UCLouvain
Institute for Information
and Communication Technologies,
Electronics and Applied Mathematics

Nonconvex and nonsmooth economic dispatch

Loïc Van Hoorebeeck

Thesis submitted in partial fulfillment
of the requirements for the degree of
Docteur en sciences de l'ingénieur

Dissertation committee:

Prof. Pierre-Antoine Absil (UCLouvain, advisor)
Prof. Anthony Papavasiliou (UCLouvain, advisor)
Prof. Philippe Lefèvre (UCLouvain, chairman)
Prof. François Glineur (UCLouvain)
Prof. Anastasios Bakirtzis (Aristotle University of Thessaloniki, Greece)
Prof. Daniel Molzahn (Georgia Institute of Technology, GA)

Version of May 16, 2022.

Abstract

LARGE gas power plants are—and will be in the years to come—a major element of power system operations. On one hand, for the flexibility they offer, which is most valuable for the massive integration of renewable energy sources in the energy mix and the associated uncertainty linked to the production of such sources. On the other hand, they can serve as a replacement for nuclear power plants, following the policy of some European countries like Germany and Belgium.

These large gas power plants often obey the valve point effect. This physical effect is due to the increase of throttling losses when operating a unit off a valve point, that is, just after opening one of the several fuel admission valves. We show how the consideration of this physical effect makes the dispatch a nonconvex and nonsmooth optimization problem and propose algorithms that aim at efficiently finding a solution as close as possible to the global optimum, along with some guarantees.

In the first part of the thesis, we present a three-step algorithm. The first step solves a relaxation of the problem to obtain an infeasible solution that we expect to be close to the feasible set, along with a lower bound to the global optimum. Then, the second step projects this infeasible solution onto the feasible set. Finally, the last step amounts to locally improving the feasible solution that is obtained in the second step *via* a Riemannian subgradient descent scheme.

In the second part of the thesis, we further analyze the second step. The problem is posed abstractly as the projection of a given point onto the intersection of a quadratic hypersurface—or quadric—and a box. We show how to compute the exact projection onto a central non-cylindrical quadric and use splitting methods for the projection onto the full set.

Acknowledgements

In this unexpected adventure that is called a thesis, I have met, discovered, and journeyed with great people to whom I am sincerely grateful.

For their thorough guidance throughout the whole trip, I would like to first thank my two advisors: Pierre-Antoine, the wizard with whom mathematical problems suddenly disappear, and Anthony for his scientific expertise and advice on how best to present scientific contributions. I am grateful as well to my thesis committee for their careful review.

I also express my gratitude to the crew of the ship that we (or is it just me?) call Œuler; some of them are experts in crosswords, others prefer to pound the pavement, lots like to play board games, and they all made this crossing a pleasant one.

Standing out of the crowd, I would like to thank the two teammates that enrolled alongside me: Cécile and Charles. I am also amazed by Emilie R. and Adrien S., who apparently manage to listen to me blather on and on, and thankful to Étienne for always encouraging me.

Because the port is as important as the ship, I want to say how grateful I am to my friends and to my roommates. Eli, Strong, Hélène, Raf, Hub, Mouss: you make me call home the place where I live.

At some port of call, I got to know a fantastic human being: Emilie, thank you for being a wonderful partner with unfailing support.

During the storm as when the sun shines, their faith in me never declines; Final call is to my family, I owe you so much undoubtedly.

This work was supported by the Fonds de la Recherche Scientifique - FNRS under Grant no. PDR T.0025.18.

Contents

II Projection onto quadrics

III Conclusions

List of Figures

List of Tables

List of Abbreviations

ACOPF	**A**lternating **C**urrent **O**ptimal **P**ower **F**low
ADMM	**A**lternating **D**irection **M**ethod of **M**ultipliers
APC	**A**lternating **P**rojection method with the **C**entre-based quasi-projection
APE	**A**lternating **P**rojection method with the **E**xact projection
APG	**A**lternating **P**rojection method with the **G**radient-based quasi-projection
APLA	**A**daptive **P**iecewise **L**inear **A**pproximation
APQUA	**A**daptive **P**iecewise **Q**uadratic **U**nder - **A**pproximation
CCGT	**C**ombined **C**ycle **G**as **T**urbines
DCOPF	**D**irect **C**urrent **O**ptimal **P**ower **F**low
DED-DCOPF	**D**ynamic **E**conomic **D**ispatch with **DC O**ptimal **P**ower **F**low
DED-R	**D**ynamic **E**conomic **D**ispatch with **R**eserves
DR	**D**ouglas - **R**achford
DR-F	**D**ouglas - **R**achford for **F**easibility problems
ED	**E**conomic **D**ispatch

FE	**F**unction **E**valuation
KKT	**K**arush **K**uhn **T**ucker
LICQ	**L**inear **I**ndependence **C**onstraint **Q**ualification
MILP	**M**ixed - **I**nteger **L**inear **P**rogramming
MINLP	**M**ixed - **I**nteger **N**on **L**inear **P**rogramming
MIP	**M**ixed - **I**nteger **P**rogramming
MIQP	**M**ixed - **I**nteger **Q**uadratic **P**rogramming
MISOCP	**M**ixed - **I**nteger **S**econd **O**rder **C**one **P**rogramming
NLP	**N**on **L**inear **P**rogramming
OA	**O**uter - **A**pproximation
POZ	**P**rohibited **O**perating **Z**one
QCLP	**Q**uadratically **C**onstrained **L**inear **P**rogram
QCQP	**Q**uadratically **C**onstrained **Q**uadratic **P**rogram
QP	**Q**uadratic **P**rogramming
RSG	**R**iemannian **S**ub**G**radient scheme
SOS	**S**pecial **O**rder **S**et

List of Symbols

$\boldsymbol{b}$ — linear coefficients of Ψ (p. 133)

c — independent term of Ψ (p. 133)

$\operatorname{co}\{\cdot\}$ — convex hull (p. 116)

$\boldsymbol{d}$ — quadric center (p. 106)

$d_{\mathcal{B}}$ — distance function to the set $\mathcal{B}$ (p. 171)

deviation — system deviation (p. 121)

$\dim(\cdot)$ — dimension of a space (p. 111)

e_{kt} — power flow (p. 67)

$\underline{f}$ — under-approximation of f (p. 57)

f — objective function (p. 16)

$f_g^{\mathsf{Q}}, f_g^{\mathsf{V}}$ — quadratic and rectified-sine parts of the cost function (p. 17)

$\hat{f}(\cdot; \boldsymbol{X})$ — piecewise-linear interpolation of f, given the knots $\boldsymbol{X}$ (p. 20)

f^{feas} — feasibility objective (p. 97)

$\boldsymbol{g}^k$ — smooth part of the projected subgradient (p. 118)

g — index of generator unit (p. 16)

grad — projected generalized gradient (p. 116)

grad_j — projected gradient of the jth pointwise maximum (p. 116)

h, h_g — surrogate objectives (p. 21)

h_{gt} — surrogate objective (p. 70)

$\underline{h}^k$ — best (known) lower bound of the surrogate problem (p. 36)

$.^k$ — iterate index (superscript) (p. 28)

$\cdot_k$ — line index (subscript) (p. 67)

n_{iter} — last APLA iterate on the unrestricted domain (p. 73)

n_g^{knot} — number of knots (p. 20)

$\boldsymbol{n}_t$ — normal of the quadric $\mathcal{Q}_t$ at $\tilde{\boldsymbol{p}}_t^0$ (p. 99)

$\overline{\boldsymbol{p}}^0$ — best returned solution of the relaxed problem (p. 97)

$\boldsymbol{p}^*$ — optimal solution (p. 28)

$\boldsymbol{p}^{**,k}$ — optimal solution to a surrogate problem (p. 36)

$\boldsymbol{p}^{*,i}$ — sub-instance optimal solution (p. 76)

$\boldsymbol{p}, p_g$ — unit productions (p. 16)

p_{gt} — unit production (p. 64)

$\boldsymbol{p}^{i,\tilde{k}}$ — sub-instance iterate (p. 74)

$\tilde{p}_{gt}^i$ — point around which the restriction is made (p. 74)

$\tilde{\boldsymbol{p}}_t^0$ — $t-$feasible solution (p. 99)

p_t^{loss} — approximation of the power losses (p. 93)

P_t^{S} — system reserve requirement (p. 65)

$\boldsymbol{p}^k$ — APLA's iterate (p. 28)

$\text{prox}(\cdot)$ — proximal operator (p. 168)

s_{gt} — spinning upward reserve requirement (p. 65)

$\text{sign}(\cdot)$ — signum function (p. 147)

$\text{spec}(\cdot)$ — spectrum of a matrix (p. 134)

t — time step (p. 92)

$\boldsymbol{v}^k$ — line-search direction (p. 109)

$\boldsymbol{x}(\mu), x_i(\mu)$ — $\boldsymbol{x}$ in function of μ (p. 136)

$\boldsymbol{x}^*$ — optimal solution (p. 134)

x_i^0 — ith component of the point to be projected (p. 135)

$\boldsymbol{x}^0$ — point to be projected (p. 134)

$\boldsymbol{x}_k^{\mathrm{d}}$ — additional KKT point (p. 149)

A_g, B_g, C_g, D_g, E_g — coefficient parameters (p. 17)

$\boldsymbol{A}$ — quadratic coefficients of Ψ (p. 133)

$APLA(\Omega_{\tilde{\boldsymbol{p}}^i}, n_{\mathrm{iter}})$ — APLA call on the set $\Omega_{\tilde{\boldsymbol{p}}^i}$ with n_{iter} iterations (p. 74)

$\boldsymbol{B}, \boldsymbol{B}_0, B_{00}$ — loss coefficients (p. 93)

$\mathcal{B}$ — box (p. 164)

B_k — susceptance of a line (p. 66)

$\boldsymbol{C}^k$ — projected active constraints (p. 118)

$\mathcal{C}^1$ — class of continuously differentiable functions (p. 138)

$\boldsymbol{D}$ — diagonal matrix of eigenvalues (p. 135)

G — set of generator units (p. 16)

I — index set (p. 147)

$\boldsymbol{I}$ — identity operator (p. 171)

$\mathcal{I}$ — search interval (p. 137)

I^+ — index set associated with a positive eigenvalue (p. 148)

J — nonempty set of indices (p. 137)

J_g — set of piece indices (p. 21)

K — index subset (p. 147)

$\mathcal{K}$ — set of lines (p. 67)

K_k — subset of L_k with associated zero components (p. 155)

L, L_g — Lipschitz constants (p. 35)

$L_\rho(\cdot)$ — augmented Lagrangian (p. 48)

$\mathcal{LB}^k$ — lower bound on the global optimum (p. 40)

$\mathcal{L}_\mathrm{I}$ — list of integer solutions (p. 73)

L_k — subset of indices associated to the same eigenvalue (p. 150)

L_i, L_{j_i} — Lipschitz constants (p. 45)

$\mathcal{N}(\mu, \sigma)$ — normal distribution with mean μ and variance σ^2 (p. 175)

$\mathrm{N}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}$ — normal space of $\mathcal{Q}^{\mathrm{tot}}$ at point $\boldsymbol{p}$ (p. 112)

P^{D} — system load demand (p. 16)

P_t^{D} — system load demand (p. 65)

$\underline{P}_g, \overline{P}_g$ — minimum and maximum power output (p. 16)

P — quasi-projection onto $\mathcal{Q}$ (p. 159)

$\mathrm{Pr}_C(\boldsymbol{x})$ — projection onto C (p. 49)

$\mathcal{Q}^{\mathrm{tot}}, \mathcal{Q}_t$ — total quadric and quadric at time step t (p. 105)

$\underline{R}_g, \overline{R}_g$ — ramp-down and ramp-up rates (p. 93)

$\mathcal{R}$ — retraction (p. 114)

$\mathbb{R}_+^n$ — positive orthant (p. 134)

$\mathbb{R}_+^{n,*}$ — positive orthant without $\mathbf{0}$ (p. 134)

$\boldsymbol{S}^k$ — nonsmooth part of the projected subgradient (p. 118)

S_t^* — shift value for the plane relaxation (p. 99)

T — set of time steps (p. 64)

TC_k — line capacity (p. 67)

$\mathrm{T}\mathcal{Q}_t$ — tangent bundle (p. 111)

$\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}$ — tangent space of $\mathcal{Q}^{\mathrm{tot}}$ at point $\boldsymbol{p}$ (p. 112)

$\mathcal{UB}^k$ — upper bound on the global optimum (p. 40)

$\boldsymbol{X}_g$ — knot (x-axis) of the piecewise interpolation (p. 20)

$X_{g,j}$ — jth knot of the piecewise interpolation (p. 20)

X^i_{gtj} — sub-instance knots (p. 74)

$\boldsymbol{X}_{gt}$, X_{gtj} — knots (x-axis) of the piecewise interpolation (p. 70)

$\boldsymbol{X}^{\text{kink}}$, $\boldsymbol{X}^{\text{kink}}_g$ — kink points (p. 25)

$\boldsymbol{X}^{\text{knot,I}}_g$ — initial knots (p. 20)

$\boldsymbol{Y}_g$ — knot (y-axis) of the piecewise interpolation (p. 21)

α^k — line-search step size (p. 109)

$\alpha_{g,j}, \beta_{g,j}$ — coefficients of the linear pieces (p. 21)

γ — tolerance set to the MIP solver (p. 36)

γ^k — effective solver tolerance (p. 28)

$\gamma^{i,\tilde{k}}$ — sub-instance solver tolerance (p. 76)

δ^k — surrogate gap (p. 32)

$\delta^{i,\tilde{k}}$ — sub-instance surrogate gap (p. 76)

$\tilde{\delta}^k$ — effective surrogate gap (p. 28)

δ^k_g — contribution of unit g to the surrogate gap (p. 32)

ϵ^k — over-approximation error (p. 28)

$\epsilon^{i,\tilde{k}}$ — sub-instance over-approximation error (p. 76)

ζ^i — sub-instance gap (p. 76)

θ_{mt}, θ_{nt} — voltage phasor angles (p. 67)

$\boldsymbol{\lambda}, \lambda_i$ — eigenvalues (p. 134)

$\lambda_{\text{N}}, \lambda_{\text{Q}}$ — parameters of the feasibility objective (p. 97)

$\overline{\boldsymbol{\lambda}}$ — sorted vector of the unique eigenvalues (p. 150)

$\boldsymbol{\lambda}^k$ — convex combination (p. 117)

μ — Lagrange multiplier (p. 135)

μ^{I} — inflection point (p. 143)

ρ — scaling parameter (p. 48)

$\sum_{k=(\cdot,n)}$ — sum over each edge toward n (p. 67)

$\sum_{k=(n,\cdot)}$ — sum over each edge from n (p. 67)

Ψ, Ψ_t — quadratic functions (p. 105)

Ω — feasible set (p. 17)

$\Omega_{\tilde{\boldsymbol{p}}^i}$ — restriction of the feasible set around $\tilde{\boldsymbol{p}}^i$ (p. 74)

$\mathbb{1}_C$ — indicator function of C (p. 48)

$\bigsqcup$ — disjoint union (p. 42)

$(\cdot)_i$ — ith component of a vector (p. 149)

$[k] = \{1, 2, \ldots, k\}$ — index function (p. 20)

1

Introduction

> In religion we renounce the wish to give words an unequivocal meaning from the outset, while in science we start with the hope—or, if you like, the illusion—that one day it may be possible to do just that.
>
> W. Heisenberg, citing N. Bohr [HH71].

One does not need to be a medium to prophesy that energy supply will be one of the major challenges of this century. The increase in the consumption of the ever-growing earth population implies a greater energy demand, which we wish to meet while decarbonizing the energy sector.

With the recent large-scale integration of renewable energy sources in the energy mix, there is an increasing need for flexible units that can counteract the inevitable uncertainties on the supply side of power systems. Therefore, large gas units such as combined cycle gas turbines (CCGT) also become an important resource in modern power system operations, due to their ability to quickly respond to renewable supply fluctuations. The European Commission foresees a slight increase in gas-based electricity production

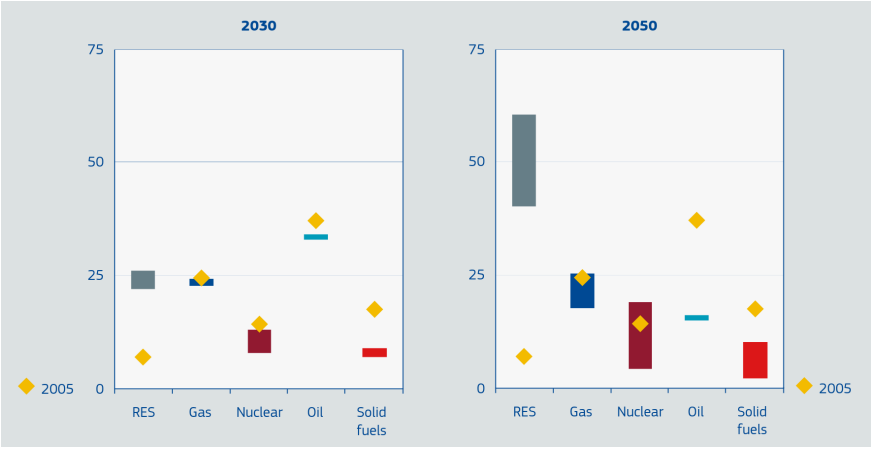


Fig. 1.1 European targets for the 2030 and 2050 energy mixes [Eur11].

for the 2030 European power mix and a stabilization around 20 % for 2050, see Fig. 1.1[1]. As stated by the commission Energy Roadmap [Eur11]:

> Natural gas will continue to play a key role in the EU's energy mix in the coming years and gas can gain importance as the *back-up* fuel for variable electricity generation.

This dependence on gas energy is also reinforced by the political decision of certain countries, *e.g.*, Belgium and Germany, to phase-out nuclear. For all these reasons, the accurate representation of the constraints and the complex cost function of such gas units is becoming increasingly important in system operations, see, *e.g.*, [Mid]. And one of the central problems of power system operations is the *economic dispatch*.

1.1 What is the economic dispatch?

The economic dispatch (ED) problem aims at the optimal scheduling of committed units to serve a given load profile at minimal cost. Two sets of constraints are considered. On the one hand, *operational constraints* ensure

[1] These predictions are regularly updated: the EU commission now foresees a 32% renewable energy share by 2030 [Eur21].

the feasibility of the dispatch and include limited power ranges, ramp rates, and prohibited operating zones. On the other hand, *balance constraints* require that the supply meets the load and guarantee that enough reserves are available.

Regarding the time of delivery, this short-term optimization problem comes after the unit commitment problem that determines—in addition to the production decisions—whether a given unit will produce at a given time; whether a given unit is *committed*. But the economic dispatch comes before the optimal power flow problem. The optimal power flow problem consists in finding "the optimal settings of a given power system network that optimize the system objective functions such as total generation cost, system loss, bus voltage deviation, emission of generating units, number of control actions, and load shedding while satisfying its power flow equations, system security, and equipment operating limits" [Zhu15].

These problems are run between once a day (for the unit commitment problem) to several times an hour (for the economic dispatch and optimal power flow problems). Various flavors of such problems exist depending on whether we consider security constraints, variable (renewable) energy productions, losses, environmental aspects, and so on.

In this thesis, we study the nonconvex and nonsmooth economic dispatch.

1.2 Why is it nonsmooth and nonconvex?

In the economic dispatch problem, only the variable part of the cost function is considered because the units are already committed. Therefore, for gas units, the cost is linked to the fuel that is being consumed for producing power. This input-output function is often modeled as a smooth convex quadratic function. However, such a function fails to accurately model large CCGT units due to the valve point effect (VPE) [DB58].

The valve point effect refers to the increase of throttling losses when operating a turbine *off* a valve point, that is, just *after* the opening of the valve. Consequently, the unit operates most efficiently when loaded *at* a valve point, that is, just *before* the next valve is open. A nonsmooth and nonconvex function, defined mathematically in (2.5), is commonly used for modeling this effect.

Losses are another source of nonconvexity of growing relevance, due

to the advent of renewable resources. Concretely, renewable resources are typically located wherever it is most geographically favorable, *e.g.*, in sites with high wind potential. These locations sometimes happen to be far from load centers. By consequence, the role of networks has become increasingly important in recent years in delivering power from remote locations to load centers [Kun13, AP17] and correspondingly losses have increased, thereby motivating the representation of such losses more accurately. Network constraints require in principle the consideration of the AC power flow equations, which are nonlinear and nonconvex. A more tractable alternative that captures an essential aspect of network operations is to focus on losses. In this context, Kron [Kro51] introduces a quadratic model for the power losses, which has been popularized by Kirchmayer [Kir58].

This nonconvexity allows for the existence of a plethora of local minima; the multimodal nature of the problem is illustrated in Fig. 2.1 for a simple 2-dimensional case. As far as the nonsmoothness of the objective function is concerned, this prevents the use of conventional derivative-based techniques.

1.3 Two main approaches for solving the economic dispatch

In order to solve this problem, the literature mostly follows two approaches: i) randomized heuristics that aim at efficiently scanning the search space to rapidly converge to a good solution and ii) deterministic methods based on approximations of the objective and the feasible set, or using logarithmic barrier functions. Instances of i) are numerous and include imperialist algorithms [MIRS13, XS18], other evolutionary algorithms [NNAA12, NAAA13, AKTH02, MA18, Bas19, LFL21], genetic algorithms [MiRSE12], and simulated annealing algorithms [WF93, PCCB06]. Examples of ii) include [PJY18, WDW+17, PJCY20] where the authors employ approximations of the objective and the feasible set without providing lower bounds, [PBN19, BSBA13] that use gradient-based algorithms with logarithmic barriers, and methods based on piecewise interpolations of the objective [YFP13, KSAA13, WDW+16, ASS18].

To be more precise, [PBN19] use a reformulation trick to get rid of the nonsmoothness of the objective (*i.e.*, they replace any absolute value with an auxiliary variable and two constraints), and then they apply a primal-dual algorithm to the unconstrained optimization problem defined with

logarithmic-barrier functions. In [BSBA13], the authors keep the original (nonsmooth) objective and design a method using subgradients. They apply this method to the (unconstrained) problem with logarithmic-barrier functions, but also propose a Riemannian subgradient descent method that does not need barrier functions. In Chapter 4, the third step of our algorithm is a direct extension of the Riemannian subgradient descent method from [BSBA13].

Similarly to what we propose in the first part of the thesis, Part I, piecewise-linear interpolation of the objective function is performed in [PJY18, WDW+17, PJCY20] (although the authors do not derive lower bounds). In the case where the interpolation error is too large, their refinement strategies amount to doubling the number of segments, thereby increasing exponentially the complexity of the problem. Our methods use instead the adaptive refinement method from [ASS18] applied to the piecewise-linear interpolation. A deeper comparison between these methods is provided in Chapter 2.

The first approach (randomized heuristics) sometimes yields faster methods, but they suffer from several problems, to wit: the lack of convergence guarantees, the non-deterministic output, the inability to exploit the structure of the problem, and the need to tune various hyperparameters.

Other deeper flaws of these metaheuristics are more fully investigated in [Sö15]: these "novel" methods are often grounded on a metaphor of a natural or artificial process. While getting some inspiration in the world around us cannot do any harm—the most fervent advocates of *biomimetics* will probably cite the invention of Velcro—it can be argued that this is not a sufficient reason to support the validity of a method. The natural behavior of bees, bats, ants, squirrels, quantum mechanics, or any hybridization of these, can be a pretext for designing a metaheuristic. Because they often introduce a brand-new vocabulary, understanding papers about a given metaheuristic is tedious for any reader stranger to this specific method. For example, the *harmony search* metaheuristic trades "harmony" for "solution", "note" (or "pitch") for "decision variable" and "sound better" for "has a better objective function value", and it is not uncommon to read such sentences in a paper on the harmony search [Sö15]:

> The harmony memory is updated whenever any of the new improvised harmonies at a given iteration sounds better (under the fitness criterion) than any of the remaining harmonies from the previous iteration.

Neither it is to read a full page of fluid mechanics in a paper of operational research. In addition to such flaws, these papers also generate much noise (because they extensively cite themselves as a matter of comparison) that yields an *up-the-wall* game. *There are no rules in this game, just a goal, which is to get higher up the wall (which translates to "obtain better results") than your opponents* [Sö15].

Finally, as pointed out in [EHBE16] for the specific case of the nonconvex economic dispatch, these metaheuristics sometimes exhibit inaccuracies in their proposed solutions. The authors of [EHBE16] divide these inaccuracies in five distinct classes. Among them, we find inaccuracies due to violating equality or inequality constraints (Class 5) and inaccuracies due to using invalidated cost functions (Class 1). We also question in Section 4.3 some papers that claim to find a solution, without providing it, of a problem that we prove to be infeasible. These inaccuracies may come from the heuristic nature of the algorithms, which also scan infeasible solutions for the purpose of finding the global solution, and by the "publish or perish" culture. Due to the up-the-wall game, this translates into "decrease-the-objective or perish" and increases scientists' bias [Fan10].

1.4 Our strategy

As these above-mentioned randomized heuristics become more and more sophisticated and computational power increases, it becomes easier to compute low-cost solutions. However, the stopping criteria of the aforementioned methods often remain basic. Indeed, without knowledge of a sufficiently good lower bound, it may be impossible to know whether the best cost found so far is within a prescribed accuracy of the globally optimal cost. Moreover, in the absence of a suitable convergence analysis, it is unknown if, given enough time, the algorithm is able to find the globally optimal cost within any prescribed accuracy.

These reasons motivate the present study that extends the methods based on piecewise approximations: on the one hand through the consideration of a piecewise-linear—instead of piecewise-quadratic—approximation and on the other hand, by adapting it to more complex dispatch problems, such as the dynamic (or multi-period) dispatch with quadratic power losses. The developed method is feasible—in the sense that all the iterates satisfy the constraints. Hence, it is possible to stop early and save computational

power. The method also returns lower bounds that rely on the solution of mixed-integer programming subproblems, defined with the piecewise-linear approximation of the objective.

Since lower and upper bounds are computed by our proposed method, it is possible to detect whether a prescribed accuracy is attained and to stop the algorithm early. Alternatively, if the difference between the upper and lower bounds does not meet the prescribed accuracy sufficiently quickly, then the user can decide to mobilize more computational power. The latter observation exploits the possibility of implementing the algorithms proposed in this work in a decentralized way. This favorable situation contrasts with most existing algorithms where a sufficiently good lower bound is unavailable and for which it is impossible to know whether the best cost found so far is close enough to the globally optimal cost.

1.5 How to read this thesis?

First, I am advising you to pour yourself a cup of coffee, tea, or any hot or cold beverage.

The common theme of this thesis is undeniably the multiple flavors of the economic dispatch in the presence of generator units that obey a valve point effect. Eight chapters split in three parts cover this rich topic. After the introduction, we explicitly deal in Part I with nonconvex and nonsmooth dispatch problems. Part II is dedicated to the projection onto quadratic hypersurfaces (or quadrics); this projection step is one of the bottlenecks of the algorithm presented at the end of Part I. Finally, the only chapter of Part III, Chapter 7, summarizes the main contributions of this thesis and provides some perspectives.

Parts I and II are relatively separate: a reader interested in the economic dispatch problem may only read Part I, and equivalently, a reader curious about the projection onto quadrics may independently read Part II.

While each chapter is written so as to be read separately, they all fit in a single framework: the method (from Chapter 4) that solves (P), the dynamic economic dispatch with quadratic power losses and valve point effect. This three-step method is denoted as APLA-RSG (standing for the combination of the adaptive piecewise-linear approximation and the Riemannian subgradient descent scheme) and is outlined in Fig. 1.2. The first step (APLA part) consists in the application of the matheuristic from Chapter 3 relying on the

APLA method from Chapter 2. The application is not direct: the feasible set must be relaxed. This relaxation is given in Chapter 4. The second step is the projection onto the feasible set, which is one of the bottlenecks for the execution time of the method. This bottleneck is at the root of Part II and in particular of Chapter 6, which is built on top of Chapter 5. The third and last step (RSG part) is given in Chapter 4.

Let us now give a short description of all chapters.

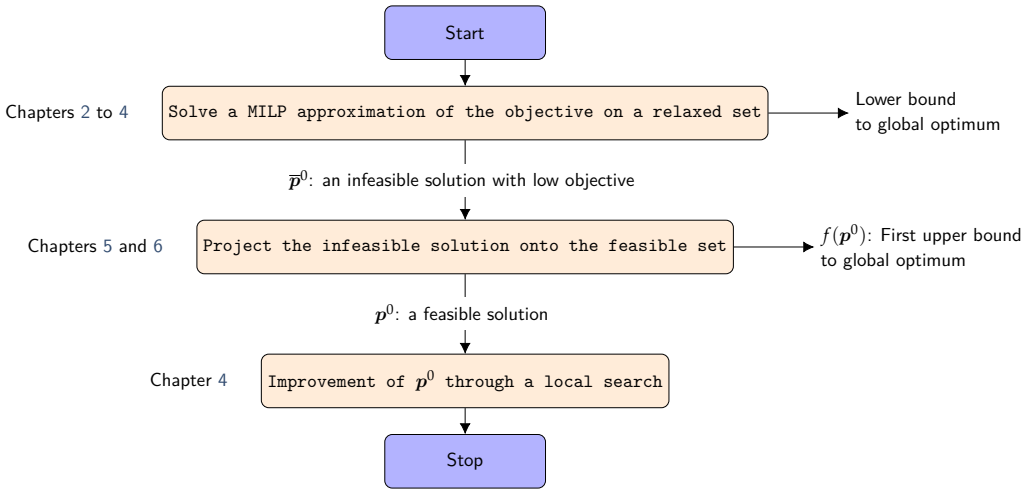


Fig. 1.2 Interplay between the chapters of this thesis.

Part I Nonconvex and nonsmooth economic dispatch

The first part of the thesis deals with nonconvex and nonsmooth economic dispatch problems as such. We design methods based on sequences of (increasing) approximations that provide sequences of (increasing) lower bounds. In this manner, the methods provide optimality guarantees.

Ch. 2 A simple economic dispatch with valve point effect

The simplest form of the economic dispatch is already a challenging problem due to the valve point effect. The valve point effect is a physical effect that occurs in large gas power plants and renders the input-output function nonsmooth and nonconvex. Because the problem is nonsmooth, most methods in the toolbox of any optimization programmer cannot be directly applied. These methods often rely on first (or higher) order information. Moreover, the nonconvex nature of the problem makes the convergence

analysis of any method difficult. Even if the method converges to a minimum, nothing prevents this critical point to be a local minimum far from the global optimum.

In the literature, most developed methods are metaheuristics that quickly scan the search space but fail to provide convergence guarantees. Here, we develop a method based on piecewise-linear approximations. The proposed method, denoted as the adaptive piecewise-linear approximation (APLA), is proven to converge up to the user-prescribed tolerance.

The original contributions of this chapter, often cited *verbatim*, are issued from the following paper.

> [VPA19] Loïc Van Hoorebeeck, Anthony Papavasiliou, and P.-A. Absil. MILP-based algorithm for the global solution of dynamic economic dispatch problems with valve-point effects. In *2019 IEEE Power Energy Society General Meeting (PESGM)*, 2019. doi:10.1109/PESGM40551.2019.8973631.

Ch. 3 A matheuristic for the dynamic economic dispatch

An extension of the method from Chapter 2 (APLA) can handle the dynamic dispatch, that is, solving an economic dispatch for several time steps. Two separate problems are studied: the dynamic dispatch with reserves (DED-R) and the dynamic dispatch with network constraints (DED-DCOPF). APLA is first adapted to solve these problems, but it suffers from a long execution time. For the purpose of reducing this execution time, a matheuristic—*i.e.*, a heuristic based on mathematical programming techniques—grounded on APLA is also presented.

The original contributions of this chapter, often cited *verbatim*, are issued from the following papers.

> [VPA19] Loïc Van Hoorebeeck, Anthony Papavasiliou, and P.-A. Absil. MILP-based algorithm for the global solution of dynamic economic dispatch problems with valve-point effects. In *2019 IEEE Power Energy Society General Meeting (PESGM)*, 2019. doi:10.1109/PESGM40551.2019.8973631.

> [VAP20a] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. Global solution of economic dispatch with valve point effects and transmission constraints. *Electric Power Systems Research*, 189:106786, 2020. doi:10.1016/j.epsr.2020.106786.

Ch. 4 Toward the consideration of quadratic power losses

The consideration of quadratic power losses adds another layer of complexity to the economic dispatch model, whose set is now described as the intersection of a polytope and a Cartesian product of quadrics. We show how to leverage the previous methods for obtaining a lower bound on the value of the global optimum, as well as a first infeasible point; this is done through a relaxation of the (now nonconvex) feasible set. This point is then projected onto the exact feasible set, yielding a first feasible iterate. The projection is done with a *black-box* solver; this will be further discussed in Part II. Finally, we show that the feasible set exhibits the structure of a manifold, and we exploit this structure by designing a Riemannian subgradient scheme to improve the first iterate. This procedure gives the three-step algorithm, outlined in Figs. 1.2 and 4.1, that summarizes most of our scientific contributions. The original contributions of this chapter, often cited *verbatim*, are issued from the following paper.

> [VAP22b] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. Solving non-convex economic dispatch with valve-point effects and losses with guaranteed accuracy. *International Journal of Electrical Power & Energy Systems*, 134:107143, January 2022. doi:10.1016/j.ijepes.2021.107143.

Part II Projection onto quadrics

Motivated by the need to quickly convert the infeasible point—obtained at the end of the first step of Fig. 1.2—into a feasible point, we analyze in the second part of this thesis the projection onto the feasible set of the dynamic economic dispatch with power losses. This set is the intersection of a polytope and the Cartesian product of quadrics. The projection onto a quadric is first studied in Chapter 5, and then it is used in the splitting methods designed to solve the full projection—the second step of Fig. 1.2—in Chapter 6. The output of this projection is the starting point for the local search in the third step of Fig. 1.2.

Ch. 5 Projection onto a quadric

A quadric is a quadratic hypersurface. Unexpectedly, the problem of the projection of a given point onto a quadric has not been studied in much detail in the literature, except for the three-dimensional case. Here, we deal with nonempty and non-cylindrical central quadrics and show how to compute this projection *via* the analysis of the Karush Kuhn Tucker (KKT) conditions. The designed algorithm is efficient because it simply consists in computing the root of a univariate scalar

function onto a definite interval. This unique root is effortlessly obtained *via* the Newton-Raphson method. However, the algorithm requires computing the eigendecomposition of some symmetric matrix, which may be expensive for large-scale problems. As a way to mitigate this issue, we also propose a heuristic based on a geometrical scheme that maps in most cases a solution onto the quadric and does not require the eigendecomposition. This heuristic is motivated by the relative closeness to the feasible set of the (infeasible) point at the end of the first step of Fig. 1.2, and the geometrical construction works particularly well for such nearly feasible points.

The original contributions of this chapter, often cited *verbatim*, are issued from the following *submitted* paper.

> [VAP22a] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. Projection onto quadratic hypersurfaces, 2022. arXiv:2204.02087.

Ch. 6 Splitting methods for the projection onto the intersection of a box and a quadric

Using the projection—or the heuristic—of the previous chapter, we use splitting methods to compute the projection onto a feasible set that is the intersection of a quadric and a hyperrectangle (also called a *box*). We study the alternating projection and the Douglas-Rachford splitting methods to solve this projection problem. Such methods need to compute the projection onto the box and onto the quadric. The former has a closed form, and the latter can be readily obtained with the method described in the previous chapter. We then extend these methods to deal with the feasible set of the problem defined in Chapter 4. The original contributions of this chapter, often cited *verbatim*, are issued from the following *submitted* paper.

> [VAP22a] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. Projection onto quadratic hypersurfaces, 2022. arXiv:2204.02087.

PART I
Nonconvex and nonsmooth economic dispatch

2

A simple economic dispatch with valve point effect

This section is devoted to the introduction of the main problem of interest of this thesis, namely the economic dispatch. We formulate in Section 2.1 the most simple version of the problem and discuss the consideration of the valve point effect—which we introduce in Section 2.2. Accounting for this effect makes the dispatch a nonconvex and nonsmooth optimization problem.

We then present in Section 2.3 a strategy to tackle this challenging optimization problem. This strategy consists in approximating the objective *via* a piecewise-linear—or piecewise-quadratic—function and successively refining the approximation in an adaptive way. This procedure gives the algorithm APLA—or its quadratic counterpart APQUA—standing for adaptive piecewise linear approximation—or adaptive piecewise-quadratic under-approximation. Doing so, we can converge to the global optimum.

Such a strategy has been introduced in [ASS18] for the static dispatch with piecewise-quadratic approximations and extended to the dynamic dispatch (see Chapter 3) and to the piecewise-linear case in [VPA19]. This explains the absence of numerical experiments with APLA in this chapter: our work starts downstream of [ASS18], where the authors already addressed the static dispatch with valve point effect. Interested readers may refer to the numerical experiments in this article.

The extension of APLA to a piecewise-smooth objective is made in Section 2.4. We also briefly discuss the use of the alternating method of multipliers (ADMM) for solving the static dispatch (Section 2.5), and we present a preprocessing method to reduce the number of integer variables used to build the piecewise approximation. This preprocessing method is called bound tightening (Section 2.6).

This chapter is based on [VPA19, ASS18] except for Section 2.2, § 2.3.3, and Sections 2.4 to 2.6.

2.1 Problem formulation

This section outlines the economic dispatch (ED) problem. This problem consists in minimizing the fuel costs of the thermal power units subject to operational constraints. The objective is defined as the sum of the individual cost functions and is therefore separable,

$$f(\boldsymbol{p}) = \sum_{g \in G} f_g(p_g), \tag{2.1}$$

where f is the total cost function (\$/h), f_g the fuel cost associated with generator g (\$/h), and $\boldsymbol{p}$ the stacked production vector of each individual generator production p_g (MW). The feasible set is restricted by the available power ranges,

$$\underline{P}_g \leq p_g \leq \overline{P}_g, \tag{2.2}$$

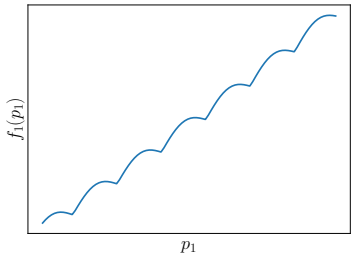
and the power balance that couples each p_g,

$$\sum_{g \in G} p_g = P^{\mathrm{D}}, \tag{2.3}$$

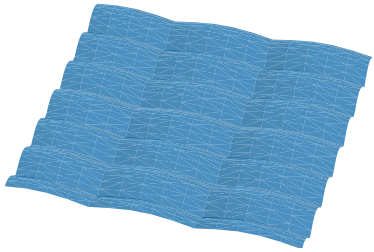
with P^{D} the load demand of the system. Using this objective and these constraints, the most simple formulation of the economic dispatch problem is given in (2.4).

Economic dispatch (ED)

$$\min_{\boldsymbol{p}} f(\boldsymbol{p}) \quad \text{subject to (2.2), (2.3)} \tag{2.4}$$



(a) Single term of (2.1).



(b) A 2-dimensional objective f as defined in (2.1). The function is depicted on the feasible set induced by (2.2).

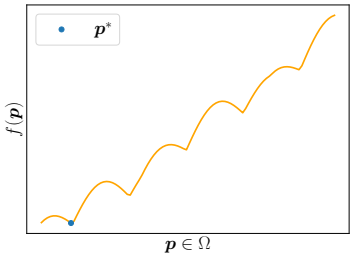
Fig. 2.1 Illustrative objective function.

From an optimization perspective, the mathematical problem defined in (2.4) seems fairly easy: the objective is separable, the number of variables is not too high—typically a few dozen to a few hundred—and, most importantly, the feasible set, which we call Ω, is *linear*. At this point, any optimizer would ask himself: *where is the catch?* The answer to this question lies in the model of the fuel cost functions that we consider throughout this thesis. Following the literature [CM06, HS11, VJ04, WS93], we model the cost function of gas units as the sum of a smooth quadratic part f_g^{Q} and a nonsmooth rectified sine f_g^{V} aimed at capturing the valve point effect (VPE):

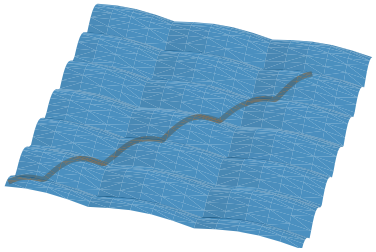
$$f_g(p_g) = \underbrace{A_g p_g^2 + B_g p_g + C_g}_{:=f_g^{\mathrm{Q}}(p_g)} + \underbrace{D_g \left|\sin E_g(p_g - \underline{P}_g)\right|}_{:=f_g^{\mathrm{V}}(p_g)}, \tag{2.5}$$

with appropriate parameters A_g, B_g, C_g, D_g, E_g (*cf.* Table A.1 for the parameter units). The impact of the VPE, namely the nonsmooth and high multimodal nature of the problem, is underlined in Figs. 2.1, 2.2 and 2.4.

This apparently trivial change makes (2.4) a challenging optimization problem, even for a relative small number of generating units. An illustrative 2D example is given in Fig. 2.2. Similarly to Fig. 2.1b, the objective is plotted on the set defined by (2.2) in Fig. 2.2b (blue surface), but here we also represent an illustrative balance constraint (2.3) (dark orange line). We also depict the restriction of f to the feasible set Ω in Fig. 2.2a.



(a) Restriction of the objective (2.1) to the feasible set of (2.4).



(b) Objective (blue surface) and feasible set (dark orange line) of (2.4).

Fig. 2.2 Restriction of the objective function to the feasible set Ω.

2.2 Valve point effect

Let us describe in this section the physics behind the valve point effect.

The steam admission to most turbines is performed with a set of valves that control the steam entering the turbine, each valve delivering to a different nozzle group. Several valves, instead of a single larger one, are used to achieve higher efficiency and reduce costs. We explain the reason for this higher efficiency below.

Admission valves are designed to operate at a given maximum output, and using a valve at another operating point implies nonzero throttling losses. Thus, while using a single large valve, the total steam flow would be throttled with correspondingly large throttling losses [HIR62]. Using a collection of smaller valves reduces this issue because each control valve is connected to a different segment of the governing stage nozzle arc (*e.g.*, in Fig. 2.3 the third valve (V_3) governs half of the nozzle arc, that is, the third nozzle group (N_3)). Hence , the major part of the total steam flow passes through fully open valves, and the throttling only occurs in the valves that partially open. Ideally, the turbine would be fed by an infinite number of valves, such that no throttling would result.

The valve point effect refers therefore to the loss of efficiency when operating a turbine off a valve point, that is, just after the previous valve opens. This loss in efficiency is depicted in Fig. 2.4. The valve point basis, modeled as f_g^{Q} (dashed line), is the ideal case of an infinite number of valves. The valve loop basis is the full cost f_g (solid line), which takes the VPE into

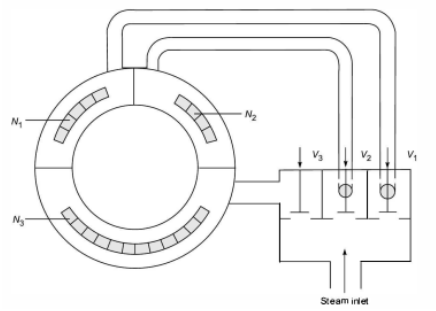


Fig. 2.3 Schematic of the governing stage of a multivalve turbine, figure from [Rat10].

account. Finally, the difference between both is the efficiency loss, modeled as f_g^{V} (dotted line). We represent the marginal cost *off* a valve point, *e.g.*, just *after* the opening of the second valve (black arrow) and *at* a valve point, *e.g.*, just *before* the third valve opens (red arrow). It is economically more interesting to be loaded at a valve point than off a valve point. This sudden change makes the derivative noncontinuous, which explains why we cannot directly rely on the largely studied class of first (or higher) order methods.

In the following section, we detail a first method to tackle (2.4), *i.e.*, an optimization problem with a separable nonconvex and nonsmooth objective defined on a polytope.

2.3 Tackling the valve point effect with piecewise interpolations

This section is devoted to the characterization of an algorithm for the solution to the economic dispatch (2.4). The method consists in solving a sequence of piecewise approximations. These approximations define the surrogate problems, which are described in Fig. 2.6a and handled by a mixed-integer programming (MIP) solver. Let us now define the surrogate problem.

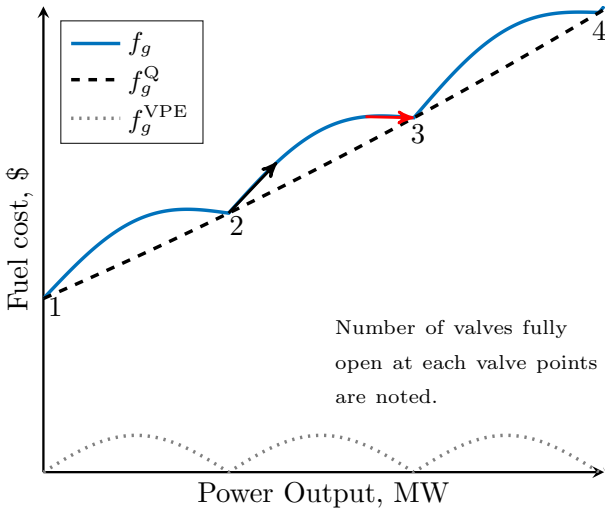


Fig. 2.4 Illustrative example of the efficiency losses due to the valve point effect. The marginal cost off the second valve point and at the third one are depicted as arrows.

2.3.1 Surrogate problem

Because the objective is a separable sum, we can consider each term independently. Let $g \in G$ and $\boldsymbol{X}_g$ be a set of n_g^{knot} points $X_{g1} < X_{g2} < \cdots < X_{gn_g^{\mathrm{knot}}}$, called *knots*, from which we construct a piecewise-linear approximation h_g of f_g. We denote as $\hat{f}_g(\cdot; \boldsymbol{X}_g)$ the piecewise interpolation of f_g given the knots $\boldsymbol{X}_g$. The original set of knots of unit g is built as the union of two subsets. The first subset is the set of kink points, which are the points where the cost function is nonsmooth, namely the set of valve points. The second subset is the set of local maxima of the rectified sine. Hence for every unit g, the set of initial knots is equal to

$$\boldsymbol{X}_g^{\mathrm{knot,I}} = \left\{ \underline{P}_g + \frac{(j-1)\pi}{2E_g} \,\middle|\, j \in [n_g^{\mathrm{knot}}] \right\}, \tag{2.6}$$

with $n_g^{\mathrm{knot}} = 1 + \lceil (\overline{P}_g - \underline{P}_g) 2E_g / \pi \rceil$ and $\lceil \cdot \rceil$ the ceiling function. We comment in § 2.3.4 on the choice of these points as the initial set of knots.

We construct the surrogate objective h_g through a binary formulation,

$$h_g(p_g) := \begin{cases} \sum_{j=1}^{n_g^{\text{knot}}-1} \alpha_{g,j}\xi_{g,j} + \eta_{g,j}\beta_{g,j}, \\ \text{with } \sum_{j=1}^{n_g^{\text{knot}}-1} \xi_{g,j} = p_g, \\ \quad \sum_{j=1}^{n_g^{\text{knot}}-1} \eta_{g,j} = 1\,, \quad \eta_{g,j} \in \{0,1\}\,, \\ \quad X_{g,j}\,\eta_{g,j} \le \xi_{g,j} \le X_{g,j+1}\,\eta_{g,j}, \end{cases} \tag{2.7}$$

where $\alpha_{g,j}$ and $\beta_{g,j}$ are the slope and vertical intercept of the linear pieces. If we denote by $\boldsymbol{Y}_g$ the image of the knots on f, we have

$$\alpha_{g,j} = \frac{Y_{g,j+1} - Y_{g,j}}{X_{g,j+1} - X_{g,j}}, \tag{2.8}$$

and

$$\beta_{g,j} = \frac{X_{g,j+1}Y_{g,j} - X_{g,j}Y_{g,j+1}}{X_{g,j+1} - X_{g,j}}. \tag{2.9}$$

The binary variables $\boldsymbol{\eta}$ act as switches that select the different pieces and corresponding continuous variables $\boldsymbol{\xi}$, see Fig. 2.5 for a visual interpretation of this selection. Let $J_g = [n_g^{\text{knot}} - 1]$ be the set of piece indices. Following (2.1), we also define $h(\boldsymbol{p}) := \sum_{g\in G} h_g(p_g)$. The surrogate problem is defined in (2.10) and corresponds to the minimization of the surrogate objective h subject to being in the feasible set Ω of (2.4).

Surrogate problem (binary form)

$$\begin{aligned} \min_{\boldsymbol{\eta},\boldsymbol{\xi},\boldsymbol{p}} \quad & \sum_{g\in G, j=1}^{n_g^{\text{knot}}-1} \alpha_{g,j}\xi_{g,j} + \beta_{g,j} \\ \text{subject to} \quad & \sum_{g\in G} p_g = P^{\text{D}} \\ & \sum_{j\in J_g} \xi_{g,j} = p_g && g \in G \\ & \sum_{j\in J_g} \eta_{g,j} = 1, && g \in G \\ & \eta_{g,j} \in \{0,1\} && g \in G, j \in J_g \\ & X_{g,j}\eta_{g,j} \le \xi_{g,j} \le X_{g,j+1}\eta_{g,j} && g \in G, j \in J_g \end{aligned} \tag{2.10}$$

This model contains $|G| + \sum_{g \in G}(n_g^{\text{knot}} - 1)$ continuous variables and $\sum_{g \in G}(n_g^{\text{knot}} - 1)$ binary variables. Remark that we can easily remove the variables p_g *via* substitution and fall down to $\sum_{g \in G}(n_g^{\text{knot}} - 1)$ continuous variables. The range constraints (2.2) are not explicitly added to the model because the lower and upper bounds are included in the knots.

The more knots we have, the better is the approximation h_g of f_g—and also h of f. However, the underlying computational complexity is directly linked to the number of binary variables. Such a comment justifies the knot update that we describe in the following subsection: an *adaptive* update.

In our context, the surrogate problem is uniquely defined by the knots $\boldsymbol{X}$, which are indexed in Algorithm 1 by k as $\boldsymbol{X}^k$. Hence, using an abuse of notation, we write $(2.10)^k$ for the surrogate problem with objective h^k that is defined *via* the knots $\boldsymbol{X}^k$. We partition these knots as

$$\boldsymbol{X}^k = \begin{pmatrix} \boldsymbol{X}_1^k \\ \boldsymbol{X}_2^k \\ \vdots \\ \boldsymbol{X}_{|G|}^k \end{pmatrix}.$$

Other models for h_g The model of h_g as presented in (2.7) and Fig. 2.5 is the easiest to explain, but other models are more efficient *in the sense that the associated MIP problem will be easier to solve by a given MIP solver*. For example, h_g can also be modeled using special ordered sets (SOS), *e.g.*, using SOS1:

$$h_g^{\text{SOS1}}(p_g) := \begin{cases} \sum\limits_{j \in J_g} \alpha_{g,j}\xi_{g,j} + \eta_{g,j}\beta_{g,j}, \\ \text{with } \sum\limits_{j \in J_g} \xi_{g,j} = p_g, \\ \quad X_{g,j}\,\eta_{g,j} \le \xi_{g,j} \le X_{g,j+1}\,\eta_{g,j}, \\ \quad \sum\limits_{j \in J_g} \eta_{g,j} = 1, \\ \quad \eta_{g,j} \text{ are SOS1.} \end{cases} \tag{2.11}$$

The SOS1 variables are a set of variables where at most *one* variable can take a nonzero value. They act as switches that select the pieces, likewise the binary variables from (2.7).

Similarly, we can use SOS2 variables:

$$h_g^{\text{SOS2}}(p_g) := \begin{cases} \sum_{j=1}^{n_g^{\text{knot}}} \lambda_j f_g(X_j), \\ \text{with } \sum_{j=1}^{n_g^{\text{knot}}} \lambda_j X_j = p_g, \\ \sum_{j=1}^{n_g^{\text{knot}}} \lambda_j = 1, \\ \lambda_j \text{ are SOS2.} \end{cases} \tag{2.12}$$

The SOS2 variables are an ordered set of nonnegative variables where at most *two* can be positive, and if two are positive they must be consecutive. Intuitively, this formulation selects the sth segment with $\lambda_s + \lambda_{s+1} = 1$ and interpolates between the points delimiting the segment through the convex combination of these two points.

Some MIP solvers (*e.g.*, Gurobi [Gur18]) allow the user to encode any piecewise-linear function by providing the software with the list of knots $\{x, f_g(x)\}_{x \in \boldsymbol{X}_g}$. The solver then internally creates the model of h_g. Such formulation changes are highly important in terms of execution time: while the represented function h_g is rigorously equivalent, an adequate formulation allows the MIP solver to branch faster on the variables. For the sake of simplicity, we stick to the binary formulations in the remainder of this thesis. However, in the numerical experiments presented in Sections 3.3 and 4.3, we use the internal model from Gurobi; it appears to be the fastest for our problems—as the branching strategy is optimized.

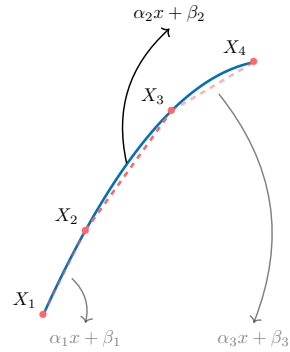
2.3.2 Knot update mechanism and algorithm statement

Assume that a solution to the surrogate problem $(2.10)^k$ has been obtained, and let us neglect the over-approximation error (defined in § 2.3.3). In this case, the surrogate problem is an under-approximation, and each feasible solution is a *lower bound* to the global optimum of (2.4).

If the gap between the true and surrogate objective function evaluated at this point is zero, it means that the optimal objective of the surrogate problem $(2.10)^k$ of our main problem (2.4) has the same value as an objective of the latter problem. In other words, the lower bound provided by this under-approximation—or *relaxation*, see remark 2.1 for our definition of

$$h_g(x) = \begin{cases} \alpha_1\xi_1 + 0\beta_1 + \alpha_2\xi_2 + 1\beta_2 \\ +\alpha_3\xi_3 + 0\beta_3, \\ \text{with } \xi_1 + \xi_2 + \xi_3 = x, \\ \qquad 0 + 1 + 0 = 1, \\ \qquad X_1\,0 \le \xi_1 \le X_2\,0, \\ \qquad X_2\,1 \le \xi_2 \le X_3\,1, \\ \qquad X_3\,0 \le \xi_3 \le X_4\,0. \end{cases}$$

(a) Mathematical interpretation.



(b) Geometric interpretation.

Fig. 2.5 Interpretation of the binary variables in (2.7) as switches for a 4-knot problem: if $x \in [X_2, X_3]$, then $b_2 = 1$ and the second segment is selected. We have successively $b_1 = b_3 = 0$, $\xi_1 = \xi_3 = 0$ and finally $\xi_2 = x$. For $x \in [X_2, X_3]$, we effectively have $h_g(x) = \alpha_2 x + \beta_2$. Here, the index g has sometimes been omitted to lighten the notation.

relaxation—matches the upper bound obtained by evaluating this solution at the true objective. If such a situation happens, *we have won*: we found a feasible solution whose objective is equal to our lower bound, that is, *one* of the optimal solutions to (2.4).

On the other hand, if the gap between both objectives is too large, an increase in the number of knots should be contemplated. Indeed, increasing the number of knots will enhance the approximation, and the same is true for the surrogate solution. In [PJY18], the adopted approach is to increase the knot sampling over the entire allowable range (*i.e.*, by doubling the number of knots). However, this *exponentially* increases the number of knots and the number of binary variables. In this work, we follow the knot update mechanism from [ASS18], *i.e.*, the previous surrogate solution is added to the knot list. As a consequence, in the new iteration, the surrogate solution differs from the old one and convergence is guaranteed, see Theorem 2.2. The proposed APLA algorithm (Algorithm 1) is substantially similar to [ASS18, Algorithm 1].

Figure 2.6 illustrates Algorithm 1: Fig. 2.6a summarizes the algorithm in a flowchart and Fig. 2.6b depicts the main and surrogate objective for a given unit g. The initial objective f_g is plotted in solid line and the smooth

part f_g^{Q} in dash-dotted line. The points where both curves meet are the points at which the sine from the nonsmooth part f_g^{V} vanishes. At these points, the initial objective is not smooth due to the absolute value in (2.5). We refer to these (valve) points as *kink points*. As detailed in Algorithm 1, the approach considered here consists in successively adding knots to the piecewise approximation to refine it. The knots $\boldsymbol{X}_g^k$ define the piecewise approximation h_g^k; therefore, they also define the surrogate problem $(2.10)^k$ at step k of the algorithm. These knots are depicted as bullets and include the kink points $\boldsymbol{X}_g^{\mathrm{kink}}$. The color difference illustrates when the points were added to the set of knots: the red ones belong to the initial set of knots, the green have been added during previous iterations and the purple is the knot to be added (which may overlap one of the current knots).

Let us now study the theoretical guarantees of this algorithm.

REMARK 2.1 (On the meaning of relaxation). A relaxation of a given optimization problem

$$\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x})$$

is, *sensu stricto*, another optimization problem

$$\min_{\boldsymbol{x}\in\mathcal{Y}} h(\boldsymbol{x})$$

such that

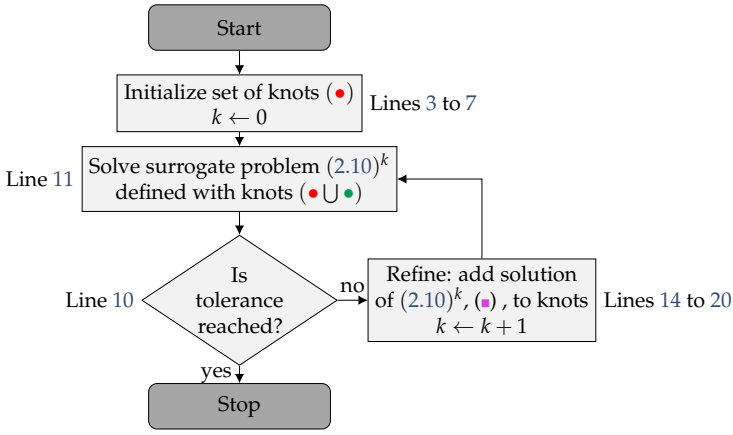
1. $h(\boldsymbol{x}) \leq f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{X}$: the relaxed objective is an under-approximation;
2. $\mathcal{X} \subseteq \mathcal{Y}$: the relaxed feasible set includes the initial feasible set.

