\lambda}_2 = 0.25$, which sum to 1:

$$0.75 \cdot 2 + 0.25 \cdot 4 = 2.5 = 0.5 \cdot 5, \qquad 0.75 \cdot 2 + 0.25 \cdot 6 = 3.$$

  The scale efficiency of DMU 4 is $0.4/0.5 = 0.8$; that of DMU 1 is 0.6667 and that of DMU 3 is 0.7778.

- **Step 5 — weak efficiency.** DMU 5 has $\tilde{\theta}^{\text{BCC}} = 1$: no convex combination uses less than 2 units of input. But DMU 1 uses the same input and produces 1 more unit: the slack $\tilde{s}_1^+ = 1$ makes DMU 5 weakly efficient.

6.8 Case study

Twelve branches of a bank ($n = 12$), with two inputs ($m = 2$: staff in FTE, *Full-Time Equivalent*, and operating costs in thousands of euros) and two outputs ($p = 2$: loans and deposits in millions of euros). The data are synthetic but realistic: the script generates them with a fixed seed, from a production function with decreasing returns to scale and a random inefficiency per branch (file `data/dea_filiali.csv`).

branch	1	2	3	4	5	6	7	8	9	10	11	12
staff (FTE)	10	25	23	18	17	27	9	24	12	10	20	30
costs (k€)	720	1420	1130	940	990	1620	640	1300	690	500	1020	1370
loans (M€)	19.6	60.7	43.5	31.7	39.6	39.7	21.9	34.5	29.2	23.7	42.8	63.5
deposits (M€)	18.5	58.7	63.0	50.5	43.2	73.1	38.0	72.7	28.7	26.3	45.7	62.9

6.9 Implementation

The core of the script (full version in `lab06_dea.py`) is the function that builds the envelopment form; one argument adds the convexity constraint of the BCC.

```
def inviluppo(X, Y, o, vrs=False):
    m_, n_ = X.shape          # inputs x DMUs
```

```
    p_ = Y.shape[0]            # outputs
    md = gp.Model("dea_envelopment")
    theta = md.addVar(lb=-GRB.INFINITY, name="theta")
    lam = md.addVars(n_, name="lambda")
    c_in = md.addConstrs(
        (theta * X[i, o] - gp.quicksum(lam[j] * X[i, j]
         for j in range(n_)) >= 0 for i in range(m_)), name="input")
    c_out = md.addConstrs(
        (gp.quicksum(lam[j] * Y[r, j] for j in range(n_))
         >= Y[r, o] for r in range(p_)), name="output")
    if vrs:                    # BCC: convex combination
        md.addConstr(lam.sum() == 1, name="convexity")
    md.setObjective(theta, GRB.MINIMIZE)
    md.optimize()
    return md, theta, lam, c_in, c_out
```

The function `moltiplicatori` builds LP (6.2) in the same way and the function `scarti` the second phase (6.5). For every branch the script solves both the primal and the dual and checks with an `assert` that the two optimal values coincide. Each model has at most 16 variables and 13 constraints.

6.10 Results

```
 branch     CCR      BCC    scale     sum lambda (CCR)    returns
      1   0.8055   0.9000   0.8950                0.671   increasing
      2   1.0000   1.0000   1.0000                1.000   constant
      3   0.9815   1.0000   0.9815                1.917   decreasing
      4   0.9233   0.9403   0.9819                1.423   decreasing
      5   0.9604   0.9611   0.9993                1.450   decreasing
      6   0.7600   1.0000   0.7600                1.924   decreasing
      7   1.0000   1.0000   1.0000                1.000   constant
      8   0.9419   1.0000   0.9419                1.913   decreasing
      9   1.0000   1.0000   1.0000                1.000   constant
     10   1.0000   1.0000   1.0000                1.000   constant
     11   0.8997   0.9470   0.9501                1.641   decreasing
     12   0.9779   1.0000   0.9779                2.679   decreasing
CCR-efficient branches: [2, 7, 9, 10]
BCC-efficient branches: [2, 3, 6, 7, 8, 9, 10, 12]
Mean CCR efficiency: 0.9375; mean BCC: 0.9790
```

For all twelve branches the optimal value of the envelopment form coincides with that of the multiplier form, and the shadow price of the normalisation constraint (6.2b) equals $\tilde{\theta}$: this is strong duality checked numerically.

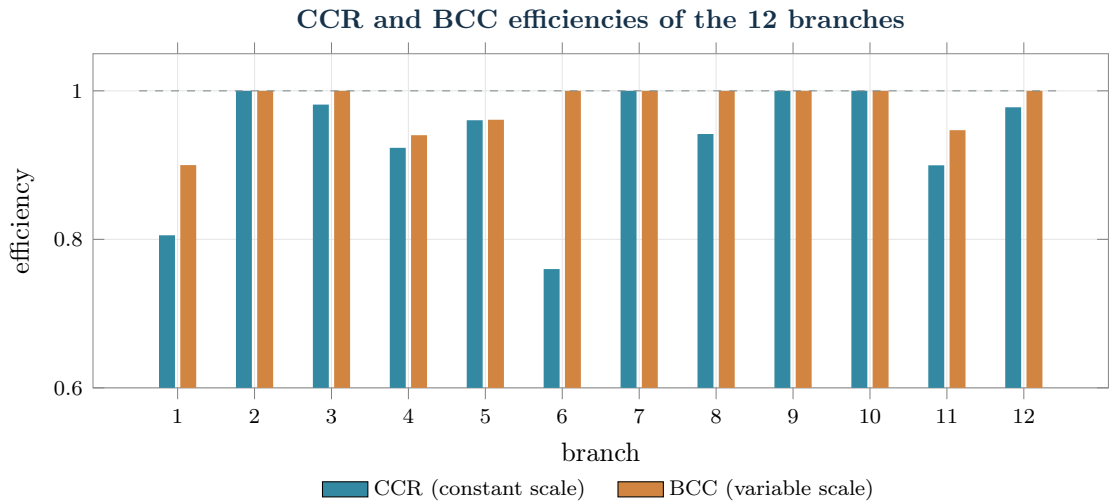


Figure 6.1: CCR efficiency (teal) and BCC efficiency (orange) of each branch; the dashed line is the value 1. Branches 3, 6, 8 and 12 are efficient in the BCC model but not in the CCR model: their inefficiency is entirely due to scale.

