Distributed Load Control in Multiphase Radial Networks

Thesis by

Lingwen Gan

In Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy

California Institute of Technology
Pasadena, California

2015
(Defended August 28, 2014)

The thesis is dedicated to

my girlfriend Tianlu,

whose love made this thesis possible,

and my parents,

who have supported me all the way.

Acknowledgements

I would like to express my deepest gratitude to my advisor, Professor Steven Low, who guided me through my years pursuing a Ph.D. He is shockingly nice and always places his students first: he allowed me to graduate ahead of time when I fell into financial crisis, he never pushed us to work for the funding, he allowed us to work from home, and he took care of a lot of dirty work himself. He is enthusiastic about research and always works on hard-core problems: he digs into the details of our work and thinks with us along the way, he consistently brings us onto the right track, and encourages us to take risks and aim big in our careers.

Next, I am grateful to my collaborators, Ufuk Topcu, Adam Wierman, Na Li, and Niangjun Chen. It is a great pleasure to work with and take the advice of these great minds. Professor Ufuk Topcu is the first person who taught me how to write a technical paper. Professor Adam Wierman taught me how to make a technical paper easy to read, and how to make a decent presentation (though I never even got close to his level).

I have greatly enjoyed studying in the department of Electrical Engineering at California Institute of Technology. We are provided an amazing working and living environment. The large and bright three-people offices are paradise for theoretical research, and all sorts of free lunches and snacks have provided us with great relaxation. I would also like to thank the helpful administrative staff, Christine Ortega, Lisa Knox, and Sydney Garstang, and many others.

I would like to thank my parents for their spiritual support during the years. They have gone through extremely difficult times: an illness that pushed us to the edge of despair, bad relationship with relatives, and grudge from the elders. Even during the most difficult times, they encouraged me to pursue my dream of being a scholar without worrying about their well-being. I know it is absolutely my duty to take care of their needs.

Finally, I would like to thank my girlfriend Tianlu Zhang who brings happiness back to my life. Having struggled long between family needs and a personal dream and suffered an emotional crisis, I finally got rid of all these annoyance because of her. She lets me realize what is the most important thing to me—family. For the rest of my life, I will dedicate myself to the well-being of her and my parents, and be a person like my advisor.

Abstract

The current power grid is on the cusp of modernization due to the emergence of distributed generation and controllable loads, as well as renewable energy. On one hand, distributed and renewable generation is volatile and difficult to dispatch. On the other hand, controllable loads provide significant potential for compensating for the uncertainties. In a future grid where there are thousands or millions of controllable loads and a large portion of the generation comes from volatile sources like wind and solar, distributed control that shifts or reduces the power consumption of electric loads in a reliable and economic way would be highly valuable.

Load control needs to be conducted with network awareness. Otherwise, voltage violations and overloading of circuit devices are likely. To model these effects, network power flows and voltages have to be considered explicitly. However, the physical laws that determine power flows and voltages are nonlinear. Furthermore, while distributed generation and controllable loads are mostly located in distribution networks that are multiphase and radial, most of the power flow studies focus on single-phase networks.

This thesis focuses on distributed load control in multiphase radial distribution networks. In particular, we first study distributed load control without considering network constraints, and then consider network-aware distributed load control.

Distributed implementation of load control is the main challenge if network constraints can be ignored. In this case, we first ignore the uncertainties in renewable generation and load arrivals, and propose a distributed load control algorithm, Algorithm 1, that optimally schedules the deferrable loads to shape the net electricity demand. Deferrable loads refer to loads whose total energy consumption is fixed, but energy usage can be shifted over time in response to network conditions. Algorithm 1 is a distributed gradient decent algorithm, and empirically converges to optimal deferrable load schedules within 15 iterations.

We then extend Algorithm 1 to a real-time setup where deferrable loads arrive over time, and only imprecise predictions about future renewable generation and load are available at the time of decision making. The real-time algorithm Algorithm 2 is based on model-predictive control: Algorithm 2 uses updated predictions on renewable generation as the true values, and computes a pseudo load to simulate future deferrable load. The pseudo load consumes 0 power at the current time step, and its total energy consumption equals the expectation of future deferrable load total energy request.