It is clear that we have $\min_{x\in\mathcal{Y}} h(\boldsymbol{x}) \leq \min_{x\in\mathcal{X}} f(\boldsymbol{x})$: the optimal solution to the relaxation is a lower bound on the original problem. In this thesis, to distinguish these two conditions, we refer to the second condition as *relaxation*, and the first condition is denoted as *under-approximation*. Therefore, each surrogate problem (2.10) with a piecewise-quadratic interpolation (*cf.* § 2.3.4) is an under-approximation of the true problem (2.4) but not a relaxation: both feasible sets are equal and the objective of the former is an under-approximation. ■

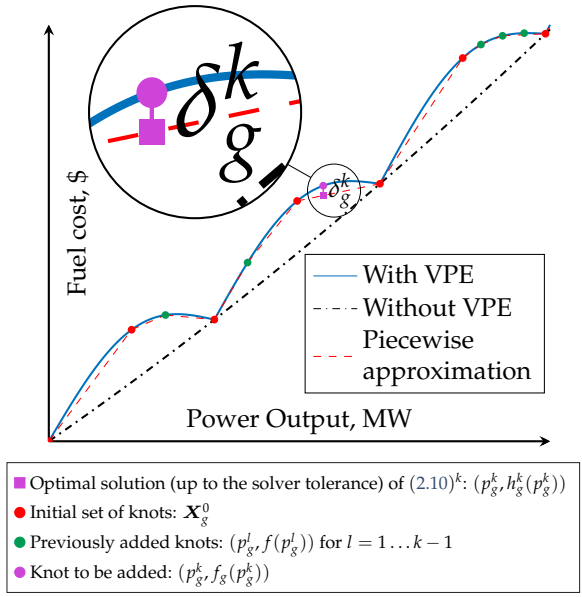
Algorithm 1 APLA: Adaptive piecewise-linear approximation

Require: Optimization problem such as (2.4)

set tolerance parameter δ_{tol}
set tolerance of the MILP solver γ
for $g \in G$ **do**
    choose set of knots ($\boldsymbol{X}_g$) including the kink points
    $\boldsymbol{Y}_g \leftarrow f_g(\boldsymbol{X}_g)$
    $n_g^{\text{knot}} \leftarrow \texttt{size}(\boldsymbol{X}_g)$
end for
$k \leftarrow 0$
$\tilde{\delta}^k \leftarrow \delta_{\text{tol}} + 1$
while $\tilde{\delta}^k > \delta_{\text{tol}}$ **do**
    $\boldsymbol{p}^k \leftarrow$ optimal solution to MILP surrogate problem $(2.10)^k$, obtained with MILP solver with tolerance γ
    $\delta^k \leftarrow f(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^k)$
    $\tilde{\delta}^k \leftarrow \min_{l \in [k]} f(\boldsymbol{p}^l) - h^k(\boldsymbol{p}^k)$
    for $g \in G$ **do**
        if $\min_{j \in [n_g^{\text{knot}}]} \left| p_g^k - X_{g,j} \right| > 0$ **then**
            $\boldsymbol{X}_g \leftarrow \text{insort}(\boldsymbol{X}_g, p_g^k)$ ▷ Ordered insertion
            $\boldsymbol{Y}_g \leftarrow \text{insert}(\boldsymbol{Y}_g, f_g(p_g^k))$ ▷ Insert at same index as previous line
            $n_g^{\text{knot}} \leftarrow n_g^{\text{knot}} + 1$
        end if
    end for
    $k \leftarrow k + 1$
end while
return $\arg\min_{l=0,\ldots,k-1} f(\boldsymbol{p}^l)$



(a) Flow chart of APLA (lines reference to Algorithm 1).



(b) Illustration of the method APLA at iteration k on a single term of the objective.

Fig. 2.6 Adaptive piecewise-linear approximation (APLA) method.

2.3.3 Bounds to the optimal solution

At step k of Algorithm 1, the objective function evaluated at the optimal solution $\boldsymbol{p}^*$ can be bounded as follows:

$$\min_{l\in[k]} f(\boldsymbol{p}^l) - \tilde{\delta}^k - \gamma^k - \epsilon^k \leq f(\boldsymbol{p}^*) \leq \min_{l\in[k]} f(\boldsymbol{p}^l), \tag{2.13}$$

where $\tilde{\delta}^k$ is the *effective surrogate gap* and is obtained as the difference between the best-known objective and the surrogate objective at iterate $\boldsymbol{p}^k$ as computed in Line 13 of Algorithm 1. The value γ^k is the *effective solver tolerance* and ϵ^k represents the over-approximation error. Let us explain more precisely each of these bounds, their evolution as k increases, and then the proof of (2.13).

The over-approximation error (ϵ^k) Algorithm 1 is based on a sequence of piecewise-linear approximations. In contrast with the APQUA method from [ASS18], the approximation is not guaranteed to be an under-approximation. One can see that the approximation is an under-approximation if the true cost function is concave on each segment delimited by the initial knots (for more details, see Proposition 2.3). However, this concavity assumption is not satisfied in our (piecewise-linear) case because the curvature of the rectified sine vanishes at the kink points (see bottom right magnification in Fig. 2.10). To be more specific, the function f_g is convex on

$$\begin{aligned}\mathcal{R}_g^{\text{convex}} := \Bigg\{ p_g \in [\underline{P}_g, \overline{P}_g] \;\Bigg|\; &\text{there is some } X_g \in \boldsymbol{X}_g^{\text{kink}} \\ &\text{with } |p_g - X_g| \leq \frac{1}{E_g} \arcsin\left(\frac{2A_g}{D_g E_g^2}\right) \Bigg\}.\end{aligned} \tag{2.14}$$

In the examples studied in this thesis (*e.g.*, in Section 3.3), these convex parts are small: about 0.5% of the whole domain. The maximal over-approximation error ϵ^k is computed as

$$\begin{aligned}\epsilon^k &:= \max_{\boldsymbol{p}} (h^k(\boldsymbol{p}) - f(\boldsymbol{p})) \\ &= \sum_{g\in G} \underbrace{\max_{p_g\in[\underline{P}_g,\overline{P}_g]} (h_g^k(p_g) - f_g(p_g))}_{:=\epsilon_g^k}.\end{aligned} \tag{2.15}$$

Note that this last equation can be easily calculated at every iteration since it can be viewed as $|G| \cdot n^{\text{knot}}$ decoupled optimization problems of a single variable. Besides, as we never remove points, $(\epsilon^k)_{k\in\mathrm{N}}$ can be bounded above; we obtain for all $k = 0, 1, 2, \ldots$

$$\begin{aligned}\epsilon^k \leq \epsilon^{\max} &:= \max_{h\in\mathcal{H}_f(\boldsymbol{X}^{\text{knot,I}})} \max_{\boldsymbol{p}\in[\underline{\boldsymbol{P}},\overline{\boldsymbol{P}}]} (h(\boldsymbol{p}) - f(\boldsymbol{p})) \\ &= \sum_{g\in G} \max_{h_g\in\mathcal{H}_{f_g}(\boldsymbol{X}_g^{\text{knot,I}})} \max_{p_g\in[\underline{P}_g,\overline{P}_g]} h_g(p_g) - f_g(p_g). \end{aligned} \tag{2.16}$$

where $\mathcal{H}_f(\boldsymbol{X}^{\text{knot,I}})$ is the set of piecewise-linear functions interpolating f that contains $\boldsymbol{X}^{\text{knot,I}}$ in its knots. In other words, $\mathcal{H}_f(\boldsymbol{X}^{\text{knot,I}})$ is the set of functions to which Algorithm 1 has access to approximate f.

Proposition 2.1 states that the maximal error is attained on a piece adjacent to a kink point $\boldsymbol{X}^{\text{L}}$ for a knot $\boldsymbol{X}^{\text{M}}$ that either verifies

$$f_g'(X_g^{\text{M}}) = \frac{f_g(X_g^{\text{M}}) - f_g(X_g^{\text{L}})}{X_g^{\text{M}} - X_g^{\text{L}}}, \tag{2.17}$$

for all $g \in G$ with such a X_g^{M}, or X_g^{M} is one of the initial knots. In other words, the slope of the line defined by X_g^{M} and X_g^{L} is equal to the derivative of the function at X_g^{M}. This situation is depicted in Fig. 2.7 with an exaggerated convex region.

Proposition 2.1. *Let us consider the initial knots $\boldsymbol{X}^{knot,\text{I}}$ consisting in the intersection of the kink points and the maxima of the rectified sine. If the maximal over-approximation error is computed as*

$$\epsilon^{\max} := \max_{h\in\mathcal{H}_f(\boldsymbol{X}^{knot,\text{I}})} \max_{\boldsymbol{x}\in[\underline{\boldsymbol{P}},\overline{\boldsymbol{P}}]} h(\boldsymbol{x}) - f(\boldsymbol{x}),$$

with $\mathcal{H}_f(\boldsymbol{X}^{knot,\text{I}})$ defined as in (2.16). Then, there exist a function

$$h^{\max} \in \operatorname*{argmax}_{h\in\mathcal{H}_f(\boldsymbol{X}^{knot,\text{I}})} \max_{\boldsymbol{x}\in[\underline{\boldsymbol{P}},\overline{\boldsymbol{P}}]} h(\boldsymbol{x}) - f(\boldsymbol{x})$$

and a point

$$\boldsymbol{x}^* \in \operatorname*{argmax}_{\boldsymbol{x}\in[\underline{\boldsymbol{P}},\overline{\boldsymbol{P}}]} h^{\max}(\boldsymbol{x}) - f(\boldsymbol{x})$$

such that for all $g \in G$, the point x_g^* belongs to a segment of $h^{\max}$ defined with a

kink point $X_g \in \boldsymbol{X}_g^{kink}$ *and a point* X_g^M *satisfying*

$$f_g'(X_g^M) = \frac{f_g(X_g^M) - f_g(X_g)}{X_g^M - X_g}, \text{ if such } X_g^M \text{ exists,}$$
$$\text{or } X_g^M \in \boldsymbol{X}_g^{knot,\mathrm{I}}.$$

Proof (sketch). We can show that the maximum occurs for some function $h^{\max} \in \max_{h \in \mathcal{H}_f(\boldsymbol{X}^{\mathrm{knot,I}})}$ at some interval such that one of the two points defining the interval is a kink point, around which the function is convex. The intuition of the proof is the following: starting from a vertical line passing through this kink point, we lower it until it reaches the function f. For a given unit $g \in G$, this happens either at (one of the first) tangent points or alternatively—if such a tangent point does not exist—at the other end of the interval. A more complete proof is provided in Appendix B.1. □

The proof of Proposition 2.1, further discussed in Appendix B.1, is constructive *provided that we know the interval on which the maximum is attained*. Nonetheless, because we have a definite number of initial intervals, we can simply look at every interval. Therefore, the procedure to compute $\epsilon^{\max}$ is the following: for every unit g we compute $\epsilon_g^{\max}$ by searching on each interval the corresponding knot X_g^{M} that maximizes the over-approximation error. Then, as shown in Appendix B.1, the value x_g^* at which the maximum is attained can be obtained by solving (on every interval) (2.17) for X_g^{M}. Once we get $\epsilon_g^{\max}$, *i.e.*, the maximal over-approximation error on all pieces of h_g, we compute $\epsilon^{\max}$ as the sum of each $\epsilon_g^{\max}$.

For the case study investigated in § 3.3.1, we get $\epsilon^{\max} = 0.32\,\$$, which is much smaller than the other bounds (*cf.* Fig. 3.6).

Note that since $\underline{P}_g$ is a kink point for all $g \in G$, we directly have $\epsilon^k \geq 0$ for every $k \in \mathbb{N}$.

REMARK 2.2 (In general $(\epsilon^k)_{k\in\mathbb{N}}$ is not decreasing). It is not true in general that the sequence of over-approximation errors is decreasing. Indeed, let us take $G = \{1\}$, $f_g^{\mathrm{Q}} = x^2/(2\pi^2)$, $f_g^{\mathrm{V}} = \sin(x)$, and $\boldsymbol{X}^0 = \{0, \pi\}$[1]. The first surrogate function is $h^0 = x/(2\pi)$. These functions are depicted in Fig. 2.8. The first (maximal) over-approximation error ϵ^0 is 0. Indeed, let

$$e^0(x) := h^0(x) - f(x) = \frac{x}{2\pi} - \frac{x^2}{2\pi^2} - \sin(x),$$

[1] As $\sin(x)$ is positive on $[0, \pi]$, we omit the absolute value.

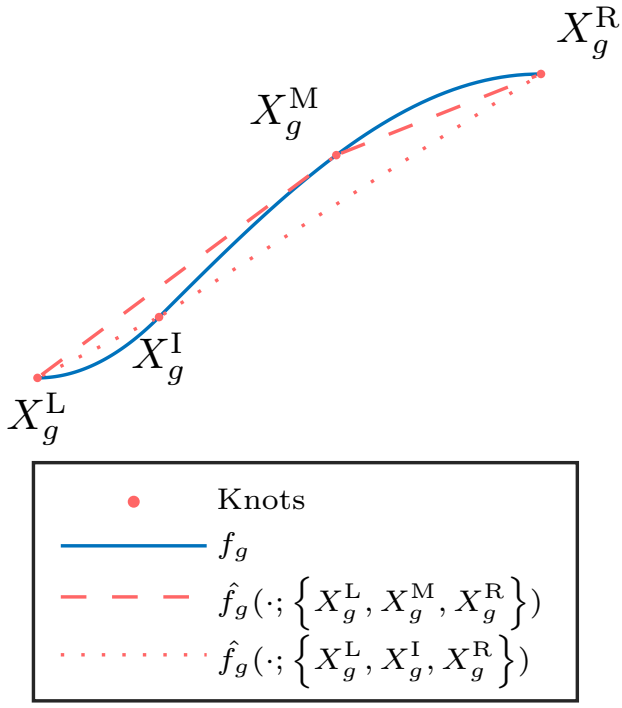


Fig. 2.7 Depiction of (2.17). The points X_g^{L} and X_g^{R} are the left and right bounds, X_g^{I} is the unique inflection point of f on $[X_g^{\mathrm{L}}, X_g^{\mathrm{R}}]$, and X_g^{M} is the knot that maximizes the over-approximation error.

we have

$$\frac{\mathrm{d}e^0}{\mathrm{d}x}(x) = \frac{1}{2\pi} - \frac{x}{\pi^2} - \cos(x),$$

whose single root on $[0, \pi]$ is $\pi/2$. Using Rolle's theorem, we show that $e^0(x)$ has at most two roots on $[0, \pi]$, and we verify that these roots are 0 and π. Finally, since

$$\frac{\mathrm{d}e^0}{\mathrm{d}x}(0) = \frac{1}{2\pi} - 1 < 0,$$

we conclude that $e^0(x)$ is nonpositive on $[0, \pi]$ and $\epsilon^0 = 0$. This nonpositive over-approximation error is depicted in Fig. 2.9 (dotted green line).

Adding the point $x = 0.1$ to the set of knots, we observe that $\epsilon^1 > 0$, see the right magnification in Fig. 2.9 (dash-dotted green line)[2]. While $(\epsilon^k)_{k\in\mathbb{N}}$ is not decreasing in general, this is not a problem in the practical cases considered here because this nonnegative sequence is upper bounded by $\epsilon^{\max}$ which is negligible in our experiments. However, one could obtain a decreasing sequence by adding the inflection points of the objective to the set of initial knots; the sequence will nonetheless not be *monotonically* decreasing and in general, $\lim_{k\to\infty} \epsilon^k = 0$ does not hold. This remark is the root of the way we deal with the convex part in Section 2.4, *i.e.*, *via* a polyhedral outer-approximation. ■

The surrogate gap and effective surrogate gap ($\delta^k, \tilde{\delta}^k$) Let us define the surrogate gap as the difference between the true and surrogate objective,

$$\delta^k := f(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^k) = \sum_{g\in G} \underbrace{f_g(p_g^k) - h_g^k(p_g^k)}_{:=\delta_g^k}, \tag{2.18}$$

and the effective surrogate gap as the difference with the best-known iterate,

$$\tilde{\delta}^k := \min_{l\in[k]} f(\boldsymbol{p}^l) - h^k(\boldsymbol{p}^k). \tag{2.19}$$

Figure 2.6b and the top right magnification in Fig. 2.10 depict δ_g^k.

[2]This choice is not innocuous. The inflection points of $e^0(x)$ are the roots of $\frac{\mathrm{d}^2 e^0(x)}{\mathrm{d}x^2}$. These roots are $\left\{2\pi n + \sin^{-1}\left(1/\pi^2\right) : n \in \mathbb{N}\right\}$ of which exactly one is in $[0, \pi]$: $x^{\mathrm{I}} := \sin^{-1}\left(1/\pi^2\right) \approx 0.1$. Hence, the function is convex on $[0, x^{\mathrm{I}}]$, and it is clear that the piecewise interpolation over-approximates f on this convex region.

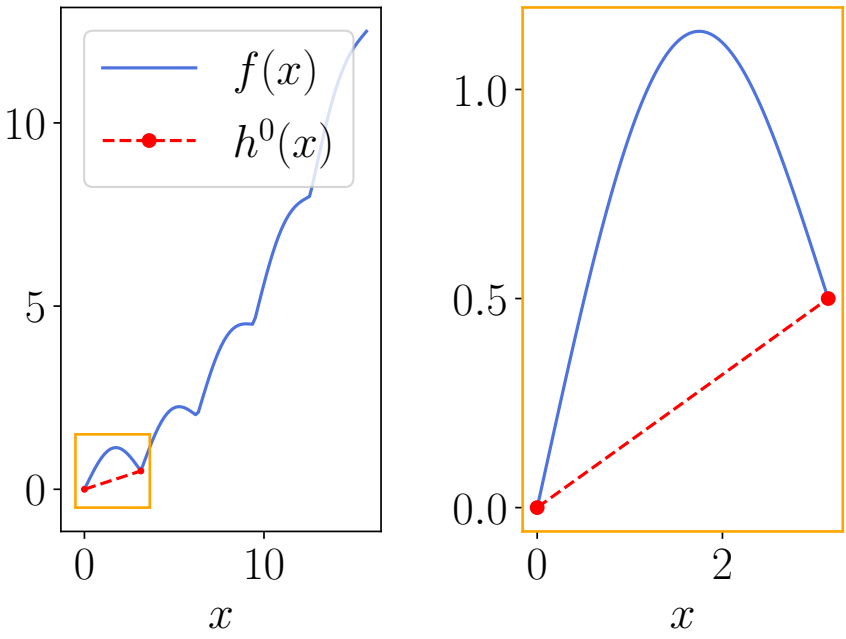


Fig. 2.8 Plot of the functions from remark 2.2, $f(x) = x^2/(2\pi^2) + |\sin(x)|$, and $h^0(x) = \hat{f}(x; \{0, \pi\}) = x/(2\pi)$. The right frame is a magnification to scale.

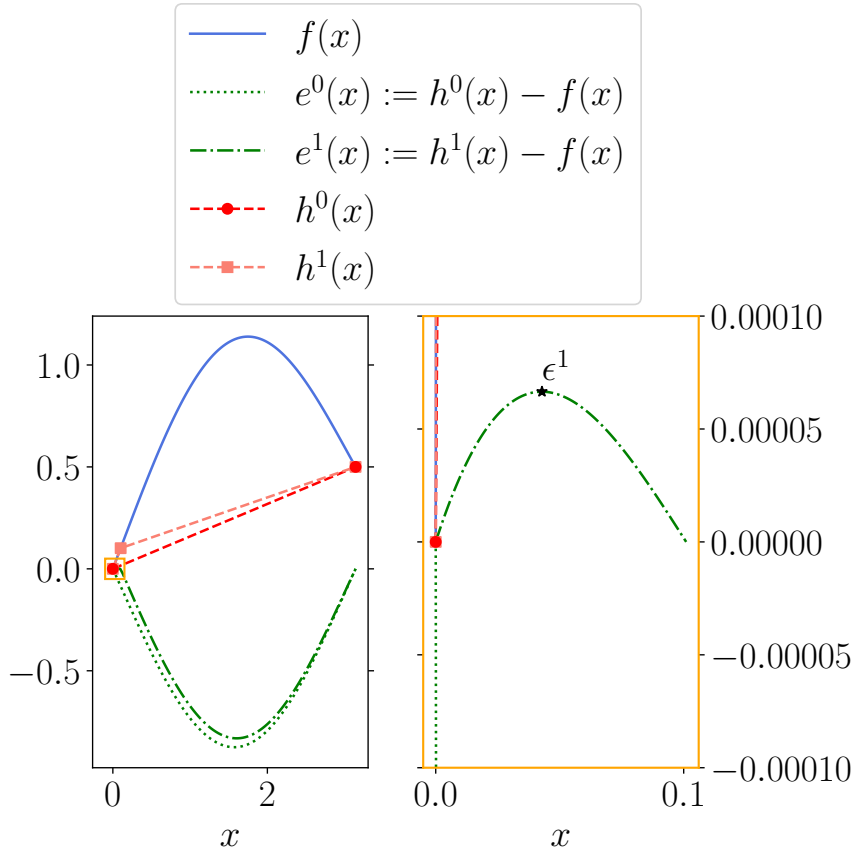


Fig. 2.9 Over-approximation errors for the functions from remark 2.2. The functions f and h^0 are defined as in Fig. 2.8 and $h^1(x) = \hat{f}(x; \{0, 0.1, \pi\})$. The right frame is a cropped magnification around $[0, 0.1]$, so as to see that $h^1(x) - f(x)$ is positive on $[0, 0.1]$ with a maximum value of $\epsilon^1 \approx 0.00007$.

Following [ASS18], we show in Theorem 2.2 that $(\delta^k)_{k\in\mathrm{N}}$, the gap between the objective and surrogate function at point $\boldsymbol{p}^k$, converges to 0, and then we use this result to prove that the limit superior of $(\tilde{\delta}^k)_{k\in\mathrm{N}}$ goes to zero as well.

Theorem 2.2. $\lim_{k\to\infty} \delta^k = 0$

Proof. We first show that f and h^k, $k = 0, 1, \ldots$ are Lipschitz continuous on the feasible set.

For each unit g and $\Delta > 0$, we have

$$\left|f_g(p_g + \Delta) - f_g(p_g)\right| \leq (2A_g\overline{P}_g + B_g + D_gE_g)\Delta := L_g\,\Delta$$

with L_g the so-called Lipschitz constant. Adding up all constants, $L := \sum_{g\in G} L_g$ is a valid Lipschitz constant for f. Since h^k_g is a continuous piecewise interpolation of f_g, it is also Lipschitz continuous and L_g (resp. L) is a valid Lipschitz constant for h^k_g (resp. h^k). Let $(\boldsymbol{p}^k)_{k\in\mathrm{N}}$ be the sequence of optimal solutions to the surrogate problem associated with function h^k, we then obtain

$$\begin{aligned}
\delta^k &= f(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^k) \\
&= f(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^{k-1}) + h^k(\boldsymbol{p}^{k-1}) - h^k(\boldsymbol{p}^k) \\
&= f(\boldsymbol{p}^k) - f(\boldsymbol{p}^{k-1}) + h^k(\boldsymbol{p}^{k-1}) - h^k(\boldsymbol{p}^k) \\
&\leq 2L\|\boldsymbol{p}^k - \boldsymbol{p}^{k-1}\|_2,
\end{aligned}$$

where the 3rd line comes from the knot updating criterion and the last line from the Lipschitz continuity.

Suppose for contradiction that $(\delta^k)_{k\in\mathrm{N}}$ does not converge to 0. Then there is $\delta^* > 0$ and an infinite subsequence $(\delta^{k_j})_{j\in\mathrm{N}}$ such that $|\delta^{k_j}| > \delta^*$ for all j. Then, for each j, we have that for all $J > j$, $\|\boldsymbol{p}^{m_J} - \boldsymbol{p}^{m_j}\|_2 \geq \delta^*/(2L)$. This last inequality implies that the subsequence $(p^{m_j})_{j\in\mathrm{N}}$ is unbounded, a contradiction with the admissible range constraints.

□

Finally, due to the definition of $\tilde{\delta}^k$, it immediately follows from Theorem 2.2 that

$$\limsup_{k\to\infty} \tilde{\delta}^k \leq 0. \tag{2.20}$$

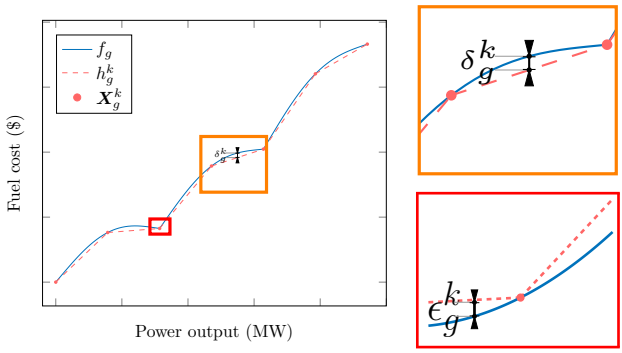


Fig. 2.10 Outline of the true f_g and surrogate h_g^k functions. The top magnification (orange frame, to scale) allows a better visualization, whereas the bottom one (red frame, not to scale) shows a tiny convex zone around a kink point where $h_g^k := h_g^{\text{MILP},k}$ is an over-approximation of f_g.

The effective solver tolerance (γ^k) Let $\boldsymbol{p}^{**,k}$ be one optimal solution to $(2.10)^k$. In general, the solution $\boldsymbol{p}^k$ of the MIP solver at Line 11 of Algorithm 1 is given up to a tolerance:

$$h^k(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^{**,k}) \leq \gamma^k := h^k(\boldsymbol{p}^k) - \underline{h}^k(\boldsymbol{p}^k), \tag{2.21}$$

where γ^k is the *effective solver tolerance*, that is, the difference between $h^k(\boldsymbol{p}^k)$ and the best-known lower bound of $(2.10)^k$, denoted as $\underline{h}^k(\boldsymbol{p}^k)$. If the time limit criterion is not attained, this effective tolerance is below the user-prescribed one:

$$\gamma^k \leq \gamma,$$

where γ is the user-prescribed tolerance fed to the MIP solver.

Hence, the sequence $\left(\gamma^k\right)_{k\in\mathrm{N}}$ is not monotonic and stays close to (and below) γ if the MIP solver does not terminate with a timeout. This phenomenon is illustrated in Fig. 3.6.

Proof of (2.13) The right-hand side of (2.13) follows from the definition of $\boldsymbol{p}^*$ as the global minimum and $\boldsymbol{p}^l$ with $l \in [k]$ as feasible solutions. For the left-hand side, let us first consider the iterate $\boldsymbol{p}^k$ instead of $\arg\min_{l\in[k]} f(\boldsymbol{p}^l)$.

We successively have

$$\begin{aligned} f(\boldsymbol{p}^k) - f(\boldsymbol{p}^*) &= f(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^k) + h^k(\boldsymbol{p}^k) - f(\boldsymbol{p}^*) \\ &= \delta^k + h^k(\boldsymbol{p}^k) - f(\boldsymbol{p}^*) \\ &\leq \delta^k + \epsilon^k + h^k(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^*) \\ &\leq \delta^k + \epsilon^k + h^k(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^{**,k}) \\ &\leq \delta^k + \epsilon^k + \gamma^k, \end{aligned} \tag{2.22}$$

where the second line comes directly from the definition of δ^k and the third line from the definition of ϵ^k. Indeed, as ϵ^k is the maximal over-approximation error (2.15), we have in particular

$$\epsilon^k = \max_{\boldsymbol{p}} (h^k(\boldsymbol{p}) - f(\boldsymbol{p})) \geq h^k(\boldsymbol{p}^*) - f(\boldsymbol{p}^*).$$

The fourth line of (2.22) follows from the definition of $\boldsymbol{p}^{**,k}$ as the global minimum of $(2.10)^k$ and the last line from the definition of γ^k as the effective solver tolerance.

Finally, to effectively recover (2.13) we use the definition of the effective surrogate gap $\tilde{\delta}^k$:

$$\min_{l \in [k]} f(\boldsymbol{p}^l) - f(\boldsymbol{p}^*) = \tilde{\delta}^k + h^k(\boldsymbol{p}^k) - f(\boldsymbol{p}^*),$$

and the rest of the proof is equivalent to (2.22).

REMARK 2.3 (What if $\delta^k < 0$). As further analyzed in remark 2.2 and § 2.3.4, there is a convex region, $\bigtimes_{g \in G} \mathcal{R}_g^{\text{convex}}$, on which f is convex. It is possible—though unlikely—that for some $\boldsymbol{p}^k$ we have $f(\boldsymbol{p}^k) - h(\boldsymbol{p}^k) = \delta^k < 0$. This case is nonetheless not an issue because from the definition of ϵ^k we immediately have

$$-\epsilon^k \leq \delta^k.$$

Therefore $\delta^k + \epsilon^k \geq 0$ and the right-hand side of (2.22) remains nonnegative. ■

2.3.4 Surrogate problem with an MIQP approximation

As seen in § 2.3.3, using a piecewise-linear approximation h_g^{MILP} to approximate h_g will not yield an under-approximation. This detail makes the

analysis of the method slightly more complicated, as we need to take into account the (small) over-approximation error ϵ.

The analysis can be simplified by considering as in [ASS18], a piecewise-quadratic *under*-approximation h_g^{MIQP} defined as:

$$h_g^{\text{MIQP}}(p_g) := f_g^{\text{Q}}(p_g) + \hat{f}_g^{\text{V}}(p_g\,; \boldsymbol{X}_g),$$

in opposition with the piecewise-linear approximation, $h_g =: h_g^{\text{MILP}}$, defined as

$$h_g^{\text{MILP}}(p_g) := \hat{f}_g(p_g; \boldsymbol{X}_g),$$

where $\hat{f}(\cdot; \boldsymbol{X})$ stands for the piecewise-linear interpolation of f, given the knots $\boldsymbol{X}$. Using such an under-approximation, we can easily build a surrogate problem in the same fashion as in § 2.3.1 by replacing $h_g^{\text{MILP}} := h_g$ with h_g^{MIQP}. We show now two useful features of the MIQP approximation: a necessary condition on the initial knots to have an under-approximation and a monotonic property. Then, we use these properties to show how to extend Theorem 2.2 and (2.13). Finally, we motivate our choice of resorting to an MILP approximation in this thesis.

Choice of the initial knots

In order for the interpolant to be lower than the original function that we wish to approximate, Proposition 2.3 shows that the kink points should be included in the set of knots.

However, this gives a poor initial approximation: the interpolant being the zero function. Indeed, since f_g^{V} vanishes at the kink points $\boldsymbol{X}_g^{\text{kink}}$, we have $\hat{f}_g^{\text{V}}(p_g\,; \boldsymbol{X}_g^{\text{kink}}) = 0$ for all p_g. Therefore, the maxima of f_g^{V} are also added to the initial set $\boldsymbol{X}_g^0$, giving the red dots of Fig. 2.6.

Proposition 2.3 (Choice of the initial knots)**.** *All kink points must belong to the set of knots $\boldsymbol{X}_g$ for the associated surrogate function $h_g^{MIQP}(p_g) := f_g^{Q}(p_g) + \hat{f}_g^{V}(p_g\,; \boldsymbol{X}_g)$ to be an under-approximation of f_g.*

Proof. We first remark that $\underline{P}_g$ and $\overline{P}_g$ must belong to the set of knots. Otherwise, the surrogate function is not defined for the whole range $[\underline{P}_g, \overline{P}_g]$. Assume for the sake of contradiction that a given kink point, $X \in (\underline{P}_g, \overline{P}_g)$, is not included in the set of knots $\boldsymbol{X}$, and let X^- and X^+ be two knots surrounding X, *i.e.*, $X^- < X < X^+$. We consider the surrogate function $h_g^{\text{MIQP}}(p_g) = f_g^{\text{Q}}(p_g) + \hat{f}_g^{\text{V}}(p_g\,; \boldsymbol{X}_g)$. Because f_g^{V} is nonnegative, we have

for all $p_g \in [\underline{P}_g, \overline{P}_g]$.

$$\hat{f}_g^{\mathrm{V}}(p_g; \boldsymbol{X}_g) \geq 0.$$

Therefore, we also have

$$f_g(X) = f_g^{\mathrm{Q}}(X) \leq h_g^{\mathrm{MIQP}}(X).$$

Since h_g^{MIQP} is by assumption an under-approximation of f_g, we also have $f_g(X) \geq h_g^{\mathrm{MIQP}}(X)$. It follows that $f_g(X) = h_g^{\mathrm{MIQP}}(X)$; this is in contradiction with the definition of $X \notin \boldsymbol{X}$. □

Knot updating criterion

One of the cornerstones of the proof of Theorem 2.2 is the knot updating criterion, *i.e.*, the last iterate is added to the set of knots and we never remove knots. This implies that

$$h^k(x^l) = f(x^l) \tag{2.23}$$

for every $l \in [k]$ and $k \in \mathbb{N}$.

Since the surrogate problem with an MIQP under-approximation also fulfills the knot updating criterion, and h_g^{MIQP} is also Lipschitz with the same constant as h_g^{MILP}. Theorem 2.2 remains valid and the proof—with $h_g := h_g^{\mathrm{MIQP}}$—is exactly the same, see [ASS18, Theorem 2].

Monotonic property

If f_g is piecewise-concave between two kink points, we have

$$\boldsymbol{X}_g^1 \subseteq \boldsymbol{X}_g^2 \implies \hat{f}_g(p_g; \boldsymbol{X}_g^1) \leq \hat{f}_g(p_g; \boldsymbol{X}_g^2) \leq f_g(p_g) \tag{2.24}$$

for each feasible point p_g and set of knots $\boldsymbol{X}_g^1$ containing the kink points of f_g.

This property is not needed for the proof of Theorem 2.2. However, it is important for the practical application of the algorithm and gives insights on why the method actually works. Recall that the *surrogate gap* is defined as

$$\delta^k = f(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^k),$$

and because a previous iteration may exhibit a lower objective than the current one, the *effective surrogate gap* is defined with the best-known solution,

$$\tilde{\delta}^k = \min_{l \in [k]} f(\boldsymbol{p}^l) - h^k(\boldsymbol{p}^k).$$

The monotonic property (2.24) implies that the sequence $(h^k(\boldsymbol{p}^k))_{k \in \mathbb{N}}$ is monotonically increasing.

The sequence $(\tilde{\delta}^k)_{k \in \mathbb{N}}$ can then be understood as the difference between a decreasing sequence of *upper bounds* $\mathcal{UB}^k := \min_{l \in [k]} f(\boldsymbol{p}^l)$ and an increasing sequence of *lower bounds* $\mathcal{LB}^k := h^k(\boldsymbol{p}^k)$.

Theorem 2.2 ensures that the infimum of $(\mathcal{UB})_{k \in \mathbb{N}}$ is equal to the supremum of $(\mathcal{LB})_{k \in \mathbb{N}}$.

REMARK 2.4 (Bound for a piecewise-quadratic under-approximation). If we use a piecewise-quadratic under-approximation h_g^{MIQP}, then $\epsilon^k = 0$ for all k. Hence (2.13) becomes

$$\min_{l \in [k]} f(\boldsymbol{p}^l) - \tilde{\delta}^k - \gamma^k \leq f(\boldsymbol{p}^*) \leq \min_{l \in [k]} f(\boldsymbol{p}^l). \tag{2.25}$$

The proof can be directly adapted from (2.22) with $\epsilon^k = 0$. Alternatively, one can observe that ϵ^k has been introduced to lower bound $f(\boldsymbol{p}^*)$ with $h^{\text{MILP},k}(\boldsymbol{p}^*)$:

$$h^{\text{MILP},k}(\boldsymbol{p}^*) - \epsilon^k \leq f(\boldsymbol{p}^*).$$

Here, since h^{MIQP} is a proper under-approximation, we directly have

$$h^{\text{MIQP},k}(\boldsymbol{p}^*) \leq f(\boldsymbol{p}^*).$$

It follows that

$$\begin{aligned} f(\boldsymbol{p}^k) - f(\boldsymbol{p}^*) &= f(\boldsymbol{p}^k) - h^{\text{MIQP},k}(\boldsymbol{p}^k) + h^{\text{MIQP},k}(\boldsymbol{p}^k) - f(\boldsymbol{p}^*) \\ &\leq \delta^k + h^{\text{MIQP},k}(\boldsymbol{p}^k) - h^{\text{MIQP},k}(\boldsymbol{p}^*) \\ &\leq \delta^k + h^{\text{MIQP},k}(\boldsymbol{p}^k) - h^{\text{MIQP},k}(\boldsymbol{p}^{**,k}) \\ &\leq \delta^k + \gamma^k. \end{aligned} \tag{2.26}$$

■

Comparison between the MILP and MIQP approximations

The surrogate problem defined with the MIQP approximation has interesting properties: i) provided that the initial set of knots contains the kink points, the first surrogate function is an under-approximation and ii) as

Table 2.1 Comparison in the execution time between the MILP and MIQP formulations for reaching different MIP gap.

	Execution time (sec)	
MIP gap (%)	MILP	MIQP
0.25	1	6
0.1	18	155
0.07	60	300

long as we never remove knots, the sequence of under-approximations is increasing.

However, for the same level of complexity (*e.g.*, the same number of decision variables) the MILP approximation is simpler to solve than the MIQP approximation. As a matter of comparison, we run the MILP and MIQP formulations to solve the initial surrogate problem detailed in § 3.3.1. We show in Table 2.1 that using the MILP formulation is faster than using the MIQP formulation. Because the running time is an important parameter of our analysis, this is a good point in favor of the MILP approximation.

The second argument in favor of the MILP formulation is the following: while the MILP formulation *over*-approximates f on a tiny region around every kink point ($\mathcal{R}^{\text{convex}}$), it better *under*-approximates f on the other larger region ($\mathcal{R}^{\text{concave}}$) [PJY17], that is,

$$h^{\text{MIQP}}(\boldsymbol{p}) \leq f(\boldsymbol{p}) \leq h^{\text{MILP}}(\boldsymbol{p}) \text{ for } \boldsymbol{p} \in \mathcal{R}^{\text{convex}},$$
$$h^{\text{MIQP}}(\boldsymbol{p}) < h^{\text{MILP}}(\boldsymbol{p}) < f(\boldsymbol{p}) \text{ for } \boldsymbol{p} \in \mathcal{R}^{\text{concave}},$$

with

$$\mathcal{R}^{\text{convex}} := \bigtimes_{g \in G} \mathcal{R}_g^{\text{convex}},$$
$$\mathcal{R}^{\text{concave}} := \bigtimes_{g \in G} [\underline{P}_g, \overline{P}_g] \setminus \mathcal{R}^{\text{convex}},$$

and $\mathcal{R}_g^{\text{convex}}$ defined as in (2.14).

By means of sacrificing the under-approximation feature on the set $\mathcal{R}^{\text{convex}}$, which is negligible in the practical instances considered here, we obtain a better under-approximation on the set $\mathcal{R}^{\text{concave}}$. Moreover, the running time of the MIP solver is reduced in the MILP case.

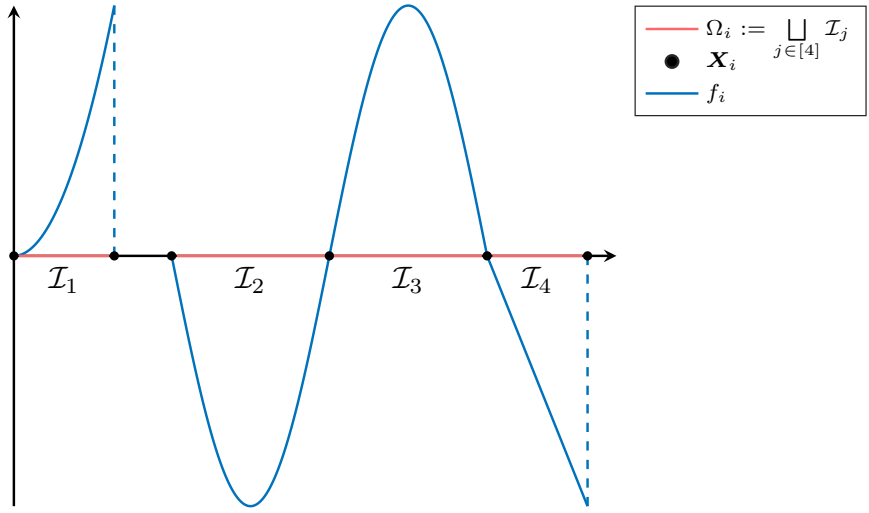


Fig. 2.11 Illustrative piecewise-smooth function.

These two arguments justify why we resort to an MILP approximation instead of an MIQP under-approximation throughout this thesis.

2.4 Extension to a broader class of functions

The method given in Algorithm 1 works for (the sum of) piecewise-smooth and piecewise-concave functions f_g. Nevertheless, it can be extended to (the sum of) piecewise-smooth functions f_i indexed with $i \in I$ and defined as

$$f_i \colon \Omega_i \to \mathbb{R}, \tag{2.27}$$

where $\Omega_i = \bigsqcup_{j_i \in J_i} \mathcal{I}_{j_i}$ and $\mathcal{I}_{j_i} \subseteq \mathbb{R}$ is a family of intervals, indexed with $j_i \in J_i$, that partition Ω_i such that $f_i|_{\mathcal{I}_{j_i}} \in \mathcal{C}^1$ and $f_i|_{\mathcal{I}_{j_i}}$ is either convex or concave. An illustrative function f_i is provided in Fig. 2.11. This illustrative function is discontinuous and nonsmooth on Ω_i but smooth and continuous on each interval $\mathcal{I}_{j_i}$.

We also consider the sum f of such univariate scalar functions:

$$f\colon \Omega \to \mathbb{R}\colon \boldsymbol{x} \mapsto f(\boldsymbol{x}) = \sum_{i\in I} f_i(x_i), \tag{2.28}$$

where $\Omega = \bigtimes_{i\in I} \Omega_i$.

As in the outer-approximation (OA) algorithm [DG86], the under-approximation of the convex part is tackled by adding constraints instead of variables. Let us apply this procedure on a simple example to illustrate the approach. We consider the problem

$$\min_{x\in[0,2\pi]} \sin(x). \tag{2.29}$$

Starting from the three knots $\{X_1, X_2, X_3\} = \{0, \pi, 2\pi\}$, the surrogate problem is written as follows:

$$\begin{aligned} \min_{\boldsymbol{\xi},\boldsymbol{\eta},t} \quad & g_1^0(\xi_1,\eta_1) + t, \\ \text{subject to } & X_0\eta_1 \le \xi_1 \le X_1\eta_1, \\ & X_1\eta_2 \le \xi_2 \le X_2\eta_2, \\ & u_1^0(\xi_2,\eta_2) \le t, \\ & u_2^0(\xi_2,\eta_2) \le t, \\ & \eta_1 + \eta_2 = 0,\ \eta_{1,2} \in \{0,1\}, \end{aligned} \tag{2.30}$$

where $g_j^k(\xi_j,\eta_j) = \alpha_j^k\xi_j + \beta_j^k\eta_j$ and (α_j^k, β_j^k) define the lines g_j^k from Figs. 2.12 and 2.13 (with parameters Eqs. (2.8) and (2.9)) for each concave segment $[X_j, X_{j+1}]$.

Let us consider $k > 0$ and a convex segment $[X_j, X_{j+1}]$. We also have $u_j^k(\xi_j,\eta_j) = \alpha_j^k\xi_j + \beta_j^k\eta_j$, except that now α_j^k is chosen such that u_j^k is tangent to f in $x^k \in (X_j, X_{j+1})$:

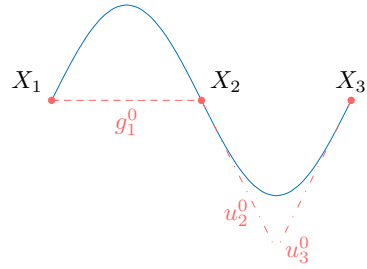
$$\alpha_j^k = f'(x^k), \tag{2.31}$$

and β_j^k is chosen such that $u_j^k(x^k) = f(x^k)$, that is,

$$\beta_j^k = f(x^k) - \alpha_j^k x^k. \tag{2.32}$$

For the case $k = 0$, one needs to carefully takes into account that f may not be differentiable in some X_j defining a convex interval and use properly the

$$h^0(x) = \begin{cases} g_1^0(x) & x \in [X_1, X_2] \\ \max\{u_2^0, u_3^0\}(x) & x \in]X_2, X_3] \end{cases}$$



(a) Mathematical formulation. **(b)** Illustration.

Fig. 2.12 Extension of APLA to piecewise-smooth functions. Initial surrogate function. The legend is the same as in Fig. 2.10.

right derivative,

$$\alpha_j^0 = \lim_{\substack{x \to X_j^+ \\ x \in [X_j, X_{j+1}]}} \frac{f(x) - f(X_j)}{x - X_j}, \tag{2.33}$$

or the left one,

$$\alpha_{j+1}^0 = \lim_{\substack{x \to X_{j+1}^- \\ x \in [X_j, X_{j+1}]}} \frac{f(x) - f(X_j)}{x - X_j}. \tag{2.34}$$

In any case, β_j^0 is chosen according to (2.32).

Let us analyze in Fig. 2.13 the procedure of refining around a point both in the concave (x^1) and in the convex region (x^2). For the sake of simplicity, we omit the binary formulation of the linear pieces, hence the initial underapproximation is defined as in Fig. 2.12a. For the concave case, we simply add x^1 to the set of knots, replacing g_1^0 by g_1^1 and g_2^1. And, for the convex case, we request t to also lie above the tangent in x^2, *i.e.*, $t \geq u_2^1(\xi_2, \eta_2)$. Doing so we can, with a few changes to Algorithm 1, adapt the method to deal with any piecewise-smooth function. This adapted method is given in Algorithm 2, which requires the knowledge of the kink points as well as the inflection points of each term of the separable objective.

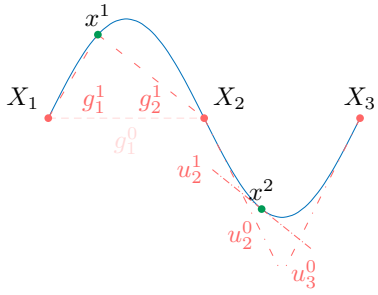
We note that the properties from § 2.3.4 are also verified for the convex region:

✓ Knot updating criterion (2.23):

$$h^2(x^2) = \max\{u_2^0, u_3^0, u_2^1\}(x^2) = f(x^2);$$

$$h^2(x) = \begin{cases} g_1^1(x) & x \in [X_1, x^1] \\ g_2^1(x) & x \in]x^1, X_2] \\ \max\{u_2^0, u_3^0, u_2^1\}(x) & x \in]X_2, X_3] \end{cases}$$

(a) Mathematical formulation. The full formulation with binary variables is provided in (2.30).



(b) Illustration.

Fig. 2.13 Same as in Fig. 2.12 after two iterations. The knots of the surrogate function contain a new point (x^1) in the concave region and a new point (x^2) in the convex region.

✓ Monotonicity (2.24):

$$h^0\Big|_{[X_2,X_3]} := \max\{u_2^0, u_3^0\} \leq \max\{u_2^0, u_3^0, u_2^1\} =: h^1\Big|_{[X_2,X_3]}.$$

We can therefore extend Theorem 2.2 into Theorem 2.4.

Theorem 2.4. *Given the minimization on a polytope of a separable function $f := \sum_{i=1}^n f_i$ with n univariate scalar functions as in (2.28), the sequence of iterates obtained by Algorithm 2 satisfies*

$$\lim_{k\to\infty} \delta^k = 0.$$

Proof. The function f is the separable sum of piecewise-smooth functions f_i. Moreover, each piece $j_i \in [n_i^{\text{knot}}]$ of f_i is also Lipschitz continuous with some Lipschitz constant L_{j_i}, and f_i is Lipschitz continuous with constant $L_i := \max_{j_i \in [n_i^{\text{knot}}]} L_{j_i}$. This constant is also valid for the corresponding $h^k\Big|_{[X_i,X_{i+1}]}$, which is also Lipschitz continuous for all k. Therefore, f is Lipschitz with constant $L := \sum_{i\in I} L_i$ and so is h^k with the same constant. Finally, we note that the definition of the surrogate function in the piecewise-

smooth case (*e.g.*, as in (2.30)) satisfies the knot updating criterion. Therefore, the remainder of the proof is similar to the proof of Theorem 2.2.

Indeed, let $(\boldsymbol{p}^k)_{k\in\mathbb{N}}$ be the sequence of optimal solutions to the surrogate problem associated with function h^k, we then also obtain

$$\begin{aligned}\delta^k &= f(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^k)\\ &= f(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^{k-1}) + h^k(\boldsymbol{p}^{k-1}) - h^k(\boldsymbol{p}^k)\\ &= f(\boldsymbol{p}^k) - f(\boldsymbol{p}^{k-1}) + h^k(\boldsymbol{p}^{k-1}) - h^k(\boldsymbol{p}^k)\\ &\leq 2L\|\boldsymbol{p}^k - \boldsymbol{p}^{k-1}\|_2.\end{aligned}$$

The last line comes from the Lipschitz continuity, and the 3rd line is also true in this extended case because so is the knot updating criterion.

Suppose now for contradiction that $(\delta^k)_{k\in\mathbb{N}}$ does not converge to 0. Then, there exists $\delta^* > 0$ and an infinite subsequence $(\delta^{k_j})_{j\in\mathbb{N}}$ with $|\delta^{k_j}| > \delta^*$ for all j. And, for each j, we have that for all $J > j$, $\|\boldsymbol{p}^{m_J} - \boldsymbol{p}^{m_j}\|_2 \geq \delta^*/(2L)$. This implies that the subsequence $(p^{m_j})_{j\in\mathbb{N}}$ is unbounded, a contradiction with the admissible range constraints. □

The optimality gap is obtained as follows

$$\begin{aligned}f(\boldsymbol{p}^k) - f(\boldsymbol{p}^*) &= f(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^k) + h^k(\boldsymbol{p}^k) - f(\boldsymbol{p}^*)\\ &= \delta^k + h^k(\boldsymbol{p}^k) - f(\boldsymbol{p}^*)\\ &\leq \delta^k + h^k(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^*)\\ &\leq \delta^k + h^k(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^{**,k})\\ &\leq \delta^k + \gamma^k,\end{aligned} \tag{2.35}$$

where the first inequality comes from the monotonic property of $(h^k)_{k\in\mathbb{N}}$, the second from the definition of $\boldsymbol{p}^{**,k}$ as the global minimizer of $(2.10)^k$, and the last inequality from the definition of γ^k, see (2.21).

The combination of (2.35) with Theorem 2.4 shows that the convergence is guaranteed up to the user-prescribed solver tolerance γ.

2.5 ADMM to solve the static dispatch

This thesis mostly focuses on methods that build sequences of lower bounds (typically obtained *via* under-approximations) and upper bounds until ter-

Algorithm 2 Extension of APLA to piecewise-smooth functions

Require: piecewise-smooth function f as in (2.28)

$I \leftarrow$ indices of variables
for $i \in I$ **do**
  $\boldsymbol{X}_i^{\texttt{kink}} \leftarrow$ kink points of f_i
  $\boldsymbol{X}_i^{\texttt{c}} \leftarrow$ inflection points of f_i
  $\boldsymbol{X}_i \leftarrow$ initial knots such that $\boldsymbol{X}_i^{\texttt{kink}} \bigcup \boldsymbol{X}_i^{\texttt{c}} \subseteq \boldsymbol{X}_i$ ▷ Union without repetition
  $\boldsymbol{Y}_i \leftarrow f_i(\boldsymbol{X}_i)$
  $n_i^{\text{knot}} \leftarrow \texttt{size}(\boldsymbol{X}_i)$
end for
$k \leftarrow 0$
Set tolerance parameter δ_{tol}
Set tolerance of the MILP solver γ
$\tilde{\delta}^k \leftarrow \delta_{\text{tol}} + 1$
while $\tilde{\delta}^k > \delta_{\text{tol}}$ **do**
  $\boldsymbol{x}^k \leftarrow$ optimal solution to MILP surrogate problem defined with knots $(\boldsymbol{X}, \boldsymbol{Y})$, obtained with MILP solver with tolerance γ
  $\delta^k \leftarrow f(\boldsymbol{x}^k) - h^k(\boldsymbol{x}^k)$
  $\tilde{\delta}^k \leftarrow \min_{l \in [k]} f(\boldsymbol{x}^l) - h^k(\boldsymbol{x}^k)$
  for $i \in I$ **do**
    if $\min_{j \in [n_i^{\text{knot}}]} \left| x_i^k - X_{i,j} \right| > 0$ **then**
      $\boldsymbol{X}_i \leftarrow \text{insort}(\boldsymbol{X}_i, x_i^k)$ ▷ Ordered insertion
      $\boldsymbol{Y}_i \leftarrow \text{insert}(\boldsymbol{Y}_i, f_i(x_i^k))$ ▷ Insert at same index as previous line
      $n_i^{\text{knot}} \leftarrow n_i^{\text{knot}} + 1$
    end if
  end for
  $k \leftarrow k + 1$
end while
return $\arg\min_{l=0,\ldots,k-1} f(\boldsymbol{x}^l)$

mination. In this section, we shortly present a different approach that has attracted a lot of attention in the last years (in particular in the power system community): the alternating direction method of multipliers (ADMM) [WYZ19, Boy10, BM17, Woh17, RNLZ18, TP20]. Such a method belongs to the larger and as popular class of *splitting methods* that also include, *e.g.*, Douglas-Rachford and Peaceman-Rachford methods, Dykstra's method, and the forward-backward method. The reason for studying ADMM here is threefold. First, due to its popularity in the power system community, we shortcut here any potential question that some readers may have. Second, because the objective is the sum of univariate scalar functions; this makes it possible to run ADMM in a decentralized fashion (see remark 2.6) and save therefore computational resources. Finally, this section serves to give a first glimpse of splitting methods, which will also be studied in Chapter 6.

2.5.1 ADMM in the general convex case

Let us consider the following general problem:

$$\begin{aligned} &\min f(x) \\ &\text{subject to } x \in C, \end{aligned} \tag{2.36}$$

with a convex set C and a convex function $f\colon C \to \mathbb{R}$.

ADMM is designed to solve an optimization problem with an objective that is the sum of two scalar functions subject to a linear equality constraint. We can write (2.36) in such a way. Let $\mathbb{1}_C$ be the indicator function of C, we have

$$\begin{aligned} &\min f(x) + \mathbb{1}_C(z) \\ &\text{subject to } x - z = 0. \end{aligned} \tag{2.37}$$

To solve (2.37) with ADMM, we first introduce the scaled *augmented Lagrangian*, see remark 2.5 for the explanation about the scaling, defined as follows:

$$L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{u}) = f(\boldsymbol{x}) + \mathbb{1}_C(\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{z} - \boldsymbol{u}\|^2, \tag{2.38}$$

with $\rho \in \mathbb{R}^+$ the penalty parameter. ADMM applies a *primal-dual* algorithm on the augmented Lagrangian and *splits* the primal step in two by successively freezing $\boldsymbol{x}$ and $\boldsymbol{z}$; this gives Algorithm 3. Since ADMM is not the primary focus of this thesis, we will not go much into the details of why the method works in the convex and some nonconvex cases. Interested readers may find further information in [BPC11, WYZ19].

Algorithm 3 ADMM

Require: f convex defined on a convex set C, inital point $(\boldsymbol{x}^0, \boldsymbol{z}^0, \boldsymbol{u}^0)$

1: $k \leftarrow 0$
2: **while** a tolerance criterion is not satisfied **do**
3: $\quad \boldsymbol{x}^{k+1} \leftarrow \arg\min_{\boldsymbol{x}} \left(\mathbb{1}_C(\boldsymbol{x}) + \frac{\rho}{2} \|\boldsymbol{z}^k - \boldsymbol{x} + \boldsymbol{u}^k\|^2 \right) = \mathrm{Pr}_C \left(\boldsymbol{z}^k + \boldsymbol{u}^k \right)$ ▷ $\mathrm{Pr}_C(\cdot)$ is the projection operator onto C
4: $\quad \boldsymbol{z}^{k+1} \leftarrow \arg\min_{\boldsymbol{z}} \left(f(\boldsymbol{z}) + \frac{\rho}{2} \|\boldsymbol{z} - \boldsymbol{x}^{k+1} + \boldsymbol{u}^k\|^2 \right)$
5: $\quad \boldsymbol{u}^{k+1} \leftarrow \boldsymbol{u}^k + \boldsymbol{x}^{k+1} - \boldsymbol{z}^{k+1}$
6: **end while**

An iteration of ADMM applied to (2.4) is quite cheap. The first primal update—with $\boldsymbol{z}$ frozen—in Line 3 of Algorithm 3 is a projection step that is easy to compute, and the second primal update—with $\boldsymbol{x}$ frozen—in Line 4 can be decoupled in our case, as f is separable. And the dual update, Line 5, is trivial.

In the remainder of the section, we first use ADMM for solving the simple—but unphysical—static dispatch with unconstrained units in § 2.5.2, then in § 2.5.3 we apply ADMM to the problem of interest of this chapter: (2.4). We discuss briefly on the convergence guarantees for nonconvex problems in § 2.5.4 and present some numerical results in § 2.5.5.

REMARK 2.5 (Scaled ADMM). Algorithm 3 is using the scaled dual variable u. In fact, the augmented Lagrangian of (2.37) is usually defined as

$$L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) := f(\boldsymbol{x}) + \mathbb{1}_C(\boldsymbol{z}) + \langle \boldsymbol{y}, \boldsymbol{x} - \boldsymbol{z} \rangle + \frac{\rho}{2} \|\boldsymbol{x} - \boldsymbol{z}\|^2,$$

where $f(\boldsymbol{x}) + \mathbb{1}_C(\boldsymbol{z}) + \langle \boldsymbol{y}, \boldsymbol{x} - \boldsymbol{z} \rangle$ is the traditional dualization of the constraint $\boldsymbol{x} - \boldsymbol{z} = 0$ with Lagrange multiplier $\boldsymbol{y}$, and $\frac{\rho}{2} \|\boldsymbol{x} - \boldsymbol{z}\|^2$ is the *augmented* term that can be seen as a regularization term. This regularization term makes the augmented Lagrangian a μ-strongly convex function with constant $\mu = \rho$, see [Nes18] for the definition of μ-strongly convex functions. It is possible to compact this formulation by means of the following variable

change: $\boldsymbol{u} := \frac{\boldsymbol{y}}{\rho}$. This yields

$$\begin{aligned} L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{u}) &= f(\boldsymbol{x}) + \mathbb{1}_C(\boldsymbol{z}) \\ &\quad + \rho \left(\frac{1}{2}\langle \boldsymbol{x} - \boldsymbol{z}, \boldsymbol{x} - \boldsymbol{z} \rangle + \langle \boldsymbol{u}, \boldsymbol{x} - \boldsymbol{z} \rangle + \frac{1}{2}\langle \boldsymbol{u}, \boldsymbol{u} \rangle - \frac{1}{2}\langle \boldsymbol{u}, \boldsymbol{u} \rangle \right) \\ &= f(\boldsymbol{x}) + \mathbb{1}_C(\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{z} + \boldsymbol{u}\|^2 + \text{const}(\boldsymbol{u}). \end{aligned}$$

Since in ADMM (Algorithm 3), the minimization is performed with respect to $\boldsymbol{x}$ and $\boldsymbol{z}$, we can drop the term that only depends on $\boldsymbol{u}$. This effectively gives (2.38). The dual update becomes

$$\begin{aligned} \boldsymbol{y}^{k+1} &:= \boldsymbol{y}^k + \rho(\boldsymbol{x}^{k+1} - \boldsymbol{z}^{k+1}) \\ &\Leftrightarrow \\ \boldsymbol{u}^{k+1} &:= \boldsymbol{u}^k + \boldsymbol{x}^{k+1} - \boldsymbol{z}^{k+1}. \end{aligned}$$

■

2.5.2 Static dispatch with unconstrained units

Let us consider a problem even easier than (2.4), the unconstrained economic dispatch:

$$\begin{aligned} &\min_{\boldsymbol{p}} f(\boldsymbol{p}) \\ &\text{subject to } \sum_{g \in G} p_g = P^{\mathrm{D}}, \end{aligned} \tag{2.39}$$

with the same objective as (2.4). We should not apply ADMM to this instance, due to the nonconvexity of f. Let us forget about this assumption for now. This is investigated in § 2.5.4. The first primal update (Line 3 of Algorithm 3) is given as the projection onto

$$C_{\text{uncstr}} := \left\{ \boldsymbol{p} \in \mathbb{R}^{|G|} : \sum_{g \in G} p_g = P^{\mathrm{D}} \right\},$$

which is readily computed as

$$\text{Pr}_{C_{\text{uncstr}}}(\boldsymbol{p}) = \boldsymbol{p} + \frac{(P^{\mathrm{D}} - \sum_{g \in G} p_g)}{|G|} \mathbb{1}^{|G|},$$

where $\mathbb{1}^{|G|}$ is a vector full of 1. The second primal update (Line 4) is obtained by solving

$$\arg\min_{\boldsymbol{p}} f(\boldsymbol{p}) + \frac{\rho}{2}\|\boldsymbol{p} - \boldsymbol{x}^{k+1} + \boldsymbol{u}^k\|^2.$$

This problem is the unconstrained minimization of a *separable* function, whose minimization can thereby be decoupled:

$$\arg\min_{p_g} f_g(p_g) + \frac{\rho}{2}(p_g - x_g^{k+1} + u_g^k)^2.$$

This function is piecewise-smooth, thus the global minimizer $z_g^{k+1,*}$ is

- Either a kink point of f_g (see § 2.3.1);
- Or satisfies $f_g'(z_g^{k+1,*}) + \rho(z_g^{k+1,*} - x_g^{k+1} + u_g^k) = 0$.