The following table reports, for the branches that are inefficient in the CCR model, the peers with their $\tilde{\lambda}_j$ and the input targets after the second phase (file `data/dea_target.csv`).

branch	$\tilde{\theta}$	peers (j: $\tilde{\lambda}_j$)	staff	target	costs	target
1	0.8055	9: 0.671	10	8.05	720	463.2
3	0.9815	7: 1.075; 10: 0.842	23	18.10	1130	1109.1
4	0.9233	7: 1.119; 10: 0.304	18	13.11	940	867.9
5	0.9604	7: 0.223; 9: 1.024; 10: 0.203	17	16.33	990	950.8
6	0.7600	7: 1.924	27	17.31	1620	1231.2
8	0.9419	7: 1.913	24	17.22	1300	1224.4
11	0.8997	2: 0.106; 10: 1.535	20	17.99	1020	917.7
12	0.9779	10: 2.679	30	26.79	1370	1339.7

What to do branch by branch

Branch 6 is the least efficient ($\tilde{\theta} = 0.76$) but it is efficient in the BCC model: it does not work badly, it is too large. Its only peer is branch 7, the smallest, taken 1.924 times: a sum of the $\tilde{\lambda}_j$ greater than 1 signals decreasing returns. The recommendation is not "cut", but to consider splitting it into two offices.
Branch 1 is the opposite case: the sum of the $\tilde{\lambda}_j$ is 0.671, so it is too small, and the BCC model also finds it inefficient (0.90). After the radial contraction a slack of 116.8 k€ remains on the costs: the cost target is 463.2 k€ against the current 720.

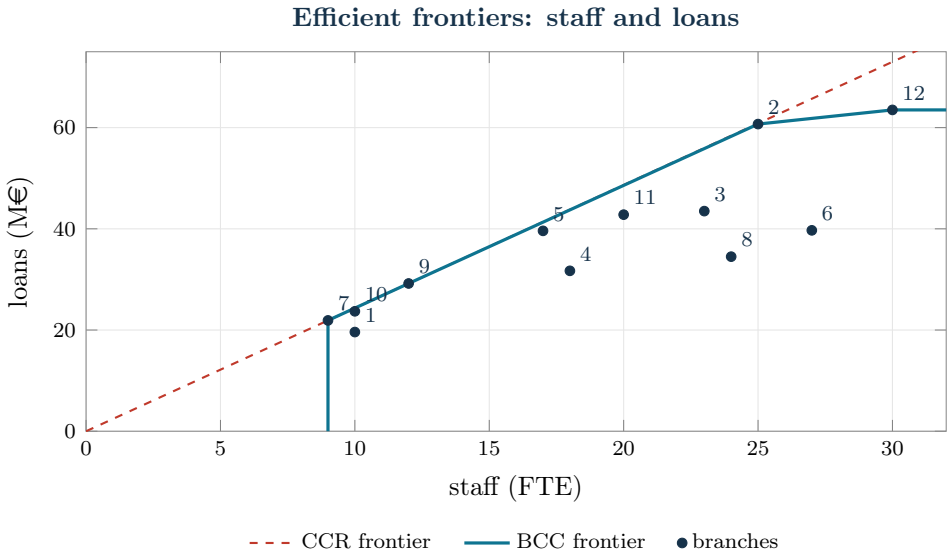


Figure 6.2: One-input, one-output view (staff and loans) of the twelve branches. The CCR frontier (dashed) is the ray from the origin with the best slope, 2.4333 M€ per FTE of branch 9; the BCC frontier (solid) goes through branches 7, 9, 2 and 12. In this reduced view the efficiencies differ from those of the full model.

The figure shows the geometric meaning of the two models. Below the CCR ray a branch pays both technical and scale inefficiency; below the BCC piecewise line it pays only the former. Branch 12 lies on the BCC frontier but far from the ray: in the view it has BCC efficiency 1 and CCR efficiency 0.870.

6.11 Sensitivity analysis

Shadow prices of the envelopment form

For branch 6 the solver gives these shadow prices of the output constraints (6.3c), with their validity ranges:

```
branch 6: theta = 0.7600
  output loans   : y = 39.7 | Pi = 0.00000 | valid for the right-hand side in
      [-inf, 42.13]
  output deposits: y = 73.1 | Pi = 0.01040 | valid for the right-hand side in
      [68.89, inf]
Check: right-hand side of deposits from 73.1 to 74.1 -> theta from 0.7600 to
    0.7704 (change +0.01040 = Pi 0.01040)
Comparison: primal weight u = (0.00000, 0.01040)
```

The shadow price is the primal weight

Increasing the right-hand side of a $\geq$ constraint shrinks the feasible region, and in a minimisation problem $\tilde{\theta}$ cannot decrease: the shadow prices are ≥ 0. The shadow price of deposits, 0.01040, is the weight $\tilde{u}_2$ of the multiplier form, as duality predicts.
The check by perturbation confirms it: requiring one more million of deposits raises $\tilde{\theta}$ by 0.0104, and the value stays valid for any right-hand side above 68.89. The shadow price of loans is zero up to 42.13: the virtual DMU already produces 2.43 million of loans more than branch 6 (the slack of the second phase), and requiring more up to that threshold costs

nothing.