Network constraints, e.g., transformer loading constraints and voltage regulation constraints, bring significant challenge to the load control problem since power flows and voltages are governed by nonlinear physical laws. Remarkably, distribution networks are usually multiphase and radial. Two approaches are explored to overcome this challenge: one based on convex relaxation and the other that seeks a locally optimal load schedule.

To explore the convex relaxation approach, a novel but equivalent power flow model, the branch flow model, is developed, and a semidefinite programming relaxation, called BFM-SDP, is obtained using the branch flow model. BFM-SDP is mathematically equivalent to a standard convex relaxation proposed in the literature, but numerically is much more stable. Empirical studies show that BFM-SDP is numerically exact for the IEEE 13-, 34-, 37-, 123-bus networks and a real-world 2065-bus network, while the standard convex relaxation is numerically exact for only two of these networks.

Theoretical guarantees on the exactness of convex relaxations are provided for two types of networks: single-phase radial alternative-current (AC) networks, and single-phase mesh direct-current (DC) networks. In particular, for single-phase radial AC networks, we prove that a second-order cone program (SOCP) relaxation is exact if voltage upper bounds are not binding; we also modify the optimal load control problem so that its SOCP relaxation is always exact. For single-phase mesh DC networks, we prove that an SOCP relaxation is exact if 1) voltage upper bounds are not binding, or 2) voltage upper bounds are uniform and power injection lower bounds are strictly negative; we also modify the optimal load control problem so that its SOCP relaxation is always exact.

To seek a locally optimal load schedule, a distributed gradient-decent algorithm, Algorithm 9, is proposed. The suboptimality gap of the algorithm is rigorously characterized and close to 0 for practical networks. Furthermore, unlike the convex relaxation approach, Algorithm 9 ensures a feasible solution. The gradients used in Algorithm 9 are estimated based on a linear approximation of the power flow, which is derived with the following assumptions: 1) line losses are negligible; and 2) voltages are reasonably balanced. Both assumptions are satisfied in practical distribution networks. Empirical results show that Algorithm 9 obtains 70+ times speed up over the convex relaxation approach, at the cost of a suboptimality within numerical precision.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The power grid is at the cusp of modernization due to the emergence of controllable loads and renewable generation. Controllable loads represented by electric vehicles have become a booming industry: over 170,000 highway-capable electric vehicles have been sold in the US from 2008 to 2013, and 16 electric vehicle models from 9 major manufacturers have been available in the US market by March 2014 [107]. This trend is forecast to speed up as major vehicle manufacturers announce their electric vehicle plans [8,47,99]. Renewable generation capacity has enjoyed an annual growth rate of 10–60% since 2004, e.g., the annual growth rate of wind generation capacity is 24% and the annual growth rate of solar generation capacity is 60%. In 2010, renewable energy consumption already occupies 16.7% of the total world energy consumption [105].

The adoption of controllable loads and renewable generation brings integration challenges to the power grid. If not coordinated wisely, the charging of electric vehicles may lead to coincidence peaks in electricity demand [62]. Consequently, power transmission/distribution lines carry much larger currents and power transformers are loaded much more heavily. As a result, circuit device lifespan will be greatly reduced [87], and network voltages will deviate significantly from their nominal values [30]. Renewable generation is not dispatchable. Furthermore, it can fluctuate severely within a short time frame, making the balance between electricity generation and demand fragile and electricity blackouts more likely.

Distributed load control can mitigate these integration challenges. For example, the charging of electric vehicles can be coordinated to compensate for the random fluctuations in renewable generation, and reactive power injections of solar photovoltaics (considered as negative loads) can be adjusted to stabilize network voltages. Consequently, frequency and voltage regulations can be achieved with less participation of the expensive ancillary services and power electronic devices. Finally, loads are located throughout the network. These millions devices can only be controlled in a distributed way.

