Solving Non-Convex Economic Dispatch with Valve-Point Effects and Losses with Guaranteed Accuracy ⋆

Loïc Van Hoorebeeck[a,*], P.-A. Absil[a], Anthony Papavasiliou[b]

[a] *ICTEAM Institute, UCLouvain, Louvain-la-Neuve, Belgium.*
[b] *CORE Institute, UCLouvain Louvain-la-Neuve, Belgium.*

Abstract

The economic dispatch problem is a fundamental problem in power system operations. An extensive body of literature has focused on providing fast and robust algorithms for solving the various instances of the economic dispatch. In order to capture physical effects such as the power losses of the network or the valve-point loading effect of combined cycle gas turbines, non-convex models of the economic dispatch have been considered. However, these features of the problem render the convergence analysis more challenging, and few methods in the literature provide insights on the global optimality of the derived solution. In this work, we propose an algorithm that efficiently provides a feasible solution, along with a lower bound, to a non-smooth and non-convex instance of the economic dispatch problem. We test our method on extensively studied test cases.

Keywords: Non-convex dispatch, Riemannian descent, Non-smooth optimization

1. Introduction

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 for the 2030 European power mix and a stabilisation around 20 % for 2050 [11]. Thus, the accurate representation of the constraints and the complex cost function of such units is becoming increasingly important in system operations, see, *e.g.*, [22].

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 the feasibility of the dispatch and include limited power ranges, ramp rates, and prohibited operation zones. On the other hand,

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

*Corresponding author
Email address: loic.vanhoorebeeck@uclouvain.be (Loïc Van Hoorebeeck)

balance constraints require that the supply meets the load and guarantee that enough reserves are available.

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 modelled as a smooth convex quadratic function. However, such a function fails to accurately model large CCGT units due to the valve-point effect (VPE) [8]. The valve-point effect refers to the increase of throttling losses when operating a turbine *off* a valve-point, *i.e.*, 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 non-smooth and non-convex function, see Eq. (1), is commonly used for modelling this effect. This non-convexity allows for the existence of a plethora of local minima and the non-smoothness of the objective function prevents the use of conventional derivative-based techniques.

In order to solve this problem, the literature mostly follows two approaches: i) randomized heuristics which aim at efficiently spanning the search space in order 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 [24, 42], other evolutionary algorithms [33, 27, 26, 4, 23], genetic algorithms [25], and simulated annealing algorithms [39, 30]. Examples of ii) include [29, 41, 28], where the authors use approximations of the objective without providing lower bounds, [31, 5] using gradient-based algorithms with logarithmic barriers, and previous work by the authors [36, 35] where power losses are neglected.

Losses are another source of non-convexity 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 often 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 [19, 2], and correspondingly losses have increased, thereby motivating a need to represent such losses more accurately. Network constraints require, in principle, the consideration of the AC power flow equations, which are highly non-linear and non-convex. A more tractable alternative which captures an essential aspect of network operations is to focus on losses. In this context, Kron [18] introduced a quadratic model for the power losses which has been popularized by Kirchmayer [17].

As these above-mentioned methods 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 if 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.

This motivates the present study, which extends previous contributions by the authors [36, 35] to non-convex feasible sets due to power losses, and develops a method that 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 problems, defined with a piece-

wise approximation of the objective. Such approximations have been studied originally in [43, 16, 40], then in [29, 1, 28].

Since lower and upper bounds are computed by our proposed method, it is possible to detect whether a prescribed accuracy is attained, and henceforth 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 fact that the algorithms that we propose in the present work are parallelizable. This favorable situation contrasts with most existing algorithms where a sufficiently good lower bound is unavailable, making it impossible to know if the best cost found so far is within a prescribed accuracy of the globally optimal cost.

The contributions of this work are the following. We first describe how to obtain an accurate approximation of the convex hull of the feasible set of an economic dispatch problem with quadratic power losses. Secondly, we extend the local method of [5] to accommodate the multi-period economic dispatch problem. This local method is a Riemannian subgradient method which takes advantage of the inherent characterization of the feasible set as a Riemannian manifold. Lastly, we show how to leverage this convex-hull approximation of the feasible set—or convex relaxation—in our previous work [35] in order to obtain a good lower bound to the global solution, and how to combine this approach with the extended local method in order to provide a high-quality objective.

The paper is organized as follows. Section 2 describes the non-smooth and non-convex optimization problem of interest, namely the economic dispatch problem with valve-point effects and quadratic losses. Different surrogate optimization problems, that will be used throughout this paper in order to tackle the main problem, are also introduced, and a study of the feasible set as well as the proposed relaxation is performed. The methods employed in our proposed approach are detailed in section 3: we briefly describe the adaptive piecewise linearization method from [36], as well as the extended Riemannian subgradient method. We also provide ingredients of differential geometry that are required for the subgradient method. Section 4 gathers the results of the methods for several test cases and compares the objectives with state-of-the-art methods. The lower bounds derived from our method allow us to assess how close the solution of our proposed method and other methods in the literature are to the optimal solution. Finally, conclusions are drawn in section 5.

2. Problem formulation

This section is organized as follows. After a brief description of the notation used in our paper, we introduce the main problem that is considered in this work in § 2.2. The full method is then outlined in § 2.3. This method depends on several auxiliary optimization problems which are introduced in § 2.4: 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 finally the optimization problems that are used for obtaining a relaxation of the feasible set. Lastly, the topology of the feasible set is studied in § 2.5.

2.1. Notation

All vectors in the paper are indicated in bold, *e.g.*, $\boldsymbol{x}$. An implicit partition is used for dealing with double indices: if $\boldsymbol{x}$ depends on both indices $g = 1, \ldots, |G|$ and $t = 1, \ldots, |T|$, then $\boldsymbol{x}$ is partitioned as follows,

$$\boldsymbol{x} = \begin{pmatrix} x_{11} & \ldots & x_{|G|1} & x_{12} & \ldots & x_{|G||T|} \end{pmatrix}^\intercal = \begin{pmatrix} \boldsymbol{x}_1{}^\intercal & \boldsymbol{x}_2{}^\intercal & \ldots & \boldsymbol{x}_{|T|}{}^\intercal \end{pmatrix}^\intercal.$$

The index g stands for the generator unit g listed in the set G and the index t stands for the time step t listed in the set T.

2.2. 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, this fails to model the inherent non-convex characteristic of the problem when the VPE is taken into account. Following the literature [7, 13, 37], we model the cost function as the sum of a smooth quadratic part, f_g^{Q}, and a non-smooth rectified sine aimed at capturing the VPE, f_g^{VPE}:

$$f_g(p_{gt}) = \underbrace{A_g p_{gt}^2 + B_g p_{gt} + C_g}_{:= f_g^{\mathrm{Q}}(p_{gt})} + \underbrace{\left| D_g \, \sin E_g (p_{gt} - P_g^-) \right|}_{:= f_g^{\mathrm{VPE}}(p_{gt})} . \tag{1}$$

Here, P_g^- is the minimum power production of generator g and A_g, B_g, C_g, D_g, E_g are parameters.

The full objective reads

$$f(\boldsymbol{p}) = \sum_{t \in T} \sum_{g \in G} f_g(p_{gt}) \,, \tag{2}$$

and a single term, $f_g(p_{gt})$, is depicted in Fig. 1.

The constraints considered in this work are the following:

- Power range limits

$$P_g^- \leqslant p_{gt} \leqslant P_g^+ \qquad \forall g = 1, \ldots, |G| \,, t = 1, \ldots, |T| \,, \tag{3}$$

  where P_g^- and P_g^+ are the minimum and maximum power output of unit g.

- Ramp rate restrictions

$$R_g^- \leqslant p_{gt} - p_{g(t-1)} \leqslant R_g^+ \qquad \forall g = 1, \ldots, |G| \,, t = 2, \ldots, |T| \,, \tag{4}$$

  where R_g^- and R_g^+ are the ramp-down and ramp-up rates of unit g, respectively.

- Power balance

$$\sum_{g \in G} p_{gt} = P_t^{\mathrm{D}} + \underbrace{\boldsymbol{p}_t{}^\intercal \boldsymbol{B} \boldsymbol{p}_t + \boldsymbol{B}_0 \boldsymbol{p}_t + B_{00}}_{:= p_t^{\mathrm{loss}}} \quad \forall \, t = 1 \ldots |T| \,. \tag{5}$$

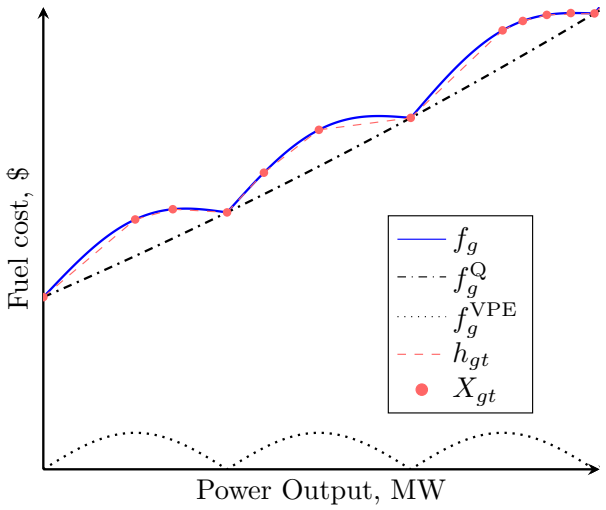


Figure 1: Illustration of a single term of the main and surrogate objectives.

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 [32], 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 [32], *e.g.*, the matrix is indefinite for the 10-unit test case in § 4.2. These coefficients are discussed in § 2.5.