Hence, an iteration of ADMM applied to (2.39) only costs $\mathcal{O}(|G|)$.

2.5.3 Static dispatch with constrained units

The previous section covers a problem that is not useful in practice, as it does not restrict the power ranges (*e.g.*, negative productions may result from (2.39)).

We can almost as easily apply ADMM to the static dispatch (2.4). The first primal update (Line 3 of Algorithm 3) becomes the projection onto the convex set

$$C_{\text{ED}} := \left\{ \boldsymbol{p} \in \mathbb{R}^{|G|} : \sum_{g \in G} p_g = P^{\text{D}},\ \underline{P}_g \leq p_g \leq \overline{P}_g \text{ for all } g \in G \right\}.$$

Computing the projection $\text{Pr}_{C_{\text{ED}}}(\boldsymbol{p})$ amounts to solving

$$\min_{\boldsymbol{x}} \|\boldsymbol{p} - \boldsymbol{x}\|^2$$
$$\text{subject to } \boldsymbol{x} \in C_{\text{ED}}.$$

This problem can be solved efficiently in $\mathcal{O}(|G|)$ using the approach developed in [PK90]. Implementation in Python is available in [Nic18].

The second primal update is the same as in § 2.5.2. We therefore also obtain a complexity of $\mathcal{O}(|G|)$ for a single iteration of ADMM.

2.5.4 Convergence guarantees for nonconvex functions

Using the convergence result from [WYZ19, Table 1, Scenario 2], we can readily show that Algorithm 3 applied to (2.4) converges, provided that the barrier parameter ρ is large enough [WYZ19, Lemma 9].

Nevertheless, while convergence is guaranteed, the convergence rate may be slow. And the larger the ρ, the smaller the convergence rate. This phenomenon is the so-called slow "tail convergence" of ADMM [EY15].

Moreover, [WYZ19] proves the convergence of ADMM in the nonconvex setting to a *critical point*. In our case, due to the high multimodal nature of the problem, Algorithm 3 may converge to some local solution $\boldsymbol{p}^*$.

We observe this phenomenon in the experiments detailed in the following subsection.

2.5.5 Numerical experiments

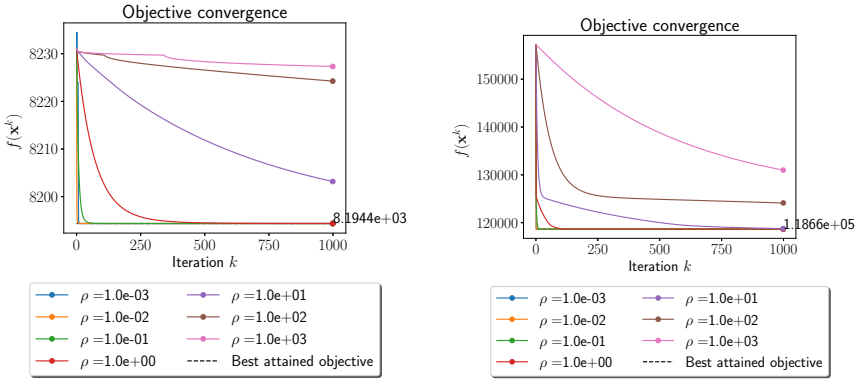
We first test ADMM in the convex case. We demonstrate that the method effectively converges in this case, and we show the slow tail convergence for large ρ. The nonconvex case is also treated, and we observe how the method may not converge if ρ is small enough. We also observe that the method can converge to local minimizers.

Convex experiments

Let us start by running ADMM on the convex economic dispatch, *i.e.*, (2.4) without VPE. We run a 3-unit and 40-unit convex dispatch with parameters from Tables A.2 and A.3, except that the VPE is removed, *i.e.*, $D_g = E_g = 0$ for all $g \in G$.

We observe in Fig. 2.14 that no matter the barrier parameter ρ, the method converges to the optimal solution. Larger ρ are associated with slower convergence rates, and there is some trade-off: a ρ too small is also associated with a slower convergence rate. For the 3-unit case, Fig. 2.14a, the optimal ρ is 0.01 and the method converges in a dozen iterations. For the 40-unit case, Fig. 2.14b, the optimal barrier parameter is also $\rho = 0.01$.

REMARK 2.6 (Running time and parallelization of ADMM). The running time is given for information in the caption of Figs. 2.14 and 2.15. One of the main advantages of ADMM is that the second part of the primal update (Line 4 of Algorithm 3) can be easily computed in a decentralized way, since



(a) 3-unit (convex) case, all instances run in less than 1.2 second.

(b) 40-unit (convex) case, all instances run in around 60 seconds.

Fig. 2.14 ADMM applied to a convex static dispatch. The starting point is the projection of $\mathbf{0}$ onto the feasible set.

this minimization problem is decoupled with respect to g. Using up to $|G|$ processors may significantly decrease the execution time.

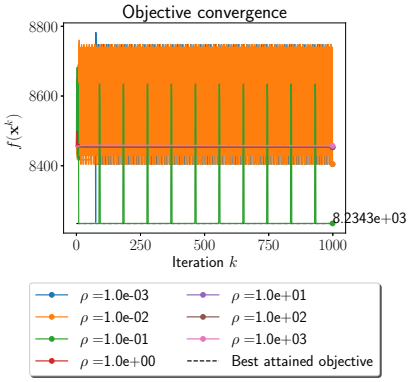
For the dynamic dispatch, which solves a dispatch for a given set of time steps T (see Chapter 3), we could use up to $|G|\,|T|$ processors. ■

Nonconvex experiments

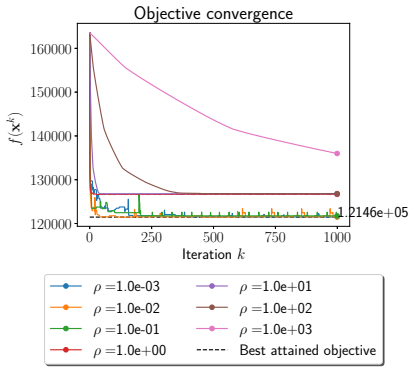
We run the same experiments in the nonconvex setting with parameters from Tables A.2 and A.3. We observe in Fig. 2.15 that for large ρ, the method smoothly converges to a local minimum and for small ρ, it oscillates. For the 3-unit case, Fig. 2.15a, only the runs with $\rho \in \{0.001, 0.01\}$ attain the global minimum of 8234 \$; this minimum is computed with APLA. For the 40-unit case, we observe a similar behavior—with smaller oscillations. In this case, the global solution has an objective (also computed with APLA) of 121 412 \$, this is slightly below every solution obtained by ADMM.

2.5.6 Conclusion of the application of ADMM on the economic dispatch

In this section, which stands alone with respect to the remainder of the manuscript, the alternating direction method of multipliers is proposed to



(a) 3-unit (nonconvex) case, all instances run in less than 14 seconds.



(b) 40-unit (nonconvex) case, all instances run in around 180 seconds.

Fig. 2.15 Same as Fig. 2.14 in the nonconvex case.

tackle the economic dispatch.

While ADMM is able to efficiently capture the solution in the convex case, it fails to properly find the solution when the valve point effect is taken into consideration. Indeed, tuning the parameter ρ to ensure convergence is difficult, and the convergence—which can be particularly slow—when it occurs, may yield a suboptimal local minimizer.

Note that we also considered heuristics for changing the parameter ρ dynamically, see, [Woh17, § IV. A.] and [TP20, § 4.1], or an accelerated version of ADMM [KCSB15] without much success. We did not include the results for the sake of keeping this discussion on ADMM as short as possible.

The lack of ability of ADMM to find the global optimal is the reason why we do not further discuss ADMM in the following chapters.

2.6 A preprocessing method: bound tightening

This subsection is devoted to the presentation of a preprocessing method with the purpose of reducing the number of variables in our optimization problem, by computing *tighter* bounds for each variable. Such a method

has been studied in the context of power systems in, *e.g.*, [CHVH16, Mol17, SDRH19].

For the purpose of illustration, let us explain this method on a toy example. Let f be a function which will be defined later on, we study the following optimization problem:

$$\begin{aligned} &\min_{x,y\in\mathbb{R}^2} \quad f(x,y) \\ &\text{subject to} \begin{array}{l} x+y=100 \\ 25\leq(x,y)\leq 200 \end{array} \Big\} := \Omega, \end{aligned} \tag{2.40}$$

and let Ω denotes the feasible set of (2.40), illustrated in Fig. 2.16. We observe that the box induced by the constraint ranges (dark gray) is much greater than the feasible set (blue line). This larger box may yield longer execution time in the *black-box* solvers that we use: such solvers often rely on relaxation solutions, *e.g.*, by solving (2.40) without the binding constraint $x+y=100$. In this context, the larger the box, the worse the relaxation.

In the next sections, we show how to improve the relaxations by reducing the size of the box. This improvement can be performed for every function f. We also observe (see Table 2.2) that it seems that this is already implemented in the preprocessing steps of Gurobi. Finally, we provide another method that exploits the specific structure of f in (2.1).

2.6.1 Bound tightening for any objective function

We can easily reduce the box by successively trying to tighten each upper bound. For y, this yields

$$\max_{x,y\in\Omega} y = 75, \tag{2.41}$$

which is illustrated in Fig. 2.17a. For x this gives

$$\max_{x,y\in\Omega} x = 75, \tag{2.42}$$

which is illustrated in Fig. 2.17b. We can do the same for the lower bounds *via* minimization. In this toy example, *the lower bounds cannot be improved*, and they stay at the same value of 25. The optimization problem at hand

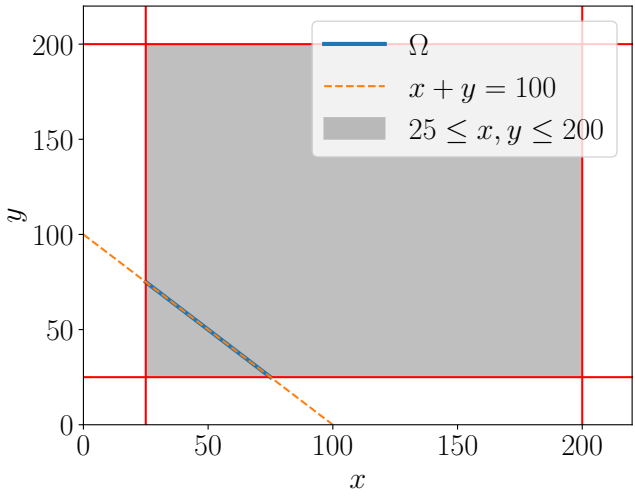


Fig. 2.16 Domain Ω of (2.40).

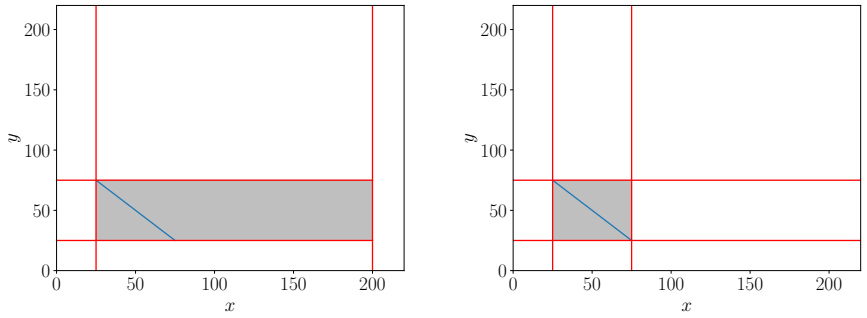
can therefore be rewritten as:

$$\begin{aligned} &\min_{x,y\in\mathbb{R}^2} \quad f(x,y) \\ &\text{subject to} \begin{array}{l} x+y=100 \\ 25 \leq (x,y) \leq 75 \end{array} \Big\} := \Omega_1, \end{aligned} \tag{2.43}$$

where the changes with respect to (2.40) have been highlighted.

REMARK 2.7 (Different formulations for the same feasible set.). The feasible set of (2.43) has a different formulation than (2.40), but both are equivalent, *i.e.*, $\Omega_1 = \Omega$. However, the relaxation obtained when removing the binding constraint $x + y = 100$ is much better with Ω_1 than with Ω. ■

A direct implication of remark 2.7 is that every (feasible) solution to (2.40) is a (feasible) solution to (2.43) with the same objective. Let us now particularize this procedure to a function f that is similar to (2.1).



(a) Bound reduction after (2.41).

(b) Bound reduction after Eqs. (2.41) and (2.42).

Fig. 2.17 Illustration of the bound tightening method for any objective function.

2.6.2 Bound tightening for structured functions

Let us consider the following objective function that belongs to the class of function defined by (2.1):

$$f(x,y) = \frac{x^2}{2} + y^2 + 50\left|\sin\frac{4\pi x}{100}\right|. \tag{2.44}$$

The toy optimization problem at hand reads

$$\begin{aligned} \min_{x,y\in\mathbb{R}^2} \quad & \frac{x^2}{2} + y^2 + 50\left|\sin\frac{4\pi x}{100}\right| \\ \text{subject to } & x + y = 100 \\ & 25 \leq (x,y) \leq 75, \end{aligned} \tag{2.45}$$

and is represented in Fig. 2.18.

Let $\underline{f}$ be an under-approximation of f, that is,

$$\underline{f}(x,y) \leq f(x,y) \text{ for all} (x,y) \in \Omega, \tag{2.46}$$

and let $\mathcal{UB}$ be some upper bound on the value of the optimal solution. We readily identify $x^2/2 + y^2$ as a possible choice of $\underline{f}$—notice that it corresponds to f_g^{Q} from (2.1). For $\mathcal{UB}$, we note that the objective of every feasible solution is a valid choice. Let $\mathcal{UB} := f(50,50) = 3750$, we consider the

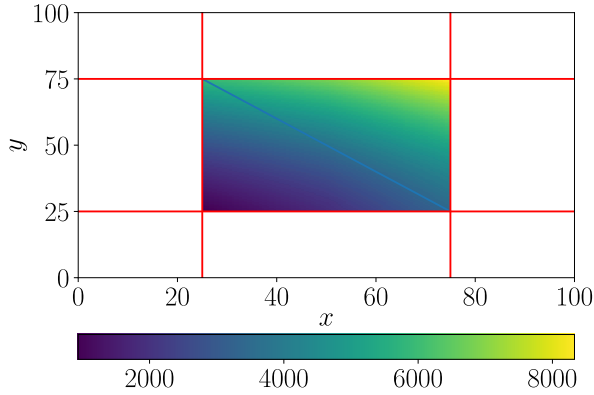


Fig. 2.18 Level curves of (2.45).

following optimization problem:

$$\left.\begin{array}{ll} & \max y \\ & x+y=100 \\ \text{subject to} & 25 \leq x, y \leq 75 \\ & \underline{f}(x,y) \leq \mathcal{UB} \end{array}\right\} := \Omega_y. \tag{2.47}$$

The colors match the illustration of this optimization problem in Fig. 2.19. The red area represents the points inside the allowable ranges that cannot be optimal because their objectives are above the lower bound $\mathcal{UB}$; the blue area is the complement. The green line is y^{**}, the solution objective of (2.47), and we show in Proposition 2.5 that adding the constraint $y \leq y^{**}$ does not cut the optimal solution off. This property is verified in this simple problem: the optimal solution (x^*, y^*) stands below the green line $y = y^{**}$.

In our example illustrated in Fig. 2.19, the new feasible set Ω_2 is the half of the feasible set: Ω_2 is the intersection of the blue line $(x + y = 100)$ and the lower half-space defined by the green line $y = y^{**}$. Intuitively, the first half of the feasible set (between the points $(25, 75)$ and $(50, 50)$), being inside the red area, yields (feasible) points that are suboptimal. We can therefore cut this part off. This observation is proven in Proposition 2.5.

Proposition 2.5. *Let $\mathcal{X}$ be a subset of $\mathbb{R}^{n-1}$ and $\mathcal{Y}$ a subset of $\mathbb{R}$, with $n \in \mathbb{N}$.*

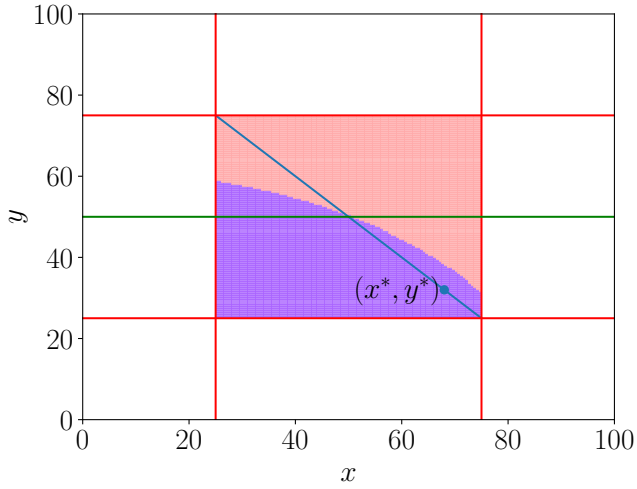


Fig. 2.19 Bound tightening using an under-approximation function.

Let $f : \underbrace{\mathcal{X} \times \mathcal{Y}}_{:=\Omega_1}$ *be the objective function of*

$$\min_{(\boldsymbol{x},y)\in\Omega_1} f(\boldsymbol{x},y),$$

$\underline{f}$ *be an under-approximation of* f *on* Ω_1*, and* $\mathcal{UB}$ *be some upper bound on the optimization problem at hand. If we define*

$$\Omega_y := \left\{(\boldsymbol{x},y) \in \Omega_1 \mid \underline{f}(\boldsymbol{x},y) \leq \mathcal{UB}\right\},$$

and

$$\Omega_2 := \Omega_1 \bigcap (y \leq \max_{(\boldsymbol{x}',y')\in\Omega_y} y'),$$

we have the two following properties:

- $\Omega_2 \subseteq \Omega_1$*;*
- $\min\limits_{(\boldsymbol{x},y)\in\Omega_1} f(\boldsymbol{x},y) = \min\limits_{(\boldsymbol{x},y)\in\Omega_2} f(\boldsymbol{x},y).$

Proof. The first property directly follows from the definition of Ω_2 as a subset of Ω_1. To prove the second property, let us assume for the sake of

contradiction that there exists $(\tilde{\boldsymbol{x}}, \tilde{y})$ a solution to

$$\min_{(\boldsymbol{x},y)\in\Omega_1} f(\boldsymbol{x}, y)$$

such that $\tilde{y} > \max_{(\boldsymbol{x},y)\in\Omega_2} y$. Then we have

- either $(\tilde{\boldsymbol{x}}, \tilde{y})$ is not feasible for Ω_1;
- or $(\tilde{\boldsymbol{x}}, \tilde{y})$ does not satisfy $\underline{f}(\tilde{\boldsymbol{x}}, \tilde{y}) \leq \mathcal{UB}$.

The first implication is not possible due to the feasibility of $(\tilde{\boldsymbol{x}}, \tilde{y})$ and for the second, we verify that

$$\underline{f}(\tilde{\boldsymbol{x}}, \tilde{y}) > \mathcal{UB} \Rightarrow f(\tilde{\boldsymbol{x}}, \tilde{y}) > \mathcal{UB},$$

in contradiction with the optimality of $(\tilde{\boldsymbol{x}}, \tilde{y})$. □

2.6.3 Example of bound tightening on a 3-unit system

To further illustrate the two bound tightening techniques—with or without taking f into account—we test them on a 3-unit system. The parameters are given in Table A.2.

We use $\underline{f} := \sum_{g\in G} f_g^{\mathrm{Q}}$ as the under-approximation function and the objective of a feasible point as the lower bound $\mathcal{UB}$.

Figure 2.20 shows the change in the power ranges. We see that both techniques allow a significant reduction of the feasible set. Moreover, we observe that whole segments are cut off. Therefore, the number of integer variables needed to describe the surrogate problem (2.10) also decreases. This reduction is illustrated in Table 2.2.

Inspection of Table 2.2 shows that using Gurobi to solve (2.10) with preprocessing yields (after preprocessing) the same number of integer and continuous variables as Ω_1. This experiment demonstrates that Gurobi performs bound tightening in the preprocessing; this makes sense, as such a procedure can be done no matter f. On the other hand, the bound tightening yielding Ω_2 outperforms the preprocessing of Gurobi, in the sense that the resulting number of variables is lower.

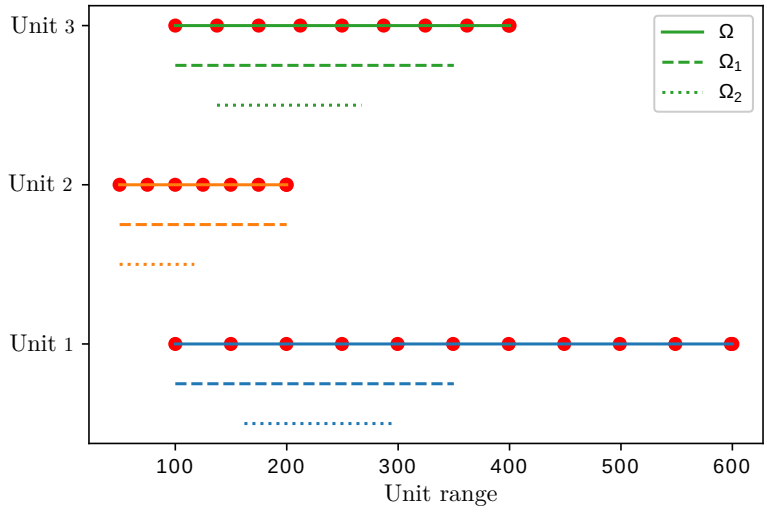


Fig. 2.20 Illustration of the two bound tightening techniques for a 3-unit system. The red dots are the initial kink points from Algorithm 1.

Table 2.2 Reduction *via* bound tightening in the number of integer variables of (2.10) for a 3-unit system.

	Ω	Ω_1	Ω_2
Continuous variable (no preprocessing)	30	23	14
Binary variable (no preprocessing)	24	17	8
Continuous variable (with preprocessing)	23	23	14
Binary variable (with preprocessing)	17	17	8

2.6.4 Discussion

The bound tightening technique is an interesting preprocessing method, that has already shown its usefulness in the power system literature, see, *e.g.*, [Mol17].

We show here that it can also be used to reduce the number of integer variables of the surrogate problem handled by APLA.

In the 3-unit case considered in this section, no major execution time improvement can be established. This is because the static dispatch has a relatively small size, and the execution time of APLA only takes a few seconds. Moreover, most of the execution time is used by Gurobi to build the model and not to actually solve it. As only the solving time benefits from the bound tightening, this explains the lack of time improvement on the preprocessed models.

We also tested this method on the larger dynamic cases of Chapter 3 with little improvement: while the first preprocessing slightly reduces the execution time, the second preprocessing does not change the number of variables for these larger cases.

Because the first preprocessing is already implemented in Gurobi, bound tightening is not further considered in this thesis.

3

A matheuristic for the dynamic economic dispatch

In this chapter, the dynamic economic dispatch problem (also called multi-period dispatch) is introduced. This problem consists in solving a dispatch for different time steps, *e.g.*, for each hour of the day. As ramping constraints couple two consecutive time steps, this more practical problem is much larger than the static dispatch.

We first introduce the dynamic economic dispatch with reserve constraints. Reserves are defined as the excess capacity that is withheld by generators (or loads) such that the system can offer uninterrupted service to customers in case of contingencies. Indeed, energy markets have such a mechanism where, along with the power production, reserves can be exchanged in a dedicated market. Reserves are one of the ancillary services used by the system operator to mitigate problems linked to disturbances, *e.g.*, the failure of a generating unit to produce. There exist several types of reserves, which is a subset of the ancillary services, depending on how fast they react. Since gas power plants can quickly change their power output, they are often used for providing reserves.

We also introduce the DC approximation of the network constraints as well as the dynamic economic dispatch with such a consideration of the network.

We then show that both problems can be solved by the APLA algorithm

from Chapter 2. To alleviate the rocketing execution time of our algorithm, we develop in the same vein as [WDW+17] a matheuristic that efficiently scans the feasible set and quickly converges to a competitive solution, while providing a lower bound. The drawback of the time saving is the loss of the guarantee of convergence to the global optimum. This last point is the reason for the title of the chapter: a *matheuristic* is a heuristic algorithm made by the interoperation of metaheuristics and mathematic programming techniques [BMRBR09]. Matheuristics are heuristics that still provide guarantees and were one of the main topics of the 31st European Conference on Operation Research (EURO 2021) where parts of the work of this chapter were presented [VAP21].

The structure of the chapter is as follows. In Section 3.1, the two dynamic dispatch models are presented as well as the associated surrogate problems. The adaptation of the APLA method is described in Section 3.2, and two versions of a matheuristic are defined. We then perform tests of the method and the matheuristics in Section 3.3. Finally, conclusions are drawn in Section 3.4.

This chapter is based on [VPA19, VAP20a].

3.1 Problem formulation

3.1.1 Dynamic economic dispatch with reserves

The dynamic economic dispatch is similar to the (static) dispatch from Chapter 2 except that now, the dispatch must be carried out on a finite set of time steps denoted as T. It aims at minimizing fuel cost f, which is defined as the sum of the production cost of every generator unit f_g at each time step. The production of unit g at time step t is denoted as p_{gt}. As in (2.5), we model the cost function as the sum of a smooth quadratic part f_g^{Q} and a nonsmooth rectified sine that captures the VPE f_g^{V}:

$$f_g(p_{gt}) = \underbrace{A_g p_{gt}^2 + B_g p_{gt} + C_g}_{:=f_g^{\mathrm{Q}}(p_{gt})} + \underbrace{\left| D_g \sin E_g(p_{gt} - \underline{P}_g) \right|}_{:=f_g^{\mathrm{V}}(p_{gt})}. \tag{3.1}$$

Here, A_g, B_g, C_g, D_g, E_g, and $\underline{P}_g$ correspond to the same cost parameters as (2.5) and are independent of t.

The full objective reads

$$f(\boldsymbol{p}) = \sum_{t \in T} \sum_{g \in G} f_g(p_{gt}), \tag{3.2}$$

and a single term $f_g(p_{gt})$ is depicted in Fig. 2.6b.

The constraints considered are the following:

- Power range limits
$$\underline{P}_g \leq p_{gt} \leq \overline{P}_g, \tag{3.3}$$
where $\underline{P}_g$ and $\overline{P}_g$ are the minimum and maximum power output of unit g.

- Ramp rate restrictions
$$\underline{R}_g \leq p_{gt} - p_{g(t-1)} \leq \overline{R}_g, \tag{3.4}$$
where $\underline{R}_g$ and $\overline{R}_g$ are the ramp-down and ramp-up rates of unit g, respectively. We consider that p_{g0} is given for every $g \in G$ as a parameter of the model. Hence, (3.4) holds for all $t \in T$.

- Power balance
$$\sum_{g \in G} p_{gt} = P_t^{\mathrm{D}}, \tag{3.5}$$
where P_t^{D} is the demand in period t.

- Spinning upward reserve constraints
$$\sum_{g \in G} s_{gt} \geq P_t^{\mathrm{S}}, \tag{3.6}$$
$$0 \leq s_{gt} \leq \overline{R}_g, \tag{3.7}$$
$$p_{gt} + s_{gt} \leq \overline{P}_g, \tag{3.8}$$
with s_{gt} the extra capacity that generator g must be able to provide and P_t^{S} the total spinning upward reserve required at time t.

Remark that (3.6) is not an equality: we expect the system to be able to provide *at least* P_t^{S} reserves at time t; it is not a problem of producing *more*. We also see that the constraint (3.8) supersedes the right-hand side of (3.3). Finally, we note that this formulation can be written without the explicit

definition of the auxiliary variables s_{gt}; this will be done in Chapter 4 (Eqs. (4.6) to (4.8)).

Using these constraints, the first problem of interest of this chapter is defined in (3.9).

Dynamic economic dispatch with spinning upward reserves (DED-R)

$$
\begin{aligned}
\min_{\boldsymbol{p},\boldsymbol{s}} f(\boldsymbol{p}) &= \sum_{t\in T}\sum_{g\in G} f_g(p_{gt}) & & \\
\text{subject to } & \sum_{g\in G} p_{gt} = P_t^{\mathrm{D}} & & t \in T \\
& \underline{R}_g \le p_{gt} - p_{g(t-1)} \le \overline{R}_g & & (g,t) \in G \times T \\
& \sum_{g\in G} s_{gt} \ge P_t^{\mathrm{S}} & & t \in T \\
& 0 \le s_{gt} \le \overline{R}_g & & (g,t) \in G \times T \\
& p_{gt} + s_{gt} \le \overline{P}_g & & (g,t) \in G \times T \\
& \underline{P}_g \le p_{gt} & & (g,t) \in G \times T
\end{aligned}
\tag{3.9}
$$

3.1.2 Dynamic economic dispatch with DCOPF

One of the challenges of power system operations comes from the network of physical lines. To illustrate this, let us consider the toy example of Fig. 3.1. If we want to ship 1 MW of power from node D, where a generating unit is located, to node A, where a load is located, we cannot directly ship it through line $k := (A, D)$. Indeed, the solution must satisfy Ohm's law such that the 1 MW will flow from D to A using the following lines: $\{(A,D),(A,B),(A,C),(C,D)\}$. This coupling notably complicates operations in large networks, especially due to the nonlinear nature of Ohm's law. In this thesis, we first model directly the network by using a linear (DC) approximation of Ohm's law, then in Chapter 4, we indirectly consider the network with quadratic power losses.

In the second problem considered in this chapter, we employ a bus angle model of the DC optimal power flow (DCOPF) problem, where the network is described as an oriented graph. The parameter B_k corresponds to the susceptance of a line $k = (m, n)$. The susceptance is the physical quantity that corresponds to the ratio between the amount of power that flows over

the line joining two nodes and their nodal phase difference. See [JEOC16] for details about the DCOPF and some limitations of this model.

Using the susceptance-based model, the flow of power e_{kt} along line $k \in \mathcal{K}$ at time t, is then described as a function of the bus angle difference along the nodes that the line is joining:

$$e_{kt} = B_k(\theta_{mt} - \theta_{nt}), \tag{3.10}$$

where θ_{mt} stands for the angle of the voltage phasor of bus m at time step t, and $\mathcal{K}$ is the set of lines. We fix the angle of the reference bus (indexed by 0), *i.e.*, $\theta_{0t} = 0$ for all $t \in T$. If losses are neglected, the main addition to the common economic dispatch models, that are employed in the literature for accounting for the valve point effect, lies in the power balance constraint:

$$-\sum_{g \in G_n} p_{gt} - \sum_{k=(\cdot,n)} e_{kt} + D_{nt} + \sum_{k=(n,\cdot)} e_{kt} = 0, \tag{3.11}$$

where G_n is the set of generators at bus n and D_{nt} the demand of bus n at time t.
This constraint must hold for each bus n and, in the multi-period case, for each time period t. Given a flow limit TC_k for line k, the flow of power along each line is constrained as follows:

$$-TC_k \leq e_{kt} \leq TC_k. \tag{3.12}$$

Using these three additional constraints and the usual nonconvex and nonsmooth objective,

$$\min \sum_{t \in T} \sum_{g \in G} f_g(p_{gt}), \tag{3.13}$$

with cost functions (2.5), we obtain the second problem of interest of the chapter: the dynamic dispatch with DCOPF. This problem is given in 3.14. Among its decision variables, the angle at each bus θ and the flow over each line e can be fully determined from the production at each node n.

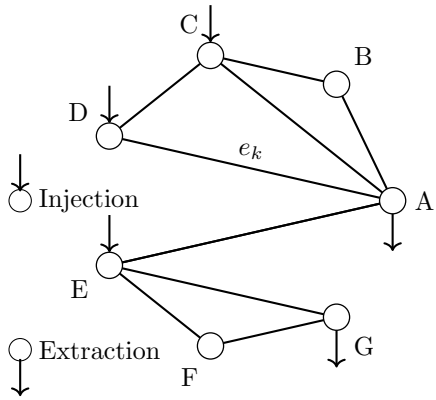


Fig. 3.1 Toy example of a network.

Dynamic economic dispatch with DCOPF based on susceptance (DED-DCOPF)

$$\min_{\boldsymbol{p},\boldsymbol{e},\boldsymbol{\theta}} f(\boldsymbol{p}) = \sum_{t\in T}\sum_{g\in G} f_g(p_{gt})$$

$$\begin{aligned}
\text{subject to } & e_{kt} = B_k(\theta_{mt} - \theta_{nt}) && k=(m,n)\in\mathcal{K}, t\in T\\
& D_{nt} - \sum_{k=(\cdot,n)} e_{kt} + \sum_{k=(n,\cdot)} e_{kt} = \sum_{g\in G_n} p_{gt} && (n,t)\in N\times T\\
& -TC_k \le e_{kt} \le TC_k && (k,t)\in\mathcal{K}\times T\\
& \underline{R}_g \le p_{gt} - p_{g(t-1)} \le \overline{R}_g && (g,t)\in G\times T\\
& \underline{P}_g \le p_{gt} \le \overline{P}_g && (g,t)\in G\times T
\end{aligned} \tag{3.14}$$

3.1.3 Comparison between the static and dynamic dispatch

Both formulations of the dynamic dispatch are similar to the simple economic dispatch (2.4): the minimization of a separable nonsmooth and nonconvex function (3.2) on a polytope Ω.

General dynamic dispatch

$$\min_{x \in \mathbb{R}^N} \sum_{i=1}^{N} f_i(x_i) \tag{3.15}$$
$$\text{subject to } x \in \Omega$$

The main difference comes from the number of variables and constraints: a few dozen (the number of generating units) for the static dispatch (2.4), a few hundred for the dynamic dispatch with reserves (3.9), and a few thousand for the DCOPF-based dynamic dispatch (3.14), see Table 3.1 for the comparison among the three. The consideration of larger networks may considerably increase these numbers. This surge in the number of decision variables—and in the overall complexity of the problem—prompts the use of another method than the one detailed in Chapter 2. Indeed, although the numerical experiments presented in § 3.3.1 shows that APLA reaches a close-to-optimal solution in a few iterations, these iterations are computationally expensive.

Table 3.1 Comparison of the number of variables and constraints of the dispatch models.

	Nr. variables	Nr. constraints
Static dispatch (2.4)	$\lvert G\rvert$	$2\lvert G\rvert + 1$
Dynamic dispatch with reserves (3.9)	$2\lvert G\rvert\lvert T\rvert$	$2\lvert T\rvert + 6\lvert G\rvert\lvert T\rvert$
Dynamic dispatch with DCOPF (3.14)	$(\lvert G\rvert + (\lvert N\rvert - 1) + \lvert K\rvert)\lvert T\rvert$	$(3\lvert\mathcal{K}\rvert + \lvert N\rvert + 4\lvert G\rvert)\lvert T\rvert$
Dynamic dispatch with reserves and losses (P)	$\lvert G\rvert\lvert T\rvert$	$4\lvert G\rvert\lvert T\rvert + 4\lvert T\rvert$

As a matter of comparison, the problems tackled in [ASS18] by the method APQUA from Chapter 2 range from 3 to 40 units, that is, up to 40 variables and 81 constraints. In § 3.3.1, the considered dynamic problem with reserves consist of $\lvert G\rvert = 10$ units over $\lvert T\rvert = 24$ time steps, *i.e.*, 480 variables and 1448 constraints. In § 3.3.2, a dynamic dispatch with DCOPF is run on the IEEE 118 case with 74 units (54 regular and 20 obeying a VPE), 24 time steps, $\lvert N\rvert = 118$ nodes and $\lvert\mathcal{K}\rvert = 77$ lines, *i.e.*, 6432 variables and 15480 constraints. Finally, in Chapter 4 the reserves and the losses are taken into account in (P) but not the network directly. This situation, detailed in Chapter 4, has a similar complexity to the dynamic dispatch with reserves in terms of number of constraints and variables. However, the consideration of the losses adds another layer of complexity: in this case, the feasible set is also nonconvex. The largest problem studied in Section 4.3 consists in the

dispatch of 15 units over 24 time steps: this gives 360 variables and 1512 constraints.

3.1.4 Surrogate problem

As in § 2.3.1, we can define a surrogate function by means of a piecewise-linear approximation and define the corresponding surrogate problem.

The feasible sets of Eqs. (3.9) and (3.14) are polytopes. The main difficulty in solving these problems is due to the nonconvex and nonsmooth objective. Thus, we can obtain for both problems a surrogate problem that is defined by approximating the objective without changing the feasible set, namely by employing the following objective function:

$$\min \sum_{t\in T} \sum_{g\in G} h_{gt}(p_{gt}), \tag{3.16}$$

subject to being in the feasible set of (3.9) or (3.14).

Given the set of knots $\boldsymbol{X}_{gt} := \left(X_{gt1}, \ldots, X_{gtn_{gt}^{\text{knot}}}\right)$, the terms of (3.16) read as

$$h_{gt}(p_{gt}) := \begin{cases} \hat{f}_g(p_g; \boldsymbol{X}_{gt}) & \text{if } f_g^{\text{V}} \neq 0, \\ f_g^{\text{Q}}(p_g) & \text{else.} \end{cases} \tag{3.17}$$

Recall that $\hat{f}(\cdot; \boldsymbol{X})$ stands for the piecewise-linear interpolation of f, given the knots $\boldsymbol{X}$, see § 2.3.1 for the modeling of piecewise-linear functions. The smooth part of the objective f_g^{Q} and the nonsmooth part f_g^{V} are defined as in (2.5).

REMARK 3.1 (On the dependence of the time step). The surrogate objective depends on the time step, although the objective f is not t–dependent, *e.g.*, the cost of producing p_{gt} is $f_g(p_{gt})$ and its surrogate objective is $h_{gt}(p_{gt})$. This difference comes from the time dependence of the knots $\boldsymbol{X}_{gt}$. ■

The novel aspects of the present chapter relative to Chapter 2 are—in addition to the consideration of a dynamic demand with ramping constraints—the integration of the reserves or the network to the model as well as the development of matheuristics (denoted as *heuristics*) that reduce the execution time and avoid timeouts. The comparison of the different methods in § 3.3.2 shows that the heuristics reach comparable solutions in a significantly reduced execution time.

3.2 Methods

This section is devoted to the description of two methods. The first one (§ 3.2.1) is largely similar to the method of Chapter 2, the difference being that the demand is dynamic and either the reserves or the network constraints are accounted for. In § 3.3.2, we demonstrate that network effects exhibit a rich interplay with the valve point effect. Therefore, it is important to consider the representation of the valve point effect in future dispatch models. However, the size of the systems considered in the present work renders the approach of Chapter 2 non-viable, hence the second method.

This second method (§ 3.2.2) is a local heuristic based on the former approach and trades optimality guarantees for a reduction in computation time. We detail two variants of the heuristic.

3.2.1 A globally convergent method

Method description

Let us first recall the globally convergent method APLA[1] from Section 2.3. The method starts with a set of knots $\boldsymbol{X}^0$ (red dots in Fig. 2.6) satisfying the property described in 2.3.4, *i.e.*, the inclusion of the kink points in the set of knots. The first surrogate problem, which is defined by the first surrogate function h^0, is then formulated as a mixed-integer problem (MIP) and solved with a predefined tolerance γ. The MIP solver returns a (first) solution $\boldsymbol{p}^0$ along with a lower bound on the global optimum. The surrogate gap δ^0 is then computed as

$$\delta^0 = f(\boldsymbol{p}^0) - h^0(\boldsymbol{p}^0). \tag{3.18}$$

This gap is visualized in Fig. 2.6. Using the surrogate gap, we compute the optimality gap by adding to it the solver tolerance. If the target accuracy is reached, the algorithm stops. Otherwise, the obtained solution $\boldsymbol{p}^0$ is added

[1] Because of the small over-approximation error, this method is not *sensu stricto* globally convergent. However, by adding some knots (two times the number of kink points) and by tackling the small convex regions around each kink point with the method described in Section 2.4, we can easily define a method that (globally) converges up to the user-prescribed tolerance. We will not do that here for three reasons: for the sake of simplicity, to save on computational power, and because this over-approximation error is much smaller than the typical solver tolerance that we consider in the practical examples of this thesis, see § 2.3.3 for more details.

to the set of knots, refining in this way the approximation, and the algorithm iterates.

This process is illustrated in Fig. 2.6a. The red and green dots represent the initial and already added knots, respectively. The purple square is the optimal solution to the surrogate problem, and the purple dot is the evaluation of the real objective at this solution. It can be seen that the method locally refines the approximation. In this sense, the method is adaptive and benefits from a lower number of knots with respect to a regular meshing. The efficiency gains of the method rely on quickly converging to a subset of the feasible space where the global optimum should lie. This follows the philosophy of other methods used in dispatch algorithms, such as *stochastic dual dynamic programming (SDDP)* [PP91], that also aim at computational savings by using locally valid representations of the objective function that is being optimized, with the purpose of quickly limiting the search to the relevant part of the feasible space by employing the information contained in the approximation of the objective function.

Convergence guarantees

Since the problem (3.15), which identifies jointly Eqs. (3.9) and (3.14), has the same form as (2.4)—the minimization of a separable sum of functions like (2.1) on a polytope—the convergence guarantees developed in § 2.3.3 remain valid. At iteration k of the method, the optimality gap can be computed as:

$$f(\boldsymbol{p}^k) - f(\boldsymbol{p}^*) \leq \overbrace{\delta^k + \gamma^k + \epsilon^k}^{\text{Optimality gap}}, \tag{3.19}$$

where $\boldsymbol{p}^*$ is the optimal solution to (3.15), γ^k stands for the gap of the kth surrogate problem returned by the MIP solver, and ϵ^k is a negligible over-approximation error. Using Theorem 2.2, we also have that for Lipschitz continuous cost function f, the sequence of iterations provided by APLA satisfies:

$$\lim_{k \to \infty} \delta^k = 0.$$

Stopping criterion

The algorithm terminates when the optimality gap, (3.19), is lower than a predefined tolerance γ: $\delta^k + \gamma^k \leq \gamma$. This γ corresponds to the targeted MIP gap, *i.e.*, the gap between the lower and upper objective bound of the surrogate problem, which is given as input to the MIP solver. Theorem 2.2 shows that the method converges, but the drawback is that it requires

several costly calls to the MIP solver that increase the computation time of the algorithm. This drawback motivates the development of a heuristic, described in the following section, as well as a stopping criterion based on i) the number of iteration and ii) the maximum time allowed to the MIP solver. This criterion is used in the experiments in § 3.3.2.

3.2.2 A local heuristic

Method description

Fig. 3.2 summarizes the heuristic that can be split into two parts: first a global search is made on the whole feasible set to find optimum candidates (blue part), and then the search is refined around these candidates (red part). In [PJY18], the authors proposed to use the solution to the initial surrogate problem as the initial point of an interior point method. Here, we follow the same approach with three main differences: a) the initial point is obtained with a few steps of APLA, b) a list of candidates is considered and, c) the local search method is also based on a local approximation. Let us explain more formally these three points.

Search for an initial point The first step of this heuristic is to select a promising candidate, around which the local search will be initialized. In order to achieve this, the APLA algorithm is used with a finite number of iterations n_{iter} over the whole search space Ω, *i.e.*, the feasible set of (3.15). This number is typically chosen to be small to avoid excessive running time. In the experiments below, we take $n_{\text{iter}} = 2$.

List of candidates During the search for an initial point, several potential candidates for the global solution to the surrogate problems are found by the MIP solver. Most of these points, called *integer solutions*, are good initial guesses for a local search. Hence, APLA is slightly modified to return a list $\mathcal{L}_{\text{I}}$ of the best integer solutions, which will serve as initial guesses. Using a list of initial guesses instead of a single starting point reduces the sensitivity of the method with respect to the initial point. To be more specific, we modify Line 11 of Algorithm 1 such that, instead of only returning the best solution $\boldsymbol{p}^k$, the algorithm returns a list $\mathcal{L}_{\text{I}}$ of the τ best integer solutions. Note that the the influence of this metaparameter τ is marginal as soon as we select a few of the best integer solutions. Here, we set this metaparameter to 50. Because in our experiments Gurobi seldom finds more than 50 integer

solutions, this implies that virtually each integer solution is tested as a starting point for the local search method. Such a situation is possible because the local search is much faster than the search for an initial point. However, our experiments show that, while the top few elements of $\mathcal{L}_{\mathrm{I}}$ yield good solutions, the remaining elements have poor objective values. Therefore, the resulting local searches of these remaining elements end up with local optima with poor objective values.

Local search method The local search in the neighborhood $\Omega_{\tilde{\boldsymbol{p}}^i}$, of a specific point $\tilde{\boldsymbol{p}}^i$, at iteration i of the inner (red) loop, proceeds as follows: the power generation constraint (3.3) is narrowed around the value $\tilde{\boldsymbol{p}}^i$ *via* the closest knots. Eq. (3.3) now reads

$$X^i_{gtj} \leq p_{gt} \leq X^i_{gtj'}, \tag{3.20}$$

for $1 \leq j < j' \leq n^{\mathrm{knot}}_{gt}$ and $X^i_{gtj} \leq \tilde{p}^i_{gt} \leq X^i_{gtj'}$. In this way, the feasible set is strongly reduced and the power ranges become $t-$dependent. In the specific case of $j' = j+1$, see Fig. 3.3a for a visual representation of this restriction, the first surrogate problem of every APLA call inside the (red) loop becomes a much simpler convex quadratic problem that can be solved efficiently.

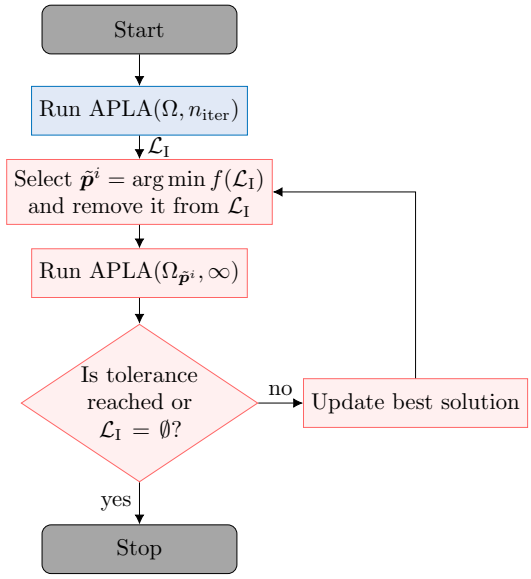
Convergence guarantees

This APLA-based heuristic is converging because every APLA call inside the loop (Fig. 3.2) converges by Theorem 2.2, and the number of loop calls is equal to the finite size of $\mathcal{L}_{\mathrm{I}}$. However, every lower bound obtained on a sub-instance $\Omega_{\tilde{\boldsymbol{p}}^i}$ is not a lower bound for (3.15). As we neglect the over-approximation error, the optimality gap is computed as follows:

$$f(\boldsymbol{p}^{i,\tilde{k}}) - f(\boldsymbol{p}^*) \leq \overbrace{f(\boldsymbol{p}^{i,\tilde{k}}) - (h^{n_{\mathrm{iter}}}(\boldsymbol{p}^{n_{\mathrm{iter}}}) - \gamma^{n_{\mathrm{iter}}})}^{\text{Optimality gap}}, \tag{3.21}$$

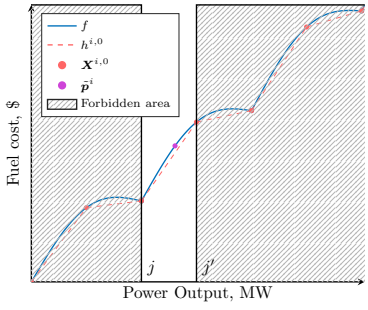
where $(\boldsymbol{p}^{i,\tilde{k}})_{\tilde{k}=0,1,\dots}$ and $(\boldsymbol{p}^k)_{k=0,1,\dots,n_{\mathrm{iter}}}$ are the solution sequences of the two APLA-calls APLA($\Omega_{\tilde{\boldsymbol{p}}^i}, \infty$) and APLA($\Omega, n_{\mathrm{iter}}$). To prove this expression, let us consider Fig. 3.4 that illustrates the different quantities of (3.21). We have:

$$\underline{\mathrm{h}}^{n_{\mathrm{iter}}} \leq h^{n_{\mathrm{iter}}}(\boldsymbol{p}^{**,n_{\mathrm{iter}}}) \leq h^{n_{\mathrm{iter}}}(\boldsymbol{p}^*) \leq f(\boldsymbol{p}^*),$$

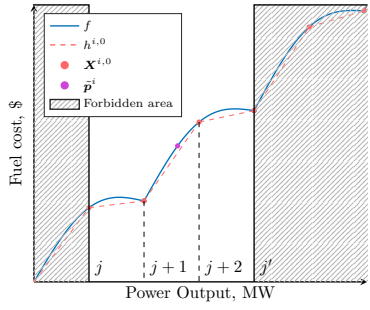


1. Optimization run on the whole domain Ω.
2. Local improvement of each candidate $\tilde{\boldsymbol{p}}^i \in \mathcal{L}_{\text{I}}$ obtained in previous step. Only δ can be improved.

Fig. 3.2 Flow chart of the APLA-based heuristic.



(a) First variant: H-local. One segment is selected.



(b) Second variant: H-full. Up to three segments are selected.

Fig. 3.3 Illustration of the heuristic restrictions.

where $\underline{h}^{n_{\text{iter}}}$ is a lower bound on the global optimum of the (full) n_{iter}th surrogate problem. This bound is provided by the MIP solver. Indeed, the lower bound of the n_{iter}th (and last) surrogate problem on the whole domain (blue box in Fig. 3.2) is lower or equal to the optimal objective of this surrogate problem (first inequality). The optimality of $\boldsymbol{p}^{**,n_{\text{iter}}}$ for the last surrogate problem implies that its objective is lower or equal to the objective of $\boldsymbol{p}^*$, one of the optimal solution to our main problem (3.15), (second inequality). The last inequality is true because $h^{n_{\text{iter}}}$ under approximates f, provided that we neglect the over-approximation error. We thus have

$$-f(\boldsymbol{p}^*) \leq -\underline{h}^{n_{\text{iter}}}.$$

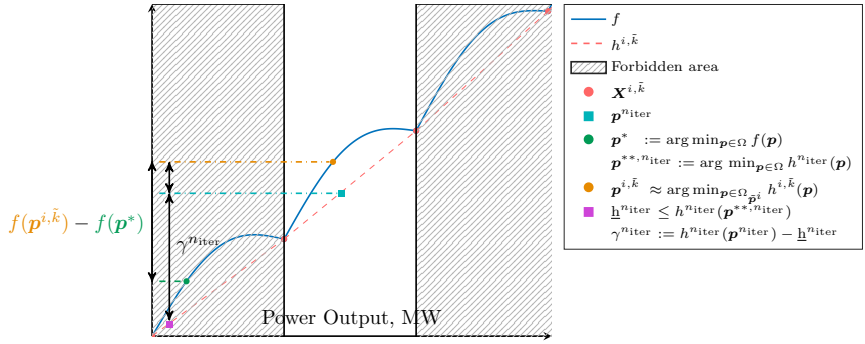
Finally, adding $f(\boldsymbol{p}^{i,\tilde{k}})$ and using the definition of $\gamma^{n_{\text{iter}}}$ (see Fig. 3.4) gives (3.21). Eq. (3.21) shows that once the heuristic enters the inner (red) loop in Fig. 3.2, it reduces the optimality gap only by decreasing the objective. An expression similar to (3.19) can also be obtained,

$$\begin{aligned} f(\boldsymbol{p}^{i,\tilde{k}}) - f(\boldsymbol{p}^*) &= f(\boldsymbol{p}^{i,\tilde{k}}) - f(\boldsymbol{p}^{*,i}) + \underbrace{f(\boldsymbol{p}^{*,i}) - f(\boldsymbol{p}^*)}_{:=\zeta^i} \\ &\leq \delta^{i,\tilde{k}} + \gamma^{i,\tilde{k}} + \epsilon^{i,\tilde{k}} + \zeta^i \\ &= \delta^{i,\tilde{k}} + \zeta^i, \end{aligned} \tag{3.22}$$

where $\boldsymbol{p}^{*,i} := \arg\min_{\boldsymbol{p}\in\Omega_{\tilde{\boldsymbol{p}}^i}} f(\boldsymbol{p})$. The first inequality holds because the application of APLA on the restricted domain $\Omega_{\tilde{\boldsymbol{p}}^i}$ is a valid instance. Hence, (3.19) can be used. Furthermore, the over-approximation error is negligible here and the restricted instances $\text{APLA}(\Omega_{\tilde{\boldsymbol{p}}^i}, \infty)$ are solved to optimality, *i.e.*, with a tolerance $\gamma^{i,\tilde{k}} \approx 0$. This sub-instance optimality implies that $\boldsymbol{p}^{i,\tilde{k}} \approx \arg\min_{\boldsymbol{p}\in\Omega_{\tilde{\boldsymbol{p}}^i}} h^{i,\tilde{k}}(\boldsymbol{p})$. This expression is visualized in Fig. 3.5.

Eq. 3.22 shows that at iteration $\tilde{k}$ of $\text{APLA}(\Omega_{\tilde{\boldsymbol{p}}^i}, \infty)$, there are two main contributions to the optimality gap. Firstly, $\delta^{i,\tilde{k}}$, *i.e.*, the (sub-instance) gap between the surrogate and true function. This gap goes to zero as $\tilde{k}$ goes to infinity by Theorem 2.2. In particular, we have $\lim_{\tilde{k}\to\infty} \boldsymbol{p}^{i,\tilde{k}} = \boldsymbol{p}^{i,*}$. Secondly, ζ^i that captures the case where no optimal solution $\boldsymbol{p}^*$ lies in $\Omega_{\tilde{\boldsymbol{p}}^i}$.

By construction $\boldsymbol{p}^{n_{\text{iter}}} \in \mathcal{L}_{\text{I}}$ and in the special case where $\tilde{\boldsymbol{p}}^i = \boldsymbol{p}^{n_{\text{iter}}}$, we have $f(\boldsymbol{p}^{*,i}) \leq f(\boldsymbol{p}^{n_{\text{iter}}})$. This inequality implies that $\zeta^i \leq \delta^{n_{\text{iter}}}$: the heuristic either improves $\boldsymbol{p}^{n_{\text{iter}}}$ or shows that it is globally optimal. Unfortunately, (3.22) is not useful in practice, as $f(\boldsymbol{p}^*)$—and also ζ^i—is not known. On the



$$f(\boldsymbol{p}^{i,\tilde{k}}) - f(\boldsymbol{p}^*) \leq f(\boldsymbol{p}^{i,\tilde{k}}) - h^{n_{\text{iter}}}(\boldsymbol{p}^{n_{\text{iter}}}) + \gamma^{n_{\text{iter}}} = f(\boldsymbol{p}^{i,\tilde{k}}) - \underline{h}^{n_{\text{iter}}}$$

Fig. 3.4 Illustration of (3.21).

other hand, all quantities in (3.21) can be computed, and we can use this bound, denoted as *optimality gap*, as a tolerance criterion.

Stopping criterion

The heuristic terminates either if it reaches the targeted optimality gap, (3.21) or after iteration over the whole list $\mathcal{L}_{\text{I}}$.

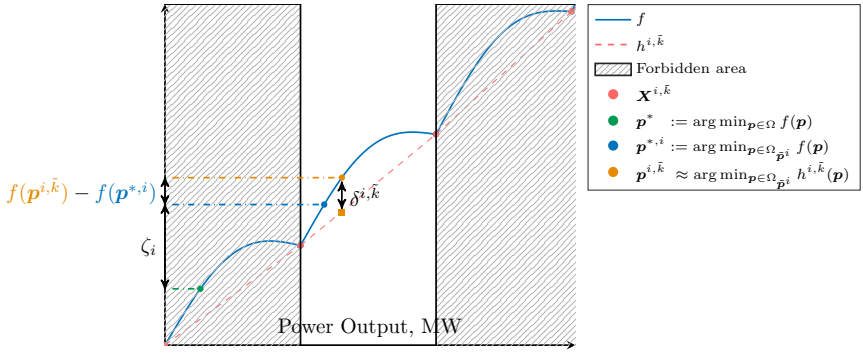
Note that all quantities in (3.21) can be computed: $\boldsymbol{p}^{i,\tilde{k}}$ is the current iterate, $h^{n_{\text{iter}}}(\boldsymbol{p}^{n_{\text{iter}}})$ is the best surrogate objective attained by APLA$(\Omega, n_{\text{iter}})$, and $\gamma^{n_{\text{iter}}}$ is the (returned) effective solver tolerance (see § 2.3.3).

Extension of the methods

As for APLA, the heuristic can be easily extended to account for piecewise-smooth functions, multiple fuels, and prohibited operation zones (POZ). These model enhancements will incur an increase in the number of integer variables and further motivate the use of an APLA-based heuristic.

3.3 Numerical experiments

We consider two batches of experiments. First, we test in § 3.3.1 the dynamic problem with reserves, as given in (3.9). We show that APLA is able to converge to the global solution, compute the over-approximation error which is negligible, and assess the price of neglecting the VPE. This experiment



$$f(\boldsymbol{p}^{i,\tilde{k}}) - f(\boldsymbol{p}^*) = f(\boldsymbol{p}^{i,\tilde{k}}) - f(\boldsymbol{p}^{*,i}) + \underbrace{f(\boldsymbol{p}^{*,i}) - f(\boldsymbol{p}^*)}_{:=\zeta^i} \leq \delta^{i,\tilde{k}} + \zeta^i$$

Fig. 3.5 Illustration of (3.22).

gives a first glimpse of the limitations of APLA: the running time of around 900 seconds is too excessive in comparison with the size of the problem that contains ten units. Most importantly, the majority of the execution time is used to improve the lower bound instead of reducing the objective.

In the second batch of experiments, § 3.3.2, we solve the dynamic dispatch on a IEEE57 and IEEE118 bus system, as defined in (3.14). We test APLA as well as the two heuristic variants and demonstrate that the heuristics capture good solutions in a fraction of the execution time of APLA. Finally, the cost of the valve point effect computed for both test cases demonstrates the interplay of the network with the VPE.

3.3.1 Dynamic dispatch with reserves

In this section, a 10-unit dynamic economic dispatch without losses over $T = 24$ hours is studied and the returned solution is compared with the solution that is obtained by ignoring the VPE. The data set used for the case study can be found in [AKTH02] and the (upward) spinning reserve constraint is set at 5 % of the demand. All parameters are given in Tables A.4 and A.5. The optimization is performed on a computer with an Intel-i7 CPU and 16 GB of RAM. Gurobi 8.0.0 has been used with a relative gap tolerance of 0.25% and the model has been coded in AMPL. Because every

feasible solution stays feasible for the surrogate problem at every iteration, we fed the MILP solver with the best-known solution to the true problem as a (feasible) starting point, thereby benefiting from the previous iterations. Algorithm 1 is also slightly improved by asking, in Line 11, the solver to return the 50 best integer solutions instead of the sole best one; this integer solution list is equivalent to $\mathcal{L}_{\mathrm{I}}$ from § 3.2.2. From this list, we select the best solution *with respect to the true objective*, and this point becomes our iterate.

After 9 iterations and 902 seconds, a solution $\tilde{\boldsymbol{p}}$ with $\tilde{\delta} = 1.42\,\$$ is found with objective 1 016 276 \$. This is an improvement over the previous best solution in the literature with objective 1 016 311 \$ [PJY18]. Neglecting the over-approximation error, which is upper-bounded by $\epsilon^{\max} = 0.32\,\$$ here, the final relative optimality gap is

$$\frac{|f(\boldsymbol{p}^*) - f(\tilde{\boldsymbol{p}})|}{f(\tilde{\boldsymbol{p}})} = \frac{\gamma + \tilde{\delta}}{f(\tilde{\boldsymbol{p}})} = 0.25\%. \tag{3.23}$$

Following standard practice of the VPE literature [CY06, PJY18, ASS18], we include the power dispatch among the generating units in Table 3.2 to validate our results.