Stability of the frontier

Branch 7 is a peer of five inefficient branches, like branch 10. Removing it from the comparison and recomputing the CCR model:

```
  branch  3: CCR 0.9815 -> 1.0000  <- changes
  branch  4: CCR 0.9233 -> 0.9795  <- changes
  branch  5: CCR 0.9604 -> 0.9791  <- changes
  branch  6: CCR 0.7600 -> 0.9156  <- changes
  branch  8: CCR 0.9419 -> 1.0000  <- changes
Without branch 7: 5 branches change efficiency; newly efficient: [3, 8]
```

DEA measures efficiency *relative* to the observed DMUs: a single DMU on the frontier can change the verdict on many others. Before acting on the targets it is worth checking the data of the efficient DMUs cited most often as peers (a recording error would make them "too good") and repeating the analysis without them.

6.12 In summary

In summary

- Efficiency is the weighted output/input ratio with the most favourable weights; fixing the weighted input to 1 gives an LP.
- The dual, the envelopment form, looks for a virtual unit that does at least as much with less: $\tilde{\theta} = \tilde{h}_o$, and the $\tilde{\lambda}_j > 0$ are the peers.
- BCC adds $\sum_{j=1}^{n} \lambda_j = 1$; scale efficiency is $\tilde{\theta}^{\text{CCR}} / \tilde{\theta}^{\text{BCC}}$.
- Efficiency is relative to the observed units: the frontier and the most cited peers must be checked.

6.13 Exercises

Exercise 6.1 — check

With the optimal weights of the multiplier form for branch 4, compute the efficiency ratio of all twelve branches. Which branches have ratio 1, and how do they relate to the peers of branch 4?

Exercise 6.2 — output orientation

Write the output-oriented CCR model (maximise the factor φ by which the outputs are expanded for given inputs). Check on the data that $\tilde{\varphi} = 1/\tilde{\theta}$ for every branch, and that in the BCC model the relation does not hold in general.

Exercise 6.3 — a weakly efficient branch

Add a branch 13 equal to branch 7 but with operating costs of 700 k€. Compute its $\tilde{\theta}$ and its slacks: is it efficient? In what sense?

Exercise 6.4 — weight restrictions

Management believes that one million of deposits is worth at least as much as one million of loans. Add the constraint $u_2 \geq u_1$ to the multiplier form and recompute the CCR efficiencies: which branches remain efficient?

Exercise 6.5 — super-efficiency

The four CCR-efficient branches all have $\tilde{\theta} = 1$. To rank them, solve the envelopment form removing DMU o from the reference set ($\lambda_o = 0$): the optimal value can exceed 1. Which branch is the most robust?

Part III

Decisions under uncertainty

Chapter 7

The Newsvendor and its variants

Class: 1D convex / scenario-based stochastic LP **Script:** `python/lab07_newsvendor.py`
Online page

Prerequisites

- From this module, Chapter 2 «Background: linear programming and sensitivity analysis»: shadow prices and their validity ranges.
- From this module, Chapter 3 «The solver: building, solving, interpreting»: building and solving an LP in `gurobipy`.

Objectives

At the end of this chapter you will be able to:

- derive and apply the quantile rule;
- turn a stochastic problem into a scenario-based LP with the positive-part trick;
- quantify the value of modelling uncertainty (VSS);
- incorporate service levels and risk aversion.

7.1 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).

7.2 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

$$\min \quad c_o\,\mathbb{E}\big[(q-D)^+\big] + c_u\,\mathbb{E}\big[(D-q)^+\big] \tag{7.1a}$$

$$\text{subject to} \quad q \geq 0. \tag{7.1b}$$

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

- the convex objective function (7.1a) 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 (7.1b) 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.

7.3 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.**

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

- **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.

7.4 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
```

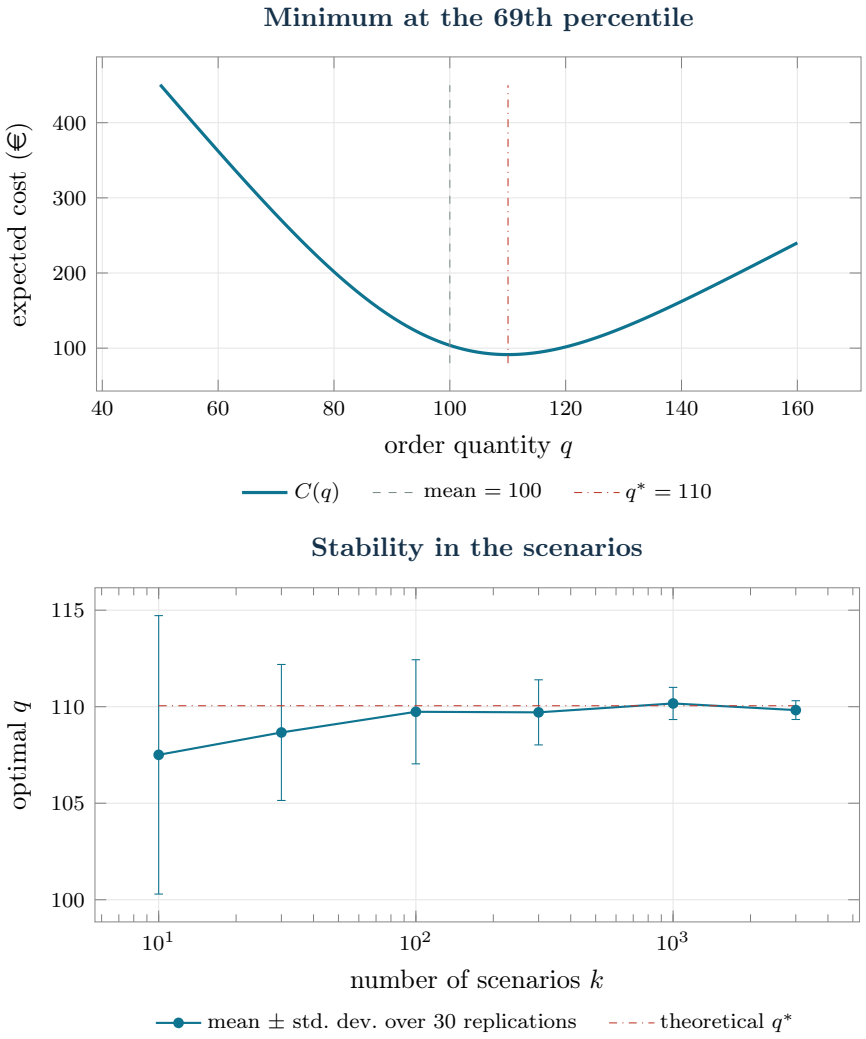


Figure 7.1: Left: the expected cost $C(q)$ is convex and the minimum falls at $q^* = 110$ (69th percentile of demand), not at the mean 100. Right: the optimal q estimated from k sampled scenarios, mean and standard deviation over 30 replications — with few scenarios the decision is unstable.

7.5 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 $s \in \{1, 2, \ldots, k\}$; 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:

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

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

Scenario-based Newsvendor LP model

$$\min \quad \sum_{s=1}^{k} \pi_s \big(c_o\, o_s + c_u\, u_s\big) \tag{7.2a}$$

$$\text{subject to} \quad o_s \geq q - d_s, \qquad \forall s \in \{1, 2, \ldots, k\}, \tag{7.2b}$$

$$u_s \geq d_s - q, \qquad \forall s \in \{1, 2, \ldots, k\}, \tag{7.2c}$$

$$q \geq 0, \tag{7.2d}$$

$$o_s,\ u_s \geq 0, \qquad \forall s \in \{1, 2, \ldots, k\}. \tag{7.2e}$$

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

- the linear objective function (7.2a) 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 (7.2b) and (7.2c) 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 (7.2d) and (7.2e) 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 (7.2b) 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 quantile regression.