- Spinning reserve constraints

  The reserve requirements are modeled as in [26], $\forall t \in T$:

$$\left(\Delta_t^{(1)} = \sum_{g \in G} P_g^+ - (P_t^{\mathrm{D}} + p_t^{\mathrm{loss}} + P_t^{\mathrm{S}})\right) \geqslant 0 \tag{6}$$

$$\left(\Delta_t^{(2)} = \sum_{g \in G} \min\left(P_g^+ - p_{gt}, R_g^+\right) - P_t^{\mathrm{S}}\right) \geqslant 0 \tag{7}$$

$$\left(\Delta_t^{(3)} = \sum_{g \in G} \min\left(P_g^+ - p_{gt}, \frac{R_g^+}{6}\right) - \frac{P_t^{\mathrm{S}}}{6}\right) \geqslant 0 \tag{8}$$

  for a given reserve requirement P_t^{S}.

  Eqs. (6) and (7) model the requirement for the spinning reserve that can respond within one hour. Eq. (8) models the requirement for the spinning reserve that can respond within 10 minutes. Note that, if the losses are neglected, *i.e.*, $p_t^{\mathrm{loss}} = 0$, then Eq. (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 § 4.2 to show that a given problem is infeasible.

Taking all these constraints into account, the optimization problem at hand reads

$$\begin{aligned} \min_{\boldsymbol{p}} \quad & f(\boldsymbol{p}) = \sum_{t \in T} \sum_{g \in G} f_g(p_{gt}) \,, \\ \text{s.t.} \quad & (3) - (8) \,. \end{aligned} \tag{P}$$

This is a non-smooth and non-convex continuous optimization problem. The feasible set is the intersection of a polytope—Eqs. (3), (4) and (6) to (8)—and a quadratic hypersurface—Eq. (5)—which is further described in § 2.5.

2.3. Outline of the Method

Before introducing the other optimization problems that will be used in the remainder of the paper, we first outline the full method. This method, denoted as APLA-RSG, is depicted in Fig. 2 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 an adaptive piecewise-linear approximation (APLA) of the objective, 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: § 2.4.1, § 2.4.2, and § 2.4.3, respectively. The last step is a Riemannian subgradient descent scheme (RSG), and depends on a quadratic subproblem, (Sub), defined in § 3.1.

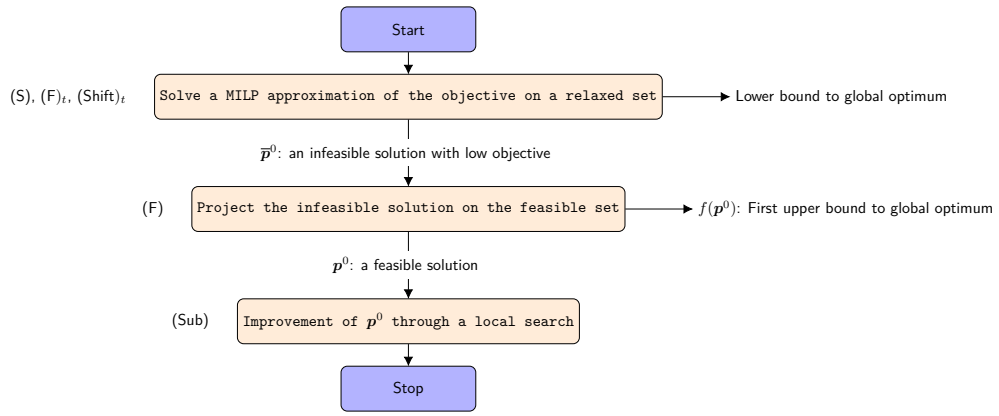


Figure 2: Block diagram of the method APLA-RSG for solving (P).

2.4. Auxiliary Optimization Problems

2.4.1. Surrogate Problem

One way of coping with the non-linearities 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 [36].

The surrogate problem, (S), aims at i) finding a good initial point for a local search and ii) providing a lower bound to the solution of (P). This is achieved through an under-approximation of the objective (as in [36]) and a relaxation of the quadratic constraint Eq. (5). This relaxation is explained in § 2.4.3.

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:

$$h_{gt}(p_{gt}) := \Pi[f_g, \boldsymbol{X}_{gt}](p_{gt})\,. \tag{9}$$

In this expression, $\Pi[f, \boldsymbol{X}]$ 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 Eq. (2), in order to form the total surrogate objective h,

$$h(\boldsymbol{p}) = \sum_{t\in T}\sum_{g\in G} h_{gt}(p_{gt})\,. \tag{10}$$

This *surrogate objective* is illustrated in Fig. 1.

Note that, even if the fuel cost f_g does not depend on the time step t, the approximation h_{gt} is dependent on t, because the set of knots which define the approximation also depends on this time index.

Equipped with this surrogate objective, we define the *surrogate problem* as

$$\begin{aligned} \min_{\boldsymbol{p}} \quad & h(\boldsymbol{p}) = \sum_{t\in T}\sum_{g\in G} h_{gt}(p_{gt})\,, \\ \text{s.t.} \quad & (3)-(4), (5)_R, (6)-(8)\,, \end{aligned} \tag{S}$$

where $(5)_R$ stands for the relaxation of constraint (5), and is defined in § 2.4.3. Two relaxations are considered, a linear relaxation and a (convex) quadratic one, depending on whether $\boldsymbol{B}$ 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 § 2.5.

2.4.2. Feasibility Problems

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

We define the feasibility objective f^{feas} as

$$f^{\text{feas}}(\boldsymbol{p}\,; \overline{\boldsymbol{p}}^0, \lambda_{\text{N}}, \lambda_{\text{Q}}) = \lambda_{\text{N}} \left|\left|\boldsymbol{p} - \overline{\boldsymbol{p}}^0\right|\right|_2^2 + \lambda_{\text{Q}} \sum_{t\in T}\sum_{g\in G} f_g^{\text{Q}}(p_{gt})\,, \tag{11}$$

for given parameters λ_{N}, $\lambda_{\text{Q}} \in \mathbb{R}_{\geqslant 0}$ and f_g^{Q} defined as in (1). We discuss these parameters hereafter.

The feasibility problem reads,

$$\begin{aligned} \min_{\boldsymbol{p}} \quad & (11) \\ \text{s.t.} \quad & (3)-(8)\,. \end{aligned} \tag{F}$$

This problem depends on the parameters λ_{N} and λ_{Q}. When $\lambda_{\mathrm{Q}} = 0$, (F) becomes a projection on the feasible set. 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. Note that (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 therefore computational resources.

The fixed-time feasibility problem is also considered. It is similar to (F), except that the problem is decoupled with respect to a given time step t. It reads as follows:

$$\begin{aligned} \min_{\boldsymbol{p}_t} \quad & \lambda_{\mathrm{Q}} \sum_{g \in G} f_g^{\mathrm{Q}}(p_{gt}) \\ \text{s.t.} \quad & (3)_t, (5)_t - (8)_t \,, \end{aligned} \tag{F}$$

where $(\cdot)_t$ indicates that the constraint must only hold for the given time step t. Note that constraint (4) is dropped, as it depends on two consecutive time steps.

2.4.3. Relaxation Problem

The goal of the relaxation problem is to compute a convex relaxation of the constraint (5), written $(5)_R$. 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. Note that this relaxation problem is decoupled with respect to the time index t. Indeed, if $(5)_{t,R}$ is the relaxation of $(5)_t$, then taking the Cartesian product of every time step yields a relaxation of equation (5). Hence, the discussion is made here for a given $t \in T$, and the full relaxation is obtained as the Cartesian product over t:

$$\bigtimes_{t \in T} (5)_{t,\mathrm{R}} \tag{5}_{\mathrm{R}}$$

Case I: $\boldsymbol{B}$ is positive definite. In this case, the feasible set generated by constraint $(5)_t$ is the surface of an ellipsoid, see § 2.5. However, the power ranges of $(3)_t$ restrict the feasible set to a box which is, in practice, very small with respect to the ellipsoid, because the losses are small. This explains why a linear approximation is often used, as in [29, 28]. In order to obtain a relaxation, the set induced by $(5)_{t,\mathrm{R}}$ should include the set induced by $(5)_t$. This condition is fulfilled if $(5)_{t,\mathrm{R}}$ is the intersection between the interior of the ellipsoid, the interior of the power ranges, and the half space induced by any secant plane that has no intersection with the feasible set of equation (P)—in Figure 3, 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 $n := |G|$ intersections between the ellipsoid and the box, the optimal secant plane (π_0 in Figure 3) is the one defined by the n intersections. Unfortunately, getting the n intersections in order to find such an ideal plane (π_0) is challenging. The simple enumeration of the vertices of the hyper-cube becomes intractable for a small number of generators $|G|$.

Note that the 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 behaviour has been studied in [21] for non-convex network constraints.

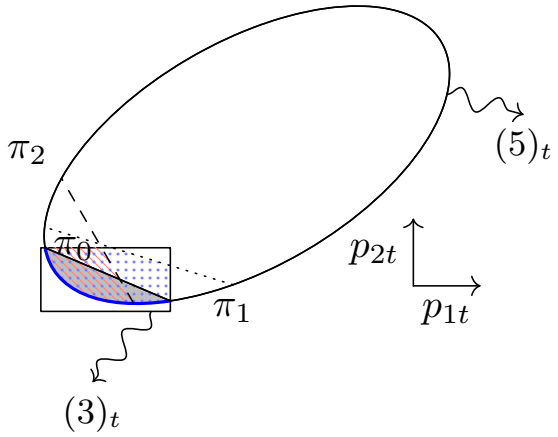


Figure 3: Illustration of the relaxation induced by the interior of the ellipsoid, the power ranges, and several secant planes, for a very simple case with only two generators at a given time step. The area induced by π_0 (gray fill) and π_1 (blue dots) are valid relaxations while the one induced by π_2 (red hashed lines) is not a valid relaxation as 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).