The practical economy of the consideration of the VPE can now be computed. If the parameters D_g and E_g from (3.1) are set to 0, *i.e.*, if the VPE is ignored, we face a convex quadratic programming (QP) problem. When the solution to the QP problem is inserted into the real objective function, we obtain an objective of 1 036 211 \$. Hence, the additional work for taking into account the VPE decreases the cost by 1.96%.

Figure 3.6 shows the evolution of the bounds γ^k and δ^k as k increases. We observe that, as predicted by Theorem 2.2, $\lim_{k\to\infty} \delta^k = 0$. However, we observe that even at the first iteration, δ^1 is smaller than γ^1, and because we do not change the solver prescribed tolerance γ, this remains true while k increases. In this sense, APLA may spend too much time trying to show that the first solution it finds $\boldsymbol{p}^1$ falls under the prescribed tolerance; APLA decreases $\delta^k := f(\boldsymbol{p}^k) - h^k(\boldsymbol{p}^k)$ by decreasing $f(\boldsymbol{p}^k)$ on the one hand and increasing $h^k(\boldsymbol{p}^k)$ on the other hand. In contrast, the heuristic stops looking at the whole search space Ω after a few iterations of APLA—typically one or two—and then it tries to improve the optimality gap only by decreasing the objective.

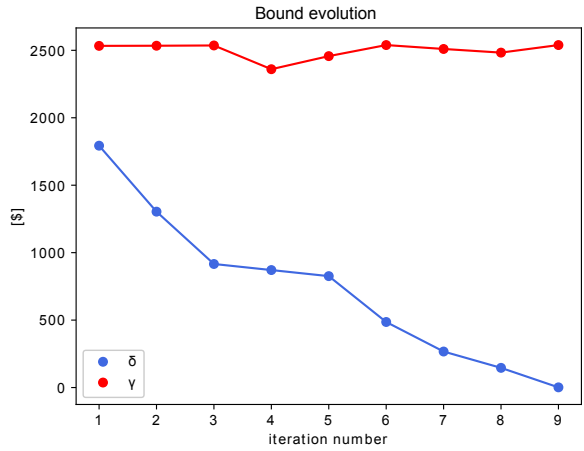


Fig. 3.6 Bound evolution of APLA's iterates (Algorithm 1) applied to the 10-unit problem over 24 hours from § 3.3.1. The bounds γ and δ are defined as in (2.13). The over-approximation error, upper bounded by $\epsilon^{\max} = 0.34\,\$$ (see (2.15)), is negligible here in comparison with the optimal objective of $1\,016\,276\,\$$.

Table 3.2 Optimal solution (with objective value of 1 016 276 $) of the problem from § 3.3.1. The production of U10 is always 55 MW.

Time step	U1 (MW)	U2	U3	U4	U5	U6	U7	U8	U9
1	150	143.56	185.533	60	122.867	122.45	129.59	47	20
2	150	223.56	229.399	60	73	122.45	129.59	47	20
3	150	303.56	297.399	60	73	122.45	129.59	47	20
4	226.624	316.799	305.67	60	122.867	122.45	129.59	47	20
5	226.624	396.799	299.67	60	122.867	122.45	129.59	47	20
6	303.248	396.799	321.179	60	172.733	122.45	129.59	47	20
7	303.248	396.799	297.399	107.913	222.6	122.45	129.59	47	20
8	379.873	396.799	297.399	149.87	172.733	122.45	129.59	52.285	20
9	456.497	396.799	297.399	191.246	172.733	122.45	129.59	82.285	20
10	456.497	396.799	303.399	241.246	222.6	160	129.59	85.312	21.556
11	456.497	396.799	297.399	291.246	222.6	160	129.59	85.312	51.556
12	456.497	460	307.698	291.246	222.6	160	129.59	85.312	52.057
13	456.497	396.799	302.899	241.246	222.6	160	129.59	85.312	22.057
14	456.497	396.799	294.373	191.246	172.733	122.45	129.59	85.312	20
15	379.873	396.799	303.397	168.624	122.867	122.85	129.59	77	20
16	303.248	393.821	291.266	118.624	73	122.45	129.59	47	20
17	303.248	313.821	297.399	68.624	122.867	122.45	129.59	47	20
18	379.873	393.821	297.399	60	122.867	122.45	129.59	47	20
19	456.497	396.799	297.399	70.219	172.733	122.45	129.59	55.312	20
20	456.497	460	332.782	120.219	222.6	160	129.59	85.312	50
21	456.497	389.533	297.399	110	222.6	158.069	129.59	85.312	20
22	379.873	309.533	297.399	60	172.733	122.45	129.59	81.422	20
23	303.248	229.533	237.89	60	122.867	122.45	129.59	51.422	20
24	226.624	222.266	178.22	60	122.867	122.45	129.59	47	20

3.3.2 Dynamic dispatch with DCOPF

In this section, the comparison is made between the APLA method (Algorithm 1) and the heuristic for solving the dynamic dispatch with DCOPF, (3.14). The benefit from taking the VPE into account is also estimated. Gurobi 8.1 [Gur18] has been used for solving the MIP problem associated with the surrogate problem. The optimization has been run on a computer with Intel-i7 3.6 GHz CPU and 16 GB of RAM. Two variants of the heuristic from § 3.2.2 are tested: H-local, which restricts the feasible region to a single segment (Fig. 3.3a), *i.e.*, $j' = j + 1$ from (3.20), and H-full, which restricts the feasible region to up to three segments (Fig. 3.3b). The impact of the VPE is highlighted by comparing the solution obtained *via* the aforementioned methods to the solution from a method which does not take the VPE into account.

The data and algorithm implementations are available on GitLab [Van19].

Data set creation

As no data set with transmission constraints and VPE is openly available in the literature, we analyze two IEEE test systems by introducing additional

generators that obey a VPE. We use IEEE test system data from the PSTCA and MatPower ([Chr99, ZMS]), while VPE generator data can be found in Table A.4. The VPE generators are added randomly in buses of the IEEE networks, represented in Figs. 3.7 and 3.10. To limit the uncertainty from this random selection and have a more robust analysis of the method, 100 algorithm trials are made with different network configurations. We further assess the scalability of the method by increasing the number of VPE units. In particular, we double the number of VPE units and introduce a 5% variation on the parameters of these units. We include this variation to avoid symmetry, which creates unnecessary computational complexity since it is rarely the case that different physical assets share an identical set of economic and technical parameters. We denote as *experiment* the simulation of 100 different *instances* of the problem.

IEEE cases with 10 added VPE units

IEEE 57-bus system The original case study with seven generators [Chr99] is extended by including ten additional generators that obey a valve point effect. The network topology is sketched in Fig. 3.7. The tolerance of the MIP solver and the maximal MIP solver time are set at 0.1% and 45 seconds when the surrogate problem is defined over the whole domain Ω and at 0.01% and 45 seconds when the problem is restricted to a subdomain $\Omega_{\tilde{p}^j}$. Figs. 3.8 and 3.9 (blue boxes) present a comparison between the execution time and the final optimality gap of the three methods: H-full, H-local and APLA. Whereas some instances of the initial APLA algorithm can be resolved in a few seconds, others require significantly more time due to timeouts. The timeout limit for APLA is set at a maximum of ten iterations of 45 seconds.

The two heuristics, H-full and H-local, perform much faster, and the majority of instances are executed within 20 seconds (H-local) and 100 seconds (H-full). This time improvement is at the cost of an increase in the optimality gap, with about half of the instances being solved at the target tolerance of 0.1%. The bottom magnification in Fig. 3.8 shows that the median execution time of H-local is approximately four times lower than the median of H-full, while the optimality gap, depicted in Fig. 3.9, is comparable. We discuss the implication of these observations in detail in section 3.3.3.

IEEE 118-bus system As in the previous case, we extended the 54-generator IEEE 118-bus system (Fig. 3.10) with the same ten generators obeying a

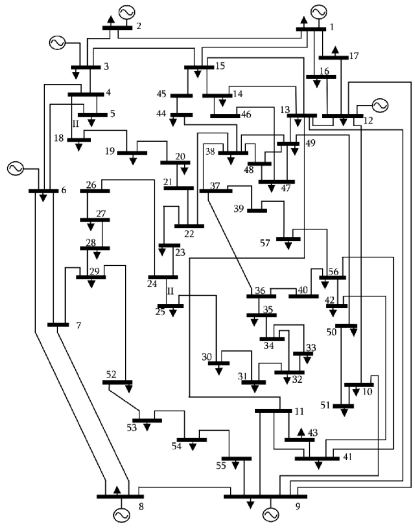


Fig. 3.7 Single-line diagram of the IEEE 57-bus system [AR15].

valve point effect [Chr99]. Tests are performed with the same methods and VPE parameters as in the IEEE 57-bus system. The comparison of the execution time, Fig. 3.8 (green boxes), shows that APLA performs faster in the IEEE 118 case, but it is still outperformed by the two heuristics.

However, the final optimality gap (Fig. 3.9) and final objective (Table 3.3) is slightly better with APLA. We discuss the implication of these results in detail in section 3.3.3.

IEEE cases with 20 added VPE units

As observed in the previous experiment, increasing the size of the network while keeping the number of VPE units constant does not pose significant computational complications. On the contrary, a larger network may result in a reduced run time. In this section, we increase the size of the problem by adding ten additional VPE units to the systems, bringing the number of generators obeying a VPE to 20.

IEEE 57-bus system We perform tests on an extension of the 57-bus case using the same methods and parameters as in the previous section but with 20 additional units. We report the results in Figs. 3.11 and 3.12. Compared to the previous tests, all three methods exhibit a similar relative increase in run

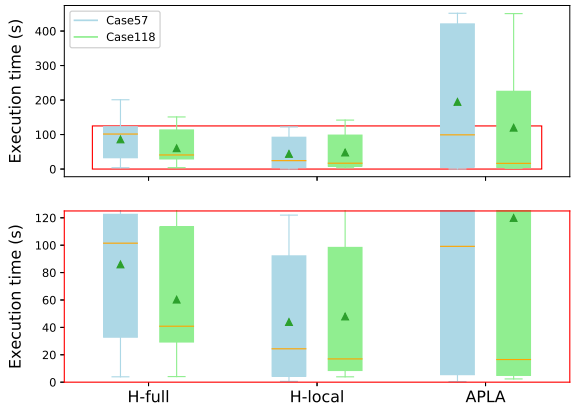


Fig. 3.8 Comparison between the execution time of the methods for 100 random instances with 10 additional VPE units. The targeted optimality gap is 0.1%. The orange lines and green triangles represent the medians and means, respectively. The bottom frame is a magnification of the upper one.

time and final optimality gap. However, the absolute values of the running time and optimality gap remain acceptable for real-time applications. The performance of the methods relative to each other is similar to that of the previous case, and the discussion is then analogous.

IEEE 118-bus system We perform tests on an extension of the 118-bus case using the same methods and parameters as in the previous section but with 20 additional units. In this case, the median final optimality gap doubles. However, the heuristics still outperform the execution time of APLA and the difference between both heuristics is reduced, with a slight dominance of H-local. Further comments on this increase are made in § 3.3.3.

Impact of the VPE

In order to assess the benefit of accounting for the VPE, each instance of both case studies is solved while ignoring the VPE by specifically removing the rectified sine term. As a result of this simplification, the objective function becomes quadratic and the problem becomes much simpler. The computation time is less than a second, but there is no guarantee for the optimal solution. The mean of the final objective is reported in Table 3.3,

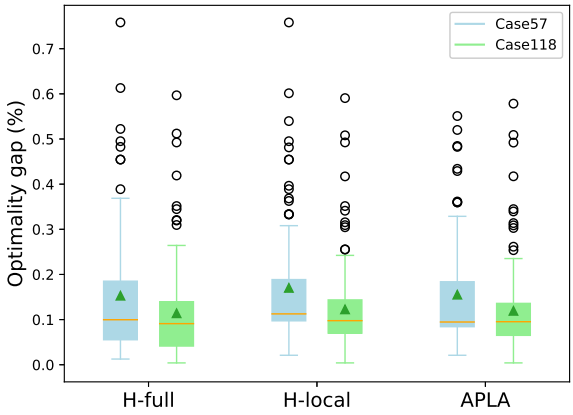


Fig. 3.9 Same as Fig. 3.8 for the optimality gap.

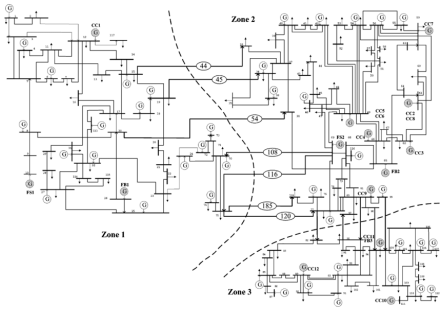


Fig. 3.10 Single-line diagram of the IEEE 118-bus system [LS05].

for this method (QP) as well as for APLA and both heuristics. The cost of neglecting the VPE can be readily estimated at 5.3% and 1% for the IEEE 57 and IEEE 118-bus system case with 10 additional VPE units, respectively. The second experiment, including 20 additional VPE units, shows larger discrepancies with 8% and 3.3% differences.

3.3.3 Discussion

The application of the APLA method to the dynamic dispatch, in particular the network-constrained case, is compromised by the significant number of instances that terminate due to timeouts. These timeouts push up the mean of the execution time. However, the heuristics based on the former method

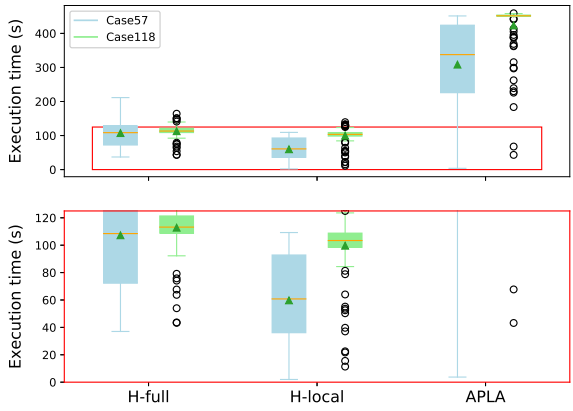


Fig. 3.11 Comparison of the execution time of the methods applied to 100 random instances of the IEEE test cases with 20 additional VPE units. The targeted optimality gap is set at 0.1%.

Table 3.3 Objective mean of each method.

	H-full	H-local	APLA	QP
Case57 (10 VPE-units)	621622	621634	**621547**	654535
Case118 (10 VPE-units)	2447475	2447540	**2447383**	2471132
Case57 (20 VPE-units)	587695	587713	**587669**	634653
Case118 (20 VPE-units)	2227536	2227592	**2227087**	2301490

can reach at least 0.3% optimality gap for 75% of the instances in reduced time, relatively to APLA.

Also, APLA reduces the optimality gap by increasing the lower bound, on the one hand, and by decreasing the optimal objective, on the other hand. On the contrary, the heuristic computes a single lower bound and then focus on the objective; for the same optimality gap, the solution to the heuristic is therefore lower. This phenomenon is highlighted by the final optimality gap depicted in Figs. 3.9 and 3.12 that is approximately 10% better for APLA and by the final APLA objective reported in Table 3.3 that is lower than the heuristics by only $\sim 0.01\%$ on average. Of the two heuristic variants considered, H-local, which restricts the local search to a smaller subset than H-full, performs faster.

The scalability of the method is assessed in two stages. Firstly, the

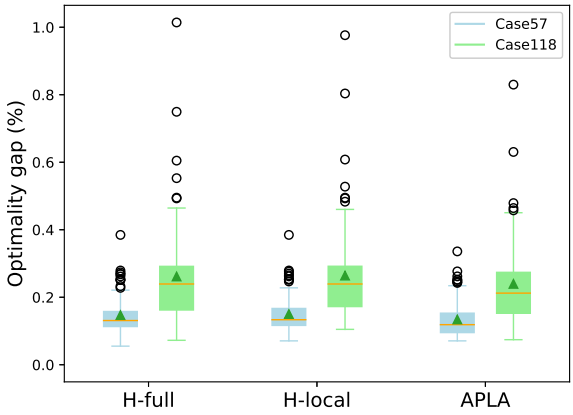


Fig. 3.12 Same as Fig. 3.11 for the optimality gap.

number of buses and generators is increased while keeping the number of VPE units constant, then this number is doubled. The first experiments (Figs. 3.8 and 3.9) have shown that the problem does not become more difficult when increasing the number of nodes and usual thermal units; the problem becomes even easier when the number of nodes is increased from 57 to 118. The second experiments (Figs. 3.11 and 3.12) show that this is not always true: with 20 additional VPE units, the 57-bus system becomes easier than the 118-bus system. In general, the execution time increases and the quality of the solution worsens when increasing the number of VPE units. However, despite this time and gap increase, the values of these quantities remain under two minutes and 0.25 %, respectively, for most instances.

As shown in the experiments, the core difficulty of the problem lies rather in the number of VPE units than in the total number of units. This motivates the application of the heuristics on real test cases where only a small fraction of the units obeys a valve point effect.

Finally, the cost of the valve point effect computed for both batches of experiments demonstrate the interplay of the network with the VPE. Ignoring the VPE costs around 2% for the dynamic dispatch with reserves, as shown in § 3.3.1. In § 3.3.2, we compute an additional cost of 5% on average for the 57-bus case and 1% for the 118-bus case. This difference between the 57 and 118-bus cases can be explained by the distinct level of VPE-units in the two test cases: more than half of the units obey a valve point effect for the modified IEEE 57 case, whereas this number is reduced

to fifteen percent in the second case. These percentages increase for the case with 20 additional VPE units. The average relative cost of neglecting the VPE units is estimated at 8% and 3% for the 57-bus and 118-bus cases, respectively.

3.4 Conclusion

In this chapter, a global method for the solution to the multi-period (or dynamic) economic dispatch problem has been studied. This method, the adaptive piecewise-linear approximation (APLA), suffers from a long execution time when applied to larger problems, *e.g.*, with network constraints. In our experiments, a ten-minute run time threshold (which is a reasonable upper bound on economic dispatch models in European market operations) is attained several times. This motivates the development of a faster method that still targets the global solution to this nonconvex problem and provides guarantees for the final solution.

The main contributions of this chapter are i) to demonstrate that VPE *matters* in instances of dynamic dispatch and ii) to develop local heuristics which accelerate the solving time.

The APLA method is tested on an instance of the dynamic dispatch with reserves and several instances of the IEEE 57-bus and IEEE 118-bus systems. Then, the heuristics are tested on the IEEE systems. We show that the heuristics produce solutions of comparable quality to those obtained by APLA, at a fraction of the computation time required by the latter. The additional cost of ignoring the VPE effect is computed for the dynamic dispatch with reserves and is measured at around two percent. It is also evaluated for the IEEE systems and is measured on average as up to eight percent.

Further work may include the extension of the method to the unit commitment problem and an analysis of the interactions of the valve point effect with the scheduling decisions. The impact of the position of the VPE-units in the network may also be of interest, as suggested by the various range of solutions obtained here for different network topologies. With the aim of improving the practicality of the model, some direct extensions could be contemplated. Firstly, the method presented here can be adapted to account for quadratic power losses, through a piecewise relaxation of the losses—this is the main topic of Chapter 4. Secondly, a more realistic power

flow model, such as a convex ACOPF, could also be used. The main change would be to resort to a mixed-integer second-order cone programming (MISOCP) solver instead of an MIP one.

4

Toward the consideration of quadratic power losses

In this thesis, we have—up to now—explored nonconvexities due to the objective, and every feasible set in the previously considered optimization problems is a polytope. This chapter is devoted to the analysis of nonconvexities that appear in the feasible set. We add a further layer of complexity on the dynamic dispatch with reserves (3.9) through the consideration of quadratic power losses. These losses naturally occur in power systems, as the production sites are often located far away from the consumption sites. Today, this phenomenon is increased with the penetration of renewable energy sources, as production is more distributed. Consideration of such losses makes the dispatch more accurate, but it adds to the difficulty of the problem. The methods from Chapter 3 cannot be directly applied to the dispatch with quadratic power losses because each surrogate problem is now an MINLP: a mixed-integer nonlinear programming problem.

The formal definition of the problem of interest in this chapter, namely the dynamic dispatch with quadratic power losses, is given hereafter in § 4.1.1. We tackle this problem, denoted as (P), through a three-step algorithm that is outlined in § 4.1.2. The first step is based on the method APLA—and the heuristic—from Chapters 2 and 3, applied on a relaxation of (P). The second step is a projection step that is further studied in a more abstract and general way in Part II. The last step, a local Riemannian subgra-

dient scheme (RSG), is the main contribution of this chapter—along with the three-step algorithm that combines everything together.

So to speak, this chapter is the pinnacle of the scientific contributions of this thesis. The three-step algorithm, coupled with one of the projection methods presented in Part II, is the most sophisticated of our algorithms that is designed to solve the most complex of all problems considered in this thesis. The interplay of all chapters in this algorithm is given in Fig. 1.2.

The organization of the present chapter is standard and similar to the previous one: Section 4.1 presents the main problem and additional subproblems that are used in the chapter. Section 4.2 describes the (three-step) method. This method is tested on Section 4.3 and discussed in Section 4.4.

This chapter is based on [VAP22b].

4.1 Problem formulation

This section is organized as follows. We first introduce the main problem that is considered in this chapter in § 4.1.1. The full method is then outlined in § 4.1.2. This method depends on several auxiliary optimization problems which are introduced in § 4.1.3: the surrogate problems, providing lower bounds to the main problem; the feasibility problems, which are used to find a feasible solution or to prove that no feasible solution exists; and the shift problems, which are used for obtaining a relaxation of the feasible set. Lastly, the topology of this feasible set is studied in § 4.1.4.

4.1.1 Main problem: economic dispatch with VPE and transmission losses

The main problem, denoted as (P), aims at minimizing fuel cost f, which is defined as the sum of the production cost of every generator unit f_g at each time step. The production of unit g at time step t is denoted as p_{gt}. A quadratic function is often used for modeling the fuel cost of a given unit. However, in the case of large gas units, this fails to model the inherent nonconvex characteristic of the problem when the VPE is taken into account, see Section 2.2. Using the same model as before for the fuel cost (2.5), we

have

$$f_g(p_{gt}) = \underbrace{A_g p_{gt}^2 + B_g p_{gt} + C_g}_{:=f_g^{\mathrm{Q}}(p_{gt})} + \underbrace{\left| D_g \sin E_g (p_{gt} - \underline{P}_g) \right|}_{:=f_g^{\mathrm{V}}(p_{gt})}, \tag{4.1}$$

with $\underline{P}_g$ the minimum power production of generator g and A_g, B_g, C_g, D_g, E_g some parameters.

The full objective reads

$$f(\boldsymbol{p}) = \sum_{t \in T} \sum_{g \in G} f_g(p_{gt}), \tag{4.2}$$

and a single term $f_g(p_{gt})$ is depicted in Figs. 2.6b and 2.10.

Most constraints considered in this chapter are the same as in § 3.1.1, with two differences. First, the reserves are formulated here without the addition of new variables, and they are defined for two different response times: one hour and ten minutes. This first point does not fundamentally make the problem more complex. The second difference comes from the consideration of the power losses, and this changes the nature of the problem: instead of a (convex) polytope, the feasible set is now the subset of a nonconvex quadratic hypersurface.

The constraints are the following:

- Power range limits

$$\underline{P}_g \leq p_{gt} \leq \overline{P}_g \qquad (g,t) \in G \times T, \tag{4.3}$$

  where $\underline{P}_g$ and $\overline{P}_g$ are the minimum and maximum power output of unit g.

- Ramp rate restrictions

$$\underline{R}_g \leq p_{gt} - p_{g(t-1)} \leq \overline{R}_g \qquad (g,t) \in G \times T, \tag{4.4}$$

  where $\underline{R}_g$ and $\overline{R}_g$ are the ramp-down and ramp-up rates of unit g, respectively.

- Power balance

$$\sum_{g \in G} p_{gt} = P_t^{\mathrm{D}} + \underbrace{\boldsymbol{p}_t^{\mathsf{T}} \boldsymbol{B} \boldsymbol{p}_t + \boldsymbol{B}_0^{\mathsf{T}} \boldsymbol{p}_t + B_{00}}_{:=p_t^{\mathrm{loss}}} \qquad t \in T, \tag{4.5}$$

where P_t^{D} is the demand and p_t^{loss} is an approximation of the transmission losses in period t. This approximation is obtained using *Kron*'s formula [Saa99] for a given matrix $\boldsymbol{B}$, vector $\boldsymbol{B}_0$, and parameter B_{00}. The matrix $\boldsymbol{B}$, which contains the *loss coefficients*, is symmetric because it is obtained as the real part of a Hermitian matrix. However, this matrix is not necessarily positive definite [Saa99, § 7.7], *e.g.*, the matrix is indefinite for the 10-unit test case in § 4.3.2. These coefficients are discussed in § 4.1.4. The feasible set induced by this constraint is a quadratic hypersurface, also referred to as *quadric*, which is the focus of Part II.

- Spinning reserve constraints

  The reserve requirements are modeled as in [NAAA13] for all $t \in T$:

$$\left(\Delta_t^{(1)} = \sum_{g \in G} \overline{P}_g - (P_t^{\mathrm{D}} + p_t^{\mathrm{loss}} + P_t^{\mathrm{S}})\right) \geq 0, \tag{4.6}$$

$$\left(\Delta_t^{(2)} = \sum_{g \in G} \min\left(\overline{P}_g - p_{gt}, \overline{R}_g\right) - P_t^{\mathrm{S}}\right) \geq 0, \tag{4.7}$$

$$\left(\Delta_t^{(3)} = \sum_{g \in G} \min\left(\overline{P}_g - p_{gt}, \frac{\overline{R}_g}{6}\right) - \frac{P_t^{\mathrm{S}}}{6}\right) \geq 0, \tag{4.8}$$

  for a given reserve requirement P_t^{S}.

  Equations (4.6) and (4.7) model the requirement for the spinning reserves that can respond within one hour. Eq. (4.8) models the requirement for the spinning reserves that can respond within 10 minutes. If the losses are neglected, *i.e.*, $p_t^{\mathrm{loss}} = 0$, then (4.6) does not depend on decision variables and is therefore simply a test on the feasibility of the problem. This feasibility test is used in § 4.3.2 to show that a given problem is infeasible.

Taking all these constraints into account, the optimization problem at hand is defined in (P).

Dynamic economic dispatch with quadratic power losses

$$\begin{aligned} \min_{\boldsymbol{p}} \quad & f(\boldsymbol{p}) = \sum_{t \in T} \sum_{g \in G} f_g(p_{gt}) \\ \text{subject to} \quad & (4.3) \text{ - } (4.8) \end{aligned} \tag{P}$$

This is a nonsmooth and nonconvex continuous optimization problem. The feasible set is the intersection of a polytope—Eqs. (4.3), (4.4) and (4.6) to (4.8)—and the Cartesian product of quadratic hypersurfaces—(4.5)—which is further described in § 4.1.4.

4.1.2 Outline of the method

Before introducing the other optimization problems that will be used in the remainder of the chapter, we first outline the full method. This method, denoted as APLA-RSG, is depicted in Fig. 4.1 and consists of the following steps:

1. Obtaining a lower bound and an initial (infeasible) candidate through the solution of a relaxation of (P);
2. Projecting this candidate onto the feasible set;
3. Improving the projected candidate with a local search.

The first step is based on the adaptive piecewise-linear approximation (APLA) algorithm from Chapters 2 and 3 and a relaxation of the feasible set. It requires solving three different optimization problems: (S), $(\text{F})_t$, and $(\text{Shift})_t$. The second step solves the feasibility problem (F). These problems are introduced in the next section: § 4.1.3. The last step is a Riemannian subgradient descent scheme (RSG) that depends on a quadratic subproblem, denoted as (Sub) and defined in § 4.2.1.

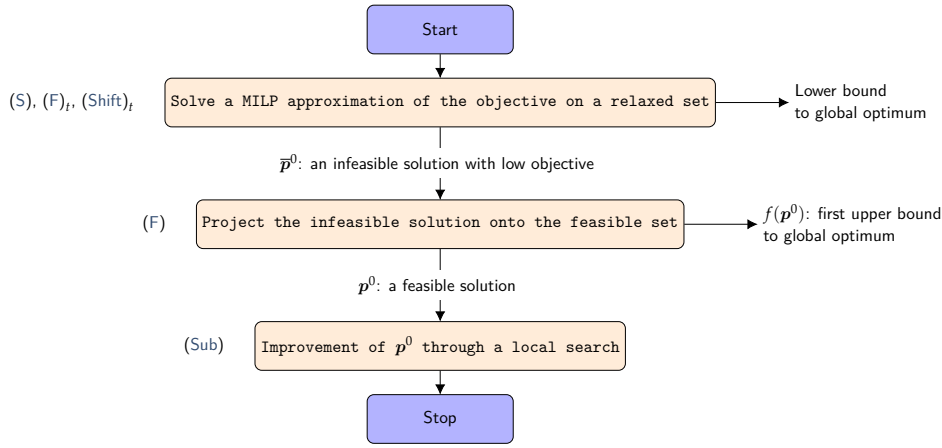


Fig. 4.1 Block diagram of the method APLA-RSG for solving (P).

4.1.3 Auxiliary optimization problems

Surrogate problem

One way of coping with the nonlinearities of the objective is to approximate it as a piecewise-linear function and then solve this *surrogate problem*, which can be formulated as a mixed-integer programming (MIP) problem, see § 2.3.1.

The surrogate problem (S) aims at i) finding a good initial point for a local search and ii) providing a lower bound on (P). This is achieved through an approximation of the objective—similar to the approximation from § 2.3.1 and 3.1.4—and a relaxation of the quadratic constraint (4.5). This relaxation is given in Algorithms 4 and 5, depending on whether $\boldsymbol{B}$ is positive definite.

Recall that the piecewise-linearization of a given function f is entirely defined by the set of knots, which refer to the points where the pieces of the cost function meet. Let $\boldsymbol{X}_{gt} := \left(X_{gt1}, \ldots, X_{gtn_{gt}^{\text{knot}}} \right)$ be the set of knots of unit g at time t. We can then approximate the cost function as follows:

$$h_{gt}(p_{gt}) := \hat{f}_g(p_{gt}; \boldsymbol{X}_{gt}).$$

In this expression, $\hat{f}(\cdot; \boldsymbol{X})$ stands for the piecewise-linear interpolation of a

function f given the knots $\boldsymbol{X}$. These approximations of the fuel costs are then aggregated as in equation (3.16), in order to form the total surrogate objective h,

$$h(\boldsymbol{p}) = \sum_{t \in T} \sum_{g \in G} h_{gt}(p_{gt}). \tag{4.9}$$

This *surrogate objective* is illustrated in, *e.g.*, Figs. 2.6b and 2.10. Equipped with this surrogate objective, we can define the *surrogate problem* (S).

Surrogate problem of (P)

$$\begin{aligned} \min_{\boldsymbol{p}} \quad & h(\boldsymbol{p}) = \sum_{t \in T} \sum_{g \in G} h_{gt}(p_{gt}) \\ \text{subject to} \quad & (4.3) - (4.4), (4.5)_{\mathrm{R}}, (4.6) - (4.8) \end{aligned} \tag{S}$$

Here, $(4.5)_{\mathrm{R}}$ stands for the relaxation of constraint (4.5) and is defined in § 4.1.3. Two relaxations are considered, a linear relaxation and a (convex) quadratic one, depending on whether $\boldsymbol{B}$ (from (4.5)) is positive definite or not. The case of a semidefinite $\boldsymbol{B}$ is not considered: this matrix is assumed to be invertible, as detailed in § 4.1.4.

Feasibility problems

The feasibility problem, which we call (F), focuses on converting an infeasible candidate $\overline{\boldsymbol{p}}^0$ into a feasible point. More specifically, this problem will be used to map the solution of (S) onto the feasible set of (P).

We define the feasibility objective f^{feas} as

$$f^{\mathrm{feas}}(\boldsymbol{p}; \overline{\boldsymbol{p}}^0, \lambda_{\mathrm{N}}, \lambda_{\mathrm{Q}}) = \lambda_{\mathrm{N}} \|\boldsymbol{p} - \overline{\boldsymbol{p}}^0\|_2^2 + \lambda_{\mathrm{Q}} \sum_{t \in T} \sum_{g \in G} f_g^{\mathrm{Q}}(p_{gt}), \tag{4.10}$$

for given parameters λ_{N}, $\lambda_{\mathrm{Q}} \in \mathbb{R}_{\geq 0}$ and f_g^{Q} defined as in (4.1). We survey these parameters hereafter.

Let us define the feasibility problem.

Feasibility problem

$$\begin{aligned} \min_{\boldsymbol{p}} \quad & (4.10) \\ \text{subject to} \quad & (4.3) \text{ - } (4.8) \end{aligned} \tag{F}$$

This problem depends on the parameters λ_{N} and λ_{Q}. When $\lambda_{\mathrm{Q}} = 0$, (F)

becomes a projection onto the feasible set. This case is studied in the second part of this thesis: Part II. If no initial guess $\overline{\boldsymbol{p}}^0$ is available, λ_{N} is set to zero and the problem is a quadratically constrained quadratic program (QCQP). Finally, if both parameters are set to zero, (F) becomes a usual feasibility problem without any objective. The problem (F) is easier than the main problem (P): on one hand because the objective is convex and on the other hand because the primary goal is to obtain a feasible solution, hence (F) will not be solved to optimality, saving thereby computational resources.

The fixed-time feasibility problem is also considered. It is similar to (F), except that the problem is decoupled with respect to the time steps.

t-feasibility problem

$$\begin{aligned} \min_{\boldsymbol{p}_t} \quad & \lambda_{\mathrm{Q}} \sum_{g \in G} f_g^{\mathrm{Q}}(p_{gt}) \\ \text{subject to} \quad & (4.3)_t, (4.5)_t - (4.8)_t \end{aligned} \tag{F}_t$$

Here, $(\,\cdot\,)_t$ indicates that the constraint must only hold for the given time step t. We drop constraint (4.4) because it depends on two consecutive time steps.

Relaxation problem

The goal of the relaxation problem is to compute a convex relaxation of the constraint (4.5), which we write as $(4.5)_{\mathrm{R}}$. Two different cases are considered: either the coefficient matrix $\boldsymbol{B}$ is positive definite or there are at least two eigenvalues of opposite sign, in which case the matrix is indefinite. This relaxation problem is decoupled with respect to the time index t. Indeed, if $(4.5)_{t,\mathrm{R}}$ is the relaxation of $(4.5)_t$, then taking the Cartesian product of every time step yields a relaxation of (4.5). Hence, the discussion is made here for a given time step $t \in T$, and the full relaxation is obtained as

$$\bigtimes_{t \in T} (4.5)_{t,\mathrm{R}}. \tag{4.5}_{\mathrm{R}}$$

Case I: B is positive definite In this case, the feasible set generated by constraint $(4.5)_t$ is the hypersurface of an ellipsoid, see § 4.1.4. However, the power ranges $(4.3)_t$ restrict the feasible set to a box (or hyperrectangle) that is, in practice, much smaller than the ellipsoid because the losses are small. This explains why a linear approximation (4.5) is often used, as in [PJY18, PJCY20]. To obtain a relaxation, the set induced by $(4.5)_{t,\mathrm{R}}$ should

include the set induced by $(4.5)_t$. This condition is fulfilled if $(4.5)_{t,\mathrm{R}}$ is the intersection of the interior of the ellipsoid, the interior of the power ranges, and the half-space induced by some secant plane. The secant plane is chosen such that its intersection with the feasible set of equation (P) is empty—in Figure 4.2, π_0 and π_1 are valid planes, while π_2 is not. Ideally, the secant plane should be chosen so as to minimize the relaxed set. However, computing the optimal secant plane can be complicated. For example, in the specific case where there are exactly $|G|$ points in the box that intersect the ellipsoid, the optimal secant plane (π_0 in Figure 4.2) is the one defined by the $|G|$ points. Unfortunately, getting these $|G|$ points of intersection is challenging. The simple enumeration of the vertices of the hyperrectangle becomes intractable for a few generators $|G|$.

This relaxation may be exact—meaning that the optimal solution of the relaxed problem is feasible for the unrelaxed one—or inexact, depending on the position of the box. This type of behavior has been studied in [Low14] for nonconvex network constraints.

The procedure to obtain a relaxed plane at a given time step t is the following: First, a feasible point for this time step, denoted as $\tilde{\boldsymbol{p}}_t^0$, is computed by solving $(\mathrm{F})_t$. Then, the slope of the plane is obtained as the tangent plane of the ellipsoid in $\tilde{\boldsymbol{p}}_t^0$. Finally, the plane is shifted toward the interior of the ellipsoid in the perpendicular direction $\hat{\boldsymbol{n}}_t$. The value of the shift is given with the following optimization problem.

(Inward) Shift subproblem

$$S_t^* := \max_{\bar{\boldsymbol{p}}_t \in \mathcal{X}_t} \hat{\boldsymbol{n}}_t \cdot (\bar{\boldsymbol{p}}_t - \tilde{\boldsymbol{p}}_t^0) \qquad (\mathrm{Shift})_t$$

Here $\mathcal{X}_t$ is the feasible set of $(\mathrm{F})_t$. This procedure is illustrated in Figure 4.3a, which is a magnification of Figure 4.2 around the available power ranges. The explicit procedure is presented in Algorithm 4.

For a given time t, the relaxed balance constraint reads

$$\begin{aligned} &\Big(\sum_{g \in G} p_{gt} \geq P_t^{\mathrm{D}} + \boldsymbol{p}_t{}^{\intercal} \boldsymbol{B} \boldsymbol{p}_t + \boldsymbol{B}_0{}^{\intercal} \boldsymbol{p}_t + B_{00} \Big) \\ &\bigcap \Big(0 \geq \boldsymbol{p}_t \cdot \boldsymbol{n}_t - (\tilde{\boldsymbol{p}}_0^t + \hat{\boldsymbol{n}}_t S_t^*) \cdot \boldsymbol{n}_t \Big). \end{aligned} \qquad (4.5)_{t,\mathrm{R}}$$

This convex relaxation is motivated by the size of the quadric—much larger than the admissible ranges. Thus, the linear relaxation is almost on the

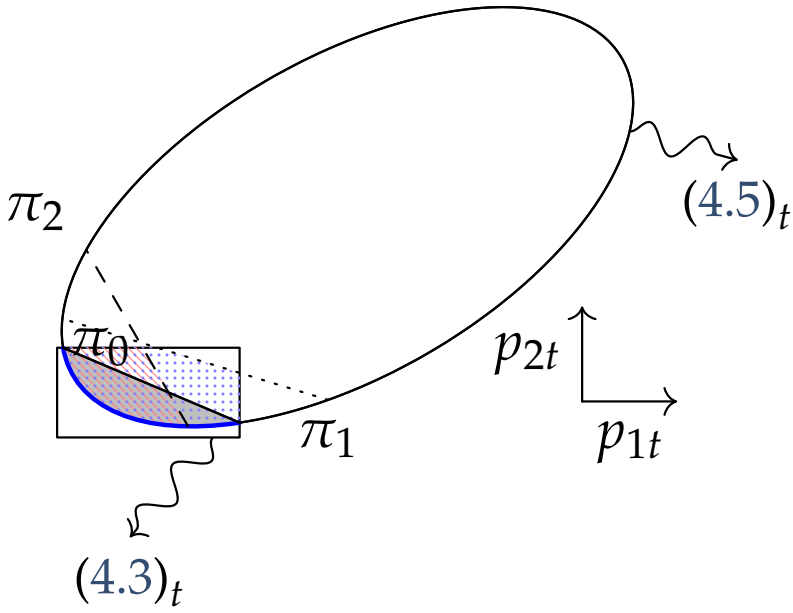
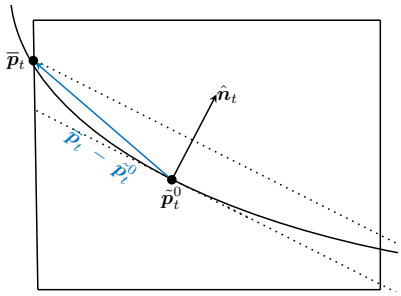
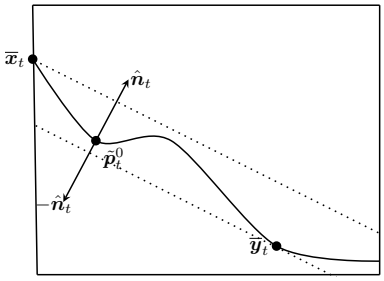


Fig. 4.2 Illustration of the relaxation induced by the interior of the ellipsoid, the power ranges, and several secant planes. The area induced by π_0 (gray fill) and π_1 (blue dots) are valid relaxations, while the one induced by π_2 (red hashed lines) is not a valid relaxation because some feasible points are cut off. The relaxation induced by π_0 is optimal and corresponds to the convex hull of the feasible set (blue line).



(a) Ellipsoid case: 1-plane relaxation.



(b) Hyperboloid case: 2-plane relaxation.

Fig. 4.3 Procedure to obtain the relaxation.

quadratic hypersurface. We illustrate this phenomenon in Figure 4.4 for a 3-unit problem at a given time step t.

Algorithm 4 Procedure to obtain relaxation at given time step t

Require: $\boldsymbol{B}$ positive definite
$\tilde{\boldsymbol{p}}_t^0 \leftarrow$ solution of $(\mathrm{F})_t$
$\boldsymbol{n}_t \leftarrow \boldsymbol{B}\tilde{\boldsymbol{p}}_t^0 + \boldsymbol{B}_0$
$\hat{\boldsymbol{n}}_t \leftarrow \boldsymbol{n}_t / \|\boldsymbol{n}_t\|_2$
$S_t^* \leftarrow \max_{\bar{\boldsymbol{p}}_t \in \mathcal{X}_t} \hat{\boldsymbol{n}}_t \cdot (\bar{\boldsymbol{p}}_t - \tilde{\boldsymbol{p}}_t^0)$
$(4.5)_{t,\mathrm{R}} \leftarrow \left(0 \geq \boldsymbol{p}_t \cdot \boldsymbol{n}_t - (\tilde{\boldsymbol{p}}_0^t + \hat{\boldsymbol{n}}_t S_t^*) \cdot \boldsymbol{n}_t\right) \cap \left(\sum_{g \in G} p_{gt} \geq P_t^{\mathrm{D}} + p_t^{\mathrm{loss}}\right)$
return $(4.5)_{t,\mathrm{R}}$

Case II: B is indefinite In this case, there are at least two eigenvalues of $\boldsymbol{B}$ of opposite sign. Therefore, the set defined by the power balance $(4.5)_t$ is no longer the boundary of a convex set. Figure 4.6 illustrates an example of this case. Figures 4.6b and 4.6c show that a single plane will not be enough to construct the relaxation: Fig. 4.6b prompts the use of an interior relaxation plane as in case I, whereas Fig. 4.6c demonstrates that en exterior plane should also be used. To tackle this issue, we solve $(\mathrm{Shift})_t$ for both directions $\hat{\boldsymbol{n}}_t$ and $-\hat{\boldsymbol{n}}_t$. The whole procedure is explicitly given in Algorithm 5 and depicted in Fig. 4.3b.

As further detailed in remark 4.2, an indefinite matrix $\boldsymbol{B}$ is possible, while still modelling nonnegative power losses.

REMARK 4.1 (Comparison of the two relaxations). In case I, the relaxation is a (convex) quadratic one. In case II, the relaxation is linear. Nonetheless, it

Fig. 4.4 Illustration of the relative size of the power ranges in comparison with the quadric (ellipsoid) for a 3-unit problem at a given time step t. The power balance $(4.5)_t$ is the surface of the blue ellipsoid. The set of admissible power ranges $(4.3)_t$ is the interior of the red box (right frame). This box is also depicted in the left frame as a (nearly indistinguishable) red dot at the bottom of the image. The right frame is a magnification around the admissible power ranges. The red points, in the right figure, are the vertices of the box.

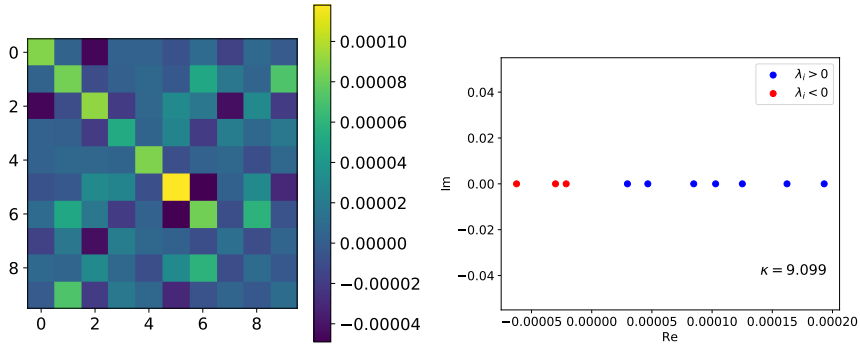
is possible to also obtain a linear relaxation for positive definite $\boldsymbol{B}$. Indeed, the application of the second relaxation to the first case also yield a (linear) relaxation but of lower quality. ■

REMARK 4.2 (On the origin of the $\boldsymbol{B}$ matrix). The loss coefficients are constructed from a solution of the power flow equation [Saa99, § 7.7]. Hence, they are a function of this (initial) operating state and can be considered as constant only if the scheduling operation is not too far away from this initial operating state. This condition explains why the matrix $\boldsymbol{B}$ can be indefinite: though negative losses $(\boldsymbol{p}_t{}^\intercal \boldsymbol{B}\boldsymbol{p}_t + \boldsymbol{B}_0{}^\intercal \boldsymbol{p}_t + B_{00})$ could be obtained by choosing some $\boldsymbol{p}_t$ such that the quadratic term dominates (while being negative), such a solution would not be close enough to the initial operating state. Throughout this thesis, we assume that the power ranges sufficiently restrict the feasible set such that the loss coefficients can be considered as constant.

■

Comparison of the optimization problems of this chapter

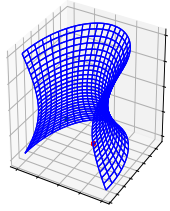
The characterization of each optimization problem is presented in Table 4.1. The last two problems are decoupled with respect to the time step, reducing significantly their size. They are considered as *easy*, relatively to the first



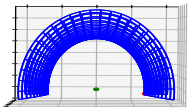
(a) Matrix entries.

(b) Matrix spectrum with a condition number $\kappa = 9.099$.

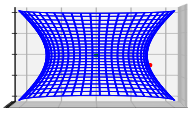
Fig. 4.5 Visualization of $\boldsymbol{B}$ for the 10-unit case from § 4.3.2.



(a) Full hyperboloid.



(b) OXZ view.



(c) OXY view.

Fig. 4.6 Illustration of the relative size of the power ranges in comparison with the quadric for a 3-unit problem at a time step t. In this example, the Kron matrix is *not* positive definite: two eigenvalues are positive, and the last one is negative. The quadric is a one-sheet hyperboloid. The set of the admissible power ranges $(4.3)_t$ is the interior of the red cube—so small that it appears as a red point. The power balance $(4.5)_t$ is the surface of the blue hyperboloid. Figures 4.6b and 4.6c show different views. The green point is the center of the quadric.

Algorithm 5 Procedure to obtain relaxation at given time step t for a nonconvex quadric.

$\tilde{\boldsymbol{p}}_t^0 \leftarrow$ solution of $(\mathrm{F})_t$
$\boldsymbol{n}_t \leftarrow \boldsymbol{B}\tilde{\boldsymbol{p}}_t^0 + \boldsymbol{B}_0$
$\hat{\boldsymbol{n}}_t \leftarrow \boldsymbol{n}_t / \|\boldsymbol{n}_t\|_2$
$S_t^{*,\text{int}} \leftarrow \max_{\bar{\boldsymbol{x}}_t \in \mathcal{X}_t} \hat{\boldsymbol{n}}_t \cdot (\bar{\boldsymbol{x}}_t - \tilde{\boldsymbol{p}}_t^0)$
$S_t^{*,\text{ext}} \leftarrow \max_{\bar{\boldsymbol{y}}_t \in \mathcal{X}_t} \; -\hat{\boldsymbol{n}}_t \cdot (\bar{\boldsymbol{y}}_t - \tilde{\boldsymbol{p}}_t^0)$
$(4.5)_{t,\mathrm{R}} \leftarrow \left(0 \geq \boldsymbol{p}_t \cdot \boldsymbol{n}_t - (\tilde{\boldsymbol{p}}_t^0 + \hat{\boldsymbol{n}}_t S_t^{*,\text{int}}) \cdot \boldsymbol{n}_t\right) \bigcup \left(0 \leq \boldsymbol{p}_t \cdot \boldsymbol{n}_t - (\tilde{\boldsymbol{p}}_t^0 + \hat{\boldsymbol{n}}_t S_t^{*,\text{ext}}) \cdot \boldsymbol{n}_t\right)$
return $(4.5)_{t,\mathrm{R}}$

three problems. In the test cases studied in Section 4.3, these two decoupled problems can be solved to optimality in less than a second. Among the three larger problems, (P) is the most difficult; it is a large nonlinear programming (NLP) problem. Problem (F) is arguably easier than (S): the reason is that each feasible solution of (F) is acceptable, since the goal is to find a feasible solution. On the other hand, (S) is a true optimization problem in the sense that we are interested in the lowest possible objective and especially a high lower bound.

Table 4.1 Comparison of the optimization problems of Chapter 4.

	(P)	(S)	(F)	$(\mathrm{F})_t$	$(\mathrm{Shift})_t$
Classification	NLP	MIQP	QCQP	QCQP	QCLP
convexity	nonconvex	nonconvex	nonconvex	nonconvex	nonconvex
objective	nonconvex, nonsmooth	piecewise-linear	quadratic	quadratic	linear
Feasible set	nonconvex	Convex[1]	nonconvex	nonconvex	nonconvex
Problem size	$\lvert T\rvert\lvert G\rvert$	$\lvert T\rvert\lvert G\rvert$	$\lvert T\rvert\lvert G\rvert$	$\lvert G\rvert$	$\lvert G\rvert$

4.1.4 Topology of the feasible set

Let us now study the feasible set defined by (4.5). To be specific, we define the quadratic surface, or quadric, and express (4.5) as a Cartesian product of quadrics. Then, we characterize this quadric as a quadric with a middle point. This middle point will be used in § 4.2.1 to compute the retraction mapping and in Section 5.7 for approximating the projection.

[1]The initial feasible set is convex. Nevertheless, the modeling of the piecewise-linear objective is made through integer variables which makes the feasible set inherently nonconvex, see [HV19] for more details about the modeling of nonconvex functions as piecewise-linear functions.

Characterization of the hypersurfaces [AHK+08] Let V be a vector space on the field $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$. A relation

$$V \to \mathbb{K} \colon \boldsymbol{x} \mapsto \Psi(\boldsymbol{x}) = \rho(\boldsymbol{x}) + 2\phi(\boldsymbol{x}) + a$$

with a quadratic form ρ, a linear form ϕ, and a constant $a \in \mathbb{R}$, is called a *quadratic function*.

Let $\Psi \colon \mathbb{R}^n \to \mathbb{R}$ be a nonzero quadratic function, then its zero set

$$\mathcal{Q} = \{\boldsymbol{x} \in \mathbb{R}^n \,|\, \Psi(\boldsymbol{x}) = 0\}$$

is a *quadric* of $\mathbb{R}^n$.

For a given time step t, $(4.5)_t$ can be written as a quadric $\mathcal{Q}_t$ by choosing the relation

$$\mathbb{R}^n \to \mathbb{R} \colon \boldsymbol{p}_t \mapsto \Psi_t(\boldsymbol{p}_t) = \boldsymbol{p}_t{}^{\mathsf{T}} \boldsymbol{B} \boldsymbol{p}_t + 2\boldsymbol{b}^{\mathsf{T}} \boldsymbol{p}_t + c_t, \tag{4.11}$$

with $n = |G|$, $\boldsymbol{b} = (\boldsymbol{B}_0 - \mathbb{1})/2$, and $c_t = B_{00} + P_t^{\mathrm{D}} > 0$. Since this constraint holds for every time step, this yields the following set:

$$\mathcal{Q}^{\mathrm{tot}} := \mathcal{Q}_1 \times \mathcal{Q}_2 \times \ldots \times \mathcal{Q}_{|T|}. \tag{4.12}$$

Let r be the rank of $\boldsymbol{B}$. If we assume that $\boldsymbol{B}$ is invertible, as it is the case in all the instances we found in the literature, then $r = n$. Following the classification of [AHK+08], the quadric is said to be of *type II* (Mittelpunktsquadrik or quadric with a middle point). Indeed, let us compute the rank of

$$\bar{\boldsymbol{B}}_t = \begin{pmatrix} \boldsymbol{B} & \boldsymbol{b} \\ \boldsymbol{b}^{\mathsf{T}} & c_t \end{pmatrix}. \tag{4.13}$$

Since $\boldsymbol{B}$ is invertible, the *Guttman rank additivity formula* yields [Zha05]

$$\operatorname{rank} \bar{\boldsymbol{B}}_t = \operatorname{rank} \boldsymbol{B} + \operatorname{rank}(c_t - \boldsymbol{b}^{\mathsf{T}} \boldsymbol{B}^{-1} \boldsymbol{b}). \tag{4.14}$$

In general, $c_t \neq \boldsymbol{b}^{\mathsf{T}} \boldsymbol{B}^{-1} \boldsymbol{b}$. Thus, it follows that $\operatorname{rank} \bar{\boldsymbol{B}}_t > r = \operatorname{rank} \boldsymbol{B}$ and henceforth $\mathcal{Q}_t$ is a type-II quadric. In Chapter 5, we equivalently classify this quadric as a nonempty and non-cylindrical central quadric [OSG20, Theorem 3.1.1].

When all the eigenvalues of the quadratic form are positive, the non-degenerate type-II quadric is an ellipsoid, illustrated in Fig. 4.4. Otherwise,

it is an elliptic hyperboloid, illustrated in Fig. 4.6.[2] A feature of the type-II quadric is the existence of a center (or middle point) computed as

$$\boldsymbol{d} = -\boldsymbol{B}^{-1}\boldsymbol{b}. \tag{4.15}$$

This center will be used in § 4.2.2 to compute the retraction on the manifold defined by the quadric.

4.2 Methods

In this section, we explain how to combine all the elements developed in Section 4.1 to solve (P). First, we describe how to derive a lower bound of the problem. Next, we show how to obtain an upper bound that is computed as the objective value of a feasible solution. Then, we improve this upper bound using a Riemannian gradient descent scheme. Finally, we bundle everything in a single algorithm.

4.2.1 Deriving a lower bound

As in § 2.3.1 and 3.1.4, the lower bound is obtained through an approximation of the objective *via* piecewise-linearization. However, this is not sufficient here, as the feasible set is the subset of a quadric. Hence, this nonconvex set is relaxed using the solution of $(\text{Shift})_t$ which requires for each time step t a point $\tilde{\boldsymbol{p}}_t^0$ feasible for $(\text{F})_t$.

The goal here is to obtain a lower bound but also a candidate which is globally efficient, meaning that its objective is close to the global optimum. In general, this candidate will not be feasible due to the relaxation of the feasible set, but we expect it to be sufficiently close to the feasible set such that when we project it back to this set, it remains close to the global optimum.

Feasible set relaxation A $t-$feasible point $\tilde{\boldsymbol{p}}_t^0$ is readily obtained for each t with Algorithm 6. This point is not globally feasible. Hence, $f(\tilde{\boldsymbol{p}}^0)$ is *not* an upper bound on the global solution. This point is simply a starting point for Algorithms 4 and 5 depending on whether $\boldsymbol{B}$ is positive definite or not.

[2] We consider that the problem is feasible. Hence, we do not study the case where all eigenvalues of $\boldsymbol{B}$ are negative.

Algorithm 6 Find $\tilde{\boldsymbol{p}}^0$ feasible for each time step t

for $t \in T$ **do**
  $\tilde{\boldsymbol{p}}_t^0 \leftarrow \arg\min (\mathrm{F})_t$
end for
return $\tilde{\boldsymbol{p}}^0$

Solution of the surrogate problem The lower bound can be obtained *via* the adaptive piecewise-linearization algorithm (APLA: Algorithm 1 from Chapter 2). However, this method suffers from a long execution time. In practice, we are not interested in spending too much time to obtain the lower bound. Therefore, in the numerical experiments of Section 4.3, we rather use the heuristic based on APLA that is described in Chapter 3. In order to simplify the discussion, the description that we provide in the present chapter is based on the APLA method.

For the sake of making this chapter self-contained, let us summarize the APLA method. Firstly, a set of knots that define the piecewise-linear approximation is defined. Then, the (first) surrogate problem $(S)^1$ defined with the (first) set of knots $\boldsymbol{X}^1$ is solved using an MIP solver. If we neglect the relaxation of the feasible set, the solution returned by the solver is non-optimal because i) optimality is guaranteed up to a given tolerance and ii) the surrogate objective approximates the real objective. To remedy the latter point, the approximation is refined around the returned solution; this adaptive refinement results in a lower number of integer variables than a global refinement that consists in doubling the number of linear pieces. It is proven in Theorem 2.2 that this method converges up to the solver tolerance, *i.e.*, the second cause of sub-optimality vanishes as the number of APLA iterations tends to infinity. The method is outlined in the dotted frame of Fig. 4.7. The surrogate objective and the knots are depicted in Figs. 2.6b and 2.10.

As a matter of fact, this method cannot be directly applied to our problem because of the nonlinear constraint (4.5). This explains the need for the relaxation developed in § 4.1.3.

4.2.2 Deriving an upper bound: Riemannian subgradient scheme

A simple and direct method for obtaining a feasible point, *i.e.*, a first upper bound on the global optimum, is to project the candidate obtained at the end of the procedure depicted in Fig. 4.7 onto the feasible set of (P). However, the projection is generally not a global optimum nor even a local optimum,

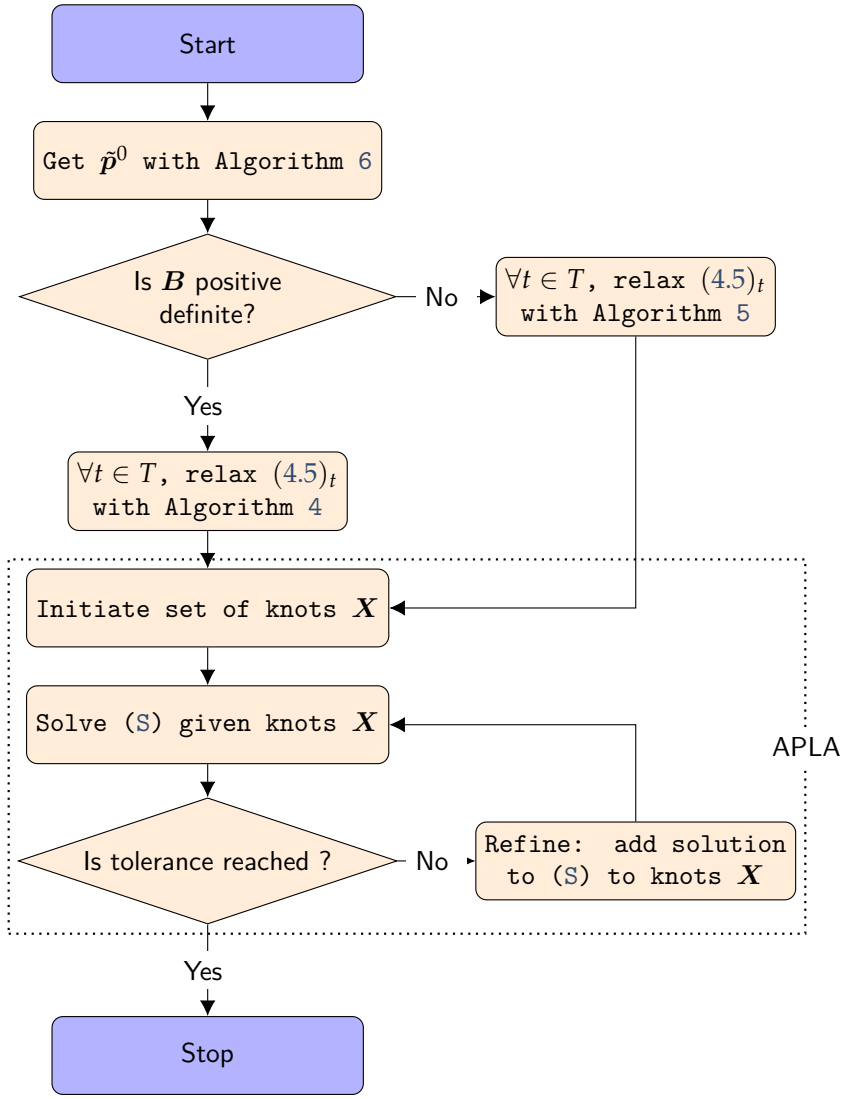


Fig. 4.7 Flowchart of the method for obtaining a lower bound along with a first (infeasible) candidate with low objective.

and it is worth trying to improve the obtained feasible solution through a local search. In [PJY18], the authors use an interior-point method as a local solver. Nevertheless, this method mildly improves the solution, and it relies on barrier parameters that are difficult to choose *a priori*. In this section, we propose to adapt the Riemannian gradient descent described in [BSBA13] to account for reserves and multiple time steps.