```
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.

7.6 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.

7.7 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
```

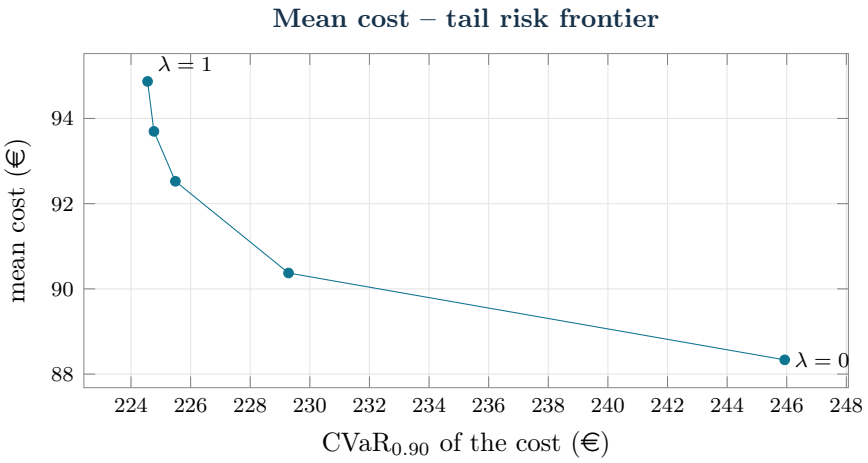


Figure 7.2: Frontier between mean cost and $\text{CVaR}_{0.90}$ of the cost, obtained by varying the weight λ of risk aversion: from $\lambda = 0$ (optimal on average, heavy tail) to $\lambda = 1$ (maximum protection). The first stretch is almost vertical: a lot of protection for very little mean cost.

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.

7.8 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.

7.9 In summary

In summary

- The quantity is chosen before observing demand, and the optimum is the quantile $q^* = F^{-1}\big(c_u/(c_u + c_o)\big)$, not the mean.
- With k scenarios the newsvendor becomes an LP with $2k + 1$ variables: the positive parts are linearized without binary variables.
- The VSS measures the gain from modelling uncertainty; decisions estimated from few scenarios are unstable.
- Cycle service level and fill rate are different measures; with mean-CVaR a little risk aversion gives a lot of protection.

7.10 Exercises

Exercise 7.1 — 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 7.2 — 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 7.3 — 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 7.4 — 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 7.5 — 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 7.6 — 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 8

VaR and CVaR: measuring and optimizing risk

Class: scenario-based LP **Script:** `python/lab08_var_cvar.py`
Online page

Prerequisites

- From this module, Chapter 7 «The Newsvendor and its variants»: the scenario-based LP, the positive-part trick and the mean-CVaR objective.
- From this module, Chapter 2 «Background: linear programming and sensitivity analysis»: shadow prices and the sensitivity protocol.

Objectives

At the end of this chapter you will be able to:

- define and compute VaR and CVaR on a discrete distribution;
- use the Rockafellar–Uryasev linear formulation;
- build portfolios and logistics plans protected against the worst scenarios;
- recognize the statistical limits of tail estimates.

8.1 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.

8.2 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 $k \in \mathbb{Z}_{\ge 1}$, indexed by $s \in \{1, 2, \dots, k\}$; for each scenario, the probability $\pi_s \in \mathbb{Q}_{\ge 0}$ (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:

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

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

Rockafellar–Uryasev LP model for the CVaR

$$\min \quad \eta + \frac{1}{1-\alpha} \sum_{s=1}^{k} \pi_s \, \xi_s \tag{8.1a}$$
$$\text{subject to} \quad \xi_s \ge \ell_s(\boldsymbol{x}) - \eta, \qquad \forall s \in \{1, 2, \dots, k\}, \tag{8.1b}$$
$$\boldsymbol{x} \in X, \tag{8.1c}$$
$$\eta \gtreqless 0, \tag{8.1d}$$
$$\xi_s \ge 0, \qquad \forall s \in \{1, 2, \dots, k\}. \tag{8.1e}$$

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

- the linear objective function (8.1a) 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 (8.1b) define the excesses: only the scenarios with a loss beyond the threshold contribute (k linear constraints);
- constraint (8.1c) collects the constraints of the underlying decision problem (polyhedral);
- constraints (8.1d) and (8.1e) define the variables: η is *free* (the symbol $\gtrless$), 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.

8.3 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: $\text{VaR}_{0.80} = 12$.
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:

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

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.

8.4 Case study 1: mean-CVaR portfolio

Eight sector stocks (ENE, FIN, TEC, IND, SAN, CON, UTL, MAT) and $k = 220$ 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 $\text{CVaR}_{0.90}$ with expected return $\geq 8\%$ per year.