1.1 Distributed Load Control

Load control falls into two categories: 1) direct load control, where a centralized load serving entity determines when and how much each load consumes electricity [41, 45, 53, 75]; and 2) price-based control, where the centralized load serving entity alters the electricity prices to incentivize the loads to change their behavior [5, 26, 70]. Direct load control has the merits of obtaining reliable responses while price-based control has to deal with uncertain human behavior. Hence, this thesis focuses on distributed direct load control mechanisms.

A distributed direct load control mechanism has to deal with the following three issues among others. 1) Since loads are controlled in a distributed manner, it is nontrivial to achieve system-level objectives. 2) Loads may have nonconvex constraints, e.g., a load may consume a fixed amount of power when it is turned on and the only flexibility in controlling the load is to decide when to turn it on. Such constraints are integer constraints and make the load control problem NP-hard in general. 3) A real-world load control mechanism requires a real-time implementation due to the uncertainties in renewable generation and load arrivals.

To minimize system-level objectives in a distributed way, we propose a gradient-decent algorithm: Algorithm 1 in Chapter 2. Algorithm 1 assumes full knowledge of renewable generation and load, and describes an iterative procedure where a centralized coordinator and a number of deferrable loads negotiate on the power consumption schedules over a period of time. In each iteration, the coordinator computes the gradients of the objective with respect to each load, and each load updates its schedule by moving along the negative direction of the gradient. We prove that Algorithm 1 converges to global minimizers of the objective function in Theorem 2.8, and case studies in Section 2.4 verify that Algorithm 1 converges to optimal load schedules within 15 iterations.

To handle nonconvex load constraints, a randomized algorithm based on the martingale theory is proposed in [45] (it is not included in this thesis since it is not related to other chapters of the thesis). It is proved in [45] that the randomized algorithm converges almost surely to certain load schedule, whose suboptimality is upper bounded by $O(1/n)$ where n is the number of loads.

Model-predictive control is adopted to obtain a real-time distributed load control algorithm: Algorithm 2 in Chapter 3. At each time step, Algorithm 2 computes a load schedule over a time horizon into the future, but only implements the schedule for the first time slot. In computing the schedule, renewable generation is approximated by its up-to-date prediction, and future deferrable loads are simulated by a pseudo load. The pseudo load consumes 0 power at the first time slot, and its total energy consumption equals the expected total energy request of future deferrable loads. Due to its advantage of using up-to-date predictions, the average suboptimality of Algorithm 2 vanishes as the time horizon expands (see Theorem 3.3). It is proved in [27] that the typical suboptimality of Algorithm 2 is within a constant time of the average suboptimality. The improvement of Algorithm

2 over the optimal static control is $O(T/\ln T)$ for two representative cases (see Corollary 3.10 and 3.11), where T is the length of the time horizon.

1.2 Optimal Power Flow

Load control has to be implemented taking into account of network constraints. For example, a load schedule should not cause transformer overloads or severe voltage deviations. To capture network power flows and voltages, the underlying physical laws have to be considered. These laws turn out to be nonlinear and complicate optimization problems involving power flow constraints.

Approaches to handle nonlinear power flow equations fall into three categories. 1) Approximate power flow equations by linear equations. 2) Look for local optima. 3) Consider convex relaxations that can be solved in polynomial time and check if the solutions are feasible and hence optimal for the original problem.

Within the scope of linear approximation, a DC approximation is adopted in the industry to estimate the real power flows in balanced transmission networks [7, 94, 95]. However, DC approximation only provides estimates for real power flows but not estimates for reactive power flows nor voltages. Furthermore, DC approximation does not apply to unbalanced multiphase networks, which are the typical configurations of distribution networks. A linear approximation of the power flows in multiphase radial networks that provides accurate estimates of the voltages, real power flows, and reactive power flows is provided in Section 4.4. Empirical studies in Section 4.5.2 show that the proposed linear approximation obtains voltage estimates within 0.0016 per unit from their true values.