The Riemannian subgradient descent can be described as a classical line-search scheme,

$$\boldsymbol{p}^{k+1} = \boldsymbol{p}^k + \alpha^k \boldsymbol{v}^k, \tag{4.16}$$

where α^k is the step size and $\boldsymbol{v}^k$ the (descent) direction at iteration k. Usually, the question remains on how to choose the step size and the descent direction to fully determine the scheme. Here, it is also required to redefine the "$+$" operation to fully define the method; since the feasible set is not a vector space, it is not true in general that (4.16) yields a feasible point, even for small α^k. A simple idea would be to project the resulting point onto the feasible set. This defines a projected line-search scheme. This is not a good idea for our problem for two reasons. Firstly, the projection onto the feasible set of (P) is a costly operation (see the classification of (F) in Table 4.1), and a usual line-search scheme typically requires a few dozen iterations. Secondly, the geometry of the feasible set exhibits a rich structure of a manifold which can be exploited.

In the following paragraphs, the general Riemannian geometry is introduced. Then, the retraction—the extension of the "$+$" operator, illustrated in Fig. 4.8—and the descent direction are described. Finally, the step size rule and some implementation details are given.

Riemannian geometry

Following [BSBA13], we define the quadric manifold, and then we use it to define the extended quadric manifold.

Proposition 4.1 (Quadric manifold)**.** *Let* $\Psi_t \colon \mathbb{R}^n \to \mathbb{R} \colon \boldsymbol{p}_t \mapsto \Psi_t(\boldsymbol{p}_t) = \boldsymbol{p}_t{}^\intercal \boldsymbol{B} \boldsymbol{p}_t + 2\boldsymbol{b}^\intercal \boldsymbol{p}_t + c_t$*, be a quadratic function. If* $\boldsymbol{B}$ *is nonsingular and* $c_t \neq \boldsymbol{b}^\intercal \boldsymbol{B}^{-1} \boldsymbol{b}$*, then the quadric* $\mathcal{Q}_t := \Psi_t^{-1}(0)$ *is an* $n-1$ *dimensional smooth manifold of* $\mathbb{R}^n$*.*

Proof. As $\Psi_t \in \mathcal{C}^\infty$, the quadric $\mathcal{Q}_t = \Psi_t^{-1}(0)$ is an algebraic variety. It is a manifold if $\mathrm{D}\Psi_t(\boldsymbol{p}_t) \neq 0$ for all $\boldsymbol{p}_t \in \mathcal{Q}_t$, that is, if the critical points of Ψ_t do not belong to the quadric. Since $\boldsymbol{B}$ is nonsingular, the only critical point is the center $\boldsymbol{d}$:

$$\boldsymbol{d} = -\boldsymbol{B}^{-1}\boldsymbol{b},$$

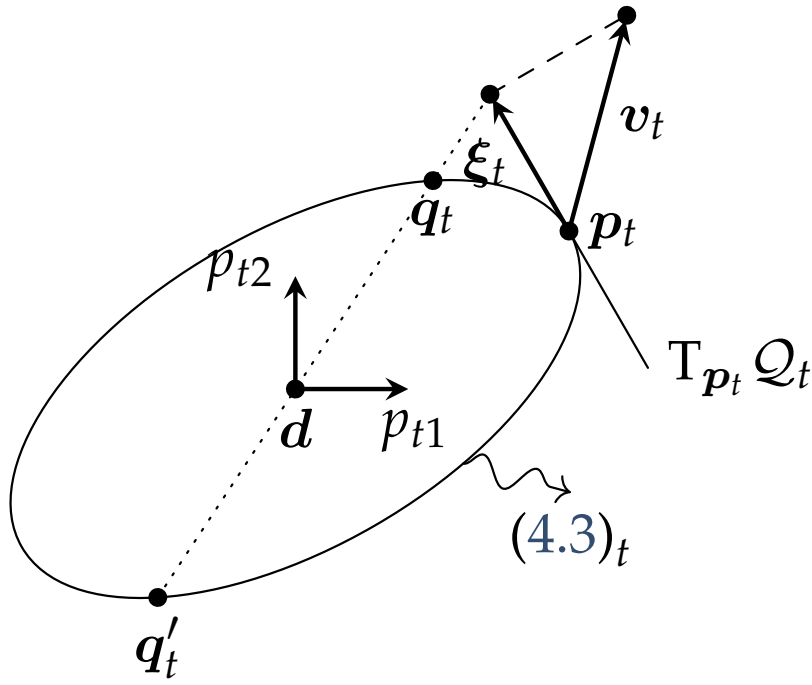


Fig. 4.8 Illustration of the retraction $\mathcal{R}_t(\boldsymbol{p}_t, \boldsymbol{\xi}_t)$.

and this point cancels Ψ_t only if $c_t = \boldsymbol{b}^\intercal \boldsymbol{B}^{-1}\boldsymbol{b}$. □

REMARK 4.3 (A type-II quadric is a manifold). The assumptions of Proposition 4.1 are equivalent to the needed assumptions for the quadric to be of *type II* in § 4.1.4. ■

Definition 4.2 (Extended quadric manifold)**.** The Cartesian product of the quadrics defined for each time step t as in (4.11) is called *extended quadric manifold* and computed as

$$\mathcal{Q}^{\text{tot}} := \mathcal{Q}_1 \times \mathcal{Q}_2 \times \ldots \times \mathcal{Q}_{|T|}.$$

The extended quadric manifold is effectively a manifold because the Cartesian product of smooth manifolds is also a manifold. In this case, the dimension of the manifold is $|T|\,(|G|-1)$.

The first important object to be described when dealing with manifolds is the tangent space, defined in general in Definition 4.3. Intuitively, it refers to the first-order approximation of the manifold at a given point $\boldsymbol{p}$. This mathematical object is used in numerous algorithms on manifolds in the following way. The tangent plane is defined at a given point $\boldsymbol{p}$ belonging to the manifold. Then, each other point $\boldsymbol{p}'$ of the manifold, sufficiently close to $\boldsymbol{p}$, is mapped to the tangent space through the *logarithmic map*. In this tangent space, the usual vector operations can be used. Eventually, the resulting vector can be mapped back to the manifold *via* the *exponential map*.

Definition 4.3. Let $\mathcal{Q}_t$ be a smooth real manifold, the *tangent space* reads

$$\mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}_t = \left\{ \boldsymbol{\xi} \in \mathbb{R}^{|G|} \,\middle|\, \text{there is some } c\colon \mathbb{R} \mapsto \mathcal{Q}_t \text{ with } c(0) = \boldsymbol{p}_t,\, c'(0) = \boldsymbol{\xi} \right\}.$$

Using the specific structure of the quadric manifold, this tangent space is computed as [BSBA13]

$$\mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}_t = \left\{ \boldsymbol{\xi} \in \mathbb{R}^{|G|} \,\middle|\, \boldsymbol{\xi}^\intercal (2\boldsymbol{B}\boldsymbol{p}_t + \boldsymbol{b}) = 0 \right\}, \tag{4.17}$$

and its dimension is $\dim\left(\mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}_t\right) = |G| - 1$. This tangent space is illustrated in Fig. 4.8 for a positive definite matrix $\boldsymbol{B}$.

Definition 4.4 (Tangent bundle)**.** The *tangent bundle* $\mathrm{T}\mathcal{Q}_t$ of a manifold $\mathcal{Q}_t$ is defined as the disjoint union of every tangent space at every point of the

manifold,

$$\mathrm{T}\mathcal{Q}_t = \bigsqcup_{\boldsymbol{p}_t \in \mathcal{Q}_t} \mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}_t.$$

Since every tangent space is a linear subspace of $\mathbb{R}^{|G|}$, each can be endowed with an inner product $\langle\cdot,\cdot\rangle_{\boldsymbol{p}_t}$ defined as the restriction of the canonical Euclidean product on the tangent space $\mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}_t$,

$$\langle\cdot,\cdot\rangle_{\boldsymbol{p}_t} \colon \mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}_t \times \mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}_t \to \mathbb{R} \colon (\boldsymbol{\xi},\boldsymbol{\zeta}) \mapsto \langle\boldsymbol{\xi},\boldsymbol{\zeta}\rangle_{\boldsymbol{p}_t} = \boldsymbol{\xi}^\intercal\boldsymbol{\zeta}. \tag{4.18}$$

Similarly, we define an inner product $\langle\cdot,\cdot\rangle_{\boldsymbol{p}}$ as the restriction of the canonical inner product on the tangent space $\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}$,

$$\langle\cdot,\cdot\rangle_{\boldsymbol{p}} \colon \mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}} \times \mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}} \to \mathbb{R} \colon (\boldsymbol{\xi},\boldsymbol{\zeta}) \mapsto \langle\boldsymbol{\xi},\boldsymbol{\zeta}\rangle_{\boldsymbol{p}} = \boldsymbol{\xi}^\intercal\boldsymbol{\zeta} = \sum_{t\in T}\langle\boldsymbol{\xi}_t,\boldsymbol{\zeta}_t\rangle_{\boldsymbol{p}_t}. \tag{4.19}$$

This inner product induces the canonical norm: $||\boldsymbol{\xi}||_{\boldsymbol{p}} = \langle\boldsymbol{\xi},\boldsymbol{\xi}\rangle_{\boldsymbol{p}}^{1/2}$. A smooth manifold equipped with an inner product on the tangent space at every point is called a *Riemannian* manifold.

The *normal space* can be computed as the orthogonal complement of the tangent space.

Definition 4.5. Let $\mathcal{Q}_t$ be a $|G|-1$ smooth manifold embedded in $\mathbb{R}^{|G|}$ and $\mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}_t$ its tangent space, the *normal space* is defined as

$$\mathrm{N}_{\boldsymbol{p}_t}\mathcal{Q}_t = \mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}_t{}^{\perp}, \tag{4.20}$$

where $\cdot^\perp$ is defined with respect to the canonical inner product on $\mathbb{R}^{|G|}$.

It follows from (4.17) that

$$\mathrm{N}_{\boldsymbol{p}_t}\mathcal{Q}_t = \{\tau(2\boldsymbol{B}\boldsymbol{p}_t + \boldsymbol{b}) \mid \tau \in \mathbb{R}\}, \tag{4.21}$$

and $\dim(\mathrm{N}_{\boldsymbol{p}_t}\mathcal{Q}_t) = 1$.

Now that an expression for the tangent and normal space of each manifold has been obtained, both can be computed for the extended quadric manifold.

Proposition 4.6. $\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{tot} = \mathrm{T}_{\boldsymbol{p}_1}\mathcal{Q}_1 \times \mathrm{T}_{\boldsymbol{p}_2}\mathcal{Q}_2 \times \ldots \times \mathrm{T}_{\boldsymbol{p}_{|T|}}\mathcal{Q}_{|T|}$

Proof. See [GP10, Chap. 1.2]. □

Proposition 4.7. $\mathrm{N}_{\boldsymbol{p}}\mathcal{Q}^{tot} = \mathrm{N}_{\boldsymbol{p}_1}\mathcal{Q}_1 \times \mathrm{N}_{\boldsymbol{p}_2}\mathcal{Q}_2 \times \ldots \mathrm{N}_{\boldsymbol{p}_{|T|}}\mathcal{Q}_{|T|}$

Proof. We first show that $\mathrm{N}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}} = (\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}})^{\perp} \supseteq \mathrm{N}_{\boldsymbol{p}_1}\mathcal{Q}_1 \times \mathrm{N}_{\boldsymbol{p}_2}\mathcal{Q}_2 \times \ldots \times \mathrm{N}_{\boldsymbol{p}_{|T|}}\mathcal{Q}_{|T|}$ and then we conclude with an argument on the dimensions.

i) Let $\boldsymbol{p} \in \mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}$ and $\boldsymbol{p}' \in \mathrm{N}_{\boldsymbol{p}_1}\mathcal{Q}_1 \times \mathrm{N}_{\boldsymbol{p}_2}\mathcal{Q}_2 \times \ldots \times \mathrm{N}_{\boldsymbol{p}_{|T|}}\mathcal{Q}_{|T|}$, both are partitioned as follows: $\boldsymbol{p} = \begin{pmatrix} \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \vdots \\ \boldsymbol{p}_{|T|} \end{pmatrix}$ and $\boldsymbol{p}' = \begin{pmatrix} \boldsymbol{p}'_1 \\ \boldsymbol{p}'_2 \\ \vdots \\ \boldsymbol{p}'_{|T|} \end{pmatrix}$. It follows from Proposition 4.6 and Definition 4.5 that $\boldsymbol{p}^{\mathsf{T}}\boldsymbol{p}' = 0$. Therefore, we have $\boldsymbol{p}' \in (\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}})^{\perp}$.

ii) Since $\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}$ is a linear subspace of $\mathbb{R}^{|G||T|}$, we have

$$\begin{aligned} \dim\left((\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}})^{\perp}\right) &= |G|\,|T| - \dim\left(\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}\right) \\ &= |G|\,|T| - |T|\,(|G|-1) = |T|\,. \end{aligned}$$

This concludes the proof as

$$\dim\left(\mathrm{N}_{\boldsymbol{p}_1}\mathcal{Q}_1 \times \mathrm{N}_{\boldsymbol{p}_2}\mathcal{Q}_2 \times \ldots \times \mathrm{N}_{\boldsymbol{p}_{|T|}}\mathcal{Q}_{|T|}\right) = |T|\,.$$

□

Since we have shown that both tangent and normal spaces of the extended quadric are the Cartesian products of the tangent and normal space of the individual manifold $\mathcal{Q}_t$, we can easily extend the projection operator from [BSBA13] by working componentwisely.

The projection $\mathrm{Pr}_{\boldsymbol{p}}(\boldsymbol{v})$ of a vector $\boldsymbol{v} \in \mathbb{R}^{|G||T|}$, partitioned as

$$(\boldsymbol{v}_1{}^{\mathsf{T}}, \boldsymbol{v}_2{}^{\mathsf{T}}, \ldots, \boldsymbol{v}_{|T|}{}^{\mathsf{T}})^{\mathsf{T}},$$

onto $\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}$ can be constructed by removing the normal component of $\boldsymbol{v}$:

$$\mathrm{Pr}_{\boldsymbol{p}}(v) = (\hat{\boldsymbol{v}}_1^{\mathsf{T}}, \ldots, \hat{\boldsymbol{v}}_{|T|}^{\mathsf{T}})^{\mathsf{T}}, \tag{4.22}$$

where $\hat{\boldsymbol{v}}_t = \boldsymbol{v}_t - \tau_t(2\boldsymbol{B}\boldsymbol{p}_t + \boldsymbol{b})$ and τ_t is chosen to ensure that $P_{\boldsymbol{p}_t}(\boldsymbol{v}_t)$ belong

to $\mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}_t$, *i.e.*,

$$\tau_t = \frac{\boldsymbol{v}_t{}^{\mathsf{T}}(2\boldsymbol{B}\boldsymbol{p}_t + \boldsymbol{b})}{\|2\boldsymbol{B}\boldsymbol{p}_t + \boldsymbol{b}\|_2^2}. \tag{4.23}$$

Retraction

Definition 4.8 (Retraction)**.** A retraction $\mathcal{R}$ is a smooth mapping from the tangent bundle of a manifold to the manifold itself,

$$\begin{aligned} &\mathcal{R}_t \colon \mathrm{T}\mathcal{Q}_t \to \mathcal{Q}_t \colon (\boldsymbol{p}_t, \boldsymbol{\xi}_t) \mapsto \boldsymbol{q}_t := \mathcal{R}_t(\boldsymbol{p}_t, \boldsymbol{\xi}_t), \\ &\text{with } \left.\frac{\mathrm{d}R_t(\boldsymbol{p}_t, \alpha\boldsymbol{\xi}_t)}{\mathrm{d}\alpha}\right|_{\alpha=0} = \boldsymbol{\xi}_t \text{ and } \mathrm{R}_t(\boldsymbol{p}_t, \boldsymbol{0}) = \boldsymbol{p}_t. \end{aligned} \tag{4.24}$$

The retraction is not unique. In fact, the retraction can be seen as an approximation of the exponential map. In the specific case of the quadric manifold, the exponential map cannot be easily computed, see the discussion in [BSBA13], but some retractions can be easily computed.

The retraction considered in this work and introduced in [BSBA13] is illustrated in Fig. 4.8: the retraction $\mathcal{R}_t(\boldsymbol{p}_t, \boldsymbol{\xi}_t)$ is obtained by looking at the intersection point, denoted as $\boldsymbol{q}_t$, of the quadric and some line. This line passes through the point $\boldsymbol{p}_t + \boldsymbol{\xi}_t$ and the quadric center $\boldsymbol{d}$, as defined in (4.15). This point of intersection is not supposed to be unique (see $\boldsymbol{q}_t'$); to remedy this, the closest point to $\boldsymbol{p}_t + \boldsymbol{\xi}_t$ is chosen. A closed-form solution of this procedure is given in [BSBA13, §3.3].

In general, the direction $\boldsymbol{v}_t$ may not lie in the tangent space of $\boldsymbol{p}_t$. An extra step of projection is then needed in (4.16): $\boldsymbol{\xi}_t = \mathrm{Pr}_{\boldsymbol{p}}(\boldsymbol{v}_t)$. This retraction can be readily extended to the multistep case: it suffices to work with each component independently:

$$\mathcal{R} \colon \mathrm{T}\mathcal{Q}^{\mathrm{tot}} \to \mathcal{Q}^{\mathrm{tot}} \colon (\boldsymbol{p}, \boldsymbol{\xi}) \mapsto \boldsymbol{q} := \begin{pmatrix} \boldsymbol{q}_1 \\ \boldsymbol{q}_2 \\ \vdots \\ \boldsymbol{q}_{|T|} \end{pmatrix}, \tag{4.25}$$

with $\boldsymbol{q}_t = \mathcal{R}_t(\boldsymbol{p}_t, \boldsymbol{\xi}_t)$. Altough the retraction is illustrated with an ellipse in Fig. 4.8, it is not limited to this specific quadric. Each quadric with a middle point (see § 4.1.4), can be considered. Also, an important feature of this procedure is that it does not require a lot of computational power: the tangent space has a closed-form (4.17) as well as the projection onto this tangent space (4.22). The retraction itself can also be efficiently computed—

it amounts to solving $|T|$ one-dimensional quadratic equations and choosing for each equation the root that is the closest to one [BSBA13].

Going back to (4.16), if the direction $\boldsymbol{v}^k$ does not belong to the tangent space of the current iterate $\boldsymbol{p}^k$, it reads

$$\boldsymbol{p}^{k+1} = \mathcal{R}(\boldsymbol{p}^k, \mathrm{Pr}_{\boldsymbol{p}^k}\left(\alpha^k \boldsymbol{v}^k\right)). \tag{4.26}$$

Descent direction on a manifold

Prior to the discussion on the *descent* direction, we first define the concept of $\mathcal{Q}-$admissible direction, which accommodates the set defined by (4.5).

Definition 4.9. A $\mathcal{Q}-$*admissible direction* defined at point $\boldsymbol{p} \in \mathcal{Q}^{\mathrm{tot}}$ is a vector $\boldsymbol{v} \in \mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}$ for which there exists $\epsilon > 0$ such that $\mathcal{R}(\boldsymbol{p}, \alpha \boldsymbol{v})$ belongs to $\mathcal{Q}^{\mathrm{tot}}$ for all $\alpha \in [0, \epsilon]$.

The gradient is inherent in the concept of steepest descent, but the function (4.1) is only smooth almost-everywhere, and the zero-measure set where it is nonsmooth is located at positions where the argument of the absolute value in (4.1) changes sign. This set simply corresponds to a multidimensional grid which can be computed as

$$\begin{aligned} S := \Big\{ \boldsymbol{p} \in \mathbb{R}^{|G||T|} \;\Big|\; & \text{there is some } g \in G, t \in T, j \in J_g, \\ & \text{with } p_{gt} = \underline{P}_g + \frac{(j-1)\pi}{2E_g} \Big\}, \end{aligned} \tag{4.27}$$

where

$$J_g := \left\{ j = 1, 2, \ldots, 1 + \left\lceil (\overline{P}_g - \underline{P}_g) \frac{2E_g}{\pi} \right\rceil \right\}. \tag{4.28}$$

For a nonsmooth function, the gradient is often replaced by the subgradient. However, this mathematical object cannot be used for the nonconvex functions (4.1). Here, we consider the closely connected concept of *generalized gradient* introduced in [Cla76]. First, let us define the generalized directional derivative $f^\circ(\boldsymbol{p}; \boldsymbol{v})$ of the Lipschitz function $f\colon X \to \mathbb{R}$ for a Banach space X in the direction $\boldsymbol{v}$ as

$$f^\circ(\boldsymbol{p}; \boldsymbol{v}) = \limsup_{\substack{\boldsymbol{h} \to 0 \\ \lambda \downarrow 0}} \frac{f(\boldsymbol{p} + \boldsymbol{h} + \lambda \boldsymbol{v}) - f(\boldsymbol{p} + \boldsymbol{h})}{\lambda}.$$

This function is convex in $\boldsymbol{v}$, regardless of the convexity of f. The *generalized gradient* of f at $\boldsymbol{p}$, written $\partial f(\boldsymbol{p})$, is defined as the subdifferential

of the convex function $f^{\circ}(\boldsymbol{p}, \cdot)$ at $\mathbf{0}$. In particular, we have

$$\partial f(\boldsymbol{p}) = \left\{ \boldsymbol{\zeta} \in X^* \mid f^{\circ}(\boldsymbol{p}; \boldsymbol{v}) \geq \langle \boldsymbol{v}, \boldsymbol{\zeta} \rangle \text{ for all } \boldsymbol{v} \in X \right\}, \tag{4.29}$$

with X^* the dual space of X. The generalized gradient shares some important properties with the subdifferential of a convex function: it is a nonempty convex and compact set and if a point $\boldsymbol{p}$ is a local minimizer of f, then $\mathbf{0} \in \partial f(\boldsymbol{p})$. Furthermore, if f is convex, then the generalized gradient coincides with the subdifferential and for a point $\boldsymbol{p}$ differentiable, we have $\partial f(\boldsymbol{p}) = \{\nabla f(\boldsymbol{p})\}$.

Function (4.2) is a Lipschitz function that can be computed as the pointwise maximum of $m := 2^{|G||T|}$ smooth functions[3], *i.e.*,

$$f(\boldsymbol{p}) = \max_{j=1,\ldots,m} f_j(\boldsymbol{p}). \tag{4.30}$$

In this specific case, [BSBA13] show that the generalized gradient can be described as

$$\partial f(\boldsymbol{p}) = \operatorname{co} \left\{ \nabla f_j(\boldsymbol{p}) \mid j \in \mathcal{I}_f(\boldsymbol{p}) \right\}, \tag{4.31}$$

where $\operatorname{co}\{\cdot\}$ denotes the convex hull and $\mathcal{I}_f$ the set of indices for which the maximum in (4.30) is attained.

This framework is valid for the unconstrained problem (P). Let us now integrate the manifold constraint (4.5) and then the other linear constraints.

Given a smooth function f_j from the pointwise maximum in (4.30), the *projected gradient* is defined as follows:

$$\operatorname{grad}_j f(\boldsymbol{p}) = \operatorname{Pr}_{\boldsymbol{p}} \left(\nabla f_j(\boldsymbol{p}) \right), \tag{4.32}$$

and the projected generalized gradient is given by

$$\operatorname{grad} f(\boldsymbol{p}) = \operatorname{co} \left\{ \operatorname{grad}_j f(\boldsymbol{p}) \mid j \in \mathcal{I}_f(\boldsymbol{p}) \right\}. \tag{4.33}$$

The steepest $\mathcal{Q}-$admissible direction $\boldsymbol{v}^k$ from iterate $\boldsymbol{p}^k$ is obtained by computing the shortest vector in $\operatorname{grad} f(\boldsymbol{p}^k)$, see [BSBA13] for more details. This can be computed by minimizing the norm of the convex combination of the projected gradients. If the coefficients of the convex combination are given by

[3] Since $|x| = \max\{x, -x\}$ and there are $|G|\,|T|$ absolute values in (4.2).

$$\begin{aligned}\boldsymbol{\lambda}^k &= \underset{\substack{\boldsymbol{\lambda}\geq 0\\ \sum\lambda_j=1}}{\arg\min} \Big\| \sum_{j\in\mathcal{I}_f(\boldsymbol{p}^k)} \lambda_j \mathrm{grad}\, f(\boldsymbol{p}^k)\Big\|^2_{\boldsymbol{p}^k} \\ &= \underset{\substack{\boldsymbol{\lambda}\geq 0\\ \sum\lambda_j=1}}{\arg\min} \Big\|\mathrm{Pr}_{\boldsymbol{p}^k}\Big(\sum_{j\in\mathcal{I}_f(\boldsymbol{p}^k)} \lambda_j \nabla f_j(\boldsymbol{p}^k)\Big)\Big\|^2_{\boldsymbol{p}^k}, \end{aligned} \tag{4.34}$$

then the steepest-descent $\mathcal{Q}-$admissible direction is computed as

$$\boldsymbol{v}^k = -\mathrm{Pr}_{\boldsymbol{p}^k}\Big(\sum_{j\in\mathcal{I}_f(\boldsymbol{p}^k)} \lambda_j^k \nabla f_j(\boldsymbol{p}^k)\Big). \tag{4.35}$$

This optimization problem is defined on a high dimensional ($2^{|G||T|}$) simplex and should be solved at each iteration. To remedy the high expected solving time, [BSBA13] also introduce a reformulation which exploits the specific form of the function (4.2) and considerably reduces the dimension of the problem. We show here how to apply this reformulation to the extended quadric manifold.

Let $\mathcal{S}(\boldsymbol{p}^k)$ be the set of indices of $\boldsymbol{p}^k$ where the sine components of the objective function evaluate to zero and $\mathcal{F}(\boldsymbol{p}^k)$ the remaining indices, *i.e.*,

$$\begin{aligned}\mathcal{S}(\boldsymbol{p}^k) &= \left\{ \underbrace{(g_1^{\mathrm{s}}, t_1^{\mathrm{s}})}_{:=s_1}, \underbrace{(g_2^{\mathrm{s}}, t_2^{\mathrm{s}})}_{:=s_2}, \ldots, \underbrace{(g_{n_{\mathrm{s}}^k}^{\mathrm{s}}, t_{n_{\mathrm{s}}^k}^{\mathrm{s}})}_{:=s_{n_{\mathrm{s}}^k}} \right\} = \bigcup_{t\in T} \mathcal{S}_t(\boldsymbol{p}_t^k) \\ &= \bigcup_{t\in T} \Big\{ (g,t) \mid g\in G, f_g^{\mathrm{V}}(p_{gt}) = 0 \Big\}, \end{aligned} \tag{4.36}$$

$$\begin{aligned}\mathcal{F}(\boldsymbol{p}^k) &= \left\{ \underbrace{(g_1^{\mathrm{f}}, t_1^{\mathrm{f}})}_{:=f_1}, \underbrace{(g_2^{\mathrm{f}}, t_2^{\mathrm{f}})}_{:=f_2}, \ldots, \underbrace{(g_{n_{\mathrm{f}}^k}^{\mathrm{f}}, t_{n_{\mathrm{f}}^k}^{\mathrm{f}})}_{:=f_{n_{\mathrm{f}}^k}} \right\} = \bigcup_{t\in T} \mathcal{F}_t(\boldsymbol{p}_t^k) \\ &= \bigcup_{t\in T} \Big\{ (g,t) \mid g\in G, (g,t) \notin \mathcal{S}(\boldsymbol{p}_t^k) \Big\}, \end{aligned} \tag{4.37}$$

and we have naturally $T\times G = \mathcal{S}(\boldsymbol{p}^k) \bigcup \mathcal{F}(\boldsymbol{p}^k)$ for all $\boldsymbol{p}^k \in \mathcal{Q}^{\mathrm{tot}}$, $\Big|\mathcal{S}(\boldsymbol{p}^k)\Big| =$

n_{s}^k and $\left|\mathcal{F}(\boldsymbol{p}^k)\right| = n_{\mathrm{f}}^k = |T|\,|G| - n_{\mathrm{s}}^k$. To lighten the notation, we sometimes omit the superscript k that denotes the dependency on k. Using these sets, the projected generalized gradient can be efficiently split between a smooth and a nonsmooth part. Let $\boldsymbol{g}^k$ be the smooth part and $\boldsymbol{S}^k$ the matrix containing the nonsmooth parts to be combined,

$$\boldsymbol{S}^k = \left[\mathrm{Pr}_{\boldsymbol{p}^k}\left(\nabla\overline{f}^{\mathrm{S}}_{s_1^k}(\boldsymbol{p}^k\right), \ldots, \mathrm{Pr}_{\boldsymbol{p}^k}\left(\nabla\overline{f}^{\mathrm{S}}_{n_{\mathrm{s}}^k}(\boldsymbol{p}^k)\right)\right] \in \mathbb{R}^{|T||G|\times n_{\mathrm{s}}^k}, \tag{4.38}$$

$$\boldsymbol{g}^k = \mathrm{Pr}_{\boldsymbol{p}^k}\left(\nabla f^{\mathrm{Q}}(\boldsymbol{p}^k) + \sum_{(g,t)\in\mathcal{F}(\boldsymbol{p}^k)} \nabla\left|\overline{f}^{\mathrm{S}}(\boldsymbol{p}^k\right|)\right) \in \mathbb{R}^{|T||G|}, \tag{4.39}$$

where f^{S} is the sine part of (4.1), *i.e.*, $f_g^{\mathrm{V}} = \left|f_g^{\mathrm{S}}\right|$, and $\overline{f}_s(\boldsymbol{p}) := f_s(p_s)$. As the fuel cost is independent of time, we define with a slight abuse of notation $f_{gt} := f_g$. The subproblem (4.34) can be rewritten as

$$\boldsymbol{\lambda}^k = \underset{-1\leq\boldsymbol{\lambda}\leq 1}{\arg\min} \|\boldsymbol{g}^k + \boldsymbol{S}^k\boldsymbol{\lambda}\|^2_{\boldsymbol{p}^k} \tag{4.40}$$

and the $\mathcal{Q}-$admissible descent direction $\boldsymbol{v}^k$ is computed as $\boldsymbol{v}^k = -(\boldsymbol{g}^k + \boldsymbol{S}^k\boldsymbol{\lambda}^k)$. The subproblem (4.40) is a convex quadratic programming (QP) problem of dimension $n_{\mathrm{s}}^k \leq |T|\,|G|$, which is much easier to solve than every problem in Table 4.1.

Until now, the unique constraint that we consider is (4.5). The generalization of the descent direction on a *constrained manifold*, such as the feasible set of (P) is presented hereafter.

Descent direction on constrained manifold

The feasible set of (P) is a manifold further constrained by q linear constraints under the form $\boldsymbol{c}_i{}^{\mathsf{T}}\boldsymbol{p} \leq 0$ with $i = 1\ldots q$. To address this situation, we define the matrix $\boldsymbol{C}^k$ of the projected active constraints at point $\boldsymbol{p}^k$ whose columns are given by

$$\boldsymbol{C}^k_{*,j} = \mathrm{Pr}_{\boldsymbol{p}^k}\left(\boldsymbol{c}_j\right) \text{ for all } j \in \{1\ldots q\} \text{ such that } \boldsymbol{c}_j{}^{\mathsf{T}}\boldsymbol{p}^k = 0. \tag{4.41}$$

We have $\boldsymbol{C}^k \in \mathbb{R}^{|T||G|\times n_{\mathrm{c}}^k}$ with $0 \leq n_{\mathrm{c}}^k \leq q$.

The subproblem (4.40) becomes

$$(\boldsymbol{\lambda}^k, \boldsymbol{\mu}^k) = \underset{\substack{-1 \leq \boldsymbol{\lambda} \leq 1 \\ \boldsymbol{\mu} \geq 0}}{\arg\min} \|\boldsymbol{g}^k + \boldsymbol{S}^k \boldsymbol{\lambda} + \boldsymbol{C}^k \boldsymbol{\mu}\|^2_{\boldsymbol{p}^k}, \tag{Sub}$$

and the descent direction $\boldsymbol{v}^k = -(\boldsymbol{g}^k + \boldsymbol{S}^k \boldsymbol{\lambda}^k + \boldsymbol{C}^k \boldsymbol{\mu}^k)$.

The dimension of (Sub), the number of decision variables, is between 0 and $|T|\,|G| + q \ll 2^{|G||T|}$. Furthermore, for a smooth point located in the interior of the domain, (Sub) is trivial, and the direction is given by the gradient: $\boldsymbol{v}^k = -\boldsymbol{g}^k = -\mathrm{Pr}_{\boldsymbol{p}^k}\left(\nabla f(\boldsymbol{p}^k)\right)$.

To complete the description of the line-search scheme, it remains to choose a step size rule and a stopping criterion.

Stopping criterion, step rule, and implementation details

It can be shown that the direction $\boldsymbol{v}^k$ at a stationary point $\boldsymbol{p}^k$ yields the zero vector [Cla76]. Thus, a natural stopping criterion is to monitor the direction norm. Unfortunately, as studied in [DDTA13], this type of criterion on the norm of the KKT violation, which is here equivalent to the norm of the descent direction, is not reliable because this norm varies nonsmoothly around stationary points. Hence, the second criterion used here is the step size α^k. If the step size becomes too small for the point to be admissible (*i.e.*, feasible for (P)) the algorithm stops.

A common practice to get the step size is to use *Armijo's rule*. This rule ensures that the step size α^k at iteration k renders the next iterate $\boldsymbol{p}^{k+1} = \mathcal{R}(\boldsymbol{p}^k, \alpha^k \boldsymbol{v}^k)$ feasible while sufficiently decreasing the objective. The explicit implementation of Armijo's rule is described in [BSBA13, Algorithm 3]. We slightly modify it such that it returns 0 if $\boldsymbol{v}^k = \boldsymbol{0}$ (up to a given tolerance) and -1 if no step size below a given threshold is found.

It appears that, for sufficiently large problems, the subproblem (Sub) becomes problematic; in the sense that the direction obtained is only admissible in a tiny neighborhood around the previous iterate. This can arise when a given component of an iterate p^k_{gt} is bound at multiple operational constraints, *e.g.*, $p^k_{gt} = \underline{P}_g$ ((4.3) is tight) and $p^k_{gt} - p^k_{g(t-1)} = \overline{R}_g$ ((4.4) is tight). To remedy this situation, the variable is frozen at its value and is no longer a decision variable—giving the right feedback loop in Fig. 4.9. This allows us to provide a temporary degree of freedom to the algorithm that may find another direction for which an admissible step size is available. This procedure, as well as the complete method for obtaining a feasible solution and improving it, is described in Fig. 4.9.

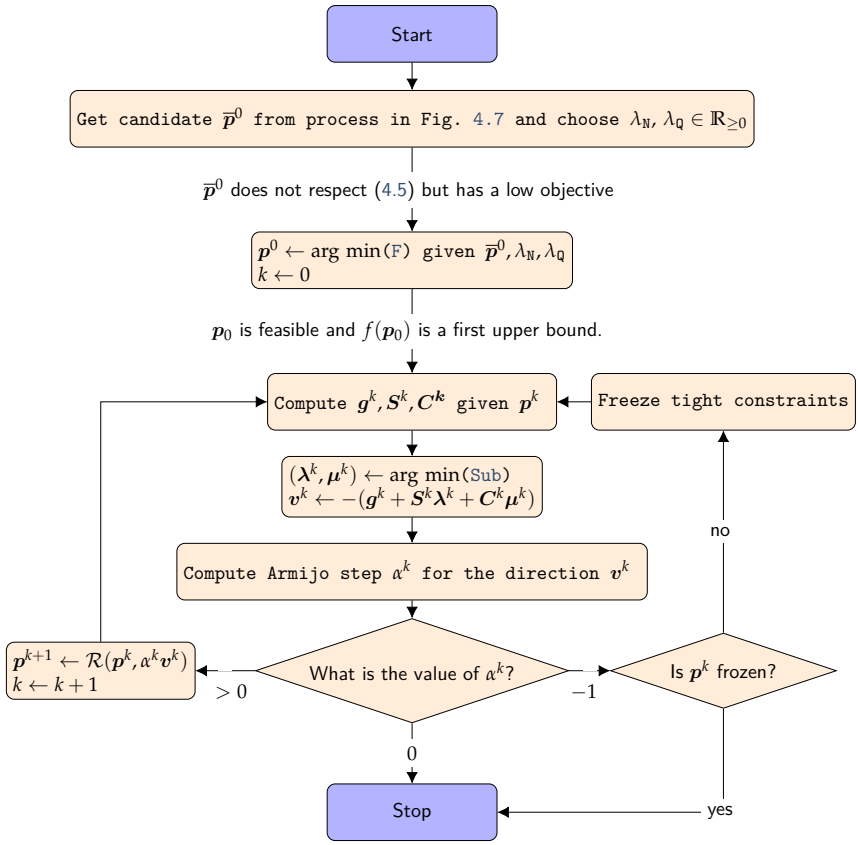


Fig. 4.9 Flowchart of the method that projects the candidate from Fig. 4.7 and improves it through a Riemannian subgradient descent.

For the projection step, *i.e.*, the second step of Fig. 4.1 or equivalently the solving of (F) in Fig. 4.9, we use Gurobi. This step is one of the bottlenecks of our method: it takes approximately half of the execution time. This motivates the second part of the thesis, Part II.

Finally, it is unlikely that a given iterate would be exactly located at a non-smooth point. This implies that $\boldsymbol{S}^k$ from (4.38) is likely to be empty. Hence, like [BSBA13], we consider that the equalities from Eqs. (4.38) and (4.41) should be satisfied within a small ϵ accuracy.

4.3 Test cases

Following [PJY18], the method is tested on several data sets with a different number of units and a time horizon of 24 hours. For each data set, the best objective (or upper bound) is reported, along with the best lower bound. The optimality gap, defined as the difference between the best-known upper and lower bound on the global optimum, is also reported.

In order to account for the different processor speeds from other methods in the literature, the scaled CPU time is used [YWY+08]:

$$\text{Scaled CPU time} = \frac{\text{Given CPU Speed}}{\text{Base CPU speed}}\text{Given CPU time}, \tag{4.42}$$

where the base CPU time used in this chapter is 3.6 GHz. However, it is important to realize that the execution time is affected by other factors than the CPU clock rate, notably the number of cores. The purpose of the S-time is thus chiefly to check whether the run time remains reasonable. The key contribution of the proposed method is to be found in the "Lower bound" column. The scaled CPU time is denoted as "S-Time" and given in minutes.

In addition to the main objective, the *deviation* and the *losses* are computed. The deviation corresponds to the mismatch between the returned solution and the ellipsoid and is computed by rearranging (4.5): deviation $= \sum_{t\in T}\left|\sum_{g\in G} p_{gt} - P_t^{\mathrm{D}} - p_t^{\mathrm{loss}}\right|$. It is reported in the column "Deviation" of Tables 4.2 and 4.3. The losses correspond to the value of $\sum_{t\in T} p_t^{\mathrm{loss}}$, reported in the column "Losses".

The data, final solutions, and algorithm implementations are available on GitLab [Van21].

4.3.1 5-unit, 24 time steps test case

We use the data from [PCCB06]. The data consists of a 5-unit case, where all units obey a valve point effect. The reserve requirement is set to 5% of the demand.

We compare the solution obtained with our proposed method to other methods from the literature in Table 4.2. The three first columns report the minimum (Min), average (Avg), and maximal (Max) solution. For deterministic methods, only the first column includes values.

To obtain the columns "Losses" and "Deviation", we plug the best

solution that is available in the literature into our model. We also take the opportunity to check whether the reported value matches the one we compute. In this experiment, we find no discrepancies between both values, unlike the experiment from § 4.3.2.

The proposed method is the only one providing a lower bound, which allows us to estimate the final (relative) optimality gap at 0.86%. Since the lower bound is only improved in the first part of the proposed method (APLA) this lower bound will always be equal to the one of the full method (APLA-RSG).

In the APLA method, we also include the projection step (F) because the solution at the end of the process described in Fig. 4.7 is not feasible. Comparing the S-Time of APLA and APLA-RSG, we observe that 24% of the execution time is spent in the RSG part. As far as the remaining execution time is concerned, about half is spent in the APLA part and the other half in the projection step.

The proposed method, APLA-RSG, achieves a competitive objective in comparison with the other methods from the literature. It is outperformed by BBOSB and MILP-IPM. Nevertheless, we note that i) APLA-RSG provides a lower bound, ii) the deviation of APLA-RSG is much smaller, and iii) a fair comparison should take the run time into account. BBOSB [XS18] only report the number of function evaluation ($\sim 2e5$). Function evaluations (FEs) allow accurate comparison among methods run on different computers. They cannot be computed in our case due to the call to the MIP solver in APLA, nonetheless, we track the FEs within the RSG method and obtain 24750 FEs for converging. We can therefore estimate the equivalent FEs for the entire APLA-RSG procedure as being equal to 100 000, which is half of the BBOSB procedure.

Table 4.2 also presents the results of Ipopt [WB06] with default parameter settings. Unfortunately, Ipopt times out and the returned objective (35592) does not match the evaluation of the returned solution at the true objective (45514). This mismatch results from the intermediate variables required for modeling the nonconvex objective (4.1) using JuMP [DHL17, JuM].

4.3.2 10-unit, 24 time steps test case

The data originates from [PCCB06] and consists of a 10-unit case. All units obey a VPE, and the matrix $\boldsymbol{B}$ is indefinite. Following [WDW+17], the reserve requirement is set to 3.5% of the demand and not 5%. As a matter

Table 4.2 Summary results: 5-unit case

Method	Cost			S-Time	Losses (MW)	Deviation (MW)	Lower bound
	Min	Avg	Max				
BBOSB[XS18]	43018	43066	43197	-	194.65	0.01	-
HIGA[MIRS13]	43125	43162	43259	1.37	194.79	0.074	-
ICA[MiRSE12]	43117	43144	43210	-	194.80	0.014	-
MILP-IPM[PJY18]	43084	-	-	0.58	195.26	0.00095	-
Ipopt	45514 (35592)	-	-	0.6	196	0.35	-
APLA	43250	-	-	0.38	193.98	1.6e-9	42527.85
APLA-RSG	43098	-	-	0.5	194.02	3e-11	42527.85

of fact, the problem with 5% reserves is not feasible: this can be shown by examining the static dispatch at the highest demand. Since (4.6) must hold for all t, we have

$$\sum_{g\in G} \overline{P}_g - (P_t^{\mathrm{D}} + \min_{\boldsymbol{p}_t\in\mathcal{P}_t} p_t^{\mathrm{loss}} + P_t^{\mathrm{S}}) \geq 0, \tag{4.43}$$

where $\mathcal{P}_t$ corresponds to the intersection of the power ranges $(4.3)_t$ with a relaxed version of the power balance $(4.5)_t$.

$$\sum_{g\in G} p_{gt} \geq P_t^{\mathrm{D}}. \tag{4.44}$$

This is a relaxation because negative losses p_t^{loss} are physically impossible. It is clear that, if the problem (P) is feasible, then $(4.6)_t$ holds for all t, which implies that $(4.43)_t$ also holds for all t. Conversely, if $(4.43)_t$ does not hold for some t then (P) must be infeasible. In this test case, the highest demand occurs for $t = 12$ with $P_t^{\mathrm{D}} = 2220$ MW and $P_t^{\mathrm{S}} = 111$ MW. The sum of the maximum power ranges is 2358 MW and the minimal power losses are computed as 49.7MW. Hence, we conclude that the 10-unit test case with a 5% reserve requirement is *not* feasible. This may explain why [PJY18], despite developing the method to account for reserves, do not test the 10-unit test case with reserves. This may also explain why [WDW+17] choose a 3.5% reserve requirement instead of the usual 5% requirement. This also raises questions about certain methods in the literature—reported in [WDW+17, Table V]—that claim to solve this infeasible problem.

Table 4.3 compares the different methods from the literature. The analysis is analogous to the previous case (§ 4.3.1); the proposed method provides a competitive objective function value in a similar amount of time. The optimality gap (0.58%) is better relative to the previous case. We also observe

Table 4.3 Summary results: 10-unit case

Method	Cost			S-Time	Losses (MW)	Deviation (MW)	Lower bound
	Min	Avg	Max				
BBOSB[XS18]	1039169[4]	1041539	1039969	-	818.22	83	-
TSMILP [WDW+17]	1037487	-	-	1.9	832.32	0.013	-
MILP-IPM[PJY18]	1040676	-	-	0.75	882.74	0.0019	-
Ipopt	1054180 (1038060)	-	-	2.8	740.3	0.015	-
APLA	1040475	-	-	1.6	882.02	1.9e-9	1032045
APLA-RSG	1038108	-	-	2.3	809.05	1.3e-11	1032045

that the returned solution exhibits the lowest power losses for both cases, except for Ipopt applied to the 10-unit case.

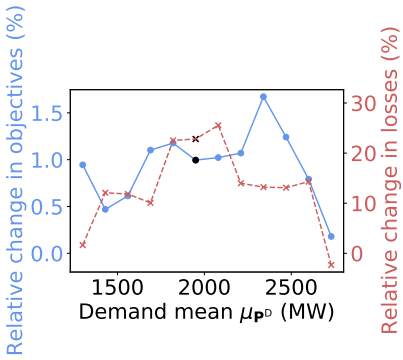
4.3.3 15-unit, 24 time steps test case

The data for this test case originates from [Zwe03]. The original instance consists in the dispatch of 15 units. All units obey a valve point effect in the original instance. As in [VAP20a], we model a demand over 24 time steps with ramping constraints. We compare the solution of APLA-RSG with Ipopt. Fig. 4.10a depicts the relative changes in objective and power losses between the returned solution of APLA-RSG and Ipopt for twelve different load profiles. The load profiles are presented in Fig. 4.10b. The objective of APLA-RSG always outperforms the objective of Ipopt, and the improvement is approximately equal to 1%. The power losses of APLA-RSG are also lower than the losses of Ipopt for eleven of the twelve problems, and the improvement goes up to 25 %. Using a lower (or higher) mean demand μ_{PD} than the one considered in the present experiments would result in an infeasible problem.

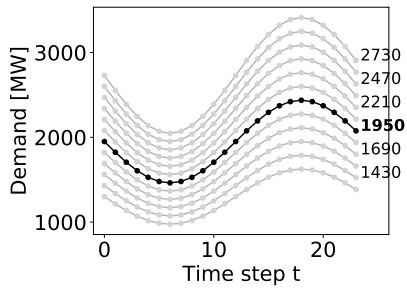
Concerning the computational time, APLA-RSG running times range from 88 to 113 seconds and Ipopt from 204 to 215 seconds. The deviation is around 1×10^{-10} MW for APLA-RSG and 0.003 MW for Ipopt. In other words, APLA-RSG obtains a solution twice as fast while strictly meeting the constraints, decreasing the losses, and reducing the objective by around 1%.

4.4 Conclusion

In this chapter, we develop a method for tackling a nonsmooth and nonconvex economic dispatch problem with power losses. In addition to the valve



(a) Relative change between APLA-RSG and Ipopt for the objective (solid blue line) and the power losses (dashed red line) for twelve different load profiles over 24 hours (with a shifted mean).



(b) Load profiles studied in § 4.3.3. Each point corresponds to the demand P_t^D at a given time step t and for a given profile. The profiles are centered around $\mu_{P_t^D}$.

Fig. 4.10 Comparison between APLA-RSG and Ipopt applied to the 15-unit test case for twelve load profiles. The profile with a mean $\mu_{P_t^D} = 1950$ MW is emphasized in black in both figures.

point effect, nonconvexities also originate from the power losses, which are modeled as a nonconvex quadratic equation.

We demonstrate that power balance with quadratic power losses can be expressed as quadrics, which exhibit the rich structure of a Riemannian manifold. The hypothesis of the positive definitiveness of the quadratic constraint is not made, as it is not always the case in practice, and we demonstrate how to construct tight relaxations whether the matrix is positive definite or not. The structure of the Riemannian manifold is exploited, and we describe how to compute all elements required for implementing a subgradient Riemannian descent algorithm.

The resulting method that we propose, referred to as APLA-RSG, consists of i) finding a lower bound and a first candidate solution through the solution of a relaxation of the problem—the APLA part—and ii) projecting this candidate to the feasible set and locally improving it with a Riemannian subgradient descent—the RSG part.

Numerical experiments illustrate that the method reaches a competitive objective as quickly as other methods from the literature. However, APLA-RSG benefits from other advantages, *e.g.*, it provides a lower bound and strictly satisfies the balance constraint. The lower bound allows the estimation of an optimality gap, even if the problem is nonconvex. Such a lower bound can also be useful for other methods, so as to assess whether derived solutions are of acceptable quality. Additionally, if the computed objective is below the lower bound, this suggests that the solution is infeasible.

Further work may include the following extensions. Firstly, we are interested in considering a more complex model: prohibited operation zones (POZ) [LB93] could be easily applied to APLA, but then the local search (RSG) will be limited to a given connected subset of the feasible set. Secondly, a better way of converting the infeasible solution of APLA to a feasible one is of interest: currently, it is possible to strongly deteriorate the performance in terms of objective value at the projection step. Moreover, this projection step costs about 35% of the total execution time, being therefore one of the bottlenecks. This second extension is the focus of the second part of the thesis. Finally, the extension of the method to the optimal power flow problem and more specifically convex or nonconvex ACOPF could be contemplated. This last extension will open the door to other interesting problems, such as the security-constrained optimal power flow (SCOPF). Indeed, if generator contingencies can be dealt with the current method—the main difference being an increase in the number of variables and constraints, *e.g.*, failing the largest unit amounts to doubling the number

of variables—dealing with line contingencies, such as the N-1 criterion, requires a direct representation of the network.

PART II
Projection onto quadrics

5

Projection onto a quadric

In this chapter, we study the projection of a given point onto a nonsingular quadratic hypersurface, or nonsingular *quadric*. Quadrics are a natural generalization of hyperplanes. The projection onto a quadric appears, *e.g.*, for the projection of a point onto the intersection of a quadric—or the Cartesian product of quadrics—and a polytope. This problem has a direct application: the projection step in the three-step method from Chapter 4. This application is tackled in Chapter 6. Other applications of quadratic projections emerge in the context of the security region of gas networks [SXZ+21] or in local learning methods [Bro20].

It is therefore intriguing that few studies of this problem can be found in the literature. Indeed, projections onto quadratic surfaces have been studied for the 2D and 3D cases, see *e.g.*, [MS13, LI14, HWCH20]. However, to the best of our knowledge, the extension to an arbitrary dimension has not been pursued, except for the short discussion at the end of [LI14] and in [SR20]. Although the method proposed in [SR20] can handle the singular case, *i.e.*, the case where the matrix that defines the quadratic surface is singular, it does not always return the exact projection. Moreover, the two-level iterative scheme that is proposed in [SR20] can be computationally expensive.

Using the method of Lagrange multipliers, we reduce this quadratically constrained quadratic program (QCQP) to the problem of finding the roots of a nonlinear real function. Then, we completely characterize the solutions

to the nonconvex projection onto a non-cylindrical central quadric and compute one of these solutions as either the (unique) root of a univariate scalar function on a computable interval or among a finite set of closed-form solutions. We also provide a suitable starting point for finding this root using Newton's method, which guarantees a quadratic convergence. In this way, the proposed method provides an efficient way to obtain the exact projection onto a nonsingular quadric. Finally, to further reduce execution time, we also introduce a heuristic based on a geometrical construction. This allows us to quickly map a point to the feasible set without having to diagonalize the matrix used to define the quadric. We detail two variants of this heuristic.

The projection considered here is not unique in general, due to the nonconvexity of the feasible set. This implies that we cannot solely rely on first or second-order line-search schemes, since such local methods may converge to a local minimum. This projection can also be handled by black-box (commercial) solvers, *e.g.*, Gurobi or Ipopt [Gur18, WB06]. However, these methods suffer from two main problems: i) the execution time rockets when the dimension of the problem increases to mid or large-scale size—this phenomenon is present in the numerical results from Chapter 6—and ii) Gurobi is not a local method and does not exploit the local structure of the problem; in certain applications, the starting point is close to the feasible set. The first problem is highlighted by the power system application that is considered here, where the projection step is only a small part of the overall procedure, which renders execution time an important factor in our analysis.

We note that our proposed approach for projecting onto a quadric is not unusual. For example, [GVL13, §6.2.1] use a similar construction for the problem of least squares minimization over a sphere. Nonetheless, this problem is easier to tackle than ours, since the (unique) solution to this convex problem is the (unique) root of the *secular equation* defined by the KKT conditions. In [CGT00, §7.3], the authors also use a similar procedure and taxonomy of secular equations for finding the ℓ_2-norm model minimizer. Their discussion is partially similar to what is proposed in this chapter, *i.e.*, searching for a specific root of a given univariate scalar-valued nonlinear function on a specific domain. However, the domain and the function are different in our work. Moreover, our analysis on degenerate cases is not present in [CGT00], since such cases do not appear in the problem that the authors study, which is linked to the trust-region subproblem.

After the formulation of the problem (Section 5.1), we compute its KKT

conditions (Section 5.2). We then deal separately with the nondegenerate ellipsoid case (Section 5.3) and the nondegenerate hyperboloid case (Section 5.4). Both degenerate cases are also tackled (Section 5.5), and we unify all these cases into a single algorithm (Section 5.6). Finally, we detail a heuristic based on a geometrical construction: the *quasi-projection* (Section 5.7). The conclusions are drawn for the two chapters of this second part together in Section 6.5.

This chapter is based on the *submitted* paper [VAP22a].

5.1 Problem formulation

In this section, the problem of interest is introduced. This problem consists in the projection of a given point $\tilde{\boldsymbol{x}}^0 \in \mathbb{R}^n$ onto a feasible set $\mathcal{Q}$:

$$\begin{aligned} &\min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{x} - \tilde{\boldsymbol{x}}^0\|_2^2 \\ &\text{subject to } \boldsymbol{x} \in \mathcal{Q}, \end{aligned} \tag{5.1}$$

where $\mathcal{Q}$ is a nonempty and non-cylindrical central quadric [OSG20, Theorem 3.1.1]. In other words, $\mathcal{Q}$ is nonempty and there exists a quadratic function

$$\Psi\colon \mathbb{R}^n \to \mathbb{R}\colon \boldsymbol{x} \mapsto \Psi(\boldsymbol{x}) = \boldsymbol{x}^\intercal \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^\intercal \boldsymbol{x} + c,$$

with $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ symmetric and nonsingular, $\boldsymbol{b} \in \mathbb{R}^n$, and $(\boldsymbol{b}^\intercal \boldsymbol{A}^{-1}\boldsymbol{b})/4 \neq c \in \mathbb{R}$, such that

$$\mathcal{Q} = \{\boldsymbol{x} \in \mathbb{R}^n \mid \Psi(\boldsymbol{x}) = 0\} = \Psi^{-1}(0). \tag{5.2}$$

See [OSG20, §3.1] and [AHK+08, Chapter 21] for a complete classification of quadrics.

This quadratic surface, or *quadric*, is denoted as type-II quadric or quadric with middle point, in the sense of [AHK+08]. The middle point or *center* $\boldsymbol{d}$, corresponding to the center of symmetry, is computed as $-(\boldsymbol{A}^{-1}\boldsymbol{b})/2$, and the condition $c \neq (\boldsymbol{b}^\intercal \boldsymbol{A}^{-1}\boldsymbol{b})/4$ is equivalent to $\boldsymbol{d} \notin \mathcal{Q}$. Under these assumptions, we can prove that the feasible set defined by (5.2) is a manifold, see remark 4.3. The quadric center and the characterization of the surface as a manifold will be used in Section 5.7 to build a fast but inexact projection mapping referred to as a *quasi-projection*.

The problem defined by (5.1) is invariant to translations and rotations. Let $\boldsymbol{\lambda} = \text{spec}(\boldsymbol{A})$ be the eigenvalues of $\boldsymbol{A}$ sorted in descending order and $\boldsymbol{x}^0$ an appropriate transformation of $\tilde{\boldsymbol{x}}^0$. Without loss of generality, we can consider the following problem in *normal form* [OSG20, Theorem 3.1.1].

Projection onto a quadric in normal form

$$\begin{aligned} &\min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{x} - \boldsymbol{x}^0\|_2^2 \\ &\text{subject to } \sum_{i=1}^n \lambda_i x_i^2 - 1 = 0 \end{aligned} \tag{5.3}$$

Since the feasible set is nonempty, we have $\lambda_1 > 0$. We will refer to the solution(s) of (5.3) as the *(true) projection(s)* of $\boldsymbol{x}^0$ onto the quadric. Note that the center $\boldsymbol{d}$ is now the origin $\mathbf{0}$.

Because this problem is also symmetric with respect to the axes, we consider without loss of generality that $\boldsymbol{x}^0 \geq 0$, *i.e.*, $\boldsymbol{x}^0$ is located inside the first orthant ($\mathbb{R}^n_+ := \{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{x} \geq 0\} = \mathbb{R}^{n,*}_+ \bigcup \{\mathbf{0}\}$).

In the case where $\lambda_n > 0$, *i.e.*, the quadric is an ellipsoid, and if we have

$$\sum_{i=1}^n \lambda_i (x_i^0)^2 - 1 > 0,$$

then the solution to (5.3) is identical to the solution to

$$\begin{aligned} &\min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{x}^0 - \boldsymbol{x}\|_2^2 \\ &\text{subject to } \sum_{i=1}^n \lambda_i x_i^2 - 1 \leq 0, \end{aligned}$$

which is a *convex* optimization problem that is easy to solve, *e.g.*, using interior-point methods (IPM)—see [BV04, Nes18] for more details—or a black-box commercial solver like Gurobi [Gur18]. On the other hand, if $\boldsymbol{A}$ is indefinite or if $\sum_{i=1}^n \lambda_i (x_i^0)^2 - 1 < 0$, then we are confronted with a nonconvex optimization problem.

Let us now prove that the quest of this chapter is not vain and show the existence of a global optimum of (5.3).