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

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

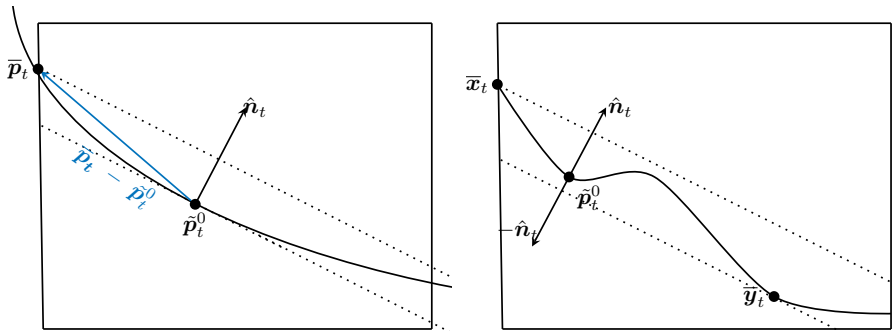
where $\mathcal{X}_t$ is the feasible set of $(\mathrm{F})_t$. This procedure is illustrated in Figure 4a which is a magnification of Figure 3 around the available power ranges. The explicit procedure is presented in Algorithm 1.

For a given time t, the relaxed balance constraint, $(5)_{t,R}$, reads

$$\begin{aligned} \sum_{g \in G} p_{gt} &\geqslant P_t^{\mathrm{D}} + \boldsymbol{p}_t{}^{\intercal} \boldsymbol{B} \boldsymbol{p}_t + \boldsymbol{B}_0 \boldsymbol{p}_t + B_{00}, \\ 0 &\geqslant \boldsymbol{p}_t \cdot \boldsymbol{n}_t - (\tilde{\boldsymbol{p}}_0^t + \hat{\boldsymbol{n}}_t S_t^*) \cdot \boldsymbol{n}_t. \end{aligned} \qquad (5)_{t,R}$$

Algorithm 1 Procedure to obtain relaxation at given time step t

Require: $\boldsymbol{B}$ positive definite
$\tilde{\boldsymbol{p}}_t^0 \leftarrow$ solution of $(\mathrm{F})_t$
$\boldsymbol{n}_t \leftarrow \boldsymbol{B}\tilde{\boldsymbol{p}}_t^0 + \boldsymbol{b}_t$
$\hat{\boldsymbol{n}}_t \leftarrow \frac{\boldsymbol{n}_t}{||\boldsymbol{n}_t||_2}$
$S_t^* \leftarrow \max_{\bar{\boldsymbol{p}}_t \in \mathcal{X}_t} \hat{\boldsymbol{n}}_t \cdot (\bar{\boldsymbol{p}}_t - \tilde{\boldsymbol{p}}_t^0)$
$(5)_{t,\mathrm{R}} \leftarrow \left(0 \geqslant \boldsymbol{p}_t \cdot \boldsymbol{n}_t - (\tilde{\boldsymbol{p}}_0^t + \hat{\boldsymbol{n}}_t S_t^*) \cdot \boldsymbol{n}_t\right) \bigcup \left(\sum_{g \in G} p_{gt} \geqslant P_t^{\mathrm{D}} + p_t^{\mathrm{loss}}\right)$
return $(5)_{t,\mathrm{R}}$



(a) Ellipsoid case: 1-plane relaxation.

(b) Hyperboloid case: 2-plane relaxation.

Figure 4: Procedure to obtain the relaxation.

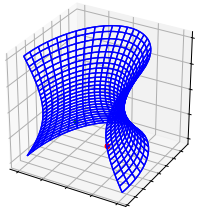
Figure 5: Illustration of the relative size of the power ranges with respect to the quadric (ellipsoid) for a 3-unit problem at a given time step t. The admissible power range, $(3)_t$, is the interior of the red cube and the power balance, $(5)_t$, the surface of the blue ellipsoid. The right figure is a magnification around the admissible power ranges. The red dots, in the right figure, are the vertices of the box.

This convex relaxation is motivated by the fact that the size of the quadric is much larger than the admissible ranges of the unit, and the linear relaxation is almost on the quadratic surface. This phenomenon is illustrated in Figure 5 for a 3-unit problem at a given time step t.

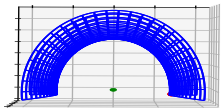
Case II: $\boldsymbol{B}$ is indefinite. In this case, there are at least two eigenvalues of $\boldsymbol{B}$ of opposite sign, and therefore the set defined by the power balance $(5)_t$ is no longer the boundary of a convex set. Figure 6 illustrates an example of this case. Figures 6b and 6c show that a single plane will not be enough to construct the relaxation: Figure 6b prompts the use of an interior relaxation plane, in a similar way as case I, however Figure 6c demonstrates that en exterior plane should also be used. To tackle this issue, we solve $(\text{Shift})_t$ for both directions $\hat{\boldsymbol{n}}_t$ and $-\hat{\boldsymbol{n}}_t$. The whole procedure is explicitly given in Algorithm 2 and depicted in Figure 4b. Remark that in this case, the relaxation is linear.

2.4.4. Comparison of the Optimization Problems

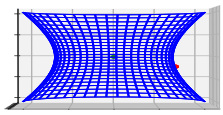
The characterization of each optimization problem is presented in Table 1. The last two problems are decoupled with respect to the time step which reduces significantly the



(a) Full hyperboloid.



(b) OXZ view.



(c) OXY view.

Figure 6: Illustration of the relative size of the power ranges with respect to the quadric for a 3-unit problem at a given time step t. In this example Kron's matrix is *not* positive definite: two eigenvalues are positive and the last one is negative. The quadric is a onesheet hyperboloid. The admissible power range, $(3)_t$, is the interior of the red cube, too small to be distinguishable, and the power balance, $(5)_t$, the surface of the blue hyperboloid. Figures 6b and 6c show different views. The green point is the center of the quadric.

Algorithm 2 Procedure to obtain relaxation at given time step t for a non-convex quadric.

$\tilde{\boldsymbol{p}}_t^0 \leftarrow$ solution of $(\mathrm{F})_t$
$\boldsymbol{n}_t \leftarrow \boldsymbol{B}\tilde{\boldsymbol{p}}_t^0 + \boldsymbol{b}_t$
$\hat{\boldsymbol{n}}_t \leftarrow \frac{\boldsymbol{n}_t}{||\boldsymbol{n}_t||_2}$
$S_t^{*,\mathrm{int}} \leftarrow \max_{\bar{\boldsymbol{x}}_t \in \mathcal{X}_t} \hat{\boldsymbol{n}}_t \cdot (\bar{\boldsymbol{x}}_t - \tilde{\boldsymbol{p}}_t^0)$
$S_t^{*,\mathrm{ext}} \leftarrow \max_{\bar{\boldsymbol{y}}_t \in \mathcal{X}_t} -\hat{\boldsymbol{n}}_t \cdot (\bar{\boldsymbol{y}}_t - \tilde{\boldsymbol{p}}_t^0)$
$(5)_{t,\mathrm{R}} \leftarrow \Big(0 \geqslant \boldsymbol{p}_t \cdot \boldsymbol{n}_t - (\tilde{\boldsymbol{p}}_t^0 + \hat{\boldsymbol{n}}_t S_t^{*,\mathrm{int}}) \cdot \boldsymbol{n}_t\Big) \bigcup \Big(0 \leqslant \boldsymbol{p}_t \cdot \boldsymbol{n}_t - (\tilde{\boldsymbol{p}}_t^0 + \hat{\boldsymbol{n}}_t S_t^{*,\mathrm{ext}}) \cdot \boldsymbol{n}_t\Big)$
return $(5)_{t,\mathrm{R}}$

size of the problem. They are considered as *easy*, relatively to the first three problems and, in the test cases studied in the present work, they can be solved to optimality in less than a second. Among the three larger problems, (P) is unquestionably the most difficult. Problem (F) is arguably easier than (S): the reason is that any 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 1: Comparison of the optimization problems.

	(P)	(S)	(F)	$(F)_t$	$(Shift)_t$
Classification	NLP	MIQP	QCQP	QCQP	QCLP
Convexity	Non-convex	Non-convex	Non-convex	Non-convex	Non-convex
Objective	Non-convex, non-smooth	Piecewise-linear	Quadratic	Quadratic	Linear
Feasible set	Non-convex	Convex[1]	Non-convex	Non-convex	Non-convex
Problem size	$\|T\|\|G\|$	$\|T\|\|G\|$	$\|T\|\|G\|$	$\|G\|$	$\|G\|$

2.5. Topology of the feasible set

Let us now study the feasible set defined by Eq. (5). In particular, we define the quadratic surface, or quadric, and express Eq. (5) as a Cartesian product of quadrics. Finally, we characterize this quadric as a quadric with middle point. This middle point will be used in § 3.1 to compute the retraction mapping.

Characterization of the hypersurfaces [3]. Let V be a vector space on the field $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$. A relation

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

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

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

$$\mathcal{Q}(\Psi) = \{\boldsymbol{x} \,|\Psi(\boldsymbol{x}) = 0\}$$

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

For a given time step t, Eq. (5) can be written as a quadric by choosing the relation

$$\mathbb{R}^n \to \mathbb{R} : \boldsymbol{p}_t \mapsto \Psi_t(\boldsymbol{p}_t) = \boldsymbol{p}_t{}^\intercal \boldsymbol{B} \boldsymbol{p}_t + 2\boldsymbol{b}^\intercal \boldsymbol{p}_t + c_t \tag{12}$$

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

$$\mathcal{Q}^{\mathrm{tot}} := \mathcal{Q}(\Psi_1) \times \mathcal{Q}(\Psi_2) \times \ldots \times \mathcal{Q}(\Psi_{|T|}) \tag{13}$$

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 [3], the

[1] The initial feasible set is convex, nevertheless the modelization of the piecewise-linear objective is made through integer variables which makes the feasible set inherently non-convex, see [14] for more details about the modelization of non-convex functions as piecewise-linear functions.

quadric hypersurface is said to be of type 2 (Mittelpunktsquadrik or quadric with middle point). Indeed, let us compute the rank of

$$\bar{\boldsymbol{B}}_t = \begin{pmatrix} \boldsymbol{B} & \boldsymbol{b} \\ \boldsymbol{b}^\intercal & c_t \end{pmatrix}. \tag{14}$$

Since $\boldsymbol{B}$ is invertible, the *Guttman rank additivity formula* yields [45]

$$\operatorname{rank} \bar{\boldsymbol{B}}_t = \operatorname{rank} \boldsymbol{B} + \operatorname{rank}(c_t - \boldsymbol{b}^\intercal \boldsymbol{B}^{-1} \boldsymbol{b}). \tag{15}$$

In general, $c_t \neq \boldsymbol{b}^\intercal \boldsymbol{B}^{-1} \boldsymbol{b}$. Thus, it follows that $\operatorname{rank} \bar{\boldsymbol{B}}_t > r = \operatorname{rank} \boldsymbol{B}$ and henceforth $\mathcal{Q}(\Psi_t)$ is a type-2 quadric. When all the eigenvalues of the quadratic form are positive, the non-degenerate type-2 quadric is an ellipsoid, illustrated in Fig. 5, otherwise it is an elliptic hyperboloid, illustrated in Fig. 6.[2] A feature of the type-2 quadric is the existence of a center, $\boldsymbol{d}$, computed as

$$\boldsymbol{d} = -\boldsymbol{B}^{-1} \boldsymbol{b}. \tag{16}$$

This center will be used in § 3.1 to compute the relaxation on the manifold defined by the quadric.