```
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
Composition: ENE 9.9 FIN 1.9 TEC 10.5 SAN 8.6 UTL 46.5 MAT 22.5
```

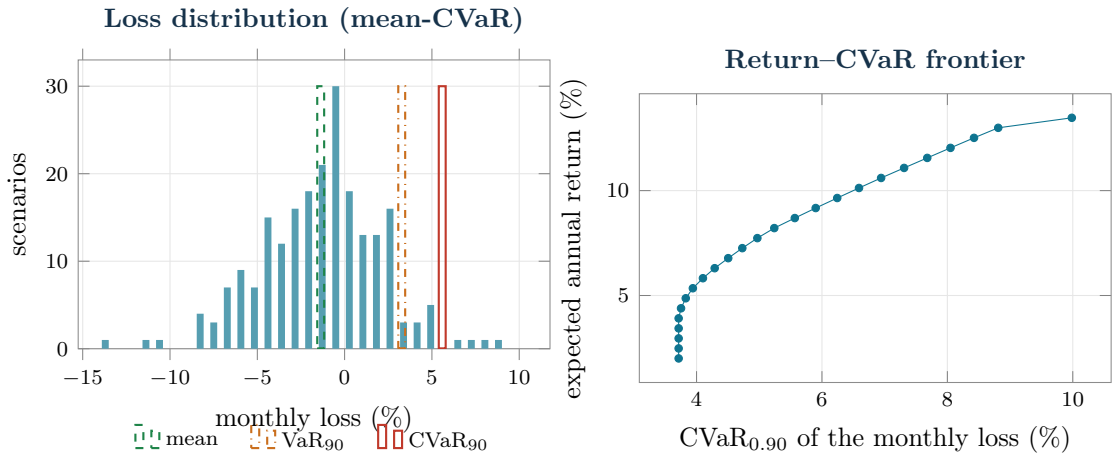


Figure 8.1: Left: histogram of the monthly losses of the mean-CVaR portfolio over the 220 scenarios, with the mean (green), VaR_{90} (orange) and CVaR_{90} (red): the CVaR is the average of the tail to the right of the VaR. Right: return–CVaR frontier obtained by varying the minimum required return.

Managerial insight

With an expected return of 8% per year the portfolio gains 0.67% per month on average, but in the worst 10% of the months it loses 4.89% on average: this is the number the mean hides. The CVaR concentrates the portfolio on UTL and MAT and keeps a share of TEC which, although mediocre on average, moves against the trend in the worst scenarios of this sample. The CVaR looks only where it hurts: the losses above the VaR.

Measuring risk with the variance instead of the CVaR leads to the Markowitz portfolio, a quadratic model: it is in the module *NLP Modelling*, page "The Markowitz portfolio".

8.5 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

$$\min \quad (1-\lambda)\Big[\sum_{a=1}^{2} c_a^{\text{pre}} x_a + \sum_{s=1}^{k} \pi_s\, \phi_s\Big] + \lambda\, \text{CVaR}_\alpha \tag{8.2a}$$

$$\text{subject to} \quad f_{as} \le \delta_{as}\, x_a, \quad \forall a \in \{1.2\},\ \forall s \in \{1,2,\dots,k\}, \tag{8.2b}$$

$$f_{1s} + f_{2s} + u_s \ge d_s, \qquad \forall s \in \{1,2,\dots,k\}, \tag{8.2c}$$

$$x_a,\ f_{as},\ u_s \ge 0, \qquad \forall a \in \{1.2\},\ \forall s \in \{1,2,\dots,k\}, \tag{8.2d}$$

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 (8.1).

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

- the objective function (8.2a) 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 (8.2b) 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 (8.2c) 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 (8.2d) 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.

8.6 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 7

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.

8.7 In summary

In summary

- VaR is a high loss threshold, the α quantile; CVaR is the average loss in the tail beyond it.
- CVaR is convex, subadditive and sensitive to the severity of the tail; VaR is not.
- The Rockafellar–Uryasev LP uses a free threshold η and excesses $\xi_s \geq 0$: at the optimum η^* is a VaR and the value is the CVaR.
- Extreme tails estimated from few scenarios are noisy: dozens of scenarios beyond the quantile are needed.

8.8 Exercises

Exercise 8.1 — 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 8.2 — subadditivity

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

Exercise 8.3 — 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 8.4 — CVaR-constrained

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

Exercise 8.5 — 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 9

Arbitrage and arbitrage-free pricing

Class: LP **Script:** `python/lab09_arbitrage.py`
Online page

Prerequisites

- From this module, Chapter 2 «Background: linear programming and sensitivity analysis»: duality, complementary slackness and strong duality.
- From this module, Chapter 3 «The solver: building, solving, interpreting»: free variables in `gurobipy` and the reading of `Pi`.

Objectives

At the end of this chapter you will be able to:

- formulate the search for an arbitrage as an LP;
- read the risk-neutral probabilities in the duals of the constraints;
- compute the interval of consistent prices of a security;
- price a new security (an option) in a complete or incomplete market.

9.1 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.

9.2 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

$$\min \quad \sum_{i=0}^{n} s_i^0 \, x_i \tag{9.1a}$$

$$\text{subject to} \quad \sum_{i=0}^{n} s_{ij}^1 \, x_i \geq 0, \qquad \forall j \in \{1, 2, \ldots, m\}, \tag{9.1b}$$

$$x_i \gtreqless 0, \qquad \forall i \in \{0, 1, \ldots, n\}. \tag{9.1c}$$

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

- the linear objective function (9.1a) minimizes the cost of the portfolio today: a negative value is an immediate cash inflow;
- the linear constraints (9.1b) impose that tomorrow the portfolio does not lose in any of the m states of the world (m linear constraints);
- constraints (9.1c) 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*.

9.3 Duality is pricing: the risk-neutral probabilities

We apply to model (9.1) the duality rules of Chapter 2 (minimization primal: table read from right to left). Each state constraint (9.1b), 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

$$\max \quad \sum_{j=1}^{m} 0 \cdot p_j \tag{9.2a}$$

$$\text{subject to} \quad \sum_{j=1}^{m} s_{ij}^1 \, p_j = s_i^0, \qquad \forall i \in \{0, 1, \ldots, n\}, \tag{9.2b}$$

$$p_j \geq 0, \qquad \forall j \in \{1, 2, \ldots, m\}. \tag{9.2c}$$

- objective (9.2a) is identically zero: of the dual only *feasibility* matters (every feasible solution is optimal);
- constraints (9.2b) 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 (9.2c) 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 (9.2b) 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 $j \in \{1, 2, \dots, m\}$.

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 $\sum_i s_i^0 x_i = \frac{1}{R} \sum_j q_j \left(\sum_i s_{ij}^1 x_i \right) \geq 0$: no arbitrage.

9.4 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.

9.5 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)
```