Proposition 5.1. *There exists a global optimum $\boldsymbol{x}^*$ of (5.3).*

Proof. The objective of (5.3) is a real-valued, continuous and coercive function defined on a nonempty closed set, implying thereby the existence of a

global optimum [Bec14, Theorem 2.32]. □

REMARK 5.1 (Two b or not two b?). For the sake of being consistent with our papers, the quadratic function used in the definition of the quadric in § 4.1.4 differs from the one in the present chapter. There is a factor two in the linear term. Hence, the "$\boldsymbol{b}$" in Chapter 4 corresponds to "$2\boldsymbol{b}$" in the present chapter. The reader should therefore not be surprised that the center previously computed as $\boldsymbol{d} = -\boldsymbol{A}^{-1}\boldsymbol{b}$ becomes here $\boldsymbol{d} = -(\boldsymbol{A}^{-1}\boldsymbol{b})/2$. ■

5.2 KKT conditions

Since the feasible set is nonconvex, the projection operator does not always return a singleton, see [FM15, Theorem 3.8]. The set of solutions may be a singleton (Fig. 5.1), a finite set (Fig. 5.5), or an infinite set (suppose that $\boldsymbol{x}^0$ is the center of a sphere, *i.e.*, $\boldsymbol{\lambda} = \mathbb{1}$ and $\boldsymbol{x}^0 = \boldsymbol{0}$).

Using the KKT conditions, we can characterize the solutions to (5.3). The Lagrangian of (5.3), with Lagrange multiplier μ and with $\boldsymbol{D} = \operatorname{diag}(\boldsymbol{\lambda}) \in \mathbb{R}^{n\times n}$, reads

$$\mathcal{L}(\boldsymbol{x}, \mu) = (\boldsymbol{x} - \boldsymbol{x}^0)^\mathsf{T}(\boldsymbol{x} - \boldsymbol{x}^0) + \mu(\boldsymbol{x}^\mathsf{T}\boldsymbol{D}\boldsymbol{x} - 1), \tag{5.4}$$

and the gradient,

$$\boldsymbol{\nabla}\mathcal{L}(\boldsymbol{x}, \mu) = \begin{pmatrix} 2(\boldsymbol{x} - \boldsymbol{x}^0) + 2\mu\boldsymbol{D}\boldsymbol{x} \\ \boldsymbol{x}^\mathsf{T}\boldsymbol{D}\boldsymbol{x} - 1 \end{pmatrix}. \tag{5.5}$$

Each point $(\boldsymbol{x}, \mu)$ that satisfies

$$\boldsymbol{\nabla}\mathcal{L}(\boldsymbol{x}, \mu) = \boldsymbol{0} \tag{5.6}$$

is referred to as a *KKT point*. For $i \in [n]$, we can write the ith equation of (5.6) as

$$x_i(1 + \mu\lambda_i) = x_i^0. \tag{5.7}$$

Every optimal solution $\boldsymbol{x}^*$ of (5.3) must either meet the KKT conditions (5.6) or fail to satisfy the linear independence constraint qualification (LICQ) criterion; more details about the optimality conditions can be found in [BSS06, Chapter 4]. In (5.3), the latter occurs if $\nabla\Psi(\boldsymbol{x}^*) = \boldsymbol{0}$. This corresponds to

the case where the center belongs to the quadric and is ruled out by the condition $c \neq (\boldsymbol{b}^\intercal \boldsymbol{A}^{-1}\boldsymbol{b})/4$.

Isolating $\boldsymbol{x}$ in the first n equations of (5.6) yields

$$\boldsymbol{x}(\mu) = (\boldsymbol{I} + \mu\boldsymbol{D})^{-1}\boldsymbol{x}^0, \tag{5.8}$$

for $\mu \notin \pi(\boldsymbol{A}) := \left\{-\frac{1}{\lambda} \mid \lambda \text{ is an eigenvalue of } \boldsymbol{A}\right\}$ and the ith component can be rewritten as

$$x_i(\mu) = \frac{x_i^0}{1 + \mu\lambda_i}. \tag{5.9}$$

We distinguish two cases:

- Case 1: $\mu \notin \pi(\boldsymbol{A})$. The matrix $(\boldsymbol{I} + \lambda\boldsymbol{D})$ is nonsingular and we have (5.8).
- Case 2: $\mu = -1/\lambda_i$ for some $i \in [n]$. The ith equation of (5.6) reads $-2x_i^0 = 0$; consequently, μ is a solution only if $x_i^0 = 0$.

We first treat the case $\boldsymbol{x}^0 > 0$, denoted as *nondegenerate case*, in Sections 5.3 and 5.4. Next, we tackle the *degenerate case* $\boldsymbol{x}^0 \geq 0$ in Section 5.5.

Inserting (5.8) in the quadric equation, $\Psi(\boldsymbol{x}) = 0$, we obtain a *univariate, extended-real-valued* function

$$\begin{aligned} f\colon \mathbb{R} \to \overline{\mathbb{R}}\colon \mu \mapsto f(\mu) = \Psi(\boldsymbol{x}(\mu)) &= \boldsymbol{x}(\mu)^\intercal \boldsymbol{D}\boldsymbol{x}(\mu) - 1 \\ &= \sum_{i=1, x_i^0 \neq 0}^{n} \lambda_i \left(\frac{x_i^0}{1 + \mu\lambda_i}\right)^2 - 1, \end{aligned} \tag{5.10}$$

of which we want to obtain the roots. Since these roots correspond to the values μ^* for which $\Psi(\boldsymbol{x}(\mu^*)) = 0$, each root can be geometrically understood as the intersection of $\{\boldsymbol{x}(\mu) : \mu \in \mathbb{R}\}$ and the quadric $\mathcal{Q} = \Psi^{-1}(0)$. This is illustrated in the examples in Sections 5.3 and 5.4. Note that the set $\pi(\boldsymbol{A})$ corresponds to the *poles* of the rational function (5.10).

In the following, we show how to efficiently solve (5.3) by computing a specific root of (5.10). We first consider the case where $\boldsymbol{x}^0 > 0$ for the ellipsoid case in Section 5.3 and the hyperboloid case in Section 5.4. We then deal with the case $\boldsymbol{x}^0 \geq \boldsymbol{0}$ in Section 5.5. Finally, we bring everything together into a single algorithm, Algorithm 8, in Section 5.6.

5.3 Nondegenerate ellipsoid case, $\mathbf{x}^0 > \mathbf{0}$

Throughout this section, we assume that the quadric is an ellipsoid ($\boldsymbol{\lambda} > 0$) and that the initial point lies (strictly) in the first orthant ($\boldsymbol{x}^0 > \mathbf{0}$).

The goal of this section is twofold. First, we derive several successive results (Propositions 5.2 to 5.5) that characterize the roots of f and the solutions to (5.3). The combination of these results yields Proposition 5.6 which states that *(5.3) can be solved by finding the unique root of f on a given interval $\mathcal{I}$*. Second, we provide a starting point for the Newton root-finding algorithm for efficiently computing this root.

Proposition 5.2. *Under the standing assumptions, every solution $\boldsymbol{x}^*$ of (5.3) satisfies $\boldsymbol{x}^* > 0$.*

Proof. Recall that since no points fulfill the LICQ criterion, each solution to (5.3) is a KKT point. Using (5.7) we see that if $(\boldsymbol{x}^*, \mu^*)$ is a KKT point, then the positivity of x_i^0 for all $i \in [n]$ implies that $x_i^* \neq 0$.

Let us suppose, for the sake of contradiction, that $\boldsymbol{x}^*$ is a minimizer of (5.3) and that there exists a nonempty set of indices $J \subseteq [n]$ with $x_j^* < 0$ for all $j \in J$. By symmetry, we can construct $\boldsymbol{x}^{**}$ defined as

$$\boldsymbol{x}^{**} := \begin{cases} x_i^* & \text{if } i \notin J, \\ -x_i^* & \text{if } i \in J. \end{cases}$$

We have $\Psi(\boldsymbol{x}^{**}) = 0$, *i.e.*, the point belongs to the quadric and $\boldsymbol{x}^{**} > 0$. The (squared) objective can be computed:

$$\begin{aligned} \|\boldsymbol{x}^{**} - \boldsymbol{x}^0\|_2^2 &= \sum_{i=1}^{n} (x_i^0 - x_i^{**})^2 \\ &= \sum_{i=1, i\notin J}^{n} (x_i^0 - x_i^*)^2 + \sum_{j \in J} \underbrace{(x_j^0 + x_j^*)^2}_{< (x_j^0 - x_j^*)^2} \\ &< \|\boldsymbol{x}^* - \boldsymbol{x}^0\|_2^2. \end{aligned}$$

This contradicts the optimality of $\boldsymbol{x}^*$. □

Proposition 5.3. *f, defined as in (5.10), is strictly decreasing on*

$$\mathcal{I} := \left]-1/\lambda_1, +\infty\right[.$$

Proof. Since $f \in \mathcal{C}^1$ on $\mathcal{I}$, we compute

$$f'(\mu) = -2 \sum_{i=1}^{n} \frac{(\lambda_i x_i^0)^2}{\underbrace{(1+\mu\lambda_i)^3}_{>0 \text{ for } \mu \in \mathcal{I}}}, \tag{5.11}$$

and this function is negative on $\mathcal{I}$. □

Proposition 5.4. *Function f restricted to $\mathcal{I}$ has one and only one zero.*

Proof. By Proposition 5.3, f is strictly decreasing. Hence, it has *at most* one zero.

Let us know prove the existence of the zero. We want to find a closed interval $[a, b] \subset \mathcal{I}$ with $f(a) \geq 0$ and $f(b) \leq 0$, so that we can conclude with the intermediate value theorem. For all $\mu \in \mathcal{I}$, the following holds

$$\begin{aligned} f(\mu) &= \sum_{i=1}^{n} \lambda_i \left(\frac{x_i^0}{1+\mu\lambda_i} \right)^2 - 1 \\ &\geq \lambda_1 \left(\frac{x_1^0}{1+\mu\lambda_1} \right)^2 - 1. \end{aligned}$$

For the left bound, we can take the largest roots of the latter formula:

$$a = \frac{x_1^0}{\sqrt{\lambda_1}} - \frac{1}{\lambda_1}.$$

We have $f(a) \geq 0$ and $a \in \mathcal{I}$. For the right bound, we verify that for $i \in [n]$ and

$$\mu \geq x_i^0 \sqrt{\frac{n}{\lambda_i}} - \frac{1}{\lambda_i},$$

we have

$$\lambda_i \left(\frac{x_i^0}{1+\mu\lambda_i} \right)^2 - 1 \leq \frac{1}{n}.$$

Therefore, if we take

$$b = \max_{i \in [n]} x_i^0 \sqrt{\frac{n}{\lambda_i}} - \frac{1}{\lambda_i},$$

then $b \in \mathcal{I}$, $b > a$, and $f(b) \leq 0$.

□

Proposition 5.5. *If μ^* is a root of f and $\mu^* \notin \mathcal{I}$, then $\boldsymbol{x}(\mu^*) \notin \mathbb{R}^n_+$.*

Proof. If μ^* is a root of f and $\mu^* \notin \mathcal{I}$, then $\mu^* < -1/\lambda_1$. The first component of $\boldsymbol{x}(\mu^*)$ reads

$$x_1(\mu^*) = \frac{x_1^0}{1 + \mu^* \lambda_1}. \tag{5.12}$$

As the denominator is negative, $\boldsymbol{x}(\mu^*)$ belongs to a different orthant than $\boldsymbol{x}^0$. □

Proposition 5.6. *If $x_i^0 \neq 0$ for all $i \in [n]$ and $\boldsymbol{\lambda} > 0$, then the optimal solution to (5.3) is given by the unique root μ^* of f restricted to $\mathcal{I}$.*

Proof. As shown in Section 5.2, the optimal solution $\boldsymbol{x}^*$ of this nondegenerate case is a KKT point, meaning that it satisfies (5.6). Using Proposition 5.2, $\boldsymbol{x}^*$ belongs to the same orthant as $\boldsymbol{x}^0$. We are therefore interested in the best KKT solution in the first orthant. However, Proposition 5.5 shows that the corresponding μ^* of the KKT solutions belonging to the same orthant as $\boldsymbol{x}^0$ are located in $\mathcal{I}$ and Proposition 5.4 proves the existence and uniqueness of a root on $\mathcal{I}$ that corresponds thereby to the optimal solution to (5.3). □

Proposition 5.7. *f is strictly convex on $\mathcal{I}$.*

Proof. $f \in \mathcal{C}^2(\mathcal{I})$ and we compute

$$f''(\mu) = 6 \sum_{i=1}^{n} \frac{(\lambda_i x_i^0)^2 \lambda_i}{(1 + \mu \lambda_i)^4},$$

which is positive on $\mathcal{I}$.

□

Proposition 5.8. *Let $\mu^0 \in \mathcal{I} =]-1/\lambda_1, +\infty[$ with $f(\mu^0) > 0$. The Newton-Raphson algorithm with starting point μ^0 converges to μ^*, the unique root of f on $\mathcal{I}$ (as in Proposition 5.6).*

Proof. We first prove by induction on the index k that the sequence $(\mu^k)_{k \in \mathbb{N}}$ provided by Newton's method is an increasing sequence upper bounded by μ^*. Next, we conclude using the strict convexity of the function f.

The sequence $(\mu^k)_{k \in \mathbb{N}}$ is increasing The Newton-Raphson iterate for $k = 0, 1, \ldots$ is given by

$$\mu^{k+1} = \mu^k - \frac{f(\mu^k)}{f'(\mu^k)}. \tag{5.13}$$

The base case $k = 0$ follows from the positiveness of $f(\mu^0)$ (by assumption) and the decrease of f on $\mathcal{I}$ (by Proposition 5.3). Using the induction hypothesis, which implies that $f(\mu^k) > 0$ for $\mu^k \neq \mu^*$, and Proposition 5.3, we have

$$\mu^k < \mu^{k+1}.$$

Remainder of the proof Since f is strictly convex on $\mathcal{I}$ (Proposition 5.7), the tangent of f at a given point is below every chord starting from this point. In particular, we have

$$f'(\mu^k) < \frac{f(\mu^k) - f(\mu^*)}{\mu^k - \mu^*},$$

Using the definition of μ^* and rearranging, we obtain

$$\begin{aligned} \mu^k - \frac{f(\mu^k)}{f'(\mu^k)} &< \mu^*, \\ \mu^{k+1} &< \mu^*. \end{aligned}$$

Since the sequence $(\mu^k)_{k\in\mathbb{N}}$ is strictly increasing (for $\mu^k \neq \mu^*$) and bounded, it must converge to a fixed point of (5.13) that corresponds to a root of f. This concludes the proof as, by Proposition 5.6, there is a unique root of f on $\mathcal{I}$ corresponding to the optimal solution to (5.3). □

2D example of a nondegenerate projection onto an ellipse Figure 5.1 presents an example of a nondegenerate projection (*i.e.*, with $\boldsymbol{x}^0 > 0$) onto an ellipse. We plot $\boldsymbol{x}(\mu)$, $f(\mu)$, $f'(\mu)$ and $\|\boldsymbol{x}(\mu) - \boldsymbol{x}^0\|_2$ for μ ranging on $]-\infty, \infty[$. Let us describe how $\boldsymbol{x}(\mu)$, in the top left subfigure, varies when μ decreases from $+\infty$ to $-\infty$. For $\mu \to +\infty$, we have $\boldsymbol{x}(\mu) = \boldsymbol{d}$, where $\boldsymbol{d} = \boldsymbol{0}$ is the quadric center depicted as a blue dot. Then, while decreasing μ to 0, we reach $\boldsymbol{x}(0) = \boldsymbol{x}^0$. For $\mu \to -1/\lambda_1$, $\boldsymbol{x}(\mu)$ follows an asymptote *and crosses the quadric on* $\boldsymbol{x}(\mu^*)$, the optimal solution to (5.3), depicted as a purple triangle. Further decreasing μ, $\boldsymbol{x}(\mu)$ reappears on the left part of the asymptote ($x_1 \to -\infty$) and tends to the asymptote ($x_2 \to +\infty$) defined by the other eigenvalue. Finally, $\boldsymbol{x}(\mu)$ converges to the quadric center when $\mu \to -\infty$, passing again through the quadric in $\boldsymbol{x}(\mu^{**})$, the maximum of (5.3), depicted as a purple square.

The function f is also depicted with its two roots. Note that, depending on the parameters of the problem, it may have one or two additional roots

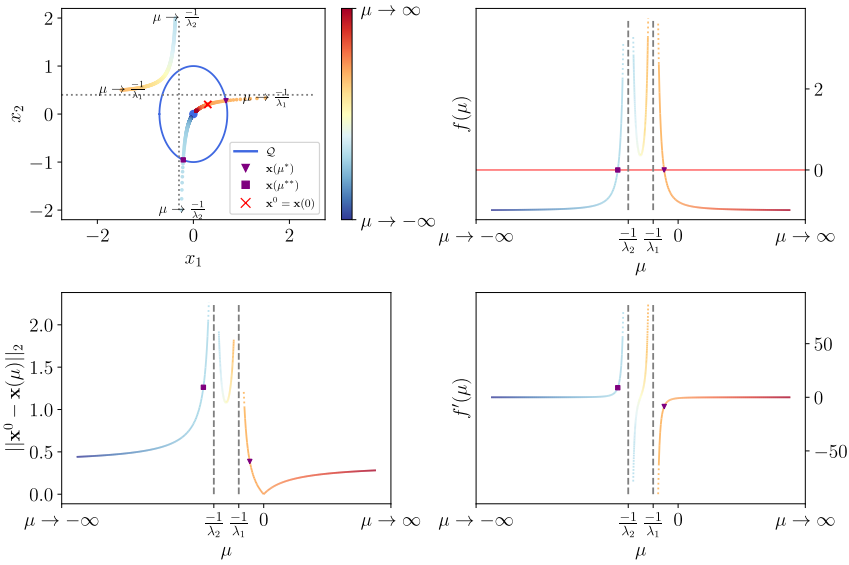


Fig. 5.1 Image of $\boldsymbol{x}(\mu)$ and graphs of $f(\mu)$, $f'(\mu)$, $\|\boldsymbol{x}(\mu) - \boldsymbol{x}^0\|_2$ for μ ranging on $]-\infty, +\infty[$ in the nondegenerate elliptic case. The unique solution to (5.3) is $\boldsymbol{x}(\mu^*)$.

corresponding to local minima or maxima. Graphically, these local optima will be located at the intersection points obtained when shifting the top left branch of the image of $\boldsymbol{x}(\mu)$. We also observe in the bottom right figure that $f'(\mu)$ is negative on $\mathcal{I}$ and show the distance to $\boldsymbol{x}^0$ for the different values of $\boldsymbol{x}(\mu)$ in the bottom left figure.

5.4 Nondegenerate hyperboloid case, $\mathbf{x}^0 > \mathbf{0}$

In the hyperboloid case, there is at least one positive and one negative eigenvalue of $\boldsymbol{A}$. Let $1 \leq p \leq n-1$ be the number of positive eigenvalues. Let us consider $e_1 := -1/\lambda_1$ and $e_2 := -1/\lambda_n$. We have $0 \in]e_1, e_2[$. We will work analogously as in the ellipsoidal case but with $\mathcal{I} :=]e_1, e_2[$. Proposition 5.2 has no assumptions about the positivity of λ and remains therefore valid. The other propositions can be successively adapted: Proposition 5.9

adapts Proposition 5.3, Proposition 5.10 adapts Proposition 5.4, Proposition 5.11 adapts Proposition 5.5, and finally the main result remains valid, *i.e.*, Proposition 5.12 adapts Proposition 5.6.

We then propose Algorithm 7, also based on Newton-Raphson, to efficiently compute the root of f in $\mathcal{I}$ and the optimal solutions to (5.3). An example of the hyperbolic case is provided in Fig. 5.3.

Proposition 5.9. *f, defined as in (5.10), is strictly decreasing on $\mathcal{I} :=]e_1, e_2[$.*

Proof. Since $f \in \mathcal{C}^1(\mathcal{I})$, we compute

$$f'(\mu) = -2\sum_{i=1}^{p} \underbrace{\frac{(\lambda_i x_i^0)^2}{(1+\mu\lambda_i)^3}}_{>0 \text{ if } \mu > e_1} - 2\sum_{i=p+1}^{n} \underbrace{\frac{(\lambda_i x_i^0)^2}{(1+\mu\lambda_i)^3}}_{>0 \text{ if } \mu < e_2}, \tag{5.14}$$

and this function is negative on $\mathcal{I}$.

□

Proposition 5.10. *Function f restricted to $\mathcal{I}$ with $\boldsymbol{x}^0 > 0$ has one and only one zero.*

Proof. By Proposition 5.9, f is strictly decreasing. Hence, it has *at most* one zero. Moreover, $\lim_{\mu \to e_1} f(\mu) = +\infty$ and $\lim_{\mu \to e_2} f(\mu) = -\infty$; the continuity of f on $\mathcal{I}$ implies the existence of the zero on $\mathcal{I}$[1].

□

Proposition 5.11. *If μ^* is a root of f and $\mu^* \notin \mathcal{I}$, then $\boldsymbol{x}(\mu^*) \notin \mathbb{R}^n_+$.*

Proof. If $\mu^* \notin \mathcal{I}$, then either $\mu^* < -1/\lambda_1$ or $\mu^* > -1/\lambda_n$. The first case is already treated in the proof of Proposition 5.5. For the second case, we note that

$$x_n(\mu^*) = \frac{x_n^0}{1+\mu^*\lambda_n}. \tag{5.15}$$

As $\lambda_n < 0$, the denominator is negative for $\mu^* > -1/\lambda_n$, and thus $\boldsymbol{x}(\mu^*)$ belongs to a different orthant than $\boldsymbol{x}^0$. □

Proposition 5.12. *If $x_i^0 \neq 0$ for all $i \in [n]$, then the unique optimal solution to (5.3) is given by the unique root μ^* of f restricted to $\mathcal{I} :=]e_1, e_2[$.*

[1] To be more rigorous, one can find an explicit closed interval $[a, b] \in \mathcal{I}$ as in the proof of Proposition 5.4.

Proof. Since Propositions 5.2, 5.4 and 5.5 are also valid in the hyperboloid case with $\mathcal{I} :=]e_1, e_2[$, the proof is identical to Proposition 5.6. □

Proposition 5.13. *There exists a unique inflection point μ^{I} of f on $\mathcal{I} :=]e_1, e_2[$.*

Proof. The existence of the inflection point follows from the continuity of f'' on $\mathcal{I}$ and because $\lim_{\mu \to e_1^+} f''(\mu) = -\lim_{\mu \to e_2^-} f''(\mu) = +\infty$. The unicity holds because f'' is strictly decreasing: $f'''(\mu) < 0$ for all $\mu \in \mathcal{I}$. □

Since there is a single inflection point μ^{I}, we can launch in parallel two Newton's algorithms and guarantee that at least one will converge.

Algorithm 7 Double Newton

```
if f(0) < 0 then
    use bisection method (see [BF01, Chapter 2.1]) to find μ_s ∈]e_1, 0[ such that f(μ_s) > 0
else if f(0) > 0 then
    use bisection method to find μ_s ∈]0, e_2[ such that f(μ_s) < 0
else
    return 0
end if
μ_0 ← Newton(0)                ▷ This and the next line are run in parallel
μ_1 ← Newton(μ_s)
return μ_0, μ_1                ▷ Returns the output solution from the first of the two parallel Newtons that is finished
```

Proposition 5.14. *One of the two Newton's methods of Algorithm 7 converges to μ^* the unique root of f.*

Proof. This proof relies on the double initiation of Newton's method in Algorithm 7: one starting from a positive value and the other from a negative value. We comment on the sign of $f(\mu^{\mathrm{I}})$.

- If $f(\mu^{\mathrm{I}}) < 0$, then $\mu^* \in]e_1, \mu^{\mathrm{I}}[$ and the function is strictly convex on this interval. The situation is similar to Proposition 5.8, and each starting point μ_s with $f(\mu_s) > 0$ is a valid starting point, in the sense that the sequence of iterates converges to μ^*.
- If $f(\mu^{\mathrm{I}}) > 0$, then $x^* \in]\mu^{\mathrm{I}}, e_2[$ and the function is strictly concave on this interval. Using the same argument as in Proposition 5.8, each starting point μ_s with $f(\mu_s) < 0$ is a valid starting point.

- If $f(\mu^{\mathrm{I}}) = 0$, each starting point in $\mathcal{I}$ is a valid starting point.

□

Note that with the knowledge of the value of μ^{I}, we could launch a single Newton scheme with the appropriate starting point. Unfortunately, computing μ^{I} amounts to computing the root of f'' which is at least as costly as finding the root of f.

In Fig. 5.2, we depict the rate of convergence of Newton's method for 100 projection instances. Each curve represents the distance to the optimal solution (assumed to be the last iterate) of a given instance. The instances that converge from the starting point 0 are depicted in blue and the other in red. We effectively observe that the convergence is superlinear and that most instances are solved in less than ten iterations (about 0.0005 s). The number of iterations does not depend on the dimension. However, increasing the dimension results in an increase in the execution time because evaluating the function f depends on the dimension of the problem: it costs $\mathcal{O}(n)$ to compute $f(\mu)$ or $f'(\mu)$. Running the same experiment as Fig. 5.2 with $n = 1000$, we obtain a (mean) execution time of about 0.1 s. This execution time is negligible with respect to the time required to compute the eigendecomposition.

Finally, if we combine the results of Section 5.3 with the present section, we can derive a sufficient condition for the uniqueness of the projection.

Proposition 5.15. *If $\boldsymbol{x}^0 > 0$ and $\mathcal{Q}$ is a non-cylindrical central quadric, then the solution to (5.3) is unique.*

Proof. The proof follows directly from the uniqueness property of Propositions 5.6 and 5.12. □

REMARK 5.2 (On the unicity of the projection). In the next section, we study the (degenerate) case where $x_i^0 = 0$ for some $i \in [n]$. This *may* yield multiple solutions, see Figs. 5.5 and 5.6 where another solution can be obtained by reflecting the green point over one of the axis, or a single one, see Fig. 5.7. ■

2D example of a nondegenerate projection onto a hyperbola Figure 5.3 shows an example of a nondegenerate projection onto a hyperbola. We observe a similar image of $\boldsymbol{x}(\mu)$, with two asymptotes. We see that $f(\mu)$ has a unique inflection point on $\mathcal{I}$. In this example, the inflection point is to the right of the root: $\mu^{\mathrm{I}} > \mu^*$. It follows from the monotonicity of the function

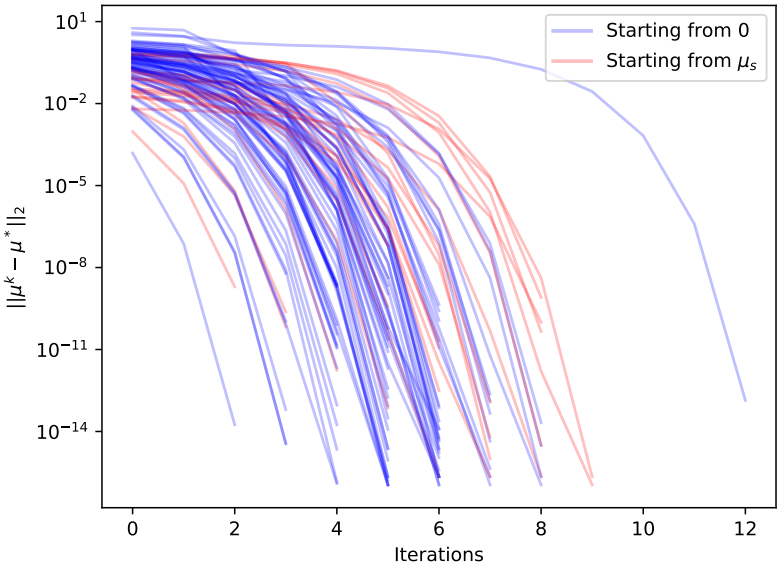


Fig. 5.2 Illustration of the superlinear convergence of Newton during 100 projection instances onto a 5-dimensional quadric.

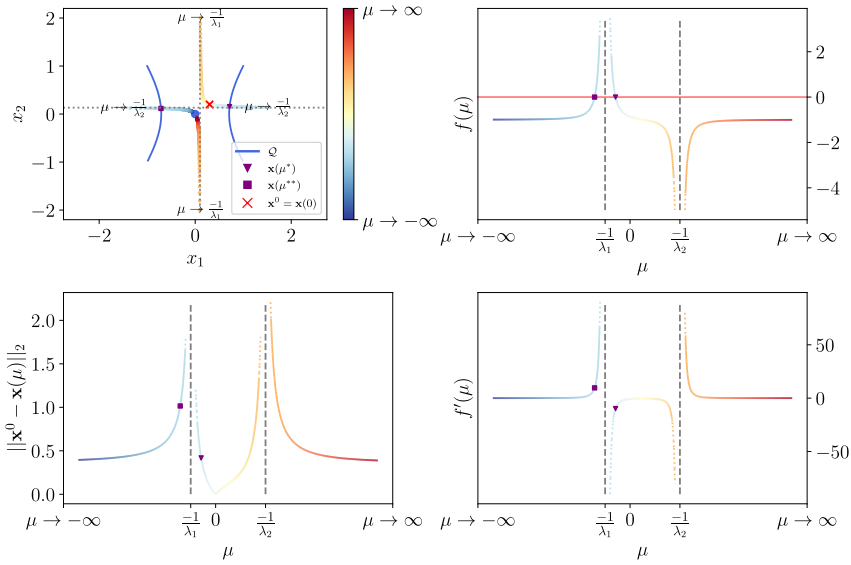


Fig. 5.3 Image of $\boldsymbol{x}(\mu)$ and graphs of $f(\mu)$, $f'(\mu)$, $\|\boldsymbol{x}(\mu) - \boldsymbol{x}^0\|_2$ for μ ranging on $]-\infty, +\infty[$ in the nondegenerate hyperbolic case.

f that $f(\mu^{\mathrm{I}}) < 0$. Thus, starting a Newton-Raphson scheme in some μ_{s} with $f(\mu_{\mathrm{s}}) > 0$ yields a sequence that converges to μ^*.

5.5 Degenerate case, $\mathbf{x}^0 \geq \mathbf{0}$

Let us first assume that all eigenvalues of $\boldsymbol{A}$ are distinct, the case with repeated eigenvalues is treated at the end of the current section.

5.5.1 All eigenvalues are distinct

This case is identical to Section 5.4 except for the following points: i) f is continuous at $-1/\lambda_i$ if the associated component of $\boldsymbol{x}^0$ is equal to zero and ii) at most two additional KKT points can be obtained for each component of $\boldsymbol{x}^0$ that is equal to zero.

We tackle these issues as follows: First, we arbitrarily decide to single out a solution in the first orthant. Second, we change the definition of e_1 and e_2 to account for the continuity of f at $-1/\lambda_i$: $\lim_{\mu \to -1/\lambda_i} f(\mu) \neq \infty$ if $x_i^0 = 0$. Finally, we show how to analytically compute these additional solutions.

Let $I := [n]$, it is clear that Proposition 5.2 is not valid anymore if some entry of $\boldsymbol{x}^0$ is equal to zero. Indeed, let $K \subseteq I$ be some nonempty subset of I, such that $x_k^0 = 0$ for all $k \in K$. If $\boldsymbol{x}^*$ is an optimal solution that belongs to the same orthant as $\boldsymbol{x}^0$, then define $\boldsymbol{x}^{**}$ in the following way:

$$x_i^{**} = \begin{cases} x_i^* & \text{for all } i \notin K, \\ -x_i^* & \text{for all } i \in K. \end{cases} \tag{5.16}$$

This feasible point is also an optimal solution. In fact, up to $2^{|K|} - 1$ solutions outside the first orthant can be obtained by mirroring $\boldsymbol{x}^*$ along a selected set of components in K.

As we are interested in finding *one* of the optimal solutions, we can also restrict our search to the first orthant.

Proposition 5.16. *Given $\boldsymbol{x}^0 \geq \mathbf{0}$, there exists an optimal solution $\boldsymbol{x}^*$ of (5.3) such that $\boldsymbol{x}^* \geq \mathbf{0}$.*

Proof. Let $\boldsymbol{x}^*$ be an optimal solution. The existence of $\boldsymbol{x}^*$ follows from Proposition 5.1. Using a similar argument to the one in the proof of Proposition 5.2, we have $\operatorname{sign}(x_j^0) = \operatorname{sign}(x_j^*)$ for all $j \in I \setminus K$. Let

$$x_i^{**} = \begin{cases} -x_i^* & \text{for all } i \in K \text{ with } \operatorname{sign}(x_i^*) \neq \operatorname{sign}(x_i^0), \\ x_i^* & \text{elsewhere.} \end{cases}$$

This feasible point has the same objective as $\boldsymbol{x}^*$ and is located in the same orthant as $\boldsymbol{x}^0$. □

For $\mu \notin -\pi(\boldsymbol{A})$, the discussion is similar to what has been derived in the previous sections. However, we change the definition of e_1, e_2 as

$$\begin{aligned} e_1 &= \max_{\{i \in I \mid \lambda_i > 0, x_i^0 \neq 0\}} -\frac{1}{\lambda_i}, \\ e_2 &= \min_{\{i \in I \mid \lambda_i < 0, x_i^0 \neq 0\}} -\frac{1}{\lambda_i}, \end{aligned} \tag{5.17}$$

and $e_1 = -e_2 = -\infty$ if the max or min is empty. This takes into account the continuity of f at $\mu = -1/\lambda_i$ if $x_i^0 = 0$ for some index i.

Let us adapt Propositions 5.9 to 5.11 to the degenerate case.

Proposition 5.17. *f, defined as in (5.10), is strictly decreasing on $\mathcal{I} :=]e_1, e_2[$.*

Proof. Since $f \in \mathcal{C}^1(\mathcal{I})$, we compute

$$f'(\mu) = -2 \sum_{i=1, i\notin K}^{p} \frac{(\lambda_i x_i^0)^2}{\underbrace{(1+\mu\lambda_i)^3}_{>0 \text{ if } \mu > e_1}} - 2 \sum_{i=p+1, i\notin K}^{n} \frac{(\lambda_i x_i^0)^2}{\underbrace{(1+\mu\lambda_i)^3}_{>0 \text{ if } \mu < e_2}}, \tag{5.18}$$

and this function is negative on $\mathcal{I}$. □

Proposition 5.18. *Function f restricted to $\mathcal{I}$ has one and only one zero, if and only if there is some $i \in I^+ := \{i \in I \mid \lambda_i > 0\}$ with $x_i^0 \neq 0$.*

Proof. We first note that the technical assumption on x_i^0 ensures that

$$\lim_{\mu \to e_1} f(\mu) > 0.$$

For $\lim_{\mu \to e_2} f(\mu)$ we distinguish two cases:

- either $e_2 = +\infty$ and $\lim_{\mu \to e_2} f(\mu) = -1$;
- or $e_2 = \min_{\{i \in I \mid \lambda_i < 0, x_i^0 \neq 0\}} -1/\lambda_i$ and $\lim_{\mu \to e_2} f(\mu) = -\infty$.

Since in both cases the limit is negative and f is continuous and strictly decreasing on $\mathcal{I}$, there exists a unique zero on this interval[2].

Let us now show the converse *via* the contrapositive. Assume that there is no $i \in I^+$ with $x_i^0 \neq 0$, then $e_1 := -\infty$ and $\lim_{\mu \to e_1} f(\mu) = -1$. Since f is continuously decreasing on $\mathcal{I}$, it has no roots on this interval. □

Proposition 5.19. *If μ^* is a root of f and $\mu^* \notin \mathcal{I}$, then $\boldsymbol{x}(\mu^*) \notin \mathbb{R}^n_+$.*

Proof. If $\mu^* \notin \mathcal{I}$, then either $e_1 \neq -\infty$ and $\mu^* < e_1$ or $e_2 \neq +\infty$ and $\mu^* > e_2$. The proof follows from the definition of the e_i's. We treat the first case; the other can be handled similarly. Let $i_1 = \operatorname{argmax}_{\{i \in I \mid \lambda_i > 0, x_i^0 \neq 0\}} -1/\lambda_i$, we have

$$x_{i_1}(\mu^*) = \frac{x_{i_1}^0}{1 + \mu^* \lambda_{i_1}}.$$

[2] See Footnote 1 (Page 142).

This implies that $\boldsymbol{x}(\mu^*)$ belongs to a different orthant than $\boldsymbol{x}^0$ since the numerator is nonzero and the denominator is negative. □

If $x_i^0 = 0$ for all $i \in I^+$, meaning that the assumption on $\boldsymbol{x}^0$ of Proposition 5.18 does not hold, then $f(\mu)$ reads

$$f(\mu) = \sum_{i \in I^-} \lambda_i \left(\frac{x_i^0}{1 + \mu\lambda_i} \right)^2 - 1,$$

where $I^- := I \setminus I^+ = \{p+1, p+2, \ldots, n\}$. This function is negative on $\mathbb{R}$. In this specific case, *f does not provide any KKT point*. Such a situation is depicted in Fig. 5.6.

However, the problem remains solvable: there are two additional KKT points that appear when $\boldsymbol{x}^0$ is located on the axes. Indeed, if $\mu = -1/\lambda_k$ for $k \in K$, then the $k-$th entry of (5.6) reads

$$2(x_k - x_k^0) + 2\mu\lambda_k x_k = 0. \tag{5.19}$$

This expression is true for all x_k. Therefore, we obtain at most *two additional* solutions to the Lagrangian system (5.6). Geometrically, this corresponds to finding the intersection of i) a line perpendicular to the axis corresponding to the component k where $x_k^0 = 0$ and ii) the quadric. These solutions, if they exist, can be computed as

$$(\boldsymbol{x}_k^{\mathrm{d}})_i = \begin{cases} \dfrac{x_i^0}{1 - \frac{\lambda_i}{\lambda_k}} & \text{if } i \neq k, \\ \pm \sqrt{\dfrac{1}{\lambda_k} \left(1 - \displaystyle\sum_{j \in I, j \neq k} \lambda_j \left(\dfrac{x_j^0}{1 - \frac{\lambda_j}{\lambda_k}} \right)^2 \right)} & \text{if } i = k, \end{cases} \tag{5.20}$$

where $(\cdot)_i$ selects the $i-$th component, and we choose the "+" solution that lies in the first orthant.

Such a situation is depicted in Fig. 5.4. We observe that $\boldsymbol{x}(\mu)$ moves around the axis corresponding to the component of $\boldsymbol{x}^0$ which is equal to zero. Moreover, the two additional solutions are depicted in green in Figs. 5.4 and 5.5. In Fig. 5.4, the optimal solution is a root of f and in Fig. 5.5, it is the two additional solutions.

There are no intersection points—and no additional solutions (for a

given index $k \in K$) to (5.6)—if

$$\frac{1}{\lambda_k}\left(1 - \sum_{j\in I, j\neq k} \lambda_j \left(\frac{x_j^0}{1-\frac{\lambda_j}{\lambda_k}}\right)^2\right) < 0.$$

This situation is depicted in Fig. 5.7.

2D examples of degenerate projections Figs. 5.4 and 5.5 show two examples of *degenerate* projections onto an ellipse. Fig. 5.4 depicts an example where the optimal solution is given by the KKT point corresponding to the root of f. Fig. 5.5 depicts an example where the optimal solution is given by the KKT point corresponding to $\mu = -\frac{1}{\lambda_2}$. Notice that in these (degenerate) cases, one of the asymptotes of the image of $\boldsymbol{x}(\mu)$ disappears, and the image is hence along one of the axes. Moreover, one of the discontinuities of f, f' and $\|\boldsymbol{x}(\mu) - \boldsymbol{x}^0\|_2$ disappears as, *e.g.*,

$$\lim_{\mu\to -\frac{1}{\lambda_k}} f(\mu) = \sum_{i=1, i\neq k}^{n} \lambda_i \left(\frac{x_i^0}{1-\frac{\lambda_i}{\lambda_k}}\right)^2 - 1 \neq \infty.$$

Figs. 5.6 and 5.7 show two examples of *degenerate* projections onto a hyperbola. Fig. 5.6 depicts an example where f has no roots. This absence of roots is not an issue, because the optimal solution is given, in this case, as one of the $\boldsymbol{x}_k^{\mathrm{d}}$ (depicted in green) which are derived in (5.20). Fig. 5.7 shows an example where there is no intersection of the gray line and the quadric; therefore, there is no $\boldsymbol{x}_k^{\mathrm{d}}$ either. This is not an issue because then there must exist a root μ^* of f on $\mathcal{I}$, which is the optimal solution (purple triangle). Remark that, if $\Psi(\boldsymbol{x}^0) > 0$, then both $\boldsymbol{x}(\mu^*)$ and $\boldsymbol{x}_k^{\mathrm{d}}$ are KKT points, and one of them is the optimal solution.

5.5.2 Some eigenvalues are repeated

Before we get to the heart of the matter, let us introduce some definitions. Let $\overline{\boldsymbol{\lambda}}$ be the vector of the unique eigenvalues of $\boldsymbol{A}$, sorted in descending order, and let $k \in \{1, \ldots, |\overline{\boldsymbol{\lambda}}|\}$ be a given component of $\overline{\boldsymbol{\lambda}}$. Let L_k be a subset of I corresponding to the same eigenvalue,

$$L_k = \{l \in I \mid \lambda_l = \overline{\lambda}_k\},$$

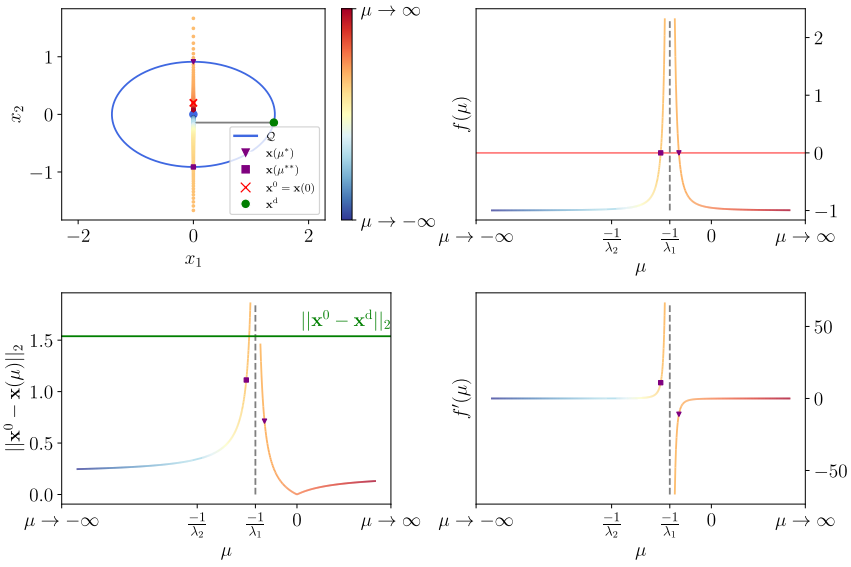


Fig. 5.4 Image of $\boldsymbol{x}(\mu)$ and graphs of $f(\mu)$, $f'(\mu)$, $\|\boldsymbol{x}(\mu) - \boldsymbol{x}^0\|_2$ for μ ranging on $]-\infty, +\infty[$ in the degenerate elliptic case. The optimal solution is the root of f and not $\boldsymbol{x}_2^{\mathrm{d}}$: the green line showing $\|\boldsymbol{x}^0 - \boldsymbol{x}_2^{\mathrm{d}}\|_2$ is above the purple triangle in the lower left figure.

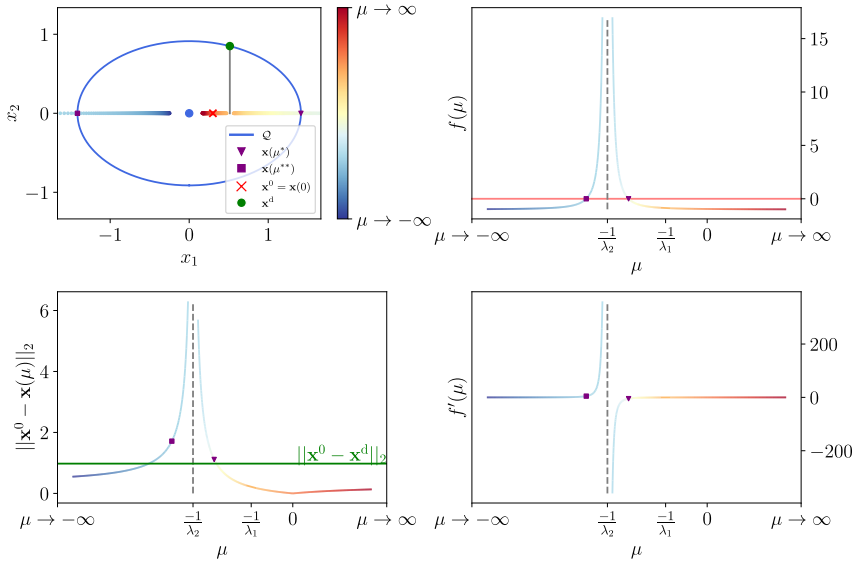


Fig. 5.5 Image of $\boldsymbol{x}(\mu)$ and graphs of $f(\mu)$, $f'(\mu)$, $\|\boldsymbol{x}(\mu) - \boldsymbol{x}^0\|_2$ for μ ranging on $]-\infty, +\infty[$ in the degenerate elliptic case. One of the optimal solutions is not the root of f but $\boldsymbol{x}_1^{\mathrm{d}}$: the green line showing $\|\boldsymbol{x}^0 - \boldsymbol{x}_1^{\mathrm{d}}\|_2$ is below the purple triangle. The reflection of the green point $\boldsymbol{x}_1^{\mathrm{d}}$ about the axis $x_2 = 0$ is also an optimal solution.

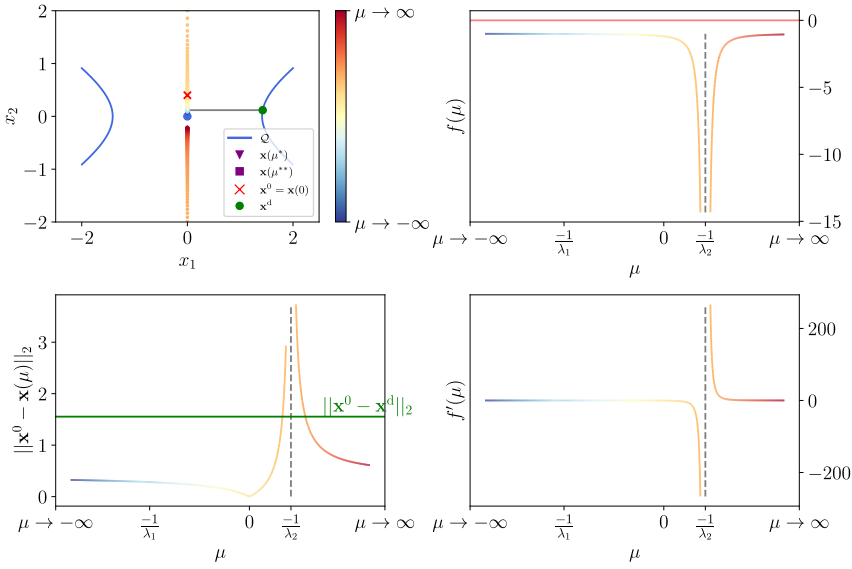


Fig. 5.6 Image of $\boldsymbol{x}(\mu)$ and graphs of $f(\mu)$, $f'(\mu)$, $\|\boldsymbol{x}(\mu) - \boldsymbol{x}^0\|_2$ for μ ranging on $]-\infty, +\infty[$ in the degenerate hyperbolic case. One of the optimal solution is $\boldsymbol{x}_1^{\mathrm{d}}$ as f has no root. The reflection of the green point $\boldsymbol{x}^{\mathrm{d}_1}$ about the axis $x_1 = 0$ is also an optimal solution.

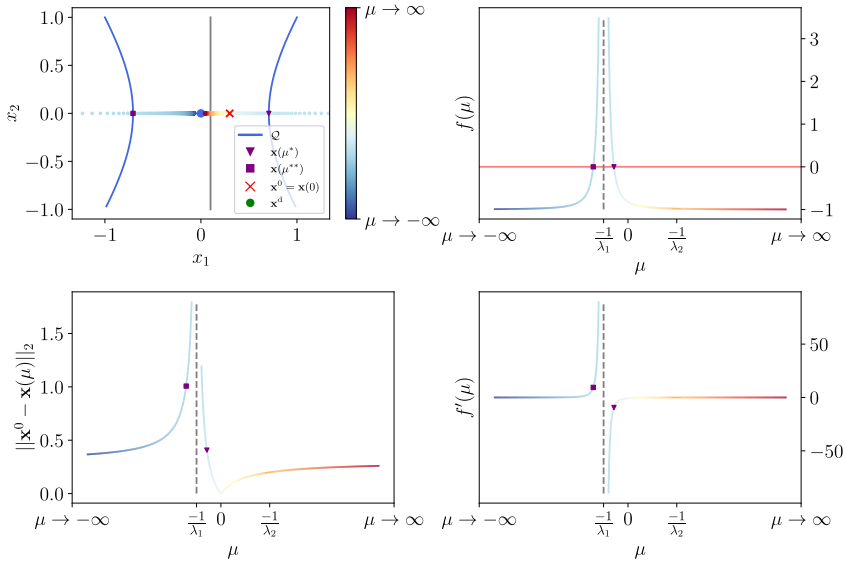


Fig. 5.7 Image of $\boldsymbol{x}(\mu)$ and graphs of $f(\mu)$, $f'(\mu)$, $\|\boldsymbol{x}(\mu) - \boldsymbol{x}^0\|_2$ for μ ranging on $]-\infty, +\infty[$ in the degenerate hyperbolic case. The optimal solution is $\boldsymbol{x}(\mu^*)$, the root of f. There are no additional KKT points $\boldsymbol{x}_k^{\mathrm{d}}$; the gray line in the upper left panel does not intersect with the quadric.

and let K_k be a subset of L_k where the associated component of $\boldsymbol{x}^0$ is equal to zero,

$$K_k := \left\{ i \in I \mid \lambda_i = \overline{\lambda}_k, x_i^0 = 0 \right\}.$$

We first show that $L_k = K_k$ if there is a solution with $\mu^* = -1/\overline{\lambda_k}$.

Proposition 5.20. *Let L_k and K_k be defined as above. There exists a solution to (5.6) with $\mu^* = -1/\overline{\lambda_k}$ only if $L_k = K_k$.*

Proof. Let $(\boldsymbol{x}^*, \mu^*)$ be a solution to (5.6) with $\mu^* = -1/\overline{\lambda_k}$. Let us assume, for the sake of contradiction, that $L_k \neq K_k$—or equivalently, that there is some $i \in I$ with $\lambda_i = \overline{\lambda}_k$ but $x_i^0 \neq 0$. The $i-$th component of (5.6) reads

$$2(x_i - x_i^0) - 2x_i = 0,$$

which does not hold. □

Remark that Proposition 5.20 is a left implication, and it is possible that no solutions to (5.6) exist with $\mu^* = -1/\overline{\lambda}_k$, an example of which is shown in Fig. 5.6.

If $|K_k| = 1$, the discussion is analogous to the previous paragraph: at most two KKT solutions are obtained as the intersection of a line and the quadric. On the other hand, if $|K_k| > 1$, we have to take the intersection of a plane π, defined as

$$\pi := \left\{ \boldsymbol{x} \in \mathbb{R}^n \mid x_i = \frac{x_i^0}{1 - \frac{\lambda_i}{\overline{\lambda}_k}} \text{ for all } i \notin K_k \right\},$$

and the quadric. Geometrically, the intersection—if nonempty, *i.e.*, if the argument of the square root below is positive—will be a $|K_k| - 1$ hypersphere in the corresponding subspace of $\mathbb{R}^n$:

$$\pi \bigcap \mathcal{Q} = \left\{ \boldsymbol{x} \in \mathbb{R}^n \text{ such that } x_i = \frac{x_i^0}{1 - \frac{\lambda_i}{\overline{\lambda}_k}} \text{if } i \notin K_k, \right.$$
$$\left. \sum_{i \in K_k} x_i^2 = \frac{1}{\overline{\lambda}_k} \left(1 - \sum_{j \in I \setminus K_k} \lambda_j \left(\frac{x_j^0}{1 - \frac{\lambda_j}{\overline{\lambda}_k}} \right)^2 \right) \right\}, \tag{5.21}$$

and every point belonging to this hypersphere is a KKT point. Moreover,

all the points in this hypersphere achieve the same value for the objective function of (5.3). Hence, for the purpose of finding one of the optimal solutions to (5.3), we can keep in our list of candidates just one element of (5.21). In particular, we can arbitrarily select *one* solution that lies in the first orthant by setting to zero all components of K_k except one (k'):

$$(\boldsymbol{x}_k^{\mathrm{d}})_i = \begin{cases} \dfrac{x_i^0}{1-\frac{\lambda_i}{\overline{\lambda}_k}} & \text{if } i \notin K_k, \\ \sqrt{\dfrac{1}{\overline{\lambda}_k}\left(1 - \sum\limits_{j \in I \setminus K_k} \lambda_j \left(\dfrac{x_j^0}{1-\frac{\lambda_j}{\overline{\lambda}_k}}\right)^2\right)} & \text{if } i = k', \\ 0 & \text{if } i \in K_k, i \neq k' \end{cases} \tag{5.22}$$

Each $k' \in K_k$ works, let us choose without loss of generality $k' := \min\limits_{i \in K_k} i$. As a matter of fact, this is equivalent to restricting the search to the subspace $\{\boldsymbol{x} \in \mathbb{R}^n \mid x_i = 0 \text{ for all } i \in K_k \setminus \{k'\}\}$ because all points in the hypersphere have the same objective. In this subspace, the problem is analogous to the case $|K_k| = 1$, *i.e.*, the intersection of a line and a quadric.

5.6 Bringing everything together

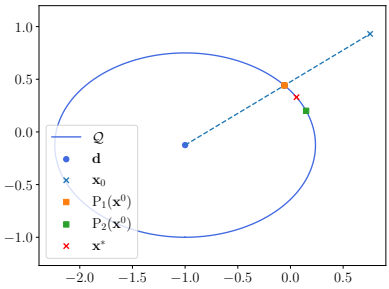
Let us give a full characterization of an optimal solution to (5.3).

Proposition 5.21. *There is an optimal solution to (5.3) in the set* $\{\boldsymbol{x}(\mu^*)\} \bigcup \boldsymbol{X}^d$ *where*

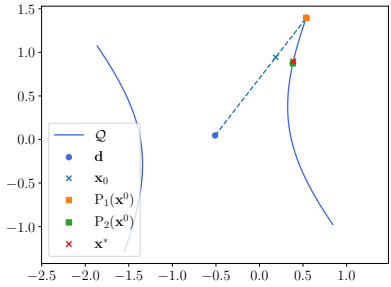
- $\boldsymbol{x}(\mu^*)$ *is defined by (5.8),* μ^* *is the unique root of* f *on* $\mathcal{I} =]e_1, e_2[$*, and* e_1*,* e_2 *are given by (5.17);*
- $\boldsymbol{X}^{\mathrm{d}} := \left\{\boldsymbol{x}_k^{\mathrm{d}} \mid k = 1, \ldots, |\overline{\boldsymbol{\lambda}}|, \text{ and } |K_k| > 0\right\}$ *as defined in (5.22).*

Proof. Since the quadric is nonsingular, no points fulfill the LICQ condition. Hence, the solutions to (5.3), which exist by Proposition 5.1, must be a KKT point. The KKT points are the solutions to (5.6); they satisfy

$$\begin{aligned} x_i(1 + \mu\lambda_i) &= x_i^0 \text{ for all } i \in I, \\ \sum_{i \in I} \lambda_i x_i^2 &= 1. \end{aligned}$$



(a) Illustration of the exact and quasi-projections for an ellipse.



(b) Illustration of the exact and quasi-projections for a hyperbola.

Fig. 5.8 Comparison between the exact and quasi-projections onto a 2D quadric.

Hence $(\boldsymbol{x}^*, \mu^*)$ is a solution to the KKT conditions (5.6) if and only if one of the following inequality holds:

(i) $\mu^* \neq -1/\lambda_i$ for all i, μ^* is a root of f defined in (5.10) and $\boldsymbol{x}^*$ satisfies (5.8).

(ii) $\mu^* = -1/\lambda_k$ for some k, $x_i^0 = 0$ for all i such that $\lambda_i = \lambda_k$. Letting K_k be the set of those i's, we have $\boldsymbol{x}^* \in \pi \bigcap \mathcal{Q}$ defined in (5.21).

In case (i), we have seen that the smallest objective function value of (5.3) is given by the— possibly nonexistent—unique root of f. In case (ii) and for a given index $k \in K$, all the points in (5.21)—which may be empty—achieve the same value for the objective of (5.3), and (5.22)—defined if and only if (5.21) is nonempty—is one of those points. Finally, recall that Proposition 5.1 proves the existence of one optimal solution to (5.3): either case (i) or case (ii) will provide a solution.

□

The full procedure to compute the projection of some point $\boldsymbol{x}^0$ onto a non-cylindrical central quadric is given in Algorithm 8.

Algorithm 8 Exact projection onto a type-II quadric in normal form

Require: $\boldsymbol{\lambda}$, the eigenvalues corresponding to (5.3) and $\boldsymbol{x}^0 \in \mathbb{R}^n_+$

$e_1 \leftarrow \max\left\{\max_{\{i\in I\,|\,\lambda_i>0, x_i^0\neq 0\}} -1/\lambda_i, -\infty\right\}$ ▷ If the inner max is empty, $e_1 = -\infty$

$e_2 \leftarrow \min\left\{\min_{\{i\in I\,|\,\lambda_i<0, x_i^0\neq 0\}} -1/\lambda_i, +\infty\right\}$ ▷ If the inner min is empty, $e_2 = +\infty$

$\boldsymbol{D} \leftarrow \operatorname{diag}(\boldsymbol{\lambda})$

$\boldsymbol{x}(\mu) \leftarrow (\boldsymbol{I} + \mu\boldsymbol{D})^{-1}\boldsymbol{x}^0$

$f(\mu) \leftarrow \sum_{i=1, x_i^0\neq 0}^{n} \lambda_i \left(x_i^0/(1+\mu\lambda_i)\right)^2 - 1$

if $e_1 \neq -\infty$ **then**

  if $e_2 = +\infty$ **then**

    $\mu_0 \leftarrow \operatorname{bisection}(f, e_1)$ ▷ Using bisection search (see [BF01, Chapter 2.1]) to find $\mu^0 > e_1$ with $f(\mu^0) > 0$

    $\mu^* \leftarrow \operatorname{root}(f, \mu_0)$ ▷ Using Newton with starting point μ^0

  else

    $\mu^* \leftarrow \operatorname{root}(f, e_1, e_2)$ ▷ Using Algorithm 7

  end if

end if ▷ See *only if* part of Proposition 5.18: $e_1 = -\infty \Rightarrow f$ has no roots on $\mathcal{I}$

$\overline{\boldsymbol{\lambda}} \leftarrow \operatorname{unique}(\boldsymbol{\lambda})$

$\boldsymbol{X}^{\mathrm{d}} \leftarrow [\,]$

for $k = 1, \ldots, \left|\overline{\boldsymbol{\lambda}}\right|$ **do**

  $K_k \leftarrow \left\{i \in I \,|\, \lambda_i = \overline{\lambda}_k, x_i^0 = 0\right\}$

  $L_k \leftarrow \left\{i \in I \,|\, \lambda_i = \overline{\lambda}_k\right\}$

  if $K_k = L_k$ **and** $\left(1 - \sum_{j\in I\setminus K_k} \lambda_j \left(x_j^0/(1-\lambda_j/\overline{\lambda}_k)\right)^2\right)/\overline{\lambda}_k > 0$ **then**

    $\boldsymbol{x}_k^{\mathrm{d}} \leftarrow$ (5.22)

    $\boldsymbol{X}^{\mathrm{d}}.\operatorname{append}(\boldsymbol{x}_k^{\mathrm{d}})$

  end if

end for

return $\underset{\boldsymbol{x}\in\{\boldsymbol{x}(\mu^*)\}\cup\boldsymbol{X}^{\mathrm{d}}}{\arg\min} \|\boldsymbol{x}^0 - \boldsymbol{x}\|_2$ ▷ The min is taken over at most $n+1$ values

5.7 Quasi-projection onto the quadric

The procedure detailed in Algorithm 8 is an exact projection, but it requires computing the full eigenvalue decomposition of $\boldsymbol{A}$, including the eigenvectors. Such a computation may be expensive for problems of large dimension. In this section, we detail a geometrical procedure that allows us under some conditions (see § 5.7.1) to map a given point to the feasible set $\mathcal{Q}$ of (5.1). We refer to this mapping as a *quasi-projection*.