3. Methods

In this section, we explain how to combine all the elements developed in Section 2 to solve (P). First, we describe how to derive a lower bound of the problem. We then show how to obtain an upper bound, *i.e.*, a feasible solution, and improve it using a Riemannian gradient descent. Finally, both steps are combined in a single algorithm.

3.1. Deriving a Lower Bound

In a similar fashion as [36, 35], the lower bound is obtained through an underapproximation of the objective via piecewise-linearization. However, this is not sufficient here as the feasible set is the subset of a quadric. Hence, this non-convex 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 3. Notice that the point is not globally feasible and hence $f(\tilde{\boldsymbol{p}}^0)$ is *not* an upper bound to the global solution. This point is simply a starting point for Algorithms 1 and 2 depending on whether $\boldsymbol{B}$ is positive definite or not.

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

Algorithm 3 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) described in [36]. However, this method suffers from 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 we rather use the heuristic based on APLA that is described in [35]. In order to simplify the discussion, the description that we provide in the present paper is based on the APLA method.

The APLA method can be summarized as follows. Firstly, a set of knots which 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 a MIP solver. If we neglect the fact that the feasible set is relaxed, 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 [36] 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. 7. The surrogate objective and the knots are depicted in Fig. 1.

Note that this method cannot be directly applied to our problem, because of the non-convex constraint Eq. (5). This explains the need of the relaxation developed in § 2.4.3.

3.2. Deriving an Upper Bound: Riemannian Subgradient Scheme

A simple and direct method for obtaining a feasible point, *i.e.*, a first upper bound to the global solution, is to project the candidate obtained at the end of the procedure depicted in Fig. 7 on the feasible set of (P). However, the projection is, in general, not a global optimum nor even a local optimum, and it is worthwhile to attempt to improve the obtained feasible solution through a local search. In [29], 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 [5] in order to account for reserves and multiple time steps.

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

$$\boldsymbol{p}^{k+1} = \boldsymbol{p}^k + \alpha^k \boldsymbol{v}^k \tag{17}$$

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 in order to fully determine the scheme. Here, it is also required to redefine the "+" operation in order to fully define the method; since the feasible set is not a vector space, it is not true in general that Eq. (17) yields a feasible point, even for small α^k. A simple idea would

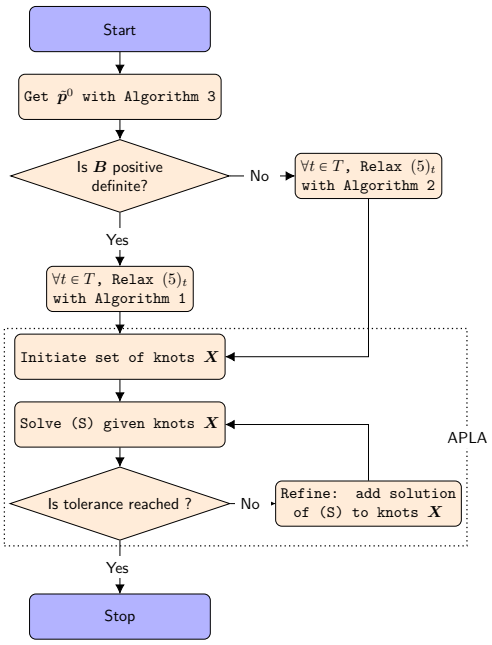


Figure 7: Flowchart of the method for obtaining a lower bound along with a first (infeasible) candidate with low objective.

be to project the resulting point on 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 classification of (F) in Table 1), and a usual line-search scheme requires at least 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. 8—and the descent direction are described. Finally, the step size rule and some implementation details are given.

3.2.1. Riemannian geometry

In a similar way as [5], we define the quadric manifold. Then, we use it to define the extended quadric manifold.

Proposition 3.1 (Quadric manifold)**.** *Let* $\Psi : \mathbb{R}^n \to \mathbb{R} : \boldsymbol{p}_t \mapsto \Psi(\boldsymbol{p}_t) = \boldsymbol{p}_t{}^\intercal \boldsymbol{B} \boldsymbol{p}_t + 2\boldsymbol{b}^\intercal \boldsymbol{p}_t + c_t$*, be a quadratic function. If* $\boldsymbol{B}$ *is nonsingular and* $c_t \neq \boldsymbol{b}^\intercal \boldsymbol{B}^{-1} \boldsymbol{b}$*, then the quadric* $\mathcal{Q}(\Psi)$ *is an* $n-1$ *dimensional smooth manifold of* $\mathbb{R}^n$*.*

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

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

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

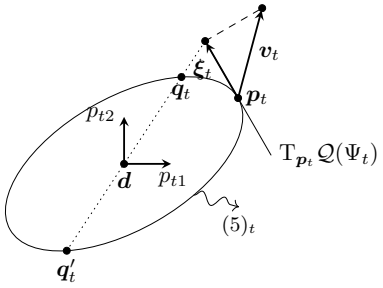


Figure 8: Illustration of the retraction, $\mathrm{R}_t(\boldsymbol{p}_t, \boldsymbol{\xi}_t)$.

Remark that the assumptions of Proposition 3.1 are equivalent to the needed assumptions for the quadric to be of *type 2* in § 2.5.

Definition 3.1 (Extended quadric manifold)**.** The Cartesian product of the quadrics defined for each time step t as in Eq. (12) is called *extended quadric manifold* and computed as

$$\mathcal{Q}^{\text{tot}} := \mathcal{Q}(\Psi_1) \times \mathcal{Q}(\Psi_2) \times \ldots \times \mathcal{Q}(\Psi_{|T|}).$$

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 $|T|\,(|G|-1)$.

A first important object to be described when dealing with manifolds is the tangent space. 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, any 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, and the resulting vector can be mapped back to the manifold via the *exponential map*.

The tangent space of the quadric manifold $\mathcal{Q}(\Psi_t)$ at a given point $\boldsymbol{p}_t$ is defined in general as

Definition 3.2. Let $\mathcal{Q}(\Psi_t)$ be a smooth real manifold, the *tangent space* reads

$$\mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}(\Psi_t) = \left\{ \boldsymbol{\xi} \in \mathbb{R}^{|G|} \mid \exists c : \mathbb{R} \mapsto \mathcal{Q}(\Psi_t) \text{ with } c(0) = 0,\, c'(0) = \boldsymbol{\xi} \right\}.$$

Using the specific structure of the quadric manifolds, this tangent space is computed as [5]

$$\mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}(\Psi_t) = \left\{ \boldsymbol{\xi} \in \mathbb{R}^{|G|} \mid \boldsymbol{\xi}^{\intercal}(2\boldsymbol{B}\boldsymbol{p}_t + \boldsymbol{b}_t) = 0 \right\}, \tag{18}$$

and $\dim\left(\mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}(\Psi_t)\right) = |G| - 1$. This tangent plane is illustrated in Fig. 8 for a positive definite matrix $\boldsymbol{B}$.

Definition 3.3 (Tangent bundle)**.** The *tangent bundle* $\mathrm{T}\mathcal{Q}(\Psi_t)$ of a manifold $\mathcal{Q}(\Psi_t)$ is defined as the disjoint union of every tangent space at every point of the manifold,

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

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 $\mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}(\Psi_t)$,

$$\langle\cdot,\cdot\rangle_{\boldsymbol{p}_t} : \mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}(\Psi_t) \times \mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}(\Psi_t) \to \mathbb{R} : (\boldsymbol{\xi}, \boldsymbol{\zeta}) \mapsto \langle \boldsymbol{\xi}, \boldsymbol{\zeta}\rangle_{\boldsymbol{p}_t} = \boldsymbol{\xi}^\intercal \boldsymbol{\zeta}. \tag{19}$$

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

$$\langle\cdot,\cdot\rangle_{\boldsymbol{p}} : \mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}} \times \mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}} \to \mathbb{R} : (\boldsymbol{\xi}, \boldsymbol{\zeta}) \mapsto \langle \boldsymbol{\xi}, \boldsymbol{\zeta}\rangle_{\boldsymbol{p}} = \boldsymbol{\xi}^\intercal \boldsymbol{\zeta} = \sum_{t\in T} \langle \boldsymbol{\xi}_t, \boldsymbol{\zeta}_t\rangle_{\boldsymbol{p}_t}. \tag{20}$$

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

$$\mathrm{N}_{\boldsymbol{p}_t}\mathcal{Q}(\Psi_t) = \mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}(\Psi_t)^\perp \tag{21}$$

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

It follows from Eq. (18) that

$$\mathrm{N}_{\boldsymbol{p}_t}\mathcal{Q}(\Psi_t) = \{\tau(2\boldsymbol{B}\boldsymbol{p}_t + \boldsymbol{b}_t) \mid \tau \in \mathbb{R}\}\,, \tag{22}$$