A variety of nonlinear programming techniques have been applied to obtain a local optimum of the underlying optimization problem, e.g., [11, 21, 32, 59, 78, 92, 100]. These algorithms respect nonlinear power flow equations and obtain physically implementable solutions if they converge, but can be computationally inefficient in comparison with the linear approximation approach. Besides, the suboptimality is difficult to quantify. The distributed algorithm Algorithm 9 proposed in Chapter 5 is an algorithm of this type, but with similar computational complexity as the linear approximation approach and a quantifiable suboptimality gap. In particular, the suboptimality gap provided in Theorem 5.12 is close to zero for practical networks. Algorithm 9 is a gradient-decent algorithm, with gradients approximated using the linearization of power flow proposed in Section 4.4. Moreover, Algorithm 9 is a distributed algorithm. Numerical studies in Section 5.5 show that a serial implementation of Algorithm 9 achieves 70+ times speedup over the convex relaxation approach.

The convex relaxation approach seeks to minimize the objective over a convex superset of the original feasible set. In general, only a lower bound on the optimal objective value can be obtained; but in practice, the optimal solution of a convex relaxation usually lands in the original feasible set.

In such cases, the convex relaxation is called exact and a global optimum of the original problem can be recovered. There are three major questions in exploring this convex relaxation approach. 1) What is the form of the convex relaxation? 2) Is there a numerically stable algorithm to solve the convex relaxation that scales to large problem sizes? 3) When is a convex relaxation exact?

To answer question 1), a standard semidefinite programming relaxation referred to as standard-SDP has been proposed in the literature for single-phase mesh networks [10]. Though a distribution network is typically multiphase and radial [63], it has a single-phase mesh equivalent circuit [29, 60] and therefore standard-SDP is applicable [33]. By exploiting the radial network topology, an equivalent SDP relaxation called BIM-SDP is proposed in Section 4.2 that reduces the computational complexity from $O(n^2)$ for standard-SDP to $O(n)$ for BIM-SDP, where n is the number of lines in the network.

To answer question 2), an SDP relaxation BFM-SDP that enhances the numerical stability of BIM-SDP is proposed in Section 4.3. BIM-SDP is ill-conditioned due to subtractions of voltages that are close in value, and BFM-SDP avoids such subtractions by adopting different variables to attain numerical stability.

To answer question 3), we prove that BIM-SDP is exact if and only if BFM-SDP is exact in Theorem 4.9, and show that BFM-SDP is numerically exact for the IEEE 13, 34, 37, 123-bus networks and a real-world 2065-bus network in Section 4.5.1. We also provide theoretical guarantees for the exactness of convex relaxations of the optimal power flow problem for two types of networks: single-phase radial alternative-current (AC) networks and single-phase mesh direct-current (DC) networks. In particular, for single-phase radial AC networks, we prove that a second-order cone programming (SOCP) relaxation is exact if voltage upper bounds are not binding (see Theorem 6.2); for single-phase mesh DC networks, we prove that a similar but different SOCP relaxation is exact if voltage upper bounds are not binding (see Theorem 7.4), or voltage upper bounds are uniform with strictly negative power injection lower bounds (see Theorem 7.5). For each type of network, a modified optimal power flow problem is proposed that has an exact SOCP relaxation (see Section 6.3 and 7.5, respectively). The modified problems are obtained by imposing additional linear constraints on the power injections such that voltage upper bounds do not bind.

1.3 Thesis Overview

The thesis is organized as follows.

1. Chapters 2 and 3 focus on distributed load control without network awareness. In particular, Chapter 2 focuses on offline scheduling of distributed loads and Chapter 3 focuses on real-time control of distributed loads. Chapter 2 is based on [44] and Chapter 3 is based on [46].

2. In Chapters 4 and 5, we explore methods for solving OPF. In particular, Chapter 4 introduces power flow models for multiphase radial networks and develops two convex relaxations of the optimal power flow problem and a linear approximation of the power flow. It is based on [43]. Chapter 5 develops a distributed gradient-decent algorithm for solving the optimal power flow problem, and has not been published yet.