9.6 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.

9.7 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

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

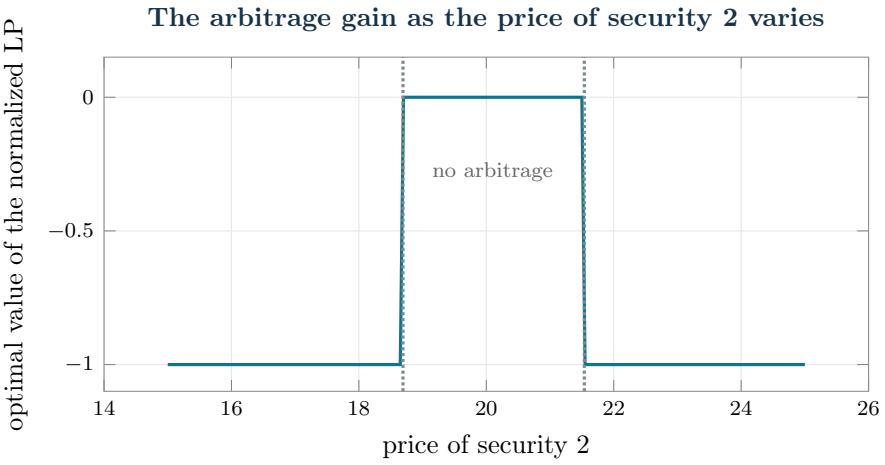


Figure 9.1: Optimal value of the normalized arbitrage LP as the price of security 2 varies (securities 0 and 1 quoted at 1 and 13). Inside the interval [18.69, 21.54] (dashed lines) the value is 0: consistent prices; outside it, the value drops to −1: an arbitrage exists, in one direction or the other.

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.

9.8 In summary

In summary

- Detecting an arbitrage is an LP with free positions and one constraint per state: payoff ≥ 0.
- Optimum < 0 is a type A arbitrage, the normalization (cost ≥ -1) avoids unboundedness; optimum 0 with a slack state constraint is type B.
- The duals, multiplied by R, are risk-neutral probabilities: no arbitrage if and only if there is q with $q_j > 0$ for every state.
- Two LPs give the interval of consistent prices: a unique price in a complete market, an interval in an incomplete one.

9.9 Exercises

Exercise 9.1 — 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 9.2 — 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 9.3 — 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 10

Robust and quantile regression: estimating parameters with an LP

Class: LP **Script:** `python/lab10_regression.py`
Online page

Prerequisites

- From this module, Chapter 7 «The Newsvendor and its variants»: the quantile rule and the critical fractile.
- From this module, Chapter 2 «Background: linear programming and sensitivity analysis»: the dual of an LP and shadow prices.

Objectives

At the end of this chapter you will be able to:

- formulate regression with absolute deviations as an LP;
- read its dual, recognising the points that support the estimate and the quantile property;
- estimate the quantity to produce conditional on the conditions of the day;
- select the relevant features with a budget on the coefficients, reading its shadow price.

10.1 Managerial motivation

Every model seen so far starts from known parameters: the costs of chapter 4, the hourly prices of chapter 5, the scenarios of chapter 7. 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.

Here we use optimization 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 7 needs.

10.2 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

$$\min \quad \sum_{i=1}^{n} \big[\tau\, u_i + (1 - \tau)\, v_i \big] \tag{10.1a}$$

$$\text{subject to} \quad \sum_{j=1}^{p} w_j\, x_{ij} + b + u_i - v_i = y_i, \qquad \forall i \in \{1, 2, \dots, n\}, \tag{10.1b}$$

$$w_j \gtreqless 0, \qquad \forall j \in \{1, 2, \dots, p\}, \tag{10.1c}$$

$$b \gtreqless 0, \tag{10.1d}$$

$$u_i,\ v_i \geq 0, \qquad \forall i \in \{1, 2, \dots, n\}. \tag{10.1e}$$

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

- the linear objective function (10.1a) 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 (10.1b) define the deviation of every observation as the difference of its two parts, $u_i - v_i = y_i -$ prediction (n linear constraints);
- constraints (10.1c), (10.1d) and (10.1e) 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 (10.1b) 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 7.