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

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

Proposition 3.2. $\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{tot} = \mathrm{T}_{\boldsymbol{p}_1}\mathcal{Q}(\Psi_1) \times \mathrm{T}_{\boldsymbol{p}_2}\mathcal{Q}(\Psi_2) \times \ldots \times \mathrm{T}_{\boldsymbol{p}_{|T|}}\mathcal{Q}(\Psi_{|T|})$

Proof. See [12, Chap. 1.2]. □

Proposition 3.3. $\mathrm{N}_{\boldsymbol{p}}\mathcal{Q}^{tot} = \mathrm{N}_{\boldsymbol{p}_1}\mathcal{Q}(\Psi_1) \times \mathrm{N}_{\boldsymbol{p}_2}\mathcal{Q}(\Psi_2) \times \ldots \mathrm{N}_{\boldsymbol{p}_{|T|}}\mathcal{Q}(\Psi_{|T|})$

Proof. We first show that $\mathrm{N}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}} = (\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}})^\perp \supseteq \mathrm{N}_{\boldsymbol{p}_1}\mathcal{Q}(\Psi_1) \times \mathrm{N}_{\boldsymbol{p}_2}\mathcal{Q}(\Psi_2) \times \ldots \times \mathrm{N}_{\boldsymbol{p}_{|T|}}\mathcal{Q}(\Psi_{|T|})$ and then we conclude with an argument on the dimensions.

i) Let $\boldsymbol{p} \in \mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}$ and $\boldsymbol{p}' \in \mathrm{N}_{\boldsymbol{p}_1}\mathcal{Q}(\Psi_1) \times \mathrm{N}_{\boldsymbol{p}_2}\mathcal{Q}(\Psi_2) \times \ldots \times \mathrm{N}_{\boldsymbol{p}_{|T|}}\mathcal{Q}(\Psi_{|T|})$, 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 3.2 and Definition 3.4 that $\boldsymbol{p}^\intercal \boldsymbol{p}' = 0$ and therefore that $\boldsymbol{p}' \in (\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}})^\perp$.

ii) Since $\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}$ is a linear subspace of $\mathbb{R}^{|G||T|}$, we have $|G|\,|T| - \dim\left(\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}\right) = \dim\left(\left(\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}\right)^{\perp}\right) = |G|\,|T| - |T|\,(|G|-1) = |T|$. This concludes the proof as

$$\dim\left(\mathrm{N}_{\boldsymbol{p}_1}\mathcal{Q}(\Psi_1) \times \mathrm{N}_{\boldsymbol{p}_2}\mathcal{Q}(\Psi_2) \times \ldots \times \mathrm{N}_{\boldsymbol{p}_{|T|}}\mathcal{Q}(\Psi_{|T|})\right) = |T|\ .$$

□

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 $\mathcal{Q}(\Psi_t)$, we can easily extend the projection operator from [5] by working componentwisely.

The projection $P_{\boldsymbol{p}}\,(v)$ of a vector $\boldsymbol{v} \in \mathbb{R}^{|G||T|}$ partitioned as $(\boldsymbol{v}_1{}^{\intercal}, \boldsymbol{v}_2{}^{\intercal}, \ldots, \boldsymbol{v}_{|T|}{}^{\intercal})^{\intercal}$ onto $\mathrm{T}_{\boldsymbol{p}}\mathcal{Q}^{\mathrm{tot}}$ can be constructed by removing the normal component of $\boldsymbol{v}$:

$$P_{\boldsymbol{p}}\,(v) = (\hat{\boldsymbol{v}}_1^{\intercal}, \ldots, \hat{\boldsymbol{v}}_{|T|}^{\intercal})^{\intercal}, \tag{23}$$

where $\hat{\boldsymbol{v}}_t = \boldsymbol{v}_t - \tau_t(2\boldsymbol{B}\boldsymbol{p}_t + \boldsymbol{b}_t)$ and τ_t is chosen to ensure that $P_{\boldsymbol{p}_t}(\boldsymbol{v}_t)$ belong to $\mathrm{T}_{\boldsymbol{p}_t}\mathcal{Q}(\Psi_t)$, *i.e.*,

$$\tau_t = \frac{\boldsymbol{v}_t{}^{\intercal}(2\boldsymbol{B}\boldsymbol{p}_t + \boldsymbol{b}_t)}{||(2\boldsymbol{B}\boldsymbol{p}_t + \boldsymbol{b}_t)||^2}. \tag{24}$$

3.2.2. Retraction

Definition 3.5 (Retraction)**.** A retraction R is a smooth mapping from the tangent bundle of a manifold to the manifold itself,

$$\begin{aligned} &\mathrm{R}_t : \mathrm{T}\mathcal{Q}(\Psi_t) \to \mathcal{Q}(\Psi_t) : (\boldsymbol{p}_t, \boldsymbol{\xi}_t) \mapsto \boldsymbol{q}_t := \mathrm{R}_t(\boldsymbol{p}_t, \boldsymbol{\xi}_t)\,, \\ &\quad \text{with } \left.\frac{\mathrm{d}R_t(\boldsymbol{p}_t, \alpha\boldsymbol{\xi}_t)}{\mathrm{d}\alpha}\right|_{\alpha=0} = \boldsymbol{\xi}_t \text{ and } \mathrm{R}_t(\boldsymbol{p}_t, \boldsymbol{0}) = \boldsymbol{p}_t. \end{aligned} \tag{25}$$

It is clear that the retraction is not unique, and in fact, the retraction can be seen as an approximation of the exponential map. Indeed, in this specific case the exponential map cannot be easily computed, see the discussion in [5], but some retractions can be easily computed.

The retraction considered in this work and introduced in [5] is illustrated in Fig. 8: the retraction $\mathrm{R}_t(\boldsymbol{p}_t, \boldsymbol{\xi}_t)$ is obtained by looking at the intersection $\boldsymbol{q}_t$ between the quadric and the line between $\boldsymbol{p}_t + \boldsymbol{\xi}_t$ and the quadric center $\boldsymbol{d}$, as defined in Eq. (16). Note that this 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 [5, §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 Eq. (17), $\boldsymbol{\xi}_t = P_{\boldsymbol{p}}\,(\boldsymbol{v}_t)$.

This retraction can be readily extended to the multistep case: it suffices to work with each component independently:

$$\mathrm{R} : \mathrm{T}\mathcal{Q}^{\mathrm{tot}} \to \mathcal{Q}^{\mathrm{tot}} : (\boldsymbol{p}, \boldsymbol{\xi}) \mapsto \boldsymbol{q} := \begin{pmatrix} \boldsymbol{q}_1 \\ \boldsymbol{q}_2 \\ \vdots \\ \boldsymbol{q}_{|T|} \end{pmatrix}, \tag{26}$$

with $\boldsymbol{q}_t = \mathrm{R}_t(\boldsymbol{p}_t, \boldsymbol{\xi}_t)$. Notice that, if the retraction is illustrated with an ellipse in Fig. 8, it is not limited to this specific quadric. Any type-2 quadric or quadric with a middle point (see § 2.5), can be considered. Also, an interesting feature of this procedure is the fact that it does not require a lot of computational power: the tangent space has a closed-form Eq. (18), as well as the projection onto this tangent space Eq. (23), and finally the retraction itself can also be efficiently computed—it amounts to solving $|T|$ one-dimensional quadratic equations and choosing for each equation the root the closest to one [5].

Going back to Eq. (17), if the direction $\boldsymbol{v}^k$ does not belong to the tangent space of the current iterate $\boldsymbol{p}^k$, it reads

$$\boldsymbol{p}^{k+1} = \mathrm{R}(\boldsymbol{p}^k, P_{\boldsymbol{p}^k}\left(\alpha^k \boldsymbol{v}^k\right)). \tag{27}$$

3.2.3. Descent direction on a manifold

Before the discussion on the *descent* direction, we define the concept of $\mathcal{Q}-$admissible direction which accommodates the set defined by Eq. (5).

Definition 3.6. 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 $\mathrm{R}(\boldsymbol{p}, \alpha\boldsymbol{v})$ 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 Eq. (1) is only smooth almost-everywhere, and the zero-measure set where it is non-smooth is located at positions where the argument of the absolute value in Eq. (1) changes sign. This set simply corresponds to a multidimentional grid which can be computed as

$$S := \left\{ \boldsymbol{p} \in \mathbb{R}^{|G||T|} \mid \exists g \in G, t \in T, j \in J_g \text{ with } p_{gt} = P_g^- + \frac{(j-1)\pi}{2E_g} \right\}, \tag{28}$$

where

$$J_g := \left\{ j = 1, 2, \ldots, 1 + \left\lceil (P_g^+ - P_g^-)\frac{2E_g}{\pi} \right\rceil \right\}. \tag{29}$$

For a non-smooth function, the gradient is often replaced by the subgradient, however this mathematical object cannot be used for the non-convex functions (1). Here, we consider the closely connected concept of *generalized gradient* introduced in [6]. First, let us define the generalized directional derivative $f^\circ(\boldsymbol{p}; \boldsymbol{v})$ of the Lipschitz function $f : X \to \mathbb{R}$, 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 $\boldsymbol{v}$, independently on 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,

$$\partial f(\boldsymbol{p}) = \left\{ \boldsymbol{\zeta} \in X^* \mid f^\circ(\boldsymbol{p}; \boldsymbol{v}) \geqslant \langle \boldsymbol{v}, \boldsymbol{\zeta} \rangle \quad \forall \boldsymbol{v} \in X \right\}, \tag{30}$$

with X^* the dual space of X. The generalized gradient shares some important properties with the subdifferential of a convex function, namely the fact that it is a nonempty convex

and compact set and that 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 (2) is a Lipschitz function that can be computed as the pointwise maximum of $m := 2^{|G||T|}$ smooth functions[3], *i.e.*,

$$f(\boldsymbol{p}) = \max_{j=1,\dots,m} f_j(\boldsymbol{p}). \tag{31}$$

In this specific case, [5] shows that the generalized gradient can be described as

$$\partial f(\boldsymbol{p}) = \operatorname{co}\left\{\nabla f_j(\boldsymbol{p}) \mid j \in \mathcal{I}_f(\boldsymbol{p})\right\}, \tag{32}$$

where $\operatorname{co}\{\cdot\}$ denotes the convex hull and $\mathcal{I}_f$ the set of indices for which the maximum in Eq. (31) is attained.