3. In Chapters 6 and 7, we provide sufficient conditions for the exactness of convex relaxations for two types of networks: single-phase radial AC networks (Chapter 6) and single-phase mesh DC networks (Chapter 7).

   For single-phase radial AC networks, it is proved in Chapter 6 that an SOCP relaxation is exact if voltage upper bound constraints do not bind. Based on this sufficient condition, a modified OPF problem is proposed by imposing additional linear constraints on power injections such that voltage upper bounds do not bind. The additional constraints only eliminate OPF feasible points that are close to voltage upper bounds, and guarantees the exactness of SOCP.

   For single-phase mesh DC networks, it is proved in Chapter 7 that an SOCP relaxation is exact if either voltage upper bounds do not bind, or voltage upper bounds are uniform with power injection lower bounds being strictly negative. A similar additional linear constraints on power injections can be imposed to ensure the exactness of the SOCP.

Chapter 2

Distributed Load Control

A distributed algorithm that optimally schedules deferrable electric loads represented by electric vehicles (EV) is presented in this chapter. We first formulate EV charging scheduling as an optimal control problem, whose objective is to impose a generalized notion of valley-filling, and study properties of optimal charging profiles. We then give a distributed algorithm to iteratively solve the optimal control problem. In each iteration, EVs update their charging profiles according to the control signal broadcast by a centralized coordinator, and the coordinator alters the control signal to guide their updates. Simulation results verify that the algorithm converges to optimal EV charging schedules within 15 iterations.

Literature A variety of algorithms have been proposed in the literature to coordinate the charging of EVs, ranging from centralized algorithms where a coordinator makes decisions for the EVs [85,86, 98,102] to distributed algorithms where each EV makes its own decisions [23,44,76,88]. Note that distributed algorithms may still require coordinators to facilitate the communication among EVs.

Centralized algorithms are mainly for cost-benefit analysis purposes when the number of EVs is large, since the computation burden would be too heavy for a single computation unit. In [102], a large number of operational distribution networks in The Netherlands are investigated to quantify the impact of EV charging on various network levels. Results show that controlled charging can reduce infrastructure update investment by half over uncontrolled charging. In [85], uncontrolled charging and smart charging of EVs are compared empirically to highlight that a second peak electricity demand during the night can be avoided by adopting smart charging. In [98] and [86], centralized linear programmings are proposed to compute the optimal charging profiles of EVs. In [98], the linear programming aims to minimize the power supply cost subject to circuit capacity constraints and the vehicle owner's requirements. The linear programming in [86] further includes a power network physical model that captures voltage deviations.

By distributing the computation burden among different units, a distributed algorithm is more suitable for scenarios where a large number of EVs need to be coordinated. In [76], a distributed algorithm is proposed to schedule the charging of EVs such that the aggregate electricity demand is made flat. It is proved in [76] that when the EVs are identical, the obtained aggregate electricity demand will be optimal in the sense that it is as flat as possible. This notion of optimal aggregate electricity demand is formalized in [44] which proposes a different distributed algorithm that always attains optimality. The algorithms proposed in [23, 88] take another perspective: instead of trying to flatten the aggregate electricity demand, they seek to minimize the charging cost given a pre-determined electricity price profile by solving a dynamic programming at each EV.

Summary The contribution of this chapter is a distributed EV charging scheduling algorithm that converges to optimal charging profiles. The algorithm is iterative. In each iteration, EVs update their charging profiles according to the control signal broadcast by the utility company, and the utility company alters the control signal to guide their updates. Imperfect information about non-EV load and EV arrivals is considered in Chapter 3, and power flows are considered in Chapters 4–7.

The rest of the chapter is organized as follows. Section 2.1 formulates EV charging scheduling as an optimal control problem. Section 2.2 studies properties of optimal charging profiles. Section 2.3 provides a distributed algorithm to iteratively solve the optimal control problem, and proves that the algorithm converges to optimal charging profiles. Case studies are presented in Section 2.4.