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 7 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 saves τ on every observation lying *above* b and costs $1-\tau$ more 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 $\tilde{\boldsymbol{\pi}} = \left(-\tfrac{4}{13}, -\tfrac{4}{13}, -\tfrac{4}{13}, -\tfrac{4}{13}, -\tfrac{2}{13}, \tfrac{9}{13}, \tfrac{9}{13}\right)$: $\tau = 9/13$ 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 7, written as an LP. With features it becomes a quantile *conditional* on the conditions of the day.

10.3 Case study: the bakery history

Sixty days of sales of the bakery of chapter 7: 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, minimising the sum of the absolute deviations (LP, $\tau = 1/2$) twice: on all the 60 days and on the 55 normal days only.

```
LP on the 60 days              : demand = 209.77 - 3.63 * temperature
LP without the 5 strike days   : demand = 210.49 - 3.58 * temperature
Mean absolute deviation over the 55 normal days: 13.08 pieces
Distance between the two intercepts: 0.72 pieces
```

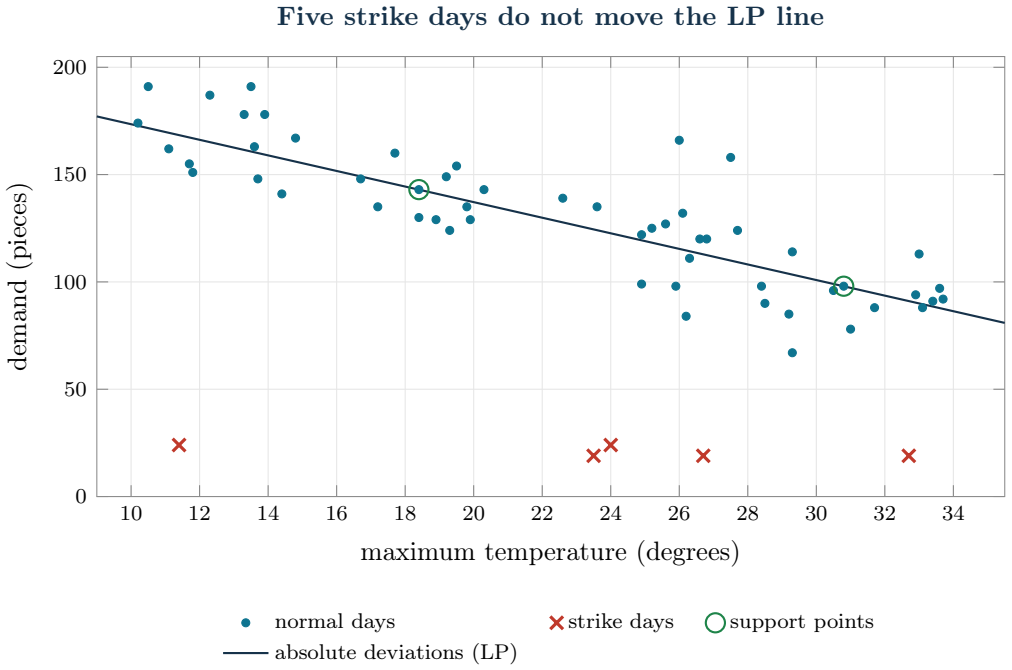


Figure 10.1: The 60 days of the bakery: the LP line (blue) ignores the five strike days and stays 0.72 pieces, on the intercept, from the line estimated on the normal days only. The two points circled in green are the support points, the only ones the LP line interpolates.

Managerial insight

The LP pays every deviation *once*, in proportion to its size: five days with deviations of about a hundred pieces weigh as much as a hundred days with deviations of five, and the line stays where the normal data are. The model reaches on its own the answer one would get after cleaning the history by hand: it is the difference between a model that needs supervision and one that can be left running every week.

Minimising instead the sum of the *squared* deviations leads to least-squares regression, a quadratic model: the comparison with the LP is in the module *NLP Modelling*, page "Least-squares regression".

10.4 The dual reading: the support points

Associating a dual π_i with each deviation constraint (10.1b), the optimality condition on the (non-negative) variables u_i and v_i confines every dual to a window that depends on τ alone:

$$-(1-\tau) \;\le\; \tilde{\pi}_i \;\le\; \tau, \qquad \forall i \in \{1, 2, \ldots, n\}, \tag{10.2}$$

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

$$\sum_{i=1}^{n} \tilde{\pi}_i \, x_{ij} = 0 \quad \forall j \in \{1, 2, \ldots, p\}, \qquad \sum_{i=1}^{n} \tilde{\pi}_i = 0. \tag{10.3}$$

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

Very few observations, the support points, sustain the solution: the others could disappear without changing it, as long as they stay on their side of the estimate. It is the complementarity of chapter 2 at work.