Definition 5.22. Quasi-projection. Let $\boldsymbol{x} \in \mathbb{R}^n$, a *quasi-projection* on the quadric $\mathcal{Q}$ is a mapping

$$\mathrm{P}\colon \mathcal{D} \to \mathcal{Q}\colon \boldsymbol{x} \mapsto \mathrm{P}(\boldsymbol{x}),$$

where $\mathcal{D}$ is a nonempty subset of $\mathbb{R}^n$.

Note that this definition is broad and includes the projection operator. Ideally, $\mathcal{D}$ should be $\mathbb{R}^n$, but we allow the quasi-projection to fail to map some points. This quasi-projection is inspired from the retraction in [BSBA13, VAP22b] and the following observation: the projection of a given point $\boldsymbol{x}^0$ onto a sphere can be analytically computed by finding the intersection of the sphere and the half-line defined by the sphere center and $\boldsymbol{x}^0$. As the quadric $\mathcal{Q}$ is by assumption a *central quadric*, it is tempting to approximate the projection by the same mechanism described above for the sphere. This yields a first variant of the *quasi-projection*. The second variant is obtained by searching for the intersection of the quadric and the line passing through $\boldsymbol{x}^0$ along the direction $\nabla\Psi(\boldsymbol{x}^0)$.

No matter the variant, the quasi-projection compute the intersection of the quadric and the line starting from $\boldsymbol{x}^0$ along some direction $\boldsymbol{\xi}$. The points of intersection are parametrized as $\boldsymbol{x}^0 + \beta\boldsymbol{\xi}$, where β satisfies $\Psi(\boldsymbol{x}^0 + \beta\boldsymbol{\xi}) = 0$—or equivalently, $b_1\beta^2 + b_2\beta + b_3 = 0$ for appropriate b_i's. The b_i's are given in Algorithm 9, see [BSBA13, §3.2] for more details. To select the point that is the closest to $\boldsymbol{x}^0$ among both intersection points, the β which is the closest to zero is chosen.

We detail two variants of our quasi-projection:

- $\boldsymbol{\xi} = \boldsymbol{d} - \boldsymbol{x}^0$: this is analogous to the retraction used in [BSBA13], is referred to as *center-based* quasi-projection and is denoted by P_1;
- $\boldsymbol{\xi} = \nabla\Psi(\boldsymbol{x}^0) = 2\boldsymbol{A}\boldsymbol{x}^0 + \boldsymbol{b}$: the direction is given by the gradient of the

level curve of Ψ at $\boldsymbol{x}^0$. We refer to it as *gradient-based* quasi-projection and denote it as P_2.

The quasi-projection procedure is given in Algorithm 9 and depicted in Fig. 5.8 for both strategies.

In an effort to maximize knowledge sharing, we packaged our method to project onto quadratic surfaces in a Python package named as `quadproj`. The package is available in the Python Package Index (PyPi) [VHa] and on conda [VHb]. The source code is open-sourced on GitLab [VHc] and the documentation is available in [VHd].

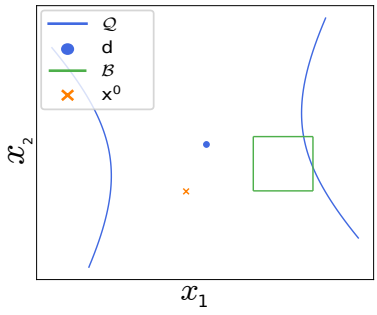
Algorithm 9 Quasi-projection onto the quadric

Require: $\boldsymbol{x}^0 \in \mathbb{R}^n$, a central quadric $\mathcal{Q}$ with center $\boldsymbol{d}$, a direction $\boldsymbol{\xi}$

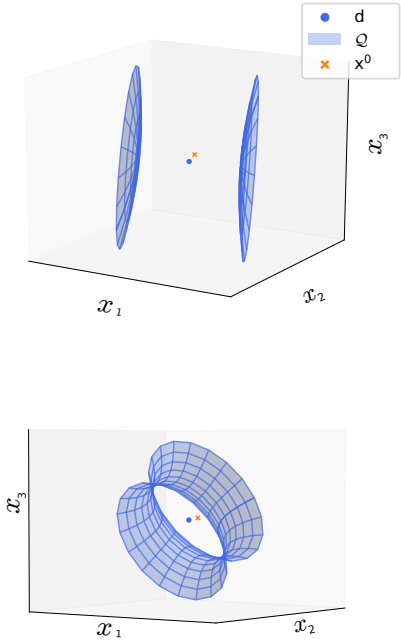
$b_1 \leftarrow \boldsymbol{\xi}^\intercal \boldsymbol{A} \boldsymbol{\xi}$
$b_2 \leftarrow 2\boldsymbol{x}^{0\intercal} \boldsymbol{A} \boldsymbol{\xi} + \boldsymbol{b}^\intercal \boldsymbol{\xi}$
$b_3 \leftarrow \boldsymbol{x}^{0\intercal} \boldsymbol{A} \boldsymbol{x}^0 + \boldsymbol{b}^\intercal \boldsymbol{x}^0 + c$
if $b_2^2 - 4b_1 b_3 < 0$ **then**
    return `None` ▷ It may happen that no intersection points are available, see Fig. 5.9
else
    $\Delta \leftarrow \sqrt{b_2^2 - 4b_1 b_3}$
    $\beta^+ \leftarrow (-b_2 + \Delta)/(2b_1)$
    $\beta^- \leftarrow (-b_2 - \Delta)/(2b_1)$
    if $b_2 > 0$ **then**
        $\beta \leftarrow \beta^+$
    else
        $\beta \leftarrow \beta^-$
    end if ▷ We select the closest to $\boldsymbol{x}^0$ of the two intersection points
    return $\boldsymbol{x}^0 + \beta \boldsymbol{\xi}$
end if

5.7.1 Failures of the quasi-projection

For $\boldsymbol{x}^0 = \boldsymbol{0}$, P_1 is not defined, and for the hyperboloid case, the set where P_1 is not defined ($\mathbb{R}^n \setminus \mathcal{D}$) includes $\boldsymbol{0}$. Examples of nontrivial points that cannot be mapped using P_1 are provided in Fig. 5.9. Indeed, in these cases, there is no intersection of the quadric and the line starting from $\boldsymbol{x}^0$. We tackle this issue by resorting to the exact projection from Algorithm 8 whenever this situation occurs.



(a) 2D (hyperbola) case.



(b) 3D (two and one-sheet hyperboloid) cases.

Fig. 5.9 Illustration of a failure of the center-based quasi-projection (P_1): the point $\boldsymbol{x}^0$ cannot be mapped to the quadric using Algorithm 9. Indeed, the line defined by $\boldsymbol{x}^0$ and $\boldsymbol{d}$ does not intersect with the quadric.

There are points $\boldsymbol{x}^0$ for which P_2 is not defined, but it returns a point when $\boldsymbol{x}^0$ is close enough to $\mathcal{Q}$.

5.7.2 Features of the quasi-projection

In general, the quasi-projection is not exact, in the sense that the resulting point is *not* the optimal solution to (5.1). However, we expect the quasi-projection to be close to optimality when the point is close enough to the quadric. Such behavior is observed in our simulations in § 6.4.2. Also, in the specific case when the quadric is a sphere, then both P_1 and P_2 solve (5.1).

6

Splitting methods for the projection onto the intersection of a box and a quadric

BUILDING on the projection from Chapter 5 and the recent advancements of splitting methods for nonconvex problems, we implement two of these methods for solving the projection onto a nonconvex set. This set is the intersection of a quadric and a polytope, similar to the feasible set of the problem (P) considered in Chapter 4. The two separate projections can be easily computed: the first one is even trivial if we consider a box, and the second one is efficiently obtained with our projection from Chapter 5 (Algorithm 8). We then leverage the rich literature on splitting methods for nonconvex programming. We consider in this work two methods: alternating projections and Douglas-Rachford splitting. References for these schemes can be found in [ABRS10, BCL02, DIL15, LM08, LLM09] and in [BCL02, LP16], respectively.

Depending on the splitting method considered and whether we use the exact projection onto the quadric or the heuristics from Section 5.7, we detail five different methods for computing the projection. We benchmark these

five methods against Ipopt in the ellipsoidal and hyperboloidal case. In these experiments, we find that one of the proposed methods, namely the alternating projections with exact projections, attains the best objectives. If the running time is to be accounted for, then one of the heuristics (the gradient-based one) produces competitive solutions in a reduced time. All the methods considered outperform Ipopt in terms of execution time, with a difference of several orders of magnitude.

Finally, we benchmark one of the proposed methods against Gurobi. We use Gurobi to find the optimal solution to a problem inspired by the power system literature. Since, in this context, the starting point is close to the feasible set, our proposed method outperforms Gurobi, both in terms of execution time and objective. Using the lower bound computed by Gurobi, we can also conclude that for this specific problem the proposed method finds the optimal solution, even if there is no guarantee for finding it in general.

The chapter organization is as follows. We formulate the projection problem in Section 6.1 and describe the different methods to tackle it in Section 6.2. The extension to the intersection of a polytope and a Cartesian product of quadrics is made in Section 6.3, and we test all methods in Section 6.4. Finally, conclusions are drawn in Section 6.5.

This chapter is based on the *submitted* paper [VAP22a].

6.1 Problem formulation

This section is devoted to the analysis of the projection of a given point $\boldsymbol{x}^0$ onto a feasible set Ω that is the intersection of a box and a non-cylindrical central (or type-II) quadric. Let $\mathcal{B}$ be a nonempty n-dimensional *box* (also called a orthotope or a hyperrectangle), aligned with the axes:

$$\mathcal{B} = \{\boldsymbol{x} \in \mathbb{R}^n \mid \underline{x}_i \leq x_i \leq \overline{x}_i \text{ for all } i \in [n]\}, \tag{6.1}$$

for given lower and upper bounds $\underline{\boldsymbol{x}}$ and $\overline{\boldsymbol{x}}$. Let$\mathcal{Q}$ be a type-II quadric defined with the same parameters as in Section 5.1:

$$\mathcal{Q} = \left\{\boldsymbol{x} \in \mathbb{R}^n \middle| \boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x} + c = 0\right\}.$$

The optimization problem at hand reads as follows.

Projection onto the intersection of a box and a quadric

$$\begin{aligned} &\min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{x}-\boldsymbol{x}^0\|_2^2 \\ &\text{subject to } \boldsymbol{x}\in\mathcal{B} \\ &\qquad\qquad \boldsymbol{x}\in\mathcal{Q} \end{aligned} \tag{6.2}$$

What is developed in this chapter can be easily extended to the intersection of a polytope and a Cartesian product of quadrics. This extension is discussed in Section 6.3.

To be specific, we study two splitting algorithms: the Douglas-Rachford (DR) scheme and the alternating projection method (AP). Along with the classical DR scheme, we study a modified version (DR-F) for the feasibility problem [LP16]. We consider three variants of the alternating projection method: one based on the exact projection and two based on the quasi-projection from Section 5.7 (which approximates the projection *via* a geometrical construction). Splitting algorithms exploit the separable structure of the problem, since the projection onto each of the sets that define the intersection, $\Omega := \mathcal{B} \bigcap \mathcal{Q}$, is easy to compute. They recently have been widely studied and perform particularly well on certain classes of nonconvex problems.

A first (local) convergence result for the solution returned by the alternating projections in the nonconvex setting is presented in [Dru13, Theorem 3.2.3] for sets that intersect transversally. This result is particularized to (nonempty and closed) semi-algebraic intersections in [DIL15, Theorem 7.3], which we use in this work. A second result that is used in this chapter is [LP16, Corollary 1], which is a convergence result for a (modified) Douglas-Rachford splitting. These two important propositions exploit the Kurdyka-Łojasiewicz properties of the indicator function of semi-algebraic sets and are unfortunately local results: the theorem from [DIL15] assumes that the starting point is *near* the intersection of the two considered sets, and the theorem from [LP16] proves the convergence to a *stationary point* of the problem of minimizing the distance to one of the sets, subject to being in the second set. We exhibit an example where both methods fail to converge to a feasible point (Figs. 6.1 and 6.4) and propose a restart heuristic in § 6.3.2.

6.2 Methods

In this section, we recall how to project onto a box in § 6.2.1. The alternating projection method and the Douglas-Rachford method are presented in § 6.2.2 and 6.2.3, respectively. Then, we compare both methods in § 6.2.4.

6.2.1 Projection onto a box

This projection is straightforward. Indeed, given a point $\boldsymbol{x}^0$, it is sufficient to check for each dimension i whether this point violates either the lower or the upper bound and to replace it accordingly. This gives Algorithm 10. For a more general polytope $\mathcal{P}$, the projection cannot be computed analytically. However, the projection can be efficiently computed by solving the convex QP optimization problem:

$$\min_{\boldsymbol{x}\in\mathcal{P}} \|\boldsymbol{x}^0 - \boldsymbol{x}\|_2^2. \tag{6.3}$$

Algorithm 10 Projection onto the box (6.1)

Require: $\boldsymbol{x}^0 \in \mathbb{R}^n$
1: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$
2: **for** $i \in [n]$ **do**
3: **if** $x_i^0 < \underline{x}_i$ **then**
4: $x_i \leftarrow \underline{x}_i$
5: **else if** $x_i^0 > \overline{x}_i$ **then**
6: $x_i \leftarrow \overline{x}_i$
7: **end if**
8: **end for**
9: **return** $\boldsymbol{x}$

6.2.2 Alternating projection method

The alternating projections is a method to find a point in the intersection of two (or more) sets by sequentially projecting onto each of the sets. This method is first introduced by Von Neumann, who proved the convergence of the method for two affine subspaces [Neu51, Theorem 13.7]. The generalization to N subspaces is made in [Hal62] and to N closed convex sets

in [Bre65]. A more in-depth history of the alternating projection method is presented in [BB96]. The case of two smooth manifolds that intersect transversally is studied in [LM08] and a more general nonconvex case in [DIL15].

We state the algorithm in Algorithm 11. Depending on whether we use the exact projection or one of the two quasi-projections detailed in § 5.7, we refer to the method as follows: alternating projections with exact projection onto the quadric (APE), alternating projections with the center-based quasi-projection (APC) or with the gradient-based quasi-projection (APG). $\mathrm{Pr}_{\mathcal{X}}$ stands for (one solution of) the projection onto a (non)convex set $\mathcal{X}$ and $\mathrm{P}_{\mathcal{Y}}$ for a quasi-projection onto a set $\mathcal{Y}$.

Algorithm 11 Alternating projections

Require: $\boldsymbol{x}^0 \in \mathbb{R}^n, \Omega = \boldsymbol{B} \cap \mathcal{Q}$
1: set maximum number of iterations n_{iter}
2: $k \leftarrow 0$
3: **while** $k < n_{\text{iter}}$ **and not** $\boldsymbol{x}^k \in \Omega$ **do**
4: $\quad \boldsymbol{y}^{k+1} \leftarrow \mathrm{Pr}_{\mathcal{B}}(\boldsymbol{x}^k)$ ▷ Using Algorithm 10
5: $\quad \boldsymbol{x}^{k+1} \leftarrow \mathrm{P}_{\mathcal{Q}}(\boldsymbol{y}^{k+1})$ ▷ Using Algorithm 8 or Algorithm 9
6: $\quad k \leftarrow k + 1$
7: **end while**
8: **return** $\boldsymbol{x}^k$

Assuming that the initial iterate is close enough to the intersection, [DIL15] provide a convergence result for APE. This result is particularized to our case in Proposition 6.1. While this proposition guarantees the convergence to a point $\boldsymbol{x}^*$ in Ω, it provides no guarantee about the optimality of this point, *i.e.*, it is not true in general that $\boldsymbol{x}^* \in \arg\min_{\boldsymbol{x} \in \Omega} \|\boldsymbol{x} - \boldsymbol{x}^0\|_2^2$.

Proposition 6.1. *If Algorithm 11 with the exact projection (APE) is initialized from $\boldsymbol{x}^0 \in \mathcal{Q}$ and near $\mathcal{B}$, then the distance of the iterates to the intersection $\mathcal{Q} \bigcap \mathcal{B}$ converges to zero, and every limit point $\boldsymbol{x}^*$ lies in $\mathcal{Q} \bigcap \mathcal{B}$.*

Proof. This follows from [DIL15, Theorem 7.3], since $\mathcal{Q}$ and $\mathcal{B}$ are semi-algebraic and $\mathcal{B}$ is bounded. □

REMARK 6.1 (Another starting point for Proposition 6.1). If $\boldsymbol{A} \succ \boldsymbol{0}$, then $\mathcal{Q}$ is also bounded, and we can as well choose $\boldsymbol{x}^0$ in $\mathcal{B}$ and near $\mathcal{Q}$. ■

Figures 6.2 and 6.3 show examples where the alternating methods converge in a single iteration or multiple iterations. Only APC is depicted.

Notice that if APE converges in a single iteration, then the obtained solution $\boldsymbol{x}^*$ is an optimal solution to (6.2), that is, $\boldsymbol{x}^* \in \arg\min_{\boldsymbol{x}\in\Omega} \|\boldsymbol{x} - \boldsymbol{x}^0\|_2^2$. Figure 6.1 shows a pathological example where none of the alternating projection methods converge to a feasible point of (5.3). We propose in § 6.3.2 certain heuristics to overcome such pathological cases.

6.2.3 Douglas-Rachford method

The Douglas-Rachford method is another popular splitting method, see [LS21] for a thorough survey of the method and its link to alternating projections and to ADMM.

Following [LP16], the Douglas-Rachford splitting algorithm aims at solving

$$\min\ f(\boldsymbol{x}) + g(\boldsymbol{x}),$$

where f has a Lipschitz continuous gradient and g is a proper closed function. The DR iteration starts at some $\boldsymbol{y}^0$ and repeats for $k = 0, 1, \ldots$

$$\begin{aligned}\boldsymbol{x}^{k+1} &= \operatorname{prox}_f(\boldsymbol{y}^k),\\ \boldsymbol{y}^{k+1} &= \boldsymbol{y}^k + \operatorname{prox}_g(2\boldsymbol{x}^{k+1} - \boldsymbol{y}^k) - \boldsymbol{x}^{k+1},\end{aligned}$$

where the prox operator (with step size 1) is defined as

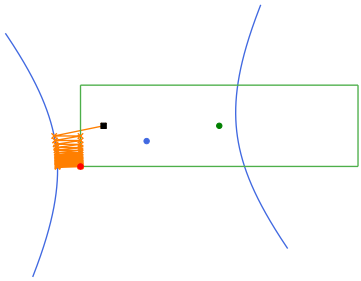
$$\operatorname{prox}_f(v) = \arg\min_{x\in\mathbb{R}^n} \left(f(x) + \frac{1}{2}\|x - v\|_2^2 \right). \tag{6.4}$$

Let $\mathbb{1}_{\mathcal{X}} : \mathcal{X} \mapsto \mathbb{B}$ be the indicator function of a set $\mathcal{X}$ defined as

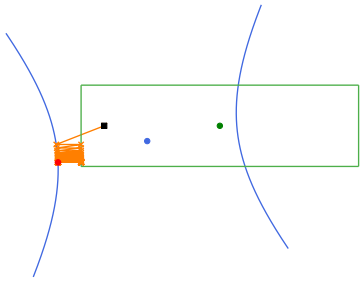
$$\mathbb{1}_{\mathcal{X}}(\boldsymbol{x}) = \begin{cases} 0 \text{ if } x \in \mathcal{X},\\ +\infty \text{ else.}\end{cases}$$

If we identify $f := \mathbb{1}_{\mathcal{B}}$ and $g := \mathbb{1}_{\mathcal{Q}}$, *i.e.*, the indicator functions of the sets that define Ω, then the DR algorithm reads

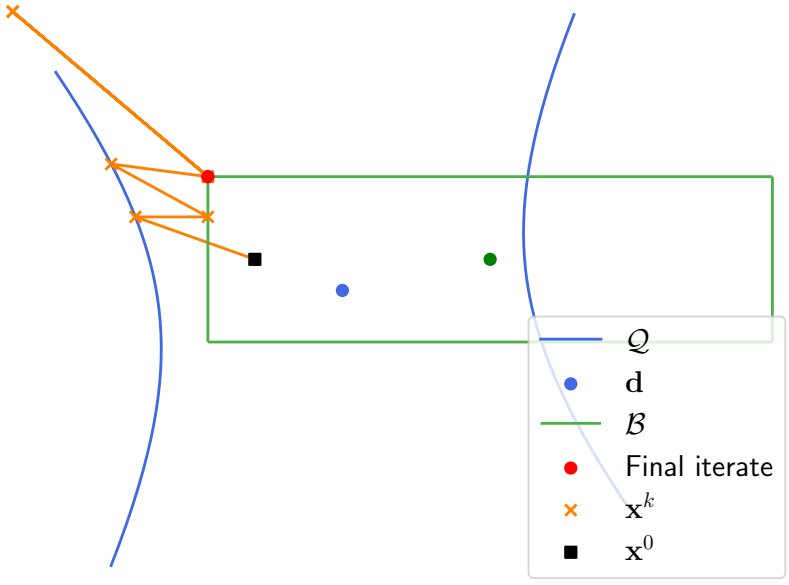
$$\begin{aligned}\boldsymbol{x}^{k+1} &= \operatorname{Pr}_{\mathcal{B}}(\boldsymbol{y}^k),\\ \boldsymbol{y}^{k+1} &= \boldsymbol{y}^k + \operatorname{Pr}_{\mathcal{Q}}(2\boldsymbol{x}^{k+1} - \boldsymbol{y}^k) - \boldsymbol{x}^{k+1}.\end{aligned}$$



(a) Alternating projections with exact projection (APE).

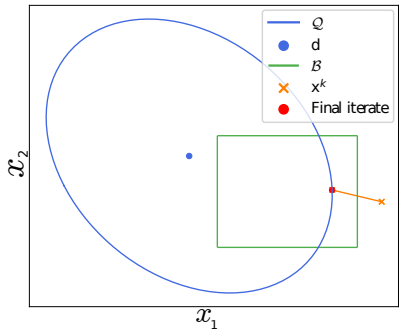


(b) Gradient-based alternating projections (APG).

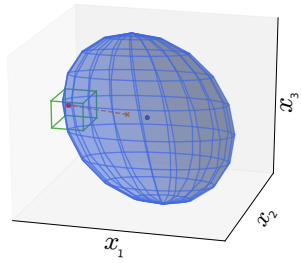


(c) Center-based alternating projections (APC).

Fig. 6.1 Illustration of a (2D) pathological case where none of the proposed alternating methods converge to a feasible point of (6.2). We represent both $\boldsymbol{x}^k$ and $\boldsymbol{y}^k$ as orange crosses. The green dot is the box center.

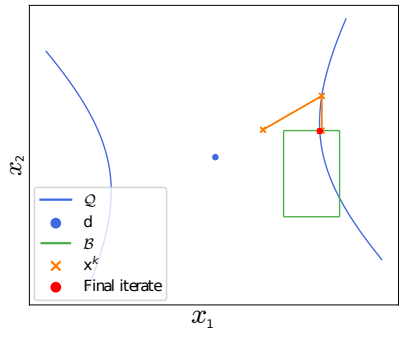


(a) 2D (ellipse) case.

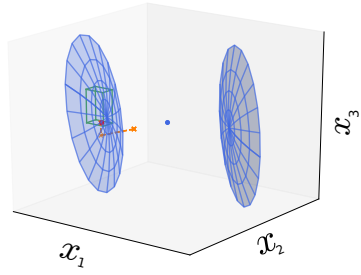


(b) 3D (ellipsoid) case.

Fig. 6.2 Illustration of the center-based alternating projection method (APC). In these examples, the method converges in a single iteration as the quasi-projection from Algorithm 9 yields a feasible point, *i.e.*, a point that is also inside the box. We represent both $\boldsymbol{x}^k$ and $\boldsymbol{y}^k$ as orange crosses.



(a) 2D (hyperbola) case: the algorithm converges in three iterations.



(b) 3D (two-sheet hyperboloid) case, the algorithm converges in three iterations.

Fig. 6.3 Illustration of the center-based alternating projection method (APC) on 2D and 3D hyperbolic cases. We represent both $\boldsymbol{x}^k$ and $\boldsymbol{y}^k$ as orange crosses.

We rewrite it in a compact way with $\boldsymbol{I}$ the identity operator [BCL02],

$$\boldsymbol{x}^{k+1} = \left(\mathrm{Pr}_{\mathcal{Q}}(2\mathrm{Pr}_{\mathcal{B}} - \boldsymbol{I}) + (\boldsymbol{I} - \mathrm{Pr}_{\mathcal{B}})\right)(\boldsymbol{x}^k). \tag{6.5}$$

Indeed, the proximal operator of an indicator function of a given set X is the projection onto this set Pr_X. We denote this method as DR and explicitly state it in Algorithm 12.

Algorithm 12 Douglas-Rachford splitting method (DR)

Require: An initial point $\boldsymbol{x}^0$
1: **while** a termination criterion is not met **do**
2: $\quad \boldsymbol{y}^{t+1} \leftarrow \mathrm{Pr}_{\mathcal{B}}(\boldsymbol{x}^t)$
3: $\quad \boldsymbol{z}^{t+1} \leftarrow \arg\min_{\boldsymbol{z}\in\mathcal{Q}} \|2\boldsymbol{y}^{t+1} - \boldsymbol{x}^t - \boldsymbol{z}\|^2$
4: $\quad \boldsymbol{x}^{t+1} \leftarrow \boldsymbol{x}^t + (\boldsymbol{z}^{t+1} - \boldsymbol{y}^{t+1})$
5: **end while**
6: **return** $\boldsymbol{z}^{t+1}$

Modified Douglas-Rachford We now present the modification of DR splitting for the feasibility problem of [LP16]. The interest of this modified version comes from its guarantee of convergence, as detailed in Proposition 6.2. Instead of using the indicator function for the convex set $\mathcal{B}$, the splitting is performed with the squared distance function $d^2_{\mathcal{B}}(\boldsymbol{x}) = \arg\min_{\boldsymbol{y}\in\mathcal{B}} \|\boldsymbol{x} - \boldsymbol{y}\|_2^2$, *i.e.*,

$$\min_{\boldsymbol{x}\in\mathcal{Q}} d^2_{\mathcal{B}}(\boldsymbol{x}), \tag{6.6}$$

which can be equivalently seen as

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} d^2_{\mathcal{B}}(\boldsymbol{x}) + \mathbb{1}_{\mathcal{Q}}(\boldsymbol{x}). \tag{6.7}$$

DR applied to (6.7) gives Algorithm 13, denoted as DR-F. We can use [LP16, Corollary 1] to obtain a convergence result for the DR-F method.

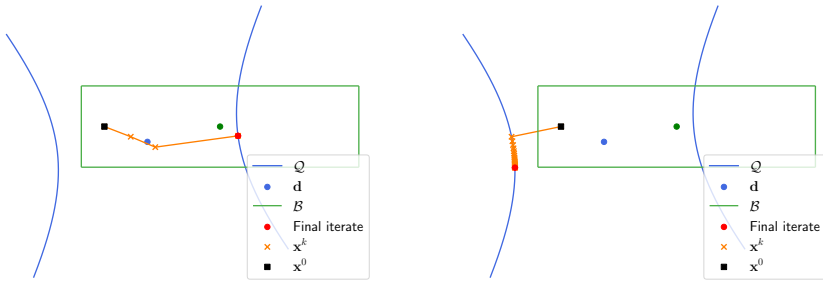
Proposition 6.2. *If $0 < \gamma < \sqrt{\frac{3}{2}} - 1$, then the sequence $\{(\boldsymbol{y}^t, \boldsymbol{z}^t, \boldsymbol{x}^t)\}$ provided by Algorithm 13 converges to a point $(\boldsymbol{y}^*, \boldsymbol{z}^*, \boldsymbol{x}^*)$ that satisfies $\boldsymbol{z}^* = \boldsymbol{y}^*$ and $\boldsymbol{z}^*$ is a stationary point of (6.6).*

Proof. Since $\mathcal{Q}$ and $\mathcal{B}$ are nonempty closed semi-algebraic sets, with $\mathcal{B}$ being convex and compact, we satisfy the hypothesis of [LP16, Corollary 1] for $0 < \gamma < \sqrt{\frac{3}{2}} - 1$. □

Algorithm 13 Douglas-Rachford splitting method for feasibility problems (DR-F)

Require: An initial point $\boldsymbol{x}^0$ and a step size parameter $\gamma > 0$

while a termination criterion is not met **do**
  $\boldsymbol{y}^{t+1} \leftarrow 1/(\gamma+1)(\boldsymbol{x}^t + \gamma P_{\mathcal{B}}(\boldsymbol{x}^t))$
  $\boldsymbol{z}^{t+1} \leftarrow \arg\min_{\boldsymbol{z}\in\mathcal{Q}} \|2\boldsymbol{y}^{t+1} - \boldsymbol{x}^t - \boldsymbol{z}\|^2$
  $\boldsymbol{x}^{t+1} \leftarrow \boldsymbol{x}^t + (\boldsymbol{z}^{t+1} - \boldsymbol{y}^{t+1})$
end while
return $\boldsymbol{z}^{t+1}$



(a) Douglas-Rachford (DR).

(b) Modified Douglas-Rachford for the feasibility problem (DR-F).

Fig. 6.4 Illustration of the behavior of the DR splitting algorithms on the same (pathological) case of Fig. 6.1. On this problem, DR does converge to a feasible point, whereas DR-F does not. The method DR-F converges to a stationary point (a local minimum) of (6.6).

6.2.4 Comparison

Table 6.1 compares the different complexities and convergence results of all the methods. The methods using the exact projection onto the quadric require the diagonalization of $\boldsymbol{A}$ as a precomputation step, which typically costs $\mathcal{O}(n^3)$ flops.

Table 6.1 Comparison of the splitting methods.

	APE	APC	APG	DR	DR-F
Complexity	$\mathcal{O}(n^3 + kn^2)$	$\mathcal{O}(kn^2)$	$\mathcal{O}(kn^2)$	$\mathcal{O}(n^3 + kn^2)$	$\mathcal{O}(n^3 + kn^2)$
Convergence Guarantees	Locally to a feasible point of (6.2): Proposition 6.1.	None	None	None[1]	Globally to a stationary point of (6.6): Proposition 6.2.

6.3 Extensions and implementation details

Let us now examine two important details: i) the extension to the intersection of a polytope and the Cartesian product of quadrics, which is exactly the feasible set of (P) from Chapter 4 and ii) a restart heuristic for the (rare) case where the method is trapped.[2]

6.3.1 Extension to the intersection of a polytope and the Cartesian product of quadrics

We can extend the splitting methods to perform the projection onto the intersection of a polytope $\mathcal{P}$ and a Cartesian product of m quadrics $\mathcal{Q}^{\text{tot}} := \bigtimes_{i=1}^{m} \mathcal{Q}_i$. We then solve

$$\begin{aligned} \min_{\boldsymbol{x} \in \mathbb{R}^n} \|\boldsymbol{x} - \boldsymbol{x}^0\|_2^2 \\ \text{subject to } \boldsymbol{x} \in \mathcal{P} \\ \boldsymbol{x} \in \mathcal{Q}^{\text{tot}}. \end{aligned} \tag{6.8}$$

The extension of all methods described in Table 6.1 is direct: instead of computing the projection on $\mathcal{B}$—now $\mathcal{P}$—analytically, we have to resort to a QP solver. And, as for the retraction from § 4.2.2, the (quasi-)projection is obtained by working individually on each quadric $\mathcal{Q}_i$:

$$\boldsymbol{x}^* \in \operatorname*{arg\,min}_{\boldsymbol{x} \in \mathcal{Q}^{\text{tot}}} \|\boldsymbol{x} - \boldsymbol{x}^0\|_2 \Leftrightarrow \boldsymbol{x}_i^* \in \operatorname*{arg\,min}_{\boldsymbol{x}_i \in \mathcal{Q}_i} \|\boldsymbol{x}_i - \boldsymbol{x}_i^0\|_2 \text{ for all } i \in [m]. \tag{6.9}$$

[1] There are, however, proofs of the convergence of DR in some nonconvex applications, see [AACT20, Section 4].

[2] It occurred about 0.01% of the time in our numerical experiments.

This problem has the same form as the projection step from Fig. 4.1 onto the feasible set of (P). This feasible set is the intersection of the Cartesian product of $|T|$ quadrics, (4.5), and the polytope that is defined by the remaining constraints, Eqs. (4.3), (4.4) and (4.6) to (4.8). The projection of a point onto the feasible set of (P) can then be obtained using the methods described in the present chapter.

Moreover, the point that has to be projected in Fig. 4.1 is obtained as the solution to a surrogate problem defined on a relaxed set, see § 4.1.4 for more details. Because this relaxation is close to the feasible set, the point that has to be projected is inside the box and near the quadric. This is the reason for the favorable performance of APC, as reported in § 6.4.2.

6.3.2 Implementation details

To address the convergence issues identified in Figs. 6.1 and 6.4, we add a restart mechanism whenever this situation arises. Such situations are easily detected: the alternating method will loop between two points, and the DR or DR-F will simply converge to an infeasible point. These problems mostly appear in the hyperboloid case and typically occur when the method is trapped on the wrong sheet of the hyperboloid. To mitigate this issue, we use the geometrical construction from the center-based quasi-projection (Algorithm 9 with $\boldsymbol{\xi} = \boldsymbol{x}^0 - \boldsymbol{d}$) and select the *largest* β. This is equivalent to transforming $\boldsymbol{x}^k \in \mathcal{Q}, \boldsymbol{x}^k \notin \mathcal{B}$ into $-\boldsymbol{x}^k =: \boldsymbol{x}^{k+1} \in \mathcal{Q}$ and continuing the method from $\boldsymbol{x}^{k+1}$. Alternatively, it is also possible to consider $\boldsymbol{x}^{k+1}$ such that at least one—instead of all—of its components is the opposite of $\boldsymbol{x}^k$. If, on the other hand, $\boldsymbol{x}^k \in \mathcal{B}$, then we work analogously with respect to the center of the box.

Using this restart mechanism is not a guarantee of convergence: the method can then be trapped into another region, or even come back to the same region. But in the few instances ($\approx$ once every 10000 trials) where the presented algorithms experienced convergence issues, the restart resulted in successful convergence.

6.4 Numerical experiments

This section is devoted to the benchmarking of the methods developed in this chapter. In § 6.4.1, we test the five methods (APE, APC, APG, DR, DR-F)

as well as Ipopt. Ipopt is an interesting method to benchmark against, as it is a natural candidate for solving (6.2). It is an open-source (OS) solver that uses by default an embedded OS linear solver. Hence, the performance of Ipopt can be enhanced through the use of a dedicated commercial linear solver. In this work, we use Pardiso [ABB+20].

We solve for small scale (Figs. 6.5 and 6.7) and larger scale (Figs. 6.6 and 6.8) instances of (6.2). For each considered dimension n, we run 100 randomized independent trials to smooth the effect of the random selection of the problem parameters. In particular, each independent trial consists of a unique (randomly generated) set of parameters $\boldsymbol{A}, \boldsymbol{b}, c$ and $\boldsymbol{x}^0 \in \mathbb{R}^n$. The problems are generated as follows: we sample (from the normal distribution) a quadric with $\boldsymbol{A} \sim \mathcal{N}(\boldsymbol{\mu} = \mathbb{1}, \Sigma = I)$, $\boldsymbol{A} = (\boldsymbol{A} + \boldsymbol{A}^\intercal)/2$, $\boldsymbol{b}$ defined with $b_i \sim \mathcal{N}(\mu = 0, \sigma = 1)$, and $c \sim \mathcal{N}(\mu = -1, \sigma = 1)$. For the ellipsoidal case, we shift $\boldsymbol{A}$ to ensure that $\boldsymbol{A} \succ \boldsymbol{0}$, and then we find one feasible point and construct the box $\mathcal{B}$ around it; this allows us to ensure that the intersection of $\mathcal{B}$ and $\mathcal{Q}$ is nonempty. Note that this feasible point is not necessarily the center of the box.

Next, in § 6.4.2, we perform two experiments to compare APC and Gurobi for a specific problem structure. This problem stems from the second step of the procedure given in Fig. 4.1 and is the initial goal of the present research. The same remarks concerning the 100 randomly generated data also apply here. For this second experiment, we use Gurobi as a benchmark because i) it may also be a natural method for solving (6.2)[3] and ii) it provides lower bounds that allow us to assess how close the returned solutions are to the global optimum. We benchmark it against APC because, even if it is the worst-performing method among the five that are presented in § 6.4.1, it still outperforms Gurobi. This behavior is explained by the relative position of the starting point with respect to the feasible set from the problem of § 6.4.2: this point is inside the box and close to the quadric.

In all experiments, whenever an algorithm terminates with a timeout and returns an infeasible point, the associated objective is meaningless. In order to avoid distorting our reported results, we omit these instances in the recorded objectives, but we count the number of timeouts and record the deviation. The deviation is computed as

$$\text{deviation} = \left|\boldsymbol{x}^\intercal \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^\intercal \boldsymbol{x} + c\right|, \tag{6.10}$$

[3]Since version 9.0, Gurobi supports nonconvex QCQP optimization, and Gurobi is employed widely in the power system optimization community.

and is an intuitive measure of how far an infeasible point is from the feasible set. The prescribed tolerance for the deviation is 10^{-6}. This deviation does not account for the box. This is not an issue here because none of the tests considered in the numerical experiments terminates outside the box.

6.4.1 Douglas-Rachford, alternating projections, and Ipopt

Two different settings are considered here. In the first setting, the matrix $\boldsymbol{A}$ is chosen such that $\boldsymbol{A} \succ \mathbf{0}$, *i.e.*, the quadric is an *ellipsoid*. This means that the quasi-projection with $\boldsymbol{\xi} = \boldsymbol{x}^0 - \boldsymbol{d}$ is well-defined: situations depicted in Fig. 5.9 cannot occur. In the second experiment, we consider the case of *hyperboloids*, *i.e.*, $\boldsymbol{A}$ is nonsingular and indefinite.

From these two settings, it appears that both DR-F and APE are the methods that find the best solution in terms of objective. However, if the execution time is taken into account, APG reaches an objective close to the one of DR-F and APE in a significantly lower run time. APG should therefore be considered, *e.g.*, if the eigendecomposition is too expensive to compute. APC works quite well in the ellipsoidal case but performs worse in the hyperboloidal case. Ipopt is the slowest method. It achieves good solution objectives in the ellipsoidal case but gives poorer results in the hyperboloidal case.

Ellipsoid experiments

In these two experiments, we run small and large-scale ellipsoidal problems. The box $\mathcal{B}$ is small in comparison with the quadric and each starting point $\boldsymbol{x}^0$ is uniformly distributed inside the box.

For small-scale ellipsoidal problems ($n \leq 100$, Fig. 6.5), we observe that all methods except DR reach comparable objectives: APE, DR-F and Ipopt obtain the same objective, APG is within 1% and DR within several percent. We also observe that none of the methods exceeds the prescribed deviation accuracy of 10^{-6} and that Ipopt provides the most feasible points (*i.e.*, with the smallest deviation).

The number of iterations required for each method remains more or less constant when the dimension increases. Considering the running time, APC is the fastest and ten times faster than APG, which is two times faster than DR. DR-F and APE require approximately the same amount of time, which is two times slower than DR. Finally, Ipopt requires much more time than all the other methods.

For large-scale ellipsoidal problems ($n \geq 100$, Fig. 6.6), the behavior

of the methods remains the same as in the small-scale case. The distance increase with n is simply due to the increase of $\|\boldsymbol{x}^* - \boldsymbol{x}^0\|_2$ with n.

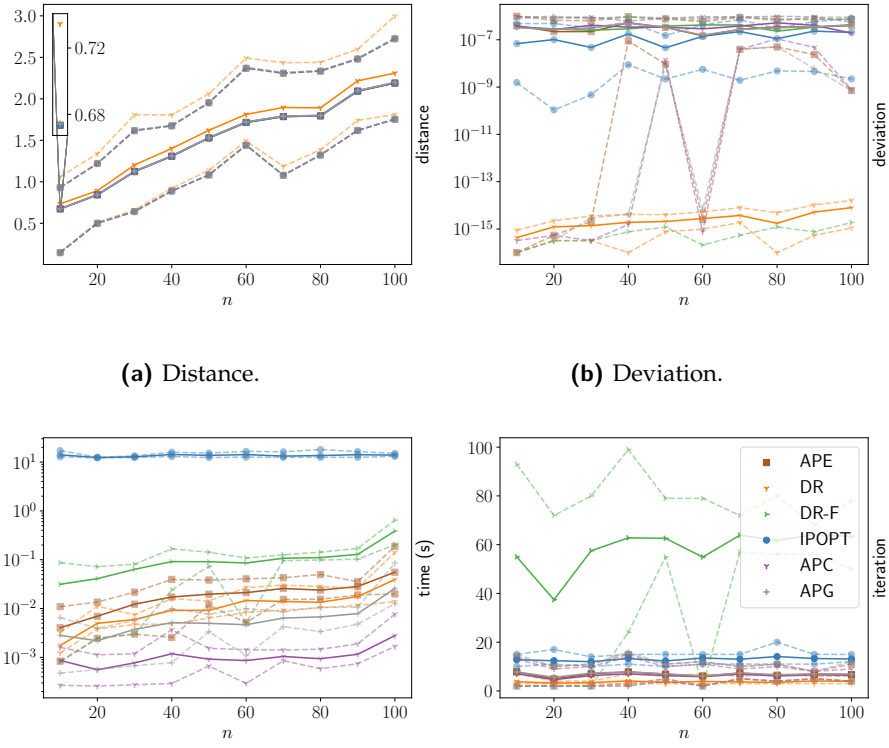
The execution time of Ipopt is remarkably stable, this is because creating the model already requires approximately 10 seconds, and this creation time does not increase much when the dimension increases. However, we note that i) for much larger dimension $n >> 1000$ the solving time of Ipopt increases significantly and ii) for such large dimension, it becomes crucial to use advanced linear algebra tools for, *e.g.*, the eigenvalue decomposition and matrix products used in the methods developed here. Hence, the comparison against Ipopt when the latter relies on dedicated linear algebra software (Pardiso) becomes less meaningful for too large n.

Hyperboloid experiments

In these two experiments, we run small and large-scale hyperboloidal problems. The box $\mathcal{B}$ is large in comparison with the quadric and the starting point $\boldsymbol{x}^0$ is uniformly distributed inside $\mathcal{B}$.

For small-scale hyperboloidal problems ($n \leq 100$, Fig. 6.7), we observe that the best objectives are obtained by APE. Then, within several percent, by DR-F, APG, DR, and Ipopt. These four methods reach most of the time the same solution as APE. However, they sometimes reach solutions that are far away from the best methods, see, *e.g.*, the maximum curve (top dashed-lines in Fig. 6.7a) that is significantly above the maximum curves of APE. Finally, APC performs poorly in terms of objective values. It is now APG which is the fastest method, despite its requirement of more iterations: the reason stems from the need of APC to resort to an exact projection whenever the situation depicted in Fig. 5.9 appears.

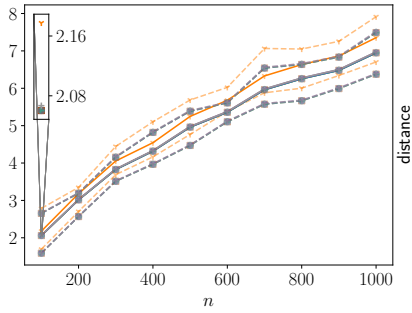
For the large-scale hyperboloidal problems ($n \geq 100$, Fig. 6.8), the best objectives are attained by APE and DR-F. The APG algorithm comes within one percent of their performance. The unmodified Douglas-Rachford finds objectives within several percent and Ipopt within 10 percent, *e.g.*, the mean objective for $n = 1000$ (solid lines in Fig. 6.8a) is around 0.51 for APE, DR-F, and APG. It is around 0.53 for DR and around 0.63 for Ipopt. We note that the number of iterations increases with n and that APG is also the fastest method. We also observe a significant increase in the execution time of Ipopt. Such an increase implies that the solving time is greater than the 10 seconds needed to create the problem. Finally, we observe that both Ipopt and APC sometimes finish with a timeout and return points above the prescribed deviation of 10^{-6}.



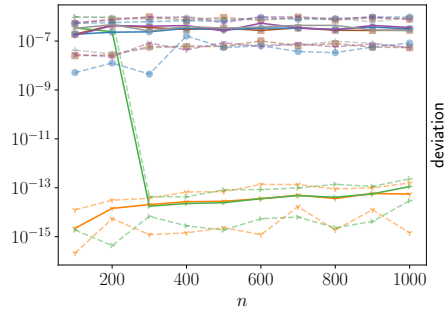
(a) Distance. **(b)** Deviation.

(c) Execution time in seconds. **(d)** Number of iterations.

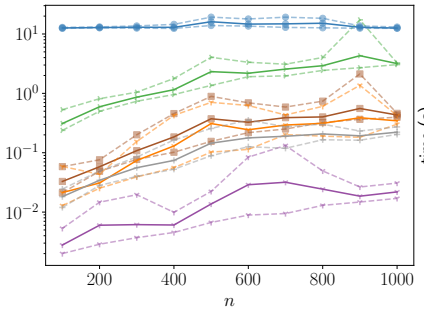
Fig. 6.5 Comparison of the different methods developed in Chapter 6: the Douglas-Rachford splitting (DR) and its modified counterpart (DR-F), the alternating projections using the exact projection (APE), and the alternating projections using the quasi-projections (either the center-based APC or the gradient-based APG). Ipopt is used as a benchmark with standard settings and with the underlying linear solver Pardiso. Ten dimensions n are considered and, for each n, 100 independent trials with $\boldsymbol{A} \succ \mathbf{0}$ are run. The top (bottom) dashed lines represent the max (min) value of the 100 trials, and the continuous line is the sample mean. The frame in the upper left of the upper left panel is a magnification around $n = 10$.



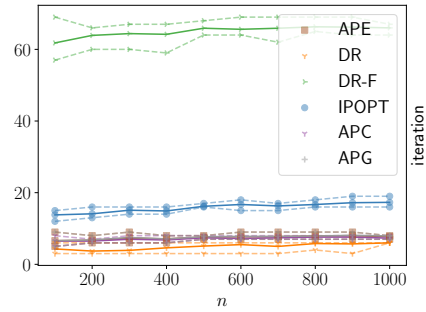
(a) Distance.



(b) Deviation.

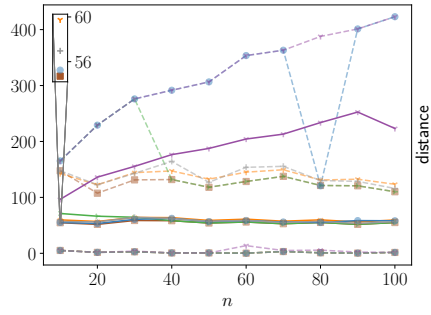


(c) Execution time in seconds.

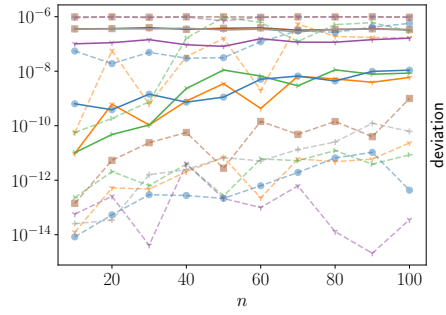


(d) Number of iterations.

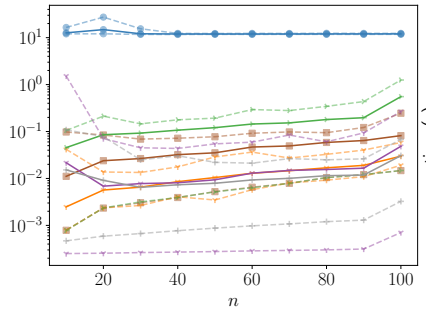
Fig. 6.6 Same as Fig. 6.5 for larger dimensions.



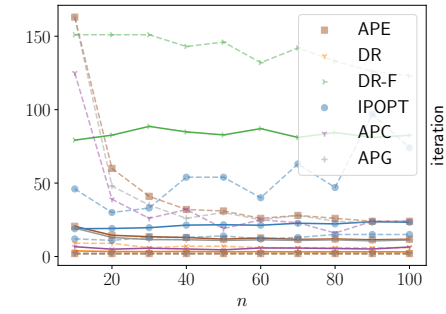
(a) Distance.



(b) Deviation.

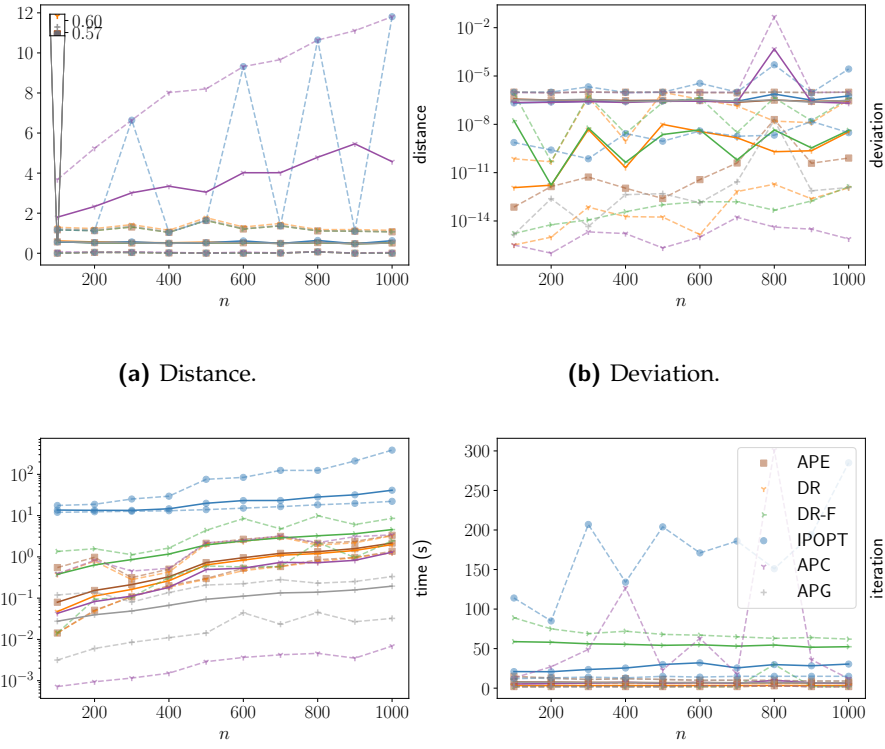


(c) Execution time in seconds.



(d) Number of iterations.

Fig. 6.7 Same as Fig. 6.5 with $\boldsymbol{A} \not\succ \mathbf{0}$.



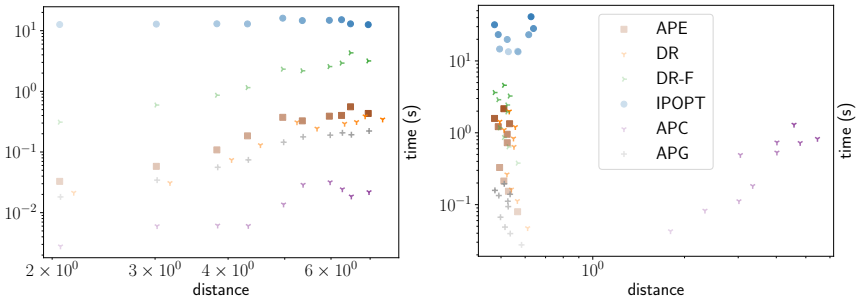
(a) Distance.

(b) Deviation.

(c) Execution time in seconds.

(d) Number of iterations.

Fig. 6.8 Same as Fig. 6.7 for larger dimensions.



(a) Ellipsoid case. **(b)** Hyperboloid case.

Fig. 6.9 Mean execution time versus mean distance for the different developed methods and for dimensions ranging from 100 (dimmest point) to 1000 (clearest point).

Summary

We gather in Fig. 6.9 most results of this section. This plot shows the mean execution time as a function of the mean objective (*e.g.*, the mean distance to the point to be projected). Hence, the closer the method is to the lower left corner, the better. Because in the ellipsoid case all methods find the same solution, we mostly observe in Fig. 6.9a a discrimination with respect to the execution time (except for DR that finds solutions worse than the other methods). The best method is therefore the fastest, *i.e.*, APC. In the hyperboloid case, depicted in Fig. 6.9b, we confirm the observation that APG is the method that finds the best trade-off between execution time and objective.

6.4.2 Alternating projections versus Gurobi

In this section, we benchmark the alternating projections with center-based quasi-projection (APC) against an *exact method*, which aims at finding the *optimal* solution to (2.4). Several methods can be used to tackle (2.4). Here, we choose to use the commercial software Gurobi [Gur18]. This tool deals with the nonconvex quadratic equality *via* piecewise linearization and solves the resulting mixed-integer quadratic programming (MIQP) problem. In this way, a solution along with a lower bound is obtained, and the optimal solution—up to a given tolerance—can be reached, assuming enough time is afforded to the solver.

The problem parameters are chosen so as to resemble problems from the power system literature: the feasible set of the economic dispatch problem with power losses (P) typically exhibits the same structure as the feasible set of (6.2). The entries of $\boldsymbol{A}$ are in the order of 10^{-5} except for diagonal entries (10^{-4}), $\boldsymbol{b}$ is close to -1, and c around 100. $\boldsymbol{A}$ represents the quadratic power losses—expected to be small—and $\boldsymbol{b}$ encodes the constraint stating that the sum of the power production must be equal to the demand ($\approx -c$).

The following quantities are compared:

$$\text{Relative time} = \log\left(1 + \frac{t_{\text{APC}} - t_{\text{Gur}}}{t_{\text{Gur}}}\right) = \log\left(\frac{t_{\text{APC}}}{t_{\text{Gur}}}\right),$$
$$\text{Relative distance} = \log\left(\frac{d_{\text{APC}}}{d_{\text{Gur}}}\right),$$

where t_{APC}, t_{Gur} are the execution times of APC and Gurobi, respectively, and d is the distance between the final iterate and $\boldsymbol{x}^0$, *i.e.*, the objective.

As a way to smooth out random effects, we run $m = 100$ slightly different instances for each dimension and report the mean, *e.g.*,

$$t_{\text{APC}} = \frac{\sum_{i=1}^{m} t^i_{\text{APC}}}{m},$$

where t^i_{APC} stands for the execution time of the $i-$th instance of the alternating projection method.

In the following experiments, it is possible that no feasible point is found. For the alternating projection method, this can occur when the method reaches the maximum number of iterations, *e.g.*, when the method is trapped in a cycle loop (see Fig. 6.1). On the other hand, Gurobi may also fail to yield a feasible point, if the time limit criterion is attained. We do not encode such points in the relative distance. In this way, we do not pollute the reported distance mean by a few instances that terminate due to a timeout. However, we also record the number of timeouts. A timeout occurs either because the method reaches the time limit or because the maximum number of iterations is attained.

One-shot experiment

In this experiment, we intend to compare the speed of both methods. We thus terminate the algorithm as soon as it finds a feasible point, hence the reference to "one-shot". For APC, this does not affect the algorithm. On the other hand, Gurobi relies on lower and upper bounds and terminates

whenever a targeted tolerance is achieved. Here, we modify the stopping criterion such that the algorithm stops as soon as a feasible solution is obtained, no matter the objective. Thus, this is a lower bound on the execution time if the method is run with the tolerance criterion.

Figure 6.10 portrays the (mean) relative execution time and distance. We observe that APC outperforms Gurobi in this experiment. For low dimensions, APC executes at least two times faster and reaches a better solution. For larger dimensions, the differences are even more significant: when the dimension is greater than 40, APC accelerates by a factor of 100 000 and reaches an objective that is 10 times lower than that of Gurobi. Moreover, the number of timeout terminations recorded for Gurobi starts to increase for dimensions greater than 40 (see the bar plot in Fig. 6.10. Hence, the relative time is limited because of the time limit. This explains the saturation of the relative time for large dimensions.

As explained above, the relative distance does not encode the infeasibility of the points that finish with a timeout because such points should have an infinite objective value. The time limit criterion is set at 600 seconds. We note that for problems of large dimension, a significant number of Gurobi's instances (up to 26%) terminate without a solution.

Multiple-shot experiment

In this experiment, we allow Gurobi to execute until it reaches the best solution (up to a given tolerance of one percent) or until timeout (600 seconds). Fig. 6.11 depicts the (mean) relative distance and execution time for 100 runs, as well as the number of timeout terminations of Gurobi. We observe that for small problem instances, *i.e.*, when the dimension is below 13, Gurobi reaches the best solution; *and this solution is close to the one obtained by APC*. Indeed, a relative distance around one means that the solutions returned by the two algorithms are comparable. Since Gurobi does not terminate with a timeout, this implies that the solution returned by APC is, as a matter of fact, also optimal—although no theoretical results support this fact. We also note that the execution time of Gurobi is 10 to 1000 greater than that of APC. For higher dimensions, the relative distance slightly decreases, and the relative time converges to 1×10^{-6}: this is due to the increasing number of timeout terminations. In other words, Gurobi fails to find the best solution in an increasing execution time.

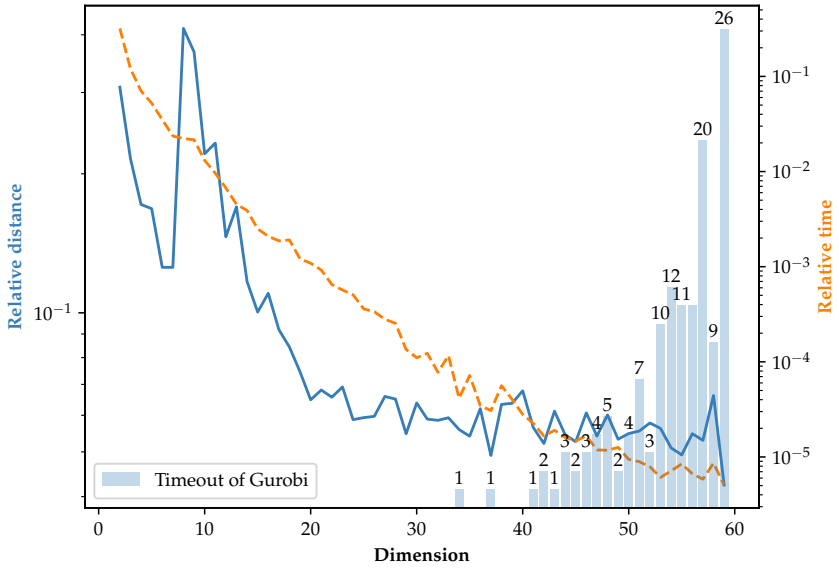


Fig. 6.10 Comparison between the alternating projection method with the center-based quasi-projection (APC) and Gurobi. In this experiment, the method terminates whenever it finds a feasible solution, no matter the objective. The timeout termination of Gurobi is set at 600 seconds, and the number of timeouts for the 100 instances is depicted as a bar plot.

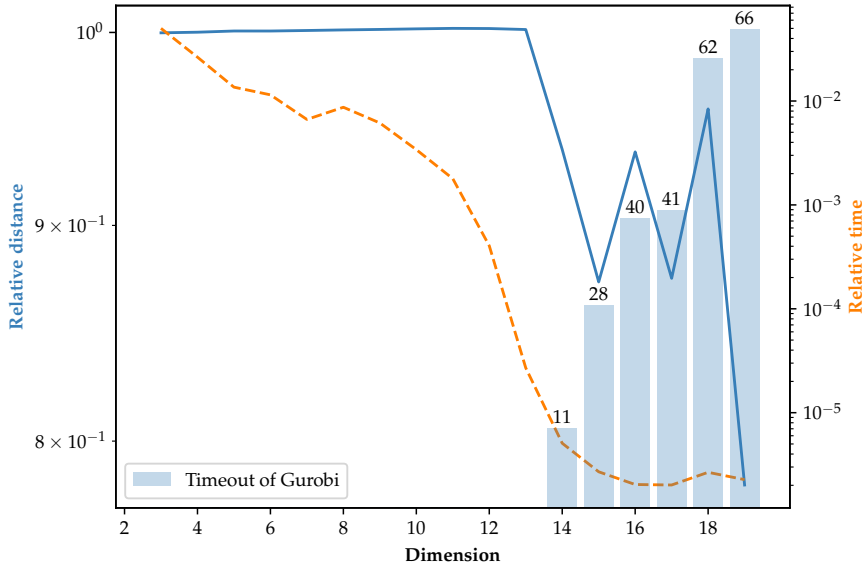


Fig. 6.11 Comparison between the alternating projection method with the center-based quasi-projection (APC) and Gurobi. In this experiment, Gurobi terminates when the objective is proven to be optimal within a 1% tolerance. The timeout termination of Gurobi is set at 600 seconds.

6.5 Conclusion

In this second part of the thesis, the projection onto quadratic hypersurfaces, or quadrics, has been investigated (Chapter 5). We assume that the quadratic hypersurface is a non-cylindrical central quadric. Using the method of Lagrange multipliers, we reduce this nonconvex optimization problem to the problem of finding the solutions to a system of nonlinear equations. We then show how one of the optimal solutions to the projection problem either lies in a finite set of computable solutions or is a root of a univariate scalar-valued nonlinear function. This unique root on a given interval is therefore simply computed with a Newton-Raphson scheme, to which we provide suitable starting points. The cost of this projection is thus cheap, and the bottleneck is the eigendecomposition. This decomposition is needed for diagonalizing the matrix that is used to define the quadric.

We also propose a heuristic, referred to as quasi-projection, based on a geometrical construction. This construction consists in finding the closest intersection of the quadric and a line passing by the point that we want to project. We detail two variants of the quasi-projection, depending on whether the direction of the line is computed as the level-curve gradient of the quadric or the vector joining the center and the point. This quasi-projection does not require the eigendecomposition, thereby economizing in computational time. On the other hand, it does not return the exact projection and fails to map some points to the quadric.

The projection and quasi-projections are then leveraged in the context of splitting algorithms (Chapter 6), namely alternating projections and Douglas-Rachford splitting. These methods allow us to project a point onto a feasible set that is the intersection of a quadric and a box. The extension to the more general case of a Cartesian product of quadrics and a polytope is also discussed. Five methods are proposed depending on whether we use standard Douglas-Rachford splitting (DR), modified Douglas-Rachford splitting (DR-F), or one of the alternating projection methods. We detail the alternating projections with the exact projection on the quadric (APE) or one of the two quasi-projections (the center-based APC or the gradient-based APG).

All methods are tested on problems of dimensions ranging from 10 to 1000 and 100 independent trials are executed for each dimension. Using Ipopt as a benchmark, we find that APE and DF reach the best objectives

and APG is within one percent, while APC and Ipopt lag behind. However, APG is much faster than the other methods and appears to achieve a good trade-off between the quality of the attained objective and the required execution time.

We also test APC on a case similar to the economic dispatch problem from the power system literature and compare it to Gurobi. We show that, in this specific case where the initial point is close to the feasible set, APC quickly reaches a solution close to or better than Gurobi, even if the execution time of Gurobi is several orders of magnitude greater. For small dimensions, Gurobi can guarantee the optimality of its returned solution. However, in this case, the APC solution is identical to Gurobi's: it is therefore also optimal. For higher dimensions, Gurobi terminates with a timeout and with a higher objective than APC. Hence, even APC, which is the poorest of the methods that we propose in Chapter 6 in terms of performance, outperforms Gurobi in these experiments.

As a mirror of the two chapters that make up this second part, the directions of future research are twofold.

For Chapter 5, namely the projection onto non-cylindrical central quadrics, the extension to cylindrical quadrics and non-central could be contemplated. In this context, the linear independence constraint qualification, LICQ, is not fulfilled anymore. We should also include such points as projection candidates. Numerical comparison with the method from [SR20] is another natural research direction for further work.

For Chapter 6, further research may include the comparison with the alternating projection method using inexact projections, *e.g.*, projecting onto the tangent space of the previous feasible point. Such methods are studied in [DL19].

PART III
Conclusions

7

Summary and perspectives

THROUGHOUT this thesis, we have considered two distinct problems: on the one hand, the economic dispatch *per se* and on the other hand, the projection onto a subset of a quadratic hypersurface. While these problems are decorrelated from each other and can thereby be considered separately, we have shown that they can also interact with each other. Indeed, the feasible set of the economic dispatch with power losses can be understood as a subset of a quadratic surface. Because a lot of methods—including ours—that deal with nonconvex sets start by solving a relaxation of the problem, being able to convert an infeasible solution into a feasible one becomes an important tool to have at hand.

Let us recap the contributions that have been proposed in this thesis and give some perspectives for further research.

7.1 What was it all about?

In Chapter 2, the first chapter of Part I, we studied the most simple economic dispatch with units that obey a valve point effect. The consideration of the valve point effect makes the economic dispatch a challenging nonconvex and nonsmooth optimization problem. In contrast to most methods solving nonconvex economic dispatch problems in the literature, *viz.* metaheuristics

that efficiently scan the feasible set for quickly finding a *good* solution, we proposed to adapt the method from [ASS18] by using a piecewise-linear approximation, so as to find the *best* solution. This method requires to solve at each iteration a difficult mixed-integer linear programming problem. Nonetheless, it is guaranteed to converge to the best solution, up to a given tolerance. Resorting to a piecewise-linear approximation, instead of a piecewise-quadratic one, reduces the execution time but complicates the convergence analysis. We studied the numerous ways of modeling such a piecewise function and showed how to extend the method, denoted as APLA, to any sum of piecewise-smooth univariate functions. Finally, we presented two sections standing alone with respect to the remaining of the thesis. In Section 2.5, we discussed the application of the adaptive method of multipliers (ADMM) to the economic dispatch. Such a method works particularly well in the convex case and thoroughly exploits the separability of the objective. However, despite these encouraging features, it is usually unable to capture the global solution. Then in Section 2.6, an interesting preprocessing method is investigated: the bound tightening. This method efficiently reduces the feasible set of the static dispatch but fails to do the same for the larger problem of the dynamic dispatch, which is the focus of Chapter 3.

The dynamic dispatch differs from the static one by one principal aspect: its much bigger size. Transforming the APLA method to cover the dynamic dispatch is straightforward: one simply needs to accordingly define the new surrogate problem. However, the practical use of the APLA method to this larger problem is not viable due to the time limit specification. A ten-minute execution time threshold, which is an acceptable upper bound on economic dispatch models in European markets, is reached several times in our simulations. Hence, we also introduced a matheuristic (*i.e.*, a heuristic that still provides guarantees with respect to the returned solution) that zooms in on subsets of the feasible set in order to quickly provide solutions with competitive objective values. The numerical experiments that are run in this chapter also show the interplay of the valve point effect with the network and demonstrate that it *matters* in these instances: neglecting it can cost up to eight percent.

The closing chapter of the first part, Chapter 4, deals with the power losses. This model improvement increases the difficulty of the problem, as now both the objective and the feasible set are nonconvex. By capitalizing on the previous chapters, we designed a three-step algorithm that i) applies the APLA-based matheuristic on the problem defined on a relaxed set, ii)

projects the (infeasible) obtained point onto the exact nonconvex feasible set, and iii) improves this (first) feasible iterate through a Riemannian subgradient method. In the experiments, we showed that our method is able to provide a low objective as quickly as other methods from the literature, and most importantly, it provides a lower bound. This lower bound allows us to assess how close our solution and any other from the literature are to the global optimum.

The second part of the thesis is motivated by the second step of the three-step algorithm presented in Chapter 4: the projection onto the intersection of the Cartesian product of quadratic surfaces (or quadrics) and a polytope. We formulated this problem in a more general way and studied firstly in Chapter 5 the projection onto a non-cylindrical central quadric. We characterized the projection *via* the KKT conditions and showed how to efficiently compute one of the solutions using the Newton-Raphson method. We also proposed a heuristic to quickly obtain a feasible solution, without the need of computing the eigendecomposition of the matrix associated with the quadratic form that defines the quadric.

Then, we exploited the ability to efficiently compute the projection onto a quadric to obtain the projection onto the intersection of a quadric and a box. To do so, we leveraged the recent advancements in splitting methods for nonconvex programming and implemented the alternating projection method as well as the Douglas-Rachford splitting method. We tested these methods on test cases inspired by our power system application and showed that they outperform black-box solvers such as Gurobi or Ipopt.

7.2 What are the perspectives?

As a reflection of the two parts of the present thesis, the perspectives are twofold.

Concerning the first part, which deals with the nonconvex and nonsmooth economic dispatch, the perspectives are mainly focused on improving the practicality of the model. Several paths should be contemplated.

First, we could take into account multiple fuels and prohibited operation zones in the dispatch. These are other forms of nonconvexities, alongside the valve point effect and the losses, that are relevant in power system operations. This model improvement does not fundamentally increases the difficulty of the problem. The resulting problem can be directly tack-

led using the algorithms proposed in the first part of the thesis, such as the adaptive piecewise-linear approximation (APLA) or the APLA-based matheuristic.

A second perspective concerns the network. Here, we modeled it directly *via* the (linear) DCOPF approximation or indirectly *via* the quadratic power losses, but it is of growing interest to have an accurate representation of the network. We discussed in the conclusion of Chapter 3 that switching from the DCOPF model to a convex relaxation of the ACOPF, *e.g.*, the second-order cone relaxation of the ACOPF (SOC-ACOPF), would be a natural research lead. It may be as simple as to resort to a mixed-integer second-order cone programming (MISOCP) solver. The next step would be to consider the exact ACOPF, but it would require further adjustments to our methods. Indeed, any surrogate problem of APLA would be an MINLP, which is difficult to solve and for which providing lower bounds is challenging.

As far as the second part is concerned, the following perspectives may be studied.

First, we provided in Chapter 5 an algorithm to provide *one* solution to the projection problem. It seems that with little work, it will be possible to provide *all* the solutions. Indeed, when our algorithm is confronted with several solutions at equal distance to the objective, it only selects one solution; even if all are characterized. Other research directions would be to extend the algorithm to all quadrics—including not only non-cylindrical central quadric—and to compare our approach to the one from [SR20].

Lastly, for the second chapter of the second part, it is of interest to study alternating projections with inexact projections, as discussed in [DL19]. It may also be possible to leverage splitting methods for projecting a point onto the feasible set of the ACOPF. In particular, by using the projection that has been developed here to project some point onto the set defined by the ACOPF (with quadratic formulation).

7.3 Any final word?

To conclude, we studied in this thesis the rich problem of nonconvex and nonsmooth economic dispatch. While the field of convex optimization—and especially smooth convex optimization—has already attained some maturity since the first development of Dantzig's simplex algorithm in the late

forties, we have recently witnessed major advancements in the algorithms for nonconvex and nonsmooth problems. Here, we considered methods based on piecewise approximations—for the guarantees of global convergence that they confer—and splitting methods—because they exploit well the structure of separable objectives. It goes without saying that research on nonsmooth and nonconvex problems has a bright future ahead, and while the methods developed here may not be the fastest in the economic dispatch literature, we hope they may be useful as a matter of benchmarking any potential new method and to figure out whether a given solution is close to the optimal one.

Appendices

A Parameters

A.1 Static dispatch

Table A.1 Parameter unit.

A_g	$\$/\mathrm{MW}^2\mathrm{h}$
B_g	$\$/\mathrm{MWh}$
C_g	$\$/\mathrm{h}$
D_g	$\$/\mathrm{h}$
E_g	MW^{-1}
$\underline{P}_g$	MW
$\overline{P}_g$	MW
$\underline{R}_g$	MW
$\overline{R}_g$	MW
P_t^{D}	MW
P_t^{S}	MW

Table A.2 Parameters of the 3-unit test case with a demand of $P^{\mathrm{D}} = 850\,\mathrm{MW}$.

Units	$\underline{P}_g$	$\overline{P}_g$	A_g	B_g	C_g	D_g	E_g
1	100	600	0.001562	7.92	561	300	0.0315
2	50	200	0.004820	7.97	78	150	0.063
3	100	400	0.001940	7.85	310	200	0.042

Table A.3 Parameters of the 40-unit test case with a demand of $P^{D} = 10\,050$ MW.

Units	A_g	B_g	C_g	D_g	E_g	$\underline{P}_g$	$\overline{P}_g$
1	0.0069	6.73	94.705	100	0.084	36	114
2	0.0069	6.73	94.705	100	0.084	36	114
3	0.02028	7.07	309.54	100	0.084	60	120
4	0.00942	8.18	369.03	150	0.063	80	190
5	0.0114	5.35	148.89	120	0.077	47	97
6	0.01142	8.05	222.33	100	0.084	68	140
7	0.00357	8.03	287.71	200	0.042	110	300
8	0.00492	6.99	391.98	200	0.042	135	300
9	0.00573	6.6	455.76	200	0.042	135	300
10	0.00605	12.9	722.82	200	0.042	130	300
11	0.00515	12.9	635.2	200	0042	94	375
12	0.00569	12.8	654.69	200	0.042	94	375
13	0.00421	12.5	913.4	300	0.035	125	500
14	0.00752	8.84	1760.4	300	0.035	125	500
15	0.00708	9.15	1728.3	300	0.035	125	500
16	0.00708	9.15	1728.3	300	0.035	125	500
17	0.00313	7.97	647.85	300	0.035	220	500
18	0.00313	7.95	649.69	300	0.035	220	500
19	0.00313	7.97	647.83	300	0.035	242	550
20	0.00313	7.97	647.81	300	0.035	242	550
21	0.00298	6.63	785.96	300	0.035	254	550
22	0.00298	6.63	785.96	300	0.035	254	550
23	0.00284	6.66	794.53	300	0.035	254	550
24	0.00284	6.66	794.53	300	0.035	254	550
25	0.00277	7.1	801.32	300	0.035	254	550
26	0.00277	7.1	801.32	300	0.035	254	550
27	0.52124	3.33	1055.1	120	0.077	10	150
28	0.52124	3.33	1055.1	120	0.077	10	150
29	0.52124	3.33	1055.1	120	0.077	10	150
30	0.0114	5.35	148.89	120	0.077	47	97
31	0.0016	6.43	222.92	150	0.063	60	190
32	0.0016	6.43	222.92	150	0.063	60	190
33	0.0016	6.43	222.92	150	0.063	60	190
34	0.0001	8.95	107.87	200	0.042	90	200
35	0.0001	8.62	116.58	200	0.042	90	200
36	0.0001	8.62	116.58	200	0.042	90	200
37	0.0161	5.88	307.45	80	0.098	25	110
38	0.0161	5.88	307.45	80	0.098	25	110
39	0.0161	5.88	307.45	80	0.098	25	110
40	0.00313	7.97	647.83	300	0.035	242	550

A.2 Dynamic dispatch

Table A.4 Parameters of the 10-unit dynamic test case.

Units	A_g	B_g	C_g	D_g	E_g	$\underline{P}_g$	$\overline{P}_g$	$\overline{R}_g$	$\underline{R}_g$
1	0.00043	21.6	958.2	450	0.041	150	470	80	80
2	0.00063	21.05	1313.6	600	0.036	135	460	80	80
3	0.00039	20.81	604.97	320	0.028	73	340	80	80
4	0.0007	23.9	471.6	260	0.052	60	300	50	50
5	0.00079	21.62	480.29	280	0.063	73	243	50	50
6	0.00056	17.87	601.75	310	0.048	57	160	50	50
7	0.00211	16.51	502.7	300	0.086	20	130	30	30
8	0.0048	23.23	639.4	340	0.082	47	120	30	30
9	0.10908	19.58	455.6	270	0.098	20	80	30	30
10	0.00951	22.54	692.4	380	0.094	55	55	30	30

Table A.5 Demand and reserve requirement of the 10-unit dynamic test case.

Time step	P_t^{D}	P_t^{S}
1	1036	51.8
2	1110	55.5
3	1258	62.9
4	1406	70.3
5	1480	74
6	1628	81.4
7	1702	85.1
8	1776	88.8
9	1924	96.2
10	2072	103.6
11	2146	107.3
12	2220	111
13	2072	103.6
14	1924	96.2
15	1776	88.8
16	1554	77.7
17	1480	74
18	1628	81.4
19	1776	88.8
20	2072	103.6
21	1924	96.2
22	1628	81.4
23	1332	66.6
24	1184	59.2

B Proofs

B.1 Proof of Proposition 2.1

Proof. As h and f are separable sums of univariate scalar functions, we consider a given asset $g \in G$; we find the corresponding maximal over-approximation error $\epsilon_g^{\max}$ and then compute $\epsilon^{\max} = \sum_{g \in G} \epsilon_g^{\max}$.

To simplify the proof, we assume that every f_g is increasing[1]. But the proof remains valid for nonincreasing f_g.

The structure of the proof is twofold: we first characterize the inner and then the outer max of (2.16).

Inner max

Let us first study the inner max, and let h_g be a piecewise-linear interpolation of f_g defined by some knots $\boldsymbol{X}_g^{\text{knot}}$, *i.e.*, $h_g(p_g) = \hat{f}_g(p_g; \boldsymbol{X}_g^{\text{knot}})$ for all $p_g \in [\underline{P}_g, \overline{P}_g]$. We assume that the set of knots includes the initial knots: $\boldsymbol{X}_g^{\text{knot,I}} \subseteq \boldsymbol{X}_g^{\text{knot}}$. Therefore, $h_g \in \mathcal{H}_{f_g}(\boldsymbol{X}_g^{\text{knot,I}})$. Let

$$x_g^* \in \underset{x \in [\underline{P}_g, \overline{P}_g]}{\operatorname{argmax}} \underbrace{h_g(x) - f_g(x)}_{:= e_g(x)} := \epsilon_g,$$

the existence of x_g^* follows from, *e.g.*, [Bec14, Theorem 2.32]: the objective is a real-valued, continuous and coercive function defined on a nonempty closed set. The uniqueness of x_g^* is not guaranteed, but it is neither needed: we are only interested in the value of the maximal error.

There exist therefore $X_g^{\text{L}}, X_g^{\text{R}} \in \boldsymbol{X}_g^{\text{knot}}$ such that $X_g^{\text{L}} \leq x_g^* \leq X_g^{\text{R}}$. Remark that if $f_g\big|_{[X_g^{\text{L}}, X_g^{\text{R}}]}$ is concave then $e_g(x) \leq 0$ and the maximum ($\epsilon_g = 0$) is attained in, *e.g.*, X_g^{L}. We have either $\epsilon_g > 0$ (and $x_g^* \in (X_g^{\text{L}}, X_g^{\text{R}})$) or $\epsilon_g = 0$. Let us show that in both cases the characterization of x_g^* is the same.

In the first case, it follows that $e_g'(x_g^*) = 0 \Leftrightarrow h_g'(x_g^*) - f_g'(x_g^*) = 0$, *i.e.*,

$$f_g'(x_g^*) = \frac{f_g(X_g^{\text{L}}) - f_g(X_g^{\text{R}})}{X_g^{\text{L}} - X_g^{\text{R}}}. \tag{A.1}$$

Geometrically, the tangent of f_g at x_g^* must be parallel to the linear piece of the current segment. Note that (A.1) also characterizes the minimizers of $e_g(x)$.

In the second case, the convexity of f_g around each kink point (which are included in the set of knots) implies that the interpolation must be tangent at this kink point, and we have (A.1) with $x_g^* = X_g^{\text{L}}$ for X_g^{L} some kink point.

Outer max

Let us now analyze the outer maximum. We consider the educated guess $h_g^{\max}$ defined with a set of knots $\boldsymbol{X}_g^{\max} = \boldsymbol{X}_g^{\text{knot,I}} \bigcap \boldsymbol{X}_g^{\text{M}}$. This set of knots is

[1] This is effectively the case for the cost functions considered in Part I: the higher the asset output, the higher the fuel input, the higher the price.

the intersection between $\boldsymbol{X}_g^{\text{knot,I}}$ and some additional knots defined as

$$\boldsymbol{X}_g^{\text{M}} := \Big\{ X_g^{\text{M}} \in (X_g^{\text{L}}, X_g^{\text{R}}) \text{ with } X_g^{\text{L}}, X_g^{\text{R}} \text{ two consecutive knots of } \boldsymbol{X}_g^{\text{knot,I}}, \\ f_g'(X_g^{\text{M}}) = \frac{f_g(X_g^{\text{M}}) - f_g(X_g^{\text{L}})}{X_g^{\text{M}} - X_g^{\text{L}}}, \\ \text{and } X_g^{\text{L}} \in \boldsymbol{X}_g^{\text{kink}} \Big\}.$$

An illustrative $h_g^{\max}$ is given in Fig. 2.7 (dashed line).

Remark that we can have $X_g^{\text{L}} < X_g^{\text{R}}$ or the opposite. For the sake of keeping this discussion as short as possible, we only treat the former, but the latter case can be treated similarly.

In the remainder of the proof, we show that no other function in $\mathcal{H}_{f_g}(\boldsymbol{X}_g^{\text{knot,I}})$ yields a greater over-approximation error than $h_g^{\max}$.

Let $\epsilon_g^{\max}$ be the maximal over-approximation error of $h_g^{\max}$—this error has been characterized in the first part of the proof—and let us consider some $h_g^* \in \mathcal{H}_f(\boldsymbol{X}^{\text{knot,I}})$ defined with some knots $\boldsymbol{X}_g^*$ so that

$$\epsilon_g^{\max} \leq \max_{x \in [\underline{P}_g, \overline{P}_g]} h_g^*(x) - f(x) := \epsilon_g^*.$$

For a function f_g with some convex region, we have $\epsilon_g^* > 0$; any interpolation defined with two knots within the convex region yields a positive over-approximation error. There exists a point $x_g^* \in (\underline{P}_g, \overline{P}_g)$ where this maximum occurs, and this point must belong to some piece.

We first establish that this piece is supported on some interval where f_g contains a convex and a concave part. Then, we show that this interval is defined by a kink point and some X_g^{M} that satisfies

$$f_g'(X_g^{\text{M}}) = \frac{f_g(X_g^{\text{M}}) - f_g(X_g^{\text{L}})}{X_g^{\text{M}} - X_g^{\text{L}}}.$$

Finally, we treat the case where such a point does not exist and provide a replacement: X_g^{R}.

The interval supporting the piece where the max is attained contains a convex and a concave part. Let h_g^* be as defined above, and let $X_g^-, X_g^+ \in \boldsymbol{X}_g^*$ be two consecutive knots with $X_g^- < x_g^* < X_g^+$. There exist $X_g^{\text{L}}, X_g^{\text{R}} \in$

$\boldsymbol{X}^{\text{knot,I}} \subseteq \boldsymbol{X}_g^*$ such that

$$X_g^{\text{L}} \leq X_g^- < X_g^+ \leq X_g^{\text{R}}.$$

Due to the definition of f_g and $\boldsymbol{X}_g^{\text{knot,I}}$, there is at most one inflection point on each piece of the piecewise interpolation. Hence, there exists a unique inflection point $X_g^{\text{I}} \in (X_g^{\text{L}}, X_g^{\text{R}})$. We consider without loss of generality that f_g is convex on $[X_g^{\text{L}}, X_g^{\text{I}}]$ and concave on $[X_g^{\text{I}}, X_g^{\text{R}}]$, as it is the case in Fig. 2.7. The opposite case can be dealt with analogously. We have:

- X_g^- and X_g^+ cannot jointly lie in $[X_g^{\text{L}}, X_g^{\text{I}})$ because the convexity of f_g on $[X_g^{\text{L}}, X_g^{\text{I}}]$ implies that the segment defined by $(X_g^{\text{L}}, f_g(X_g^{\text{L}}))$ and $(X_g^{\text{I}}, f_g(X_g^{\text{I}}))$ is above each other chord;
- X_g^- and X_g^+ cannot jointly lie in $[X_g^{\text{I}}, X_g^{\text{R}}]$ because the concavity of f_g on $[X_g^{\text{I}}, X_g^{\text{R}}]$ implies that each chord ranging from X^- to X^+ under-approximates f_g, therefore yielding a zero over-approximation error.

Hence $X_g^- \in [X_g^{\text{L}}, X_g^{\text{I}})$ and $X_g^+ \in [X_g^{\text{I}}, X_g^{\text{R}}]$.

Let us show that $X_g^- = X_g^{\text{L}}$ and $X_g^+ = X_g^{\text{M}}$, with X_g^{M} satisfying

$$f_g'(X_g^{\text{M}}) = \frac{f_g(X_g^{\text{M}}) - f_g(X_g^{\text{L}})}{X_g^{\text{M}} - X_g^{\text{L}}}. \tag{A.2}$$

We treat the case where such X_g^{M} does not exist at the end.

If there is some point X_g^{M} satisfying (A.2) on the piece where the max is attained, then $\epsilon_g^{\max} = \epsilon_g^*$. Assume that $X_g^{\text{L}} \leq X_g^- < X_g^{\text{I}}$ and that ϵ_g^* is attained at some $x_g^* \in (X_g^-, X_g^+)$ with $X_g^+ \in [X_g^{\text{I}}, X_g^{\text{R}}]$. We have

$$h_g^*\Big|_{[X_g^-, X_g^+]}(\cdot) = \hat{f}_g(\cdot; \{X_g^-, X_g^+\}).$$

We can construct another interpolation

$$h_g^{**}(\cdot) := \hat{f}_g(\cdot; \{X_g^{\text{L}}, X_g^+\}).$$

The convexity of the increasing function f_g on $[X_g^{\text{L}}, X_g^{\text{I}}]$ implies that

$$h_g^{**}(X_g^-) \geq h_g^*(X_g^-) = f_g(X_g^-).$$

It follows that $h_g^{**}(x) \geq h_g^*(x)$ for all $x \in [X_g^-, X_g^{\mathrm{R}}]$ and in particular $h_g^{**}(x_g^*) \geq h_g^*(x_g^*)$. Hence, $h_g^{**}(x_g^*)$ is also optimal and we can consider $X_g^- = X_g^{\mathrm{L}}$.

Assume now that $X_g^+ \in [X_g^{\mathrm{I}}, X_g^{\mathrm{M}}]$ and that ϵ_g^* is attained at some $x_g^* \in (X_g^{\mathrm{L}}, X_g^+)$. We have

$$h_g^*\Big|_{[X_g^{\mathrm{L}}, X_g^+]}(\cdot) = \hat{f}_g(\cdot; \{X_g^{\mathrm{L}}, X_g^+\}).$$

The concavity of f_g on $[X_g^{\mathrm{I}}, X_g^{\mathrm{R}}]$ implies that each tangent line to a point in $[X_g^{\mathrm{I}}, X_g^{\mathrm{R}}]$ over-approximates f_g. This explains the definition of X_g^{M} such that the slope of the piece it defines matches the tangent of f_g at X_g^{M}. Let

$$h_g^{**}(\cdot) := \hat{f}_g(\cdot; \{X_g^{\mathrm{L}}, X_g^{\mathrm{M}}\}),$$

the concavity of f_g on $[X_g^{\mathrm{I}}, X_g^{\mathrm{R}}]$ implies that $h_g^{**}(X_g^+) \geq h_g^*(X_g^+) = f(X_g^+)$. Therefore, we have in a similar fashion as before, $h_g^{**}(x) \geq h_g^*(x)$ for all $x \in [X_g^{\mathrm{L}}, X_g^+]$ and in particular $h_g^{**}(x_g^*) \geq h_g^*(x_g^*)$. Hence, $h_g^{**}(x_g^*)$ is also optimal. We can therefore consider $X_g^+ = X_g^{\mathrm{M}}$.

Note that $X_g^+ > X_g^{\mathrm{M}}$ is not possible because either there is some $X_g^c \in [X_g^{\mathrm{L}}, X_g^{\mathrm{M}}]$ with $f_g(X_g^c) = h_g^*(X_g^c)$—in contradiction with the definition of X_g^{L} as the left knot of the piece—or $f_g(X_g^+) < (X_g^+ - X_g^{\mathrm{L}}) f_g'(X_g^{\mathrm{L}})$ and this cannot hold because f_g is increasing.

In this section, we proved that if the maximal over-approximation error of h_g^* occurs in some closed interval $[X_g^-, X_g^+] \subseteq [X_g^{\mathrm{L}}, X_g^{\mathrm{R}}]$ where X_g^{M} exists, then this over-approximation error is similar to the one obtained with an interpolation defined by the two knots X_g^{L} and X_g^{M}. However, we defined $h_g^{\max}$ such that X_g^{L} and X_g^{M} are two consecutive knots of this interpolation. Therefore, $h_g^{\max}(x_g^*) = h_g^*(x_g^*)$, and we have

$$\epsilon_g^{\max} = \epsilon_g^*.$$

What if there is no point satisfying (A.2) on the piece where the max is attained? Finally, let us assume that no point verifies

$$f_g'(X_g^{\mathrm{M}}) = \frac{f_g(X_g^{\mathrm{M}}) - f_g(X_g^{\mathrm{L}})}{X_g^{\mathrm{M}} - X_g^{\mathrm{L}}}$$

within the interval $[X_g^{\mathrm{L}}, X_g^{+}]$ on which the maximal over-approximation error of h_g^* occurs. Let $X_g^{\mathrm{M}} = X_g^{\mathrm{R}}$, and let us assume that ϵ_g^* is attained at some $x_g^* \in (X_g^{\mathrm{L}}, X_g^{+})$ with $X_g^{+} \in [X_g^{\mathrm{I}}, X_g^{\mathrm{R}}]$. Let

$$h_g^*\Big|_{[X_g^{\mathrm{L}}, X_g^{+}]}(\cdot) = \hat{f}_g(\cdot; \{X_g^{\mathrm{L}}, X_g^{+}\}),$$

be the corresponding interpolation. The concavity of f_g on $[X_g^{\mathrm{I}}, X_g^{\mathrm{R}}]$ implies that its slope is decreasing. Thus, let us consider

$$h_g^{**}(\cdot) := \hat{f}_g(\cdot; \{X_g^{\mathrm{L}}, X_g^{\mathrm{R}}\}).$$

Because the slope of this segment is above $f_g'(X_g^{\mathrm{R}})$ and X_g^{R} is the right limit of the segment, h_g^{**} over-approximates the concave part. The rest of the proof is similar as before: we show that $h_g^{**}(X_g^{+}) \geq h_g^*(X_g^{+}) = f(X_g^{+})$, and therefore that h_g^{**} is above h_g^* on $(X_g^{\mathrm{L}}, X_g^{+})$. Thus, we can as well take $X_g^{+} = X_g^{\mathrm{M}} = X_g^{\mathrm{R}}$.

Similarly as the previous section, the definition of $h_g^{\max}$ implies that $h_g^{\max}(x_g^*) = h_g^*(x_g^*)$, and we have

$$\epsilon_g^{\max} = \epsilon_g^*.$$

□

List of Publications

⋆ Publications submitted

[VAP22a] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. Projection onto quadratic hypersurfaces, 2022. arXiv:2204.02087.

⋆ Journal papers

[VAP22b] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. Solving non-convex economic dispatch with valve-point effects and losses with guaranteed accuracy. *International Journal of Electrical Power & Energy Systems*, 134:107143, January 2022. doi:10.1016/j.ijepes.2021.107143.

[VAP20a] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. Global solution of economic dispatch with valve point effects and transmission constraints. *Electric Power Systems Research*, 189:106786, 2020. doi:10.1016/j.epsr.2020.106786.

⋆ Refereed conference papers

[VAP20b] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. Global Solution of Economic Dispatch with Valve Point Effects and Trans-

mission Constraints. In *2020 Power Systems Computation Conference (PSCC)*, pages 1–8, 2020. Published as [VAP20a].

[VPA19] Loïc Van Hoorebeeck, Anthony Papavasiliou, and P.-A. Absil. MILP-based algorithm for the global solution of dynamic economic dispatch problems with valve-point effects. In *2019 IEEE Power Energy Society General Meeting (PESGM)*, 2019. doi:10.1109/PESGM40551.2019.8973631.

⋆ Abstracts

[VAP21] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. A matheuristic for solving non-convex economic dispatches. European Conference on Operational Research (EURO), 2021.

[VAP19a] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. MILP-Based Algorithm for the Global Solution of Dynamic Economic Dispatch with Valve-Point Effects. The International Council for Industrial and Applied Mathematics (ICIAM), 2019.

[VAP19b] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. MILP-Based Algorithm for the Global Solution of Dynamic Economic Dispatch with Valve-Point Effects. European Conference on Operational Research (EURO), 2019.

Bibliography

[AACT20] Francisco J. Aragón Artacho, Rubén Campoy, and Matthew K. Tam. The Douglas–Rachford algorithm for convex and nonconvex feasibility problems. *Mathematical Methods of Operations Research*, 91(2):201–240, April 2020. doi:10.1007/s00186-019-00691-9.

[ABB+20] Christie Alappat, Achim Basermann, Alan R. Bishop, Holger Fehske, Georg Hager, Olaf Schenk, Jonas Thies, and Gerhard Wellein. A recursive algebraic coloring technique for hardware-efficient symmetric sparse matrix-vector multiplication. *ACM Trans. Parallel Comput.*, 7(3), June 2020. doi:10.1145/3399732.

[ABRS10] Hédy Attouch, Jérôme Bolte, Patrick Redont, and Antoine Soubeyran. Proximal Alternating Minimization and Projection Methods for Nonconvex Problems: An Approach Based on the Kurdyka-Łojasiewicz Inequality. *Mathematics of Operations Research*, 35(2):438–457, May 2010. doi:10.1287/moor.1100.0449.

[AHK+08] Tilo Arens, Frank Hettlich, Christian Karpfinger, Ulrich Kockelkorn, Klaus Lichtenegger, and Hellmuth Stachel. *Mathematik*. Spektrum Akademischer Verlag, 1. aufl. 2008 edition, 2 2008.

[AKTH02] P. Attaviriyanupap, H. Kita, E. Tanaka, and J. Hasegawa. A hybrid EP and SQP for dynamic economic dispatch with nonsmooth fuel cost function. *IEEE Transactions on Power Systems*, 17(2):411–416, May 2002. doi:10.1109/TPWRS.2002.1007911.

[AP17] I. Aravena and A. Papavasiliou. Renewable energy integration in zonal markets. *IEEE Transactions on Power Systems*, 32(2):1334–1349, 2017. doi:10.1109/TPWRS.2016.2585222.

[AR15] Ali R. Al-Roomi. Power Flow Test Systems Repository, 2015. Online; Accessed: 2019-08-07. URL: al-roomi.org/power-flow.

[ASS18] P.-A. Absil, B. Sluysmans, and N. Stevens. MIQP-based algorithm for the global solution of economic dispatch problems with valve-point effects. In *2018 Power Systems Computation Conference (PSCC)*, pages 1–7, June 2018. doi:10.23919/PSCC.2018.8450877.

[Bas19] M. Basu. Squirrel search algorithm for multi-region combined heat and power economic dispatch incorporating renewable energy sources. *Energy*, 182:296–305, September 2019. doi:10.1016/j.energy.2019.06.087.

[BB96] Heinz H. Bauschke and Jonathan M. Borwein. On Projection Algorithms for Solving Convex Feasibility Problems. *SIAM Review*, 38(3):367–426, September 1996. doi:10.1137/S0036144593251710.

[BCL02] Heinz H. Bauschke, Patrick L. Combettes, and D. Russell Luke. Phase retrieval, error reduction algorithm, and Fienup variants: a view from convex optimization. *Journal of the Optical Society of America A*, 19(7):1334, July 2002. doi:10.1364/JOSAA.19.001334.

[Bec14] Amir Beck. *Introduction to Nonlinear Optimization*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, October 2014. doi:10.1137/1.9781611973655.

[BF01] Richard L. Burden and J. Douglas Faires. *Numerical analysis*. Brooks/Cole, 7th edition, 2001.

[BM17] Sören Bartels and Marijo Milicevic. Alternating direction method of multipliers with variable step sizes, April 2017. arXiv:1704.06069.

[BMRBR09] Marco A. Boschetti, Vittorio Maniezzo, Matteo Roffilli, and Antonio Bolufé Röhler. Matheuristics: Optimization, Simulation and Control. In *Hybrid Metaheuristics*, Lecture Notes in Computer Science, pages 171–177, Berlin, Heidelberg, 2009. Springer. doi:10.1007/978-3-642-04918-7_13.

[Boy10] Stephen Boyd. Alternating Direction Method of Multipliers, 2010.

[BPC11] Stephen Boyd, Neal Parikh, and Eric Chu. *Distributed Optimization and Statistical Learning Via the Alternating Direction Method of Multipliers*. Now Publishers Inc, 2011.

[Bre65] Lev Meerovich Bregman. Finding the common point of convex sets by the method of successive projection. In *Doklady Akademii Nauk*, volume 162, pages 487–490. Russian Academy of Sciences, 1965.

[Bro20] Scott Brown. *Local Model Feature Transformations*. PhD thesis, The University of South Alabama, may 2020.

[BSBA13] Pierre B. Borckmans, S. Easter Selvan, Nicolas Boumal, and P.-A. Absil. A Riemannian subgradient algorithm for economic dispatch with valve-point effect. *J. Comput. Applied. Math.*, 255:848–866, 2013. doi:10.1016/j.cam.2013.07.002.

[BSS06] Mokhtar S. Bazaraa, Hanif D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley-Interscience, Hoboken, N.J, 3rd edition edition, May 2006.

[BV04] Stephen P. Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, UK ; New York, 2004.

[CGT00] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. *Trust Region Methods*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, January 2000. doi:10.1137/1.9780898719857.

[Chr99] Richard D. Christie. Power systems test case archive, 1999. Online; Accessed: 2019-08-07. URL: http://labs.ece.uw.edu/pstca.

[CHVH16] Carleton Coffrin, Hassan L. Hijazi, and Pascal Van Hentenryck. The QC Relaxation: A Theoretical and Computational Study on Optimal Power Flow. *IEEE Transactions on Power Systems*, 31(4):3008–3018, July 2016. doi:10.1109/TPWRS.2015.2463111.

[Cla76] Frank H. Clarke. A New Approach to Lagrange Multipliers. *Mathematics of Operations Research*, 1(2):165–174, 1976. doi:10.1287/moor.1.2.165.

[CM06] Leandros dos Santos Coelho and Viviana Cocco Mariani. Combining of chaotic differential evolution and quadratic programming for economic dispatch optimization with valve-point effect. *IEEE Transactions on power systems*, 21(2):989–996, 2006. doi:10.1109/TPWRS.2006.873410.

[CY06] C. H. Chen and S. N. Yeh. Particle swarm optimization for economic power dispatch with valve-point effects. In *2006 IEEE/PES Transmission Distribution Conference and Exposition: Latin America*, pages 1–5, Aug 2006. doi:10.1109/TDCLA.2006.311397.

[DB58] G. L. Decker and A. D. Brooks. Valve point loading of turbines. *Electrical Engineering*, 77(6):501–501, June 1958. doi:10.1109/EE.1958.6445133.

[DDTA13] Joydeep Dutta, Kalyanmoy Deb, Rupesh Tulshyan, and Ramnik Arora. Approximate KKT points and a proximity measure for termination. *Journal of Global Optimization*, 56(4):1463–1499, August 2013. doi:10.1007/s10898-012-9920-5.

[DG86] Marco A. Duran and Ignacio E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Math. Program.*, 36(3):307–339, October 1986. doi:10.1007/BF02592064.

[DHL17] Iain Dunning, Joey Huchette, and Miles Lubin. JuMP: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017. doi:10.1137/15M1020575.

[DIL15] D. Drusvyatskiy, A. D. Ioffe, and A. S. Lewis. Transversality and Alternating Projections for Nonconvex Sets. *Foundations*

of Computational Mathematics, 15(6):1637–1651, December 2015. doi:10.1007/s10208-015-9279-3.

[DL19] D. Drusvyatskiy and A. S. Lewis. Local Linear Convergence for Inexact Alternating Projections on Nonconvex Sets. *Vietnam Journal of Mathematics*, 47(3):669–681, September 2019. doi:10.1007/s10013-019-00357-3.

[Dru13] D. Drusvyatskiy. *Slope And Geometry In Variational Mathematics*. PhD thesis, Cornell university, August 2013.

[EHBE16] W. T. Elsayed, Y. G. Hegazy, F. M. Bendary, and M. S. El-Bages. A review on accuracy issues related to solving the non-convex economic dispatch problem. *Electric Power Systems Research*, 141:325–332, December 2016. doi:10.1016/j.epsr.2016.08.002.

[Eur11] European Commission. Energy roadmap 2050, Dec 2011.

[Eur21] European Environment Agency (EEA). Trends and Projections in Europe - EEA report, November 2021.

[EY15] Jonathan Eckstein and Wang Yao. Understanding the Convergence of the Alternating Direction Method of Multipliers: Theoretical and Computational Perspectives. *Pacific Journal of Optimization*, 11(4):39, 2015.

[Fan10] Daniele Fanelli. Do Pressures to Publish Increase Scientists' Bias? An Empirical Support from US States Data. *PLoS ONE*, 5(4):e10271, April 2010. doi:10.1371/journal.pone.0010271.

[FM15] James Fletcher and Warren B. Moors. Chebyshev sets. *Journal of the Australian Mathematical Society*, 98(2):161–231, April 2015. doi:10.1017/S1446788714000561.

[GP10] Victor Guillemin and Alan Pollack. *Differential topology*, volume 370. American Mathematical Soc., 2010.

[Gur18] Gurobi Optimization Inc. Gurobi Optimizer Reference Manual, 2018. Online; Accessed: 2019-08-07. URL: http://www.gurobi.com.

[GVL13] Gene H. Golub and Charles F. Van Loan. *Matrix computations*. Johns Hopkins studies in the mathematical sciences. The Johns Hopkins University Press, Baltimore, fourth edition edition, 2013.

[Hal62] Israel Halperin. The product of projection operators. *Acta Sci. Math.(Szeged)*, 23(1):96–99, 1962.

[HH71] Edward T. Heise and Werner Heisenberg. *Physics and Beyond: Encounters and Conversations*. Harper & Row, 1971.

[HIR62] H. H. Happ, W. B. Ille, and R. H. Reisinger. Economic System Operation Considering Valve Throttling Losses I-Method Computing Valve-Loop Heat Rates on Multivalve Turbines. *Transactions of the American Institute of Electrical Engineers. Part III: Power Apparatus and Systems*, 81(3):609–615, April 1962. doi:10.1109/AIEEPAS.1962.4501374.

[HS11] S. Hemamalini and Sishaj P. Simon. Dynamic economic dispatch using artificial immune system for units with valve-point effect. *International Journal of Electrical Power & Energy Systems*, 33(4):868 – 874, 2011. doi:10.1016/j.ijepes.2010.12.017.

[HV19] Joey Huchette and Juan Pablo Vielma. Nonconvex piecewise linear functions: Advanced formulations and simple modeling tools, 2019. arXiv:1708.00050.

[HWCH20] Shih-Feng Huang, Yung-Hsuan Wen, Chi-Hsiang Chu, and Chien-Chin Hsu. A Shape Approximation for Medical Imaging Data. *Sensors*, 20(20):5879, January 2020. doi:10.3390/s20205879.

[JEOC16] Johns Hopkins Univ., Baltimore, MD (United States), Brent Eldridge, Richard O'Neill, and Andrea Castillo. Marginal Loss Calculations for the DCOPF. Technical Report SAND2017-0563R, 1340633, 650548, Federal Energy Regulatory Commission (FERC), December 2016. doi:10.2172/1340633.

[JuM] JuMP documentation: Nonlinear modeling. URL: https://jump.dev/JuMP.jl/stable/nlp/.

[KCSB15] Mojtaba Kadkhodaie, Konstantina Christakopoulou, Maziar Sanjabi, and Arindam Banerjee. Accelerated Alternating

Direction Method of Multipliers. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 497–506, New York, NY, USA, August 2015. Association for Computing Machinery. doi:10.1145/2783258.2783400.

[Kir58] L.K. Kirchmayer. *Economic Operation of Power Systems*. General Electric series. Wiley, 1958.

[Kro51] G. Kron. Tensorial analysis of integrated transmission systems part i. the six basic reference frames. *Transactions of the American Institute of Electrical Engineers*, 70(2):1239–1248, 1951. doi:10.1109/T-AIEE.1951.5060553.

[KSAA13] M. Karami, H.A. Shayanfar, J. Aghaei, and A. Ahmadi. Scenario-based security-constrained hydrothermal coordination with volatile wind power generation. *Renewable and Sustainable Energy Reviews*, 28:726 – 737, 2013. doi:10.1016/j.rser.2013.07.052.

[Kun13] Friedrich Kunz. Improving congestion management: How to facilitate the integration of renewable generation in germany. *The Energy Journal*, Volume 34(Number 4), 2013.

[LB93] F. N. Lee and A. M. Breipohl. Reserve constrained economic dispatch with prohibited operating zones. *IEEE Transactions on Power Systems*, 8(1):246–254, 1993. doi:10.1109/59.221233.

[LFL21] Xueping Li, Linhai Fu, and Zhigang Lu. A novel constraints handling mechanism based on virtual generator unit for economic dispatch problems with valve point effects. *International Journal of Electrical Power & Energy Systems*, 129:106825, July 2021. doi:10.1016/j.ijepes.2021.106825.

[LI14] Gus K. Lott III. Direct Orthogonal Distance to Quadratic Surfaces in 3D. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(9):1888–1892, September 2014. doi:10.1109/TPAMI.2014.2302451.

[LLM09] A. S. Lewis, D. R. Luke, and Jérôme Malick. Local Linear Convergence for Alternating and Averaged Nonconvex Projections. *Foundations of Computational Mathematics*, 9(4):485–513, August 2009. doi:10.1007/s10208-008-9036-y.

[LM08] A. S. Lewis and Jérôme Malick. Alternating Projections on Manifolds. *Mathematics of Operations Research*, 33(1):216–234, February 2008. doi:10.1287/moor.1070.0291.

[Low14] S. H. Low. Convex relaxation of optimal power flow—part II: Exactness. *IEEE Transactions on Control of Network Systems*, 1(2):177–189, June 2014. doi:10.1109/TCNS.2014.2323634.

[LP16] Guoyin Li and Ting Kei Pong. Douglas–Rachford splitting for nonconvex optimization with application to nonconvex feasibility problems. *Mathematical Programming*, 159(1):371–401, September 2016. doi:10.1007/s10107-015-0963-5.

[LS05] Bo Lu and M Shahidehpour. Unit commitment with flexible generating units. *Power Systems, IEEE Transactions on*, 20:1022 – 1034, 06 2005. doi:10.1109/TPWRS.2004.840411.

[LS21] Scott B. Lindstrom and Brailey Sims. SURVEY: SIXTY YEARS OF DOUGLAS–RACHFORD. *Journal of the Australian Mathematical Society*, 110(3):333–370, June 2021. doi:10.1017/S1446788719000570.

[MA18] Farid Mohammadi and Hamdi Abdi. A modified crow search algorithm (MCSA) for solving economic load dispatch problem. *Applied Soft Computing*, 71:51 – 65, 2018. doi:10.1016/j.asoc.2018.06.040.

[Mid] Midcontinent Independent System Operator (MISO). Enhanced modeling of combined cycle generators. [Online; Accessed 2020-05-28]. URL: https://www.misoenergy.org/stakeholder-engagement/issue-tracking/enhanced-modeling-of-combined-cycle-generators.

[MIRS13] Behnam Mohammadi-Ivatloo, Abbas Rabiee, and Alireza Soroudi. Nonconvex Dynamic Economic Power Dispatch Problems Solution Using Hybrid Immune-Genetic Algorithm. *IEEE Systems Journal*, 7(4):777–785, December 2013. doi:10.1109/JSYST.2013.2258747.

[MiRSE12] Behnam Mohammadi-ivatloo, Abbas Rabiee, Alireza Soroudi, and Mehdi Ehsan. Imperialist competitive algorithm for solving non-convex dynamic economic power dispatch. *Energy*,

44(1):228–240, August 2012. doi:10.1016/j.energy.2012.06.034.

[Mol17] Daniel K. Molzahn. Computing the Feasible Spaces of Optimal Power Flow Problems. *IEEE Transactions on Power Systems*, 32(6):4752–4763, November 2017. doi:10.1109/TPWRS.2017.2682058.

[MS13] D. Martínez Morera and J. Estrada Sarlabous. On the distance from a point to a quadric surface. *Investigación Operacional*, 24(2):153–161, September 2013.

[NAAA13] Taher Niknam, Rasoul Azizipanah-Abarghooee, and Jamshid Aghaei. A new modified teaching-learning algorithm for reserve constrained dynamic economic dispatch. *IEEE Transactions on Power Systems*, 28(2):749–763, May 2013. doi:10.1109/TPWRS.2012.2208273.

[Nes18] Yurii Nesterov. *Lectures on Convex Optimization*. Springer Optimization and Its Applications. Springer International Publishing, 2 edition, 2018. doi:10.1007/978-3-319-91578-4.

[Neu51] John von Neumann. *Functional Operators, Volume II: The Geometry of Orthogonal Spaces: 2*. Princeton University Press, January 1951.

[Nic18] Vlad Niculae. Pardalos_kovoor. https://gist.github.com/vene/83bee4552339bd184cfaec0606be529c, 2018.

[NNAA12] Taher Niknam, Mohammad Rasoul Narimani, and Rasoul Azizipanah-Abarghooee. A new hybrid algorithm for optimal power flow considering prohibited zones and valve point effect. *Energy Conversion and Management*, 58:197 – 206, 2012. doi:10.1016/j.enconman.2012.01.017.

[OSG20] Boris Odehnal, Hellmuth Stachel, and Georg Glaeser. *The Universe of Quadrics*. Springer-Verlag, Berlin Heidelberg, 2020. doi:10.1007/978-3-662-61053-4.

[PBN19] Ricardo B.N.M. Pinheiro, Antonio R. Balbo, and Leonardo Nepomuceno. Solving network-constrained nonsmooth economic dispatch problems through a gradient-based approach.

International Journal of Electrical Power & Energy Systems, 113:264 – 280, 2019. doi:10.1016/j.ijepes.2019.05.046.

[PCCB06] C. K. Panigrahi, P. K. Chattopadhyay, R. N. Chakrabarti, and M. Basu. Simulated annealing technique for dynamic economic dispatch. *Electric Power Components and Systems*, 34(5):577–586, 2006. doi:10.1080/15325000500360843.

[PJCY20] Shanshan Pan, Jinbao Jian, Huangyue Chen, and Linfeng Yang. A full mixed-integer linear programming formulation for economic dispatch with valve-point effects, transmission loss and prohibited operating zones. *Electric Power Systems Research*, 180:106061, 2020. doi:10.1016/j.epsr.2019.106061.

[PJY17] Shanshan Pan, Jinbao Jian, and Linfeng Yang. A Mixed Integer Linear Programming Method for Dynamic Economic Dispatch with Valve Point Effect, 2017. arXiv:1702.04937.

[PJY18] Shanshan Pan, Jinbao Jian, and Linfeng Yang. A hybrid MILP and IPM approach for dynamic economic dispatch with valve-point effects. *International Journal of Electrical Power & Energy Systems*, 97:290 – 298, 2018. doi:10.1016/j.ijepes.2017.11.004.

[PK90] P. M. Pardalos and N. Kovoor. An algorithm for a singly constrained class of quadratic programs subject to upper and lower bounds. *Mathematical Programming*, 46(1-3):321–328, January 1990. doi:10.1007/BF01585748.

[PP91] M. V. F. Pereira and L. M. V. G. Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52(1):359–375, May 1991. doi:10.1007/BF01582895.

[Rat10] Mahesh Rathore. *Thermal Engineering*. McGraw Hill Education, New Delhi, April 2010.

[RNLZ18] Jose S. Rodriguez, Bethany Nicholson, Carl Laird, and Victor M. Zavala. Benchmarking ADMM in nonconvex NLPs. *Computers & Chemical Engineering*, 119:315–325, November 2018. doi:10.1016/j.compchemeng.2018.08.036.

[Saa99] Hadi Saadat. *Power System Analysis*. McGraw-Hill, New York, 1999.

[SDRH19] Dmitry Shchetinin, Tomas Tinoco De Rubira, and Gabriela Hug. Efficient Bound Tightening Techniques for Convex Relaxations of AC Optimal Power Flow. *IEEE Transactions on Power Systems*, 34(5):3848–3857, September 2019. doi:10.1109/TPWRS.2019.2905232.

[SR20] Wilfredo Sosa and Fernanda MP Raupp. An algorithm for projecting a point onto a level set of a quadratic function. *Optimization*, pages 1–19, October 2020. doi:10.1080/02331934.2020.1807545.

[SXZ+21] Chenhui Song, Jun Xiao, Guoqiang Zu, Ziyuan Hao, and Xinsong Zhang. Security region of natural gas pipeline network system: Concept, method and application. *Energy*, 217:119283, February 2021. doi:10.1016/j.energy.2020.119283.

[Sö15] Kenneth Sörensen. Metaheuristics—the metaphor exposed. *International Transactions in Operational Research*, 22(1):3–18, 2015. doi:10.1111/itor.12001.

[TP20] Andreas Themelis and Panagiotis Patrinos. Douglas–Rachford Splitting and ADMM for Nonconvex Optimization: Tight Convergence Results. *SIAM Journal on Optimization*, 30(1):149–181, January 2020. doi:10.1137/18M1163993.

[Van19] Loïc Van Hoorebeeck. Adaptive piecewise approximation. https://gitlab.com/Loicvh/apla, 2019.

[Van21] Loïc Van Hoorebeeck. Adaptive Piecewise Linear Approximation and Riemannian Subgradient Descent. https://gitlab.com/Loicvh/apla-rsg, 2021.

[VAP19a] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. MILP-Based Algorithm for the Global Solution of Dynamic Economic Dispatch with Valve-Point Effects. The International Council for Industrial and Applied Mathematics (ICIAM), 2019.

[VAP19b] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. MILP-Based Algorithm for the Global Solution of Dynamic Economic Dispatch with Valve-Point Effects. European Conference on Operational Research (EURO), 2019.

[VAP20a] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. Global solution of economic dispatch with valve point effects and transmission constraints. *Electric Power Systems Research*, 189:106786, 2020. doi:10.1016/j.epsr.2020.106786.

[VAP20b] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. Global Solution of Economic Dispatch with Valve Point Effects and Transmission Constraints. In *2020 Power Systems Computation Conference (PSCC)*, pages 1–8, 2020. Published as [VAP20a].

[VAP21] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. A matheuristic for solving non-convex economic dispatches. European Conference on Operational Research (EURO), 2021.

[VAP22a] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. Projection onto quadratic hypersurfaces, 2022. arXiv:2204.02087.

[VAP22b] Loïc Van Hoorebeeck, P.-A. Absil, and Anthony Papavasiliou. Solving non-convex economic dispatch with valve-point effects and losses with guaranteed accuracy. *International Journal of Electrical Power & Energy Systems*, 134:107143, January 2022. doi:10.1016/j.ijepes.2021.107143.

[VHa] Loïc Van Hoorebeeck. Quadproj: A package to project onto quadratic hypersurface. URL: https://pypi.org/project/quadproj/.

[VHb] Loïc Van Hoorebeeck. Quadproj: A package to project onto quadratic hypersurface. URL: https://anaconda.org/loicvh/quadproj.

[VHc] Loïc Van Hoorebeeck. Quadproj: A package to project onto quadratic hypersurface. URL: https://gitlab.com/Loicvh/quadproj.

[VHd] Loïc Van Hoorebeeck. Quadproj: A package to project onto quadratic hypersurface. URL: https://loicvh.gitlab.io/quadproj.

[VJ04] T.Aruldoss Albert Victoire and A.Ebenezer Jeyakumar. Hybrid PSO–SQP for economic dispatch with valve-point effect. *Electric Power Systems Research*, 71(1):51 – 59, 2004. doi:10.1016/j.epsr.2003.12.017.

[VPA19] Loïc Van Hoorebeeck, Anthony Papavasiliou, and P.-A. Absil. MILP-based algorithm for the global solution of dynamic economic dispatch problems with valve-point effects. In *2019 IEEE Power Energy Society General Meeting (PESGM)*, 2019. doi:10.1109/PESGM40551.2019.8973631.

[WB06] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, May 2006. doi:10.1007/s10107-004-0559-y.

[WDW+16] Z.L. Wu, J.Y. Ding, Q.H. Wu, Z.X. Jing, and X.X. Zhou. Two-phase mixed integer programming for non-convex economic dispatch problem with spinning reserve constraints. *Electric Power Systems Research*, 140:653 – 662, 2016. doi:10.1016/j.epsr.2016.05.006.

[WDW+17] Z. Wu, J. Ding, Q. H. Wu, Z. Jing, and J. Zheng. Reserve constrained dynamic economic dispatch with valve-point effect: A two-stage mixed integer linear programming approach. *CSEE Journal of Power and Energy Systems*, 3(2):203–211, 2017. doi:10.17775/CSEEJPES.2017.0025.

[WF93] K. P. Wong and C. C. Fung. Simulated annealing based economic dispatch algorithm. *IEEE Proceedings C - Generation, Transmission and Distribution*, 140(6):509–515, Nov 1993. doi:10.1049/ip-c.1993.0074.

[Woh17] Brendt Wohlberg. ADMM Penalty Parameter Selection by Residual Balancing, April 2017. arXiv:1704.06209.

[WS93] D.C. Walters and G.B. Sheble. Genetic algorithm solution of economic dispatch with valve point loading. *IEEE Transactions on Power Systems*, 8(3):1325–1332, August 1993. doi:10.1109/59.260861.

[WYZ19] Yu Wang, Wotao Yin, and Jinshan Zeng. Global Convergence of ADMM in Nonconvex Nonsmooth Optimization. *Journal of Scientific Computing*, 78(1):29–63, January 2019. doi:10.1007/s10915-018-0757-z.

[XS18] Guojiang Xiong and Dongyuan Shi. Hybrid biogeography-based optimization with brain storm optimization for non-convex dynamic economic dispatch with valve-point effects. *Energy*, 157:424–435, August 2018. doi:10.1016/j.energy.2018.05.180.

[YFP13] Lingjian Yang, Eric S. Fraga, and Lazaros G. Papageorgiou. Mathematical programming formulations for non-smooth and non-convex electricity dispatch problems. *Electric Power Systems Research*, 95:302 – 308, 2013. doi:10.1016/j.epsr.2012.09.015.

[YWY+08] Xiaohui Yuan, Liang Wang, Yanbin Yuan, Yongchuan Zhang, Bo Cao, and Bo Yang. A modified differential evolution approach for dynamic economic dispatch with valve-point effects. *Energy Conversion and Management*, 49(12):3447 – 3453, 2008. doi:10.1016/j.enconman.2008.08.016.

[Zha05] Fuzhen Zhang. *The Schur Complement and its Applications*, volume 4 of *Numerical Methods and Algorithms*. Springer, New York, 2005. doi:10.1007/b105056.

[Zhu15] J. Zhu. *Optimization of Power System Operation*. IEEE Press Series on Power and Energy Systems. Wiley, 2015.

[ZMS] R. D. Zimmerman and C. E. Murillo-Sanchez. Matpower (version 7.0). https://matpower.org.

[Zwe03] Zwe-Lee Gaing. Particle swarm optimization to solving the economic dispatch considering the generator constraints. *IEEE Transactions on Power Systems*, 18(3):1187–1195, 2003. doi:10.1109/TPWRS.2003.814889.

Index