2.1 Problem Formulation

This chapter studies the design and analysis of EV charging scheduling algorithms to flatten the aggregate electricity demand. Throughout, we consider a discrete-time model over a finite time horizon. The time horizon is divided into T time slots of equal length and indexed $1, \ldots, T$. In practice, the time horizon could be one day and the length of a time slot could be 10 minutes.

Let base load $b = \{b(\tau)\}_{\tau=1}^{T}$ denote the aggregate of non-EV load and assume that b is precisely known by the load serving entity. In practice, base load b is a stochastic process due to the uncertainty in both demand and renewable generation. This will be studied in the next chapter.

Consider a setting where n EVs arrive over the time horizon, each requiring a certain amount of electricity by a given deadline. We use EV and deferrable load interchangeably in this thesis. Assume that the load serving entity can negotiate with the EVs on their charging profiles over the time horizon even if EVs have not arrived for charging. Imperfect information about the arrival times and electricity requirements of these EVs will be considered in the next chapter.

For each EV, its arrival time and deadline, as well as other constraints on its power consumption, are captured via upper and lower bounds on its possible power consumption during each time. Specifically, let $i = 1, 2, \ldots, n$ denote these EVs. The power consumption of EV i at time t, denoted by $p_i(t)$, must be between given lower and upper bounds $\underline{p}_i(t)$ and $\overline{p}_i(t)$, i.e.,

$$\underline{p}_i(t) \le p_i(t) \le \overline{p}_i(t), \quad i = 1, \ldots, n, \; t = 1, \ldots, T. \tag{2.1}$$

These are specified exogenously to our algorithms. For example, if an electric vehicle plugs in with level II charging, then its power consumption must be within $[0, 3.3]$kW, i.e., $\underline{p}_i(t) = 0$ and $\overline{p}_i(t) = 3.3$; if it is not plugged in (has either not arrived yet or has already departed), then its power consumption is 0kW, i.e., $\underline{p}_i = \overline{p}_i = 0$. Further, we assume that an EV i must withdraw a fixed amount of energy P_i by its deadline, i.e.,

$$\sum_{t=1}^{T} p_i(t) = P_i, \quad i = 1, \ldots, n. \tag{2.2}$$

The objective of EV charging control is to "flatten" the *aggregate load* $p_0 = \{p_0(t)\}_{t=1}^T$ that the substation draws from the main grid [52], which equals

$$p_0(t) = b(t) + \sum_{i=1}^{n} p_i(t), \quad t = 1, \ldots, T \tag{2.3}$$

assuming power loss is negligible. This objective is captured by minimizing the *variance*

$$V(p_0) := \frac{1}{T} \sum_{t=1}^{T} \left(p_0(t) - \frac{1}{T} \sum_{\tau=1}^{T} p_0(\tau) \right)^2 \tag{2.4}$$

of aggregate load p_0.

To summarize, the optimal deferrable load control (ODLC) problem is as follows. Let $\mathcal{T} := \{1, 2, \ldots, T\}$, $\mathcal{N} := \{0, 1, \ldots, n\}$, and $\mathcal{N}^+ := \mathcal{N} \backslash \{0\}$ for convenience.

$$\textbf{ODLC:} \ \min \ \sum_{t \in \mathcal{T}} \left(p_0(t) - \frac{1}{T} \sum_{\tau \in \mathcal{T}} p_0(\tau) \right)^2$$

$$\text{over} \ \ p_i(t), \quad i \in \mathcal{N}, \; t \in \mathcal{T};$$

$$\text{s.t.} \ \ p_0(t) = b(t) + \sum_{i=1}^{n} p_i(t), \qquad t \in \mathcal{T}; \tag{2.5a}$$

$$\underline{p}_i(t) \le p_i(t) \le \overline{p}_i(t), \qquad i \in \mathcal{N}^+, \; t \in \mathcal{T}; \tag{2.5b}$$

$$\sum_{t \in \mathcal{T}} p_i(t) = P_i, \qquad i \in \mathcal{N}^+. \tag{2.5c}$$

In the ODLC problem (2.5), the objective is simply the variance $V(p_0)$ of the aggregate load p_0, and the constraints correspond to equations (2.3), (2.1), and (2.2), respectively.