This framework is valid for the unconstrained problem (P). Let us now integrate the manifold constraint Eq. (5), and then the other linear constraints.

Given a smooth function f_j from the pointwise maximum in Eq. (31), the *projected gradient* is defined as follows

$$\operatorname{grad} f_j(\boldsymbol{p}) = P_{\boldsymbol{p}}\left(\nabla f_j(\boldsymbol{p})\right), \tag{33}$$

and the projected generalized gradient is given by

$$\operatorname{grad} f(\boldsymbol{p}) = \operatorname{co}\left\{\operatorname{grad} f_j(\boldsymbol{p}) \mid j \in \mathcal{I}_f(\boldsymbol{p})\right\}. \tag{34}$$

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 [5] 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

$$\boldsymbol{\lambda}^k = \underset{\substack{\boldsymbol{\lambda}\geqslant 0 \\ \sum \lambda_j = 1}}{\arg\min} \left\| \sum_{j\in\mathcal{I}_f(\boldsymbol{p}^k)} \lambda_j \operatorname{grad} f_j(\boldsymbol{p}^k) \right\|^2 = \underset{\substack{\boldsymbol{\lambda}\geqslant 0 \\ \sum \lambda_j = 1}}{\arg\min} \left\| P_{\boldsymbol{p}^k}\left(\sum_{j\in\mathcal{I}_f(\boldsymbol{p}^k)} \lambda_j \nabla f_j(\boldsymbol{p}^k) \right) \right\|^2 \tag{35}$$

then the steepest-descent $\mathcal{Q}-$admissible direction is computed as

$$\boldsymbol{v}^k = -P_{\boldsymbol{p}^k}\left(\sum_{j\in\mathcal{I}_f(\boldsymbol{p}^k)} \lambda_j^k \nabla f_j(\boldsymbol{p}^k) \right). \tag{36}$$

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, [5] also introduces a reformulation which exploits the specific form of the function (2) and considerably reduces the dimension of the problem. We show here how to apply this reformulation to the extended quadric manifold.

[3]Since $|x| = \max\{x, -x\}$ and there are $|G|\,|T|$ absolute values in Eq. (2).

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.*,

$$\mathcal{S}(\boldsymbol{p}^k) = \left\{ \underbrace{(g_1^{\mathrm{s}}, t_1^{\mathrm{s}})}_{:=s_1}, \underbrace{(g_2^{\mathrm{s}}, t_2^{\mathrm{s}})}_{:=s_2}, \ldots, \underbrace{(g_{n_{\mathrm{s}}^k}^{\mathrm{s}}, t_{n_{\mathrm{s}}^k}^{\mathrm{s}})}_{:=s_{n_{\mathrm{s}}^k}} \right\} = \bigcup_{t \in T} \mathcal{S}_t(\boldsymbol{p}_t^k) = \bigcup_{t \in T} \left\{ (g,t) \mid g \in G, f_g^{\mathrm{VPE}}(p_{gt}) = 0 \right\} \tag{37}$$

$$\mathcal{F}(\boldsymbol{p}^k) = \left\{ \underbrace{(g_1^{\mathrm{f}}, t_1^{\mathrm{f}})}_{:=f_1}, \underbrace{(g_2^{\mathrm{f}}, t_2^{\mathrm{f}})}_{:=f_2}, \ldots, \underbrace{(g_{n_{\mathrm{f}}^k}^{\mathrm{f}}, t_{n_{\mathrm{f}}^k}^{\mathrm{f}})}_{:=f_{n_{\mathrm{f}}^k}} \right\} = \bigcup_{t \in T} \mathcal{F}_t(\boldsymbol{p}_t^k) = \bigcup_{t \in T} \left\{ (g,t) \mid g \in G, (g,t) \notin \mathcal{S}(\boldsymbol{p}_t^k) \right\} \tag{38}$$

and we have naturally $T \times G = \mathcal{S}(\boldsymbol{p}^k) \bigcup \mathcal{F}(\boldsymbol{p}^k)$ for all $\boldsymbol{p}^k \in \mathcal{Q}^{\mathrm{tot}}$, $\left|\mathcal{S}(\boldsymbol{p}^k)\right| = n_{\mathrm{s}}^k$ and $\left|\mathcal{F}(\boldsymbol{p}^k)\right| = n_{\mathrm{f}}^k = |T|\,|G| - n_{\mathrm{s}}^k$. Note that, in order 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 non-smooth part. Let $\boldsymbol{g}^k$ be the smooth part and $\boldsymbol{S}^k$ the matrix containing the non-smooth parts to be combined,

$$\boldsymbol{S}^k = \left[P_{\boldsymbol{p}^k}\left(\nabla \hat{f}_{s_1^k}^{\mathrm{S}}(\boldsymbol{p}^k) \right), \ldots, P_{\boldsymbol{p}^k}\left(\nabla \hat{f}_{n_{\mathrm{s}}^k}^{\mathrm{S}}(\boldsymbol{p}^k) \right) \right] \in \mathbb{R}^{|T||G| \times n_{\mathrm{s}}^k} \tag{39}$$

$$\boldsymbol{g}^k = P_{\boldsymbol{p}^k}\left(\nabla f^{\mathrm{Q}}(\boldsymbol{p}^k) + \sum_{(g,t) \in \mathcal{F}(\boldsymbol{p}^k)} \nabla \left| \hat{f}^{\mathrm{S}}(\boldsymbol{p}^k \right|) \right) \in \mathbb{R}^{|T||G|} \tag{40}$$

where f^{S} is the sine part of Eq. (1), and $\hat{f}_s(\boldsymbol{p}) := f_s(p_s)$. Note that, as the fuel cost is independent of time, we define with a slight abuse of notation $f_{gt} := f_g$.

The subproblem Eq. (35) can be rewritten as

$$\boldsymbol{\lambda}^k = \underset{-1 \leqslant \boldsymbol{\lambda} \leqslant 1}{\arg\min} \left|\left| \boldsymbol{g}^k + \boldsymbol{S}^k \boldsymbol{\lambda} \right|\right|_2^2, \tag{41}$$

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 Eq. (41) is a convex quadratic programming (QP) problem of dimension $n_{\mathrm{s}}^k \leqslant |T|\,|G|$, which is much easier to solve than any problem in Table 1.

Notice that, until now, the unique constraint that we consider is Eq. (5). The generalization of the descent direction on a *constrained manifold*, such as the feasible set of (P) is presented hereafter.

3.2.4. Descent direction on constrained manifold

If, as in (P), the feasible set is a manifold further constrained by q linear constraints under the form $\boldsymbol{c}_i{}^{\intercal} \boldsymbol{p} \leqslant 0$ with $i = 1 \ldots q$, we define the matrix $\boldsymbol{C}^k$ of the projected active constraints at point $\boldsymbol{p}^k$, whose columns are given by

$$\boldsymbol{C}_{*,j}^k = P_{\boldsymbol{p}^k}(\boldsymbol{c}_j) \quad \text{for all } j \in \{1 \ldots q\} \text{ such that } \boldsymbol{c}_j{}^{\intercal} \boldsymbol{p}^k = 0. \tag{42}$$

We have $\boldsymbol{C}^k \in \mathbb{R}^{|T||G| \times n_c^k}$ with $0 \leqslant n_c^k \leqslant q$.

The subproblem Eq. (41) becomes

$$(\boldsymbol{\lambda}^k, \boldsymbol{\mu}^k) = \underset{\substack{-1 \leqslant \boldsymbol{\lambda} \leqslant 1 \\ \boldsymbol{\mu} \geqslant 0}}{\arg\min} \left|\left| \boldsymbol{g}^k + \boldsymbol{S}^k \boldsymbol{\lambda} + \boldsymbol{C}^k \boldsymbol{\mu} \right|\right|_2^2, \tag{Sub}$$

and the descent direction $\boldsymbol{v}^k = -(\boldsymbol{g}^k + \boldsymbol{S}^k \boldsymbol{\lambda}^k + \boldsymbol{C}^k \boldsymbol{\mu}^k)$.

Note that 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: $\boldsymbol{v}^k = -\boldsymbol{g}^k = -P_{\boldsymbol{p}^k}\left(\nabla f(\boldsymbol{p}^k)\right)$.

In order to complete the description of the line-search scheme, it remains to choose a step-size rule and a stopping criterion.

3.2.5. 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 [6]. Thus, a natural stopping criterion is to monitor the direction norm. Unfortunately, as studied in [10], 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 as this norm varies non-smoothly 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 for 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 $\boldsymbol{p}^{k+1} = \mathrm{R}(\boldsymbol{p}^k, \alpha^k \boldsymbol{v}^k)$ feasible, while sufficiently decreasing the objective. An explicit implementation of Armijo's rule is described in [5, 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 above a given treshold is found.

It appears that, for problems which are sufficiently large, the subproblem (Sub) becomes problematic, in the sense that the direction obtained is only admissible, *i.e.*, feasible for (P), in a tiny neighbourhood around the previous iterate. This can arise when a given component of an iterate p_{gt}^k is binding at multiple operational constraints, *e.g.*, $p_{gt}^k = P_g^-$ (Eq. (3) is tight) and $p_{gt}^k - p_{g(t-1)}^k = R_g^+$ (Eq. (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. 9. This allows us to provide a temporary degree of freedom to the algorithm, which may find an other 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. 9.