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
```

10.5 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

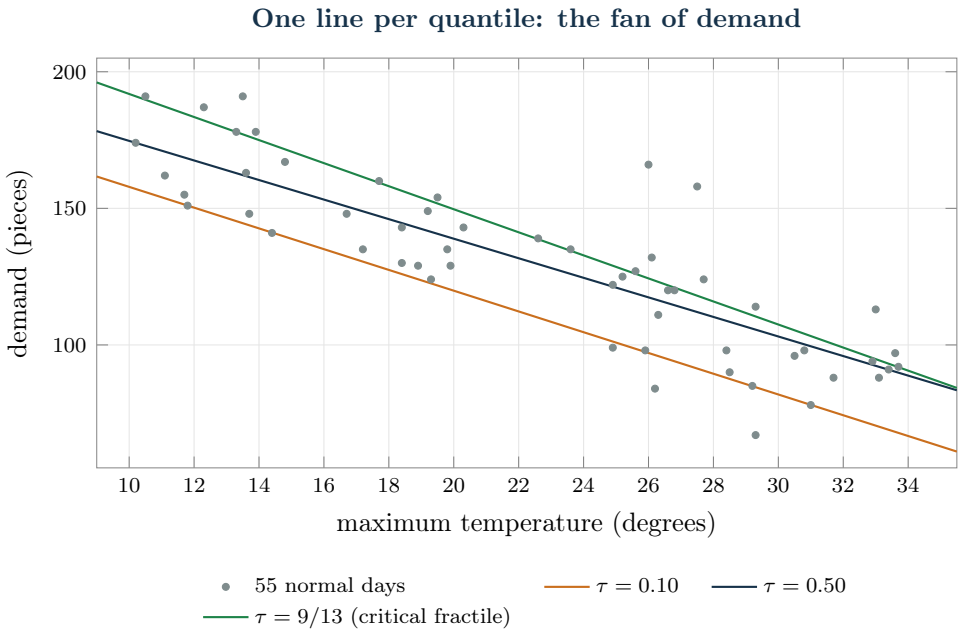


Figure 10.2: The lines estimated with temperature alone for three values of τ: each leaves the corresponding fraction of observations below it. The green line, with τ equal to the critical fractile of the bakery, is the quantity to bake day by day.

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 7 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.

10.6 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,

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

and the datum $t \in \mathbb{Q}_{\geq 0}$, the total budget. Model (10.1) gains the constraints

$$-z_j \leq w_j \leq z_j, \qquad \forall j \in \{1, 2, \dots, p\}, \tag{10.4a}$$

$$\sum_{j=1}^{p} z_j \leq t, \tag{10.4b}$$

$$z_j \geq 0, \qquad \forall j \in \{1, 2, \dots, p\}. \tag{10.4c}$$

Constraints (10.4a) 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 (10.4b) 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 (10.4b) 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
```

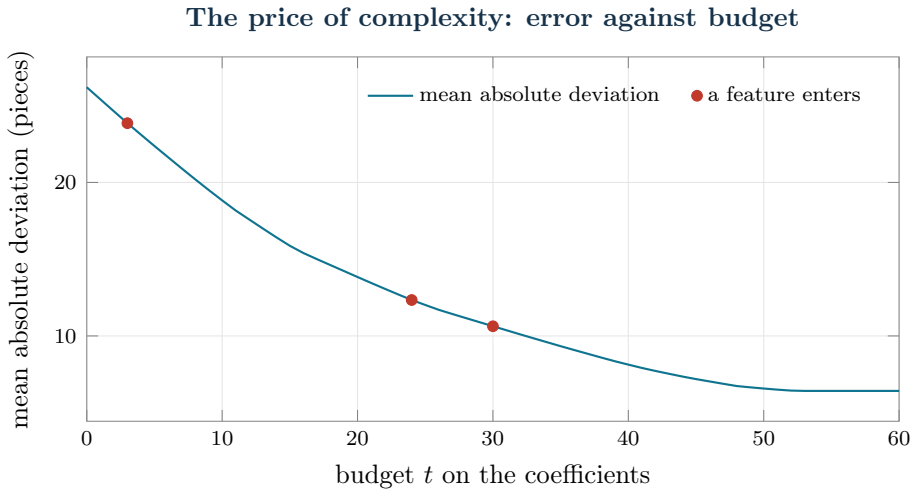


Figure 10.3: The mean absolute deviation as a function of the budget on the coefficients: a convex decreasing curve, like the optimal value of an LP as the right-hand side varies. The red dots mark the budget at which temperature ($t = 3$), price ($t = 24$) and weekend ($t = 30$) enter, in this order.

Managerial insight

The shadow price of constraint (10.4b) 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.

10.7 In summary

In summary

- With absolute deviations regression is an LP: deviation $= u_i - v_i$ with $u_i, v_i \geq 0$, and the line stays where the normal data are.
- Paying τ above and $1 - \tau$ below gives the quantile of level τ; with τ equal to the critical fractile it gives the quantity to produce.
- Every dual lies in $[-(1 - \tau), \tau]$; the interpolated observations are the support points that determine the estimate.
- A budget on $\sum_j |w_j|$ selects the features and keeps an LP; its shadow price is the price of complexity.

10.8 Exercises

Exercise 10.1 — 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 10.2 — 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 10.3 — 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 10.4 — 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 10.5 — 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 11

Organization of the lab

11.1 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	Data Envelopment Analysis (Ch. 6)	build one LP per unit; read the envelopment form (peers and targets); tell weak from strong efficiency
Lab 3	Newsvendor *or* CVaR (Chs. 7–8)	formulate a scenario LP; read the optimal quantile; measure the cost of protection
Lab 4	Project of your choice	EV charging, arbitrage or robust and quantile regression; managerial presentation of the results

Chapters 7–10 can replace lab sessions 3–4 in an edition oriented towards uncertainty and parameter estimation, or become in-depth material for individual projects.

11.2 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 regression, 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 5).
6. Optimizing a single objective when the problem has two (pure minimax with no cost).
7. Choosing the hyperparameters (τ, λ, the budget t) 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.

11.3 Course materials

lecture notes	these lecture notes, as a PDF on the "Downloads" page of the site
`python/`	one script per chapter; `run_all.py` regenerates data and figures
`data/`	all the case study data in CSV format
`notebooks/`	one Colab notebook per chapter, to open with nothing to install
`GUROBI_GUIDE.md`	standalone guide to the solver
slides	one slide deck per chapter, as PDFs on the "Downloads" page

Reproducibility: the two commands that rebuild data, tables and figures.

```
python3 -m pip install gurobipy matplotlib pandas scipy
python3 python/run_all.py
```

Of these, `scipy` is needed only for the statistical functions, not for the optimization.