Discussion on the Objective. The objective function in (2.5) can be simplified. Note that

$$\sum_{t\in\mathcal{T}} p_0(t) = \sum_{t\in\mathcal{T}} b(t) + \sum_{i=1}^{n} P_i =: P$$

is a constant that does not depend on $(p_1,\ldots,p_n)$. Hence, the objective can be simplified as

$$\sum_{t\in\mathcal{T}} \left(p_0(t) - \frac{1}{T}\sum_{\tau\in\mathcal{T}} p_0(\tau) \right)^2 \quad = \quad \sum_{t\in\mathcal{T}} p_0^2(t) - \frac{1}{T}P^2$$

and it suffices to minimize

$$L(p_0) := \sum_{t\in\mathcal{T}} p_0^2(t).$$

Remarkably, if $f : \mathbb{R} \mapsto \mathbb{R}$ is strictly convex, then minimizing

$$L_f(p_0) := \sum_{t\in\mathcal{T}} f(p_0(t))$$

is equivalent to minimizing $L(p_0)$, as stated in the following theorem. Let $p := (p_0, p_1, \ldots, p_n)$.

Theorem 2.1. *Let f be strictly convex. Then*

$$\begin{array}{c} p^* \in \text{argmin } L(p_0) \\ \text{s.t. } (2.5\text{a}) - (2.5\text{c}) \end{array} \quad \Longleftrightarrow \quad \begin{array}{c} p^* \in \text{argmin } L_f(p_0) \\ \text{s.t. } (2.5\text{a}) - (2.5\text{c}). \end{array}$$

Theorem 2.1 implies that the objective in (2.5) can be changed to $L_f(p_0)$ for arbitrary strictly convex f without changing its optimal solutions.

Proof. We prove that $p^* \in \text{argmin } L_f(p_0)$ implies $p^* \in \text{argmin } L(p_0)$. The proof of $p^* \in \text{argmin } L(p_0)$ implies $p^* \in \text{argmin } L_f(p_0)$ is similar and omitted for brevity.

Let p^* be a minimizer of $L_f(p_0)$, then

$$(p_1^*, p_2^*, \ldots, p_n^*) \in \text{argmin } \sum_{t\in\mathcal{T}} f\left(b(t) + \sum_{i=1}^{n} p_i(t) \right) \text{ s.t. } (2.5\text{b}) - (2.5\text{c}).$$

This is equivalent to

$$\sum_{i=1}^{n} \sum_{t\in\mathcal{T}} f'(p_0^*(t))[p_i(t) - p_i^*(t)] \geq 0 \quad \forall (p_1,\ldots,p_n) \text{ satisfying } (2.5\text{b}) - (2.5\text{c}).$$

according to the first order optimality condition. Note that (2.5b)–(2.5c) are decoupled in terms of

$(p_1, p_2, \ldots, p_n)$. Hence, if we define

$$\mathcal{P}_i := \left\{ p_i \in \mathbb{R}^T \mid \underline{p}_i(t) \le p_i(t) \le \overline{p}_i(t), \ \sum_{t \in \mathcal{T}} p_i(t) = P_i \right\}$$

for $i \in \mathcal{N}^+$, then

$$\sum_{t \in \mathcal{T}} f'(p_0^*(t))[p_i(t) - p_i^*(t)] \ge 0 \quad \forall p_i \in \mathcal{P}_i$$

for $i \in \mathcal{N}^+$.