Finally, it should be noted that in general, it is very unlikely that a given iterate would be exactly located at a non-smooth point. This implies that $\boldsymbol{S}^k$ from Eq. (39) is likely to be empty. Hence, and in a similar fashion as [5], we consider that the equalities from Eqs. (39) and (42) should be satisfied within a small ϵ accuracy.

4. Test cases

In a similar fashion as [29], 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

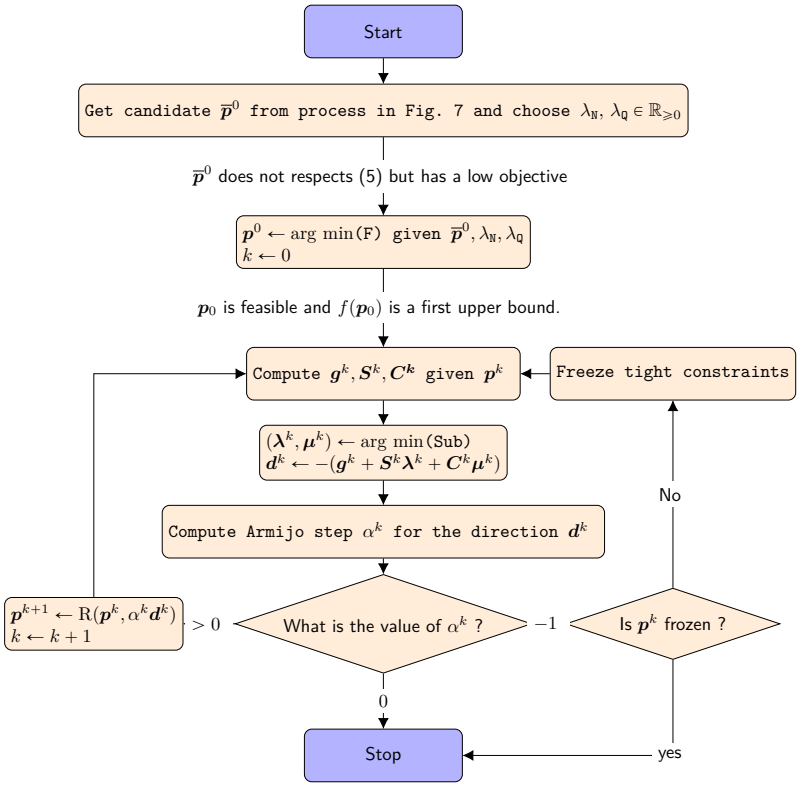


Figure 9: Flowchart of the method that projects the candidate from Fig. 7 and improves it through a Riemannian subgradient descent.

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, is also reported.

In order to account for the different processor speeds from other methods in the literature, the scaled CPU time is used [44]:

$$\text{Scaled CPU time} = \frac{\text{Given CPU Speed}}{\text{Base CPU speed}} \text{Given CPU time}\,, \tag{43}$$

where the base CPU time used in this paper 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 if the run time remains reasonable. The key contribution of the proposed method is to be found in the "lower bound" column; see also Section 1 for a discussion of its purpose. The scaled CPU time is denoted as S-time and given in minutes.

In addition to the main objective, the *deviation* ans the *losses* are computed. The deviation corresponds to the mismatch between the point and the ellipsoid, and is computed by rearanging Eq. (5): deviation $= \sum_{t\in T} \left| \sum_{g\in G} p_{gt} - P_t^{\mathrm{D}} - p_t^{\mathrm{loss}} \right|$, and the losses correspond to the value of $\sum_{t\in T} p_t^{\mathrm{loss}}$.

The data, final solution, and algorithm implementations are available on GitLab [34].

4.1. 5-unit, 24 time steps test case

We use the data from [30]. The data consists of a 5-unit case, where all units obey a valve-point effect. The reserve is set to 5% of the demand.

We compare the solution obtained with our proposed method to other methods from the literature in Table 2. The three first columns report the minimum, average and maximal solution. For deterministic methods, only the first column includes values. The best solution that is available in the literature is plugged into our model, in order to compute the losses and the demand deviation. The latter is defined as the violation of Eq. (5). Only the proposed method provides a lower bound which allows us to bound the final optimality gap at 1.3 %. Remark that, since the lower bound is only improved in the first part of the proposed method, *i.e.*, APLA, this lower bound will always be equal to the one of the full method APLA-RSG.

The proposed method, APLA-RSG, achieves a competitive objective with respect to other methods in 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 [42] only reports the number of function evaluation ($\sim 2e5$). Function evaluations (FEs) allow an accurate comparison between methods run on different computers, however they cannot be computed in our case due to the call to the MIP solver in APLA. We note, nonetheless, that the RSG method requires 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.

Note that Table 2 also presents the results of Ipopt [38] 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 non-convex objective Eq. (1) using JuMP [9, 15].

Table 2: Summary results: 5-unit case

Method	Cost			S-Time	Loss (MW)	Deviation (MW)	Lower bound
	Min	Avg	Max				
BBOSB[42]	43018	43066	43197	-	194.65	0.01	-
HIGA[24]	43125	43162	43259	1.37	194.79	0.074	-
ICA[25]	43117	43144	43210	-	194.80	0.014	-
MILP-IPM[29]	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

Table 3: Summary results: 10-unit case

Method	Cost			S-Time	Loss (MW)	Deviation (MW)	Lower bound
	Min	Avg	Max				
BBOSB[42]	1039169[4]	-	-	-	818.22	83	-
TSMILP [41]	1037487	-	-	1.9	832.32	0.013	-
MILP-IPM[29]	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

4.2. 10-unit, 24 time steps test case

The data originates from [30] and consists of a 10-unit case. All units obey a VPE, and the matrix $\boldsymbol{B}$ is indefinite. Similarly as [41], the reserve is set to 3.5% of the demand and not 5%. As a matter of fact, the problem with 5% reserve is not feasible: this can be shown by examining the static dispatch at the highest demand. Since Eq. (6) must hold for all t, we have

$$\sum_{g\in G} P_g^+ - (P_t^{\mathrm{D}} + \min_{\boldsymbol{p}_t\in\mathcal{P}_t} p_t^{\mathrm{loss}} + P_t^{\mathrm{S}}) \geqslant 0, \tag{44}$$

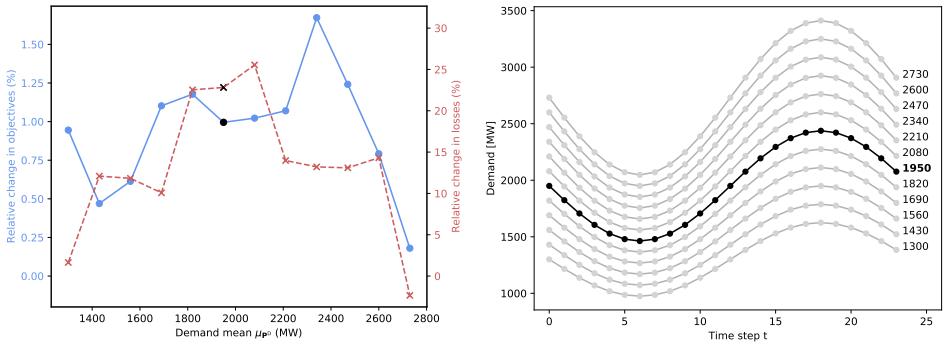
where $\mathcal{P}_t$ corresponds to the intersection of the power ranges Eq. $(3)_t$ with a relaxed version of the power balance Eq. $(5)_t$,

$$\sum_{g\in G} p_{gt} \geqslant P_t^{\mathrm{D}}. \tag{45}$$

This is a relaxation because negative losses p_t^{loss} are physically impossible. It is clear that, if the problem (P) is feasible, then Eq. $(6)_t$ holds for all t, which implies that Eq. $(44)_t$ also holds for all t. Conversely, if Eq. $(44)_t$ does not hold for any 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 [29], despite developing the method to account for reserves, do not test the 10-unit test case with reserve. This may also explain why [41] choose a 3.5% reserve instead of the usual 5% requirement. This also raises questions about certain methods in the literature, reported in [41, Table V], which claim to solve this infeasible problem.

Table 3 compares the different methods from the literature. The discussion is analogous to the previous case (§ 4.1).

[4]This value differs from the reported value of [42] and has been computed from the given solution of [42]. This may come from a mistake in the solution reported. This mistake could explain the high balance deviation, meaning that this reported solution is not feasible.



(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 that each have a different mean.

(b) Load profiles studied in § 4.3. 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 $\mu_{\boldsymbol{P}^{\mathrm{D}}}$.

Figure 10: Comparison between APLA-RSG and Ipopt for the 15-unit test case for twelve load profiles. The profile with $\mu_{\boldsymbol{P}^{\mathrm{D}}} = 1950$ is emphasized in black in both figures.

4.3. 15-unit, 24 time steps test case

The data for this test case originates from [46]. The original instance consists of 15 units in a static dispatch. All units obey a valve-point effect in the original instance. As in [35], we model a demand over 24 time steps with ramping constraints. We compare the solution of APLA-RSG with Ipopt. Fig. 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. 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 %. Note that a lower (or higher) mean demand $\mu_{\boldsymbol{P}^{\mathrm{D}}}$ than the one considered in the present experiments results 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, the solution from APLA-RSG is obtained twice as fast while strictly meeting the constraints, decreasing the losses, and reducing the objective of around 1%.

5. Conclusion

In this work, we develop a method for tackling a non-smooth and non-convex economic dispatch problem. Non-convexities originate from the inclusion of the valve-point effect, which is an important effect in the operation of large gas units, and from the consideration of power losses, which are modelled as a non-convex 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 as APLA-RSG, consists of i) finding a lower bound and 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 in a similar amount of time as other methods from the literature. However, APLA-RSG benefits from other advantages, namely the fact that it provides a lower bound and strictly satisfies the balance constraint. The lower bound allows the estimation of an optimality gap, despite the fact that the problem is non-convex. Such a lower bound can also prove useful for other methods, so as to assess whether derived solutions are of acceptable quality.