It follows that for each $i \in \mathcal{N}^+$, there exists μ_i such that

$$p_i^*(t) = \begin{cases} \overline{p}_i(t) & \text{if } f'(p_0^*(t)) < \mu_i \\ \underline{p}_i(t) & \text{if } f'(p_0^*(t)) > \mu_i. \end{cases}$$

Since f is strictly convex, f' is strictly increasing. Therefore, there exists λ_i such that

$$p_i^*(t) = \begin{cases} \overline{p}_i(t) & \text{if } p_0^*(t) < \lambda_i \\ \underline{p}_i(t) & \text{if } p_0^*(t) > \lambda_i. \end{cases}$$

It follows that

$$\sum_{t \in \mathcal{T}} p_0^*(t)[p_i(t) - p_i^*(t)] \ge 0 \quad \forall p_i \in \mathcal{P}_i$$

for $i \in \mathcal{N}^+$, and therefore

$$\sum_{i=1}^{n} \sum_{t \in \mathcal{T}} p_0^*(t)[p_i(t) - p_i^*(t)] \ge 0 \quad \forall (p_1, \ldots, p_n) \text{ satisfying (2.5b)} - \text{(2.5c)}.$$

This is the first order optimality condition for

$$(p_1^*, p_2^*, \ldots, p_n^*) \in \text{argmin} \sum_{t \in \mathcal{T}} \left(b(t) + \sum_{i=1}^{n} p_i(t) \right)^2 \text{ s.t. (2.5b)} - \text{(2.5c)},$$

i.e., p^* is a minimizer of $L(p_0)$. This completes the proof of Theorem 2.1. □

Remark 2.1. *If the objective is to track a given load profile G rather than to flatten the total load profile, then the objective function in* (2.5) *can be modified as $\sum_{t \in \mathcal{T}} [p_0(t) - G(t)]^2$. This is equivalent to having a base load $b - G$.*

2.2 Optimal Charging Profile

We study properties of optimal charging profiles in this section.

Definition 2.1. *A charging profile $p = (p_0, p_1, \dots, p_n)$ is*

1) feasible*, if it satisfies the constraints* (2.5a)–(2.5c)*;*

2) optimal*, if it solves the ODLC problem* (2.5)*;*

3) valley-filling*, if it is feasible, and there exists $A \in \mathbb{R}$ such that*

$$p_0(t) = \left[A - b(t) - \sum_{i=1}^{n} \underline{p}_i(t)\right]^+, \quad t \in \mathcal{T}.$$

Property 2.2. *A valley-filling charging profile is optimal.*

Proof. Consider the relaxed optimal deferrable load control (R-ODLC) problem

$$\begin{aligned}
\textbf{R-ODLC:}\ \min\ & \sum_{t\in\mathcal{T}} \left(p_0(t) - \frac{1}{T}P\right)^2 \\
\text{over}\ & p_0(t), \quad t \in \mathcal{T}; \\
\text{s.t.}\ & \sum_{t\in\mathcal{T}} p_0(t) = P; && \text{(2.6a)} \\
& p_0(t) \geq b(t) + \sum_{i=1}^{n} \underline{p}_i(t), \qquad t \in \mathcal{T}. && \text{(2.6b)}
\end{aligned}$$

For any feasible charging profile $p = (p_0, p_1, \dots, p_n)$ of (2.5), p_0 is feasible for the R-ODLC problem (2.6). Besides, the objective function of (2.5) evaluated at p equals the objective function of (2.6) evaluated at p_0. Therefore, if p_0 solves (2.6), then p solves (2.5).

It can be verified that for any valley-filling charging profile $p = (p_0, \dots, p_n)$, p_0 solves (2.6). Hence, p solves (2.5). □

Let $\mathcal{F}_i := \left\{p_i \mid \underline{p}_i \leq p_i \leq \overline{p}_i,\ \sum_{t\in\mathcal{T}} p_i(t) = P_i\right\}$ denote the set of feasible p_i for $i \in \mathcal{N}^+$, and $\mathcal{F}$ denote the feasible set of the ODLC problem (2.5).

Property 2.3. *Optimal charging profiles exist if feasible charging profiles exist, i.e., $\mathcal{F} \neq \emptyset$.*

Proof. It is straightforward to verify that the set $\mathcal{F}$ is compact and that the objective function in (2.5) is continuous. Hence, optimal solutions of (2.5) exist. □