Further work may include the following extensions. Firstly, we are interested in considering a more complex model: prohibited operation zones (POZ) [20] 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 deteriotate the performance in term of objective value at the projection step. And finally, the extension of the method to optimal power flow and more specifically convex or non-convex ACOPF is of interest.

References

[1] Absil, P.A., Sluysmans, B., Stevens, N.: MIQP-based algorithm for the global solution of economic dispatch problems with valve-point effects. In: 2018 Power Systems Computation Conference (PSCC), pp. 1–7 (2018). DOI 10.23919/PSCC.2018.8450877

[2] Aravena, I., Papavasiliou, A.: Renewable energy integration in zonal markets. IEEE Transactions on Power Systems **32**(2), 1334–1349 (2017). DOI 10.1109/TPWRS.2016.2585222

[3] Arens, T., Hettlich, F., Karpfinger, C., Kockelkorn, U., Lichtenegger, K., Stachel, H.: Mathematik, 1. aufl. 2008 edn. Spektrum Akademischer Verlag (2008)

[4] Attaviriyanupap, P., Kita, H., Tanaka, E., Hasegawa, J.: A hybrid EP and SQP for dynamic economic dispatch with nonsmooth fuel cost function. IEEE Transactions on Power Systems **17**(2), 411–416 (2002). DOI 10.1109/TPWRS.2002.1007911

[5] Borckmans, P.B., Selvan, S.E., Boumal, N., Absil, P.A.: 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

[6] Clarke, F.H.: A New Approach to Lagrange Multipliers. Mathematics of Operations Research **1**(2), 165–174 (1976). DOI 10.1287/moor.1.2.165

[7] Coelho, L.S., Mariani, V.C.: 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)

[8] Decker, G.L., Brooks, A.D.: Valve point loading of turbines. Electrical Engineering **77**(6), 501–501 (1958). DOI 10.1109/EE.1958.6445133

[9] Dunning, I., Huchette, J., Lubin, M.: JuMP: A modeling language for mathematical optimization. SIAM Review **59**(2), 295–320 (2017). DOI 10.1137/15M1020575

[10] Dutta, J., Deb, K., Tulshyan, R., Arora, R.: Approximate KKT points and a proximity measure for termination. Journal of Global Optimization **56**(4), 1463–1499 (2013). DOI 10.1007/s10898-012-9920-5

[11] European Commission: Energy roadmap 2050 (2011). URL https://ec.europa.eu/energy/sites/ener/files/documents/2012_energy_roadmap_2050_en_0.pdf. [Online; accessed 2020-05-28]
[12] Guillemin, V., Pollack, A.: Differential topology, vol. 370. American Mathematical Soc. (2010)
[13] Hemamalini, S., Simon, S.P.: 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 https://doi.org/10.1016/j.ijepes.2010.12.017
[14] Huchette, J., Vielma, J.P.: Nonconvex piecewise linear functions: Advanced formulations and simple modeling tools (2019)
[15] JuMP documentation: Nonlinear modeling. URL https://jump.dev/JuMP.jl/stable/nlp/. Accessed: 2021-08-01
[16] Karami, M., Shayanfar, H., Aghaei, J., Ahmadi, A.: Scenario-based security-constrained hydrothermal coordination with volatile wind power generation. Renewable and Sustainable Energy Reviews **28**, 726 – 737 (2013). DOI https://doi.org/10.1016/j.rser.2013.07.052
[17] Kirchmayer, L.: Economic Operation of Power Systems. General Electric series. Wiley (1958)
[18] Kron, G.: 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
[19] Kunz, F.: Improving congestion management: How to facilitate the integration of renewable generation in germany. The Energy Journal **Volume 34**(Number 4) (2013)
[20] Lee, F.N., Breipohl, A.M.: Reserve constrained economic dispatch with prohibited operating zones. IEEE Transactions on Power Systems **8**(1), 246–254 (1993). DOI 10.1109/59.221233
[21] Low, S.H.: Convex relaxation of optimal power flow—part II: Exactness. IEEE Transactions on Control of Network Systems **1**(2), 177–189 (2014). DOI 10.1109/TCNS.2014.2323634
[22] Midcontinent Independent System Operator (MISO): Enhanced modeling of combined cycle generators. URL https://www.misoenergy.org/stakeholder-engagement/issue-tracking/enhanced-modeling-of-combined-cycle-generators. [Online; accessed 2020-05-28]
[23] Mohammadi, F., Abdi, H.: A modified crow search algorithm (MCSA) for solving economic load dispatch problem. Applied Soft Computing **71**, 51 – 65 (2018). DOI https://doi.org/10.1016/j.asoc.2018.06.040
[24] Mohammadi-Ivatloo, B., Rabiee, A., Soroudi, A.: Nonconvex Dynamic Economic Power Dispatch Problems Solution Using Hybrid Immune-Genetic Algorithm. IEEE Systems Journal **7**(4), 777–785 (2013). DOI 10.1109/JSYST.2013.2258747. Conference Name: IEEE Systems Journal
[25] Mohammadi-ivatloo, B., Rabiee, A., Soroudi, A., Ehsan, M.: Imperialist competitive algorithm for solving non-convex dynamic economic power dispatch. Energy **44**(1), 228–240 (2012). DOI 10.1016/j.energy.2012.06.034
[26] Niknam, T., Azizipanah-Abarghooee, R., Aghaei, J.: A new modified teaching-learning algorithm for reserve constrained dynamic economic dispatch. IEEE Transactions on Power Systems **28**(2), 749–763 (2013). DOI 10.1109/TPWRS.2012.2208273
[27] Niknam, T., Narimani, M.R., Azizipanah-Abarghooee, R.: A new hybrid algorithm for optimal power flow considering prohibited zones and valve point effect. Energy Conversion and Management **58**, 197 – 206 (2012). DOI https://doi.org/10.1016/j.enconman.2012.01.017
[28] Pan, S., Jian, J., Chen, H., Yang, L.: 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**, 106,061 (2020). DOI https://doi.org/10.1016/j.epsr.2019.106061
[29] Pan, S., Jian, J., Yang, L.: 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 https://doi.org/10.1016/j.ijepes.2017.11.004
[30] Panigrahi, C.K., Chattopadhyay, P.K., Chakrabarti, R.N., Basu, M.: Simulated annealing technique for dynamic economic dispatch. Electric Power Components and Systems **34**(5), 577–586 (2006). DOI 10.1080/15325000500360843
[31] Pinheiro, R.B., Balbo, A.R., Nepomuceno, L.: Solving network-constrained nonsmooth economic dispatch problems through a gradient-based approach. International Journal of Electrical Power & Energy Systems **113**, 264 – 280 (2019). DOI https://doi.org/10.1016/j.ijepes.2019.05.046
[32] Saadat, H.: Power System Analysis. McGraw-Hill, New York (1999)
[33] Swamp, K.S., Natarajan, A.: Constrained optimization using evolutionary programming for dynamic economic dispatch. In: Proceedings of International Conference on Intelligent Sensing and Information Processing, pp. 314–319 (2005). DOI 10.1109/ICISIP.2005.1529468
[34] Van Hoorebeeck, L.: Adaptive Piecewise Linear Approximation and Riemannian Subgradient Descent. https://gitlab.com/Loicvh/apla-rsg (2021)

[35] Van Hoorebeeck, L., Absil, P.A., Papavasiliou, A.: Global solution of economic dispatch with valve point effects and transmission constraints. Electric Power Systems Research **189**, 106,786 (2020). DOI https://doi.org/10.1016/j.epsr.2020.106786

[36] Van Hoorebeeck, L., Papavasiliou, A., Absil, P.A.: MILP-based algorithm for the global solution of dynamic economic dispatch problems with valve-point effects. IEEE Power and Energy Society General Meeting (2019)

[37] Victoire, T.A., Jeyakumar, A.: Hybrid pso–sqp for economic dispatch with valve-point effect. Electric Power Systems Research **71**(1), 51 – 59 (2004). DOI https://doi.org/10.1016/j.epsr.2003.12.017

[38] Wächter, A., Biegler, L.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming **106**(1), 25–57 (2006). DOI 10.1007/s10107-004-0559-y

[39] Wong, K.P., Fung, C.C.: Simulated annealing based economic dispatch algorithm. IEEE Proceedings C - Generation, Transmission and Distribution **140**(6), 509–515 (1993). DOI 10.1049/ip-c.1993.0074

[40] Wu, Z., Ding, J., Wu, Q., Jing, Z., Zhou, X.: Two-phase mixed integer programming for non-convex economic dispatch problem with spinning reserve constraints. Electric Power Systems Research **140**, 653 – 662 (2016). DOI https://doi.org/10.1016/j.epsr.2016.05.006

[41] Wu, Z., Ding, J., Wu, Q.H., Jing, Z., Zheng, J.: 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)

[42] Xiong, G., Shi, D.: Hybrid biogeography-based optimization with brain storm optimization for non-convex dynamic economic dispatch with valve-point effects. Energy **157**, 424–435 (2018). DOI 10.1016/j.energy.2018.05.180

[43] Yang, L., Fraga, E.S., Papageorgiou, L.G.: Mathematical programming formulations for non-smooth and non-convex electricity dispatch problems. Electric Power Systems Research **95**, 302 – 308 (2013). DOI https://doi.org/10.1016/j.epsr.2012.09.015

[44] Yuan, X., Wang, L., Yuan, Y., Zhang, Y., Cao, B., Yang, B.: A modified differential evolution approach for dynamic economic dispatch with valve-point effects. Energy Conversion and Management **49**(12), 3447 – 3453 (2008). DOI https://doi.org/10.1016/j.enconman.2008.08.016

[45] Zhang, F.: The Schur Complement and its Applications, *Numerical Methods and Algorithms*, vol. 4. Springer, New York (2005). DOI 10.1007/b105056

[46] 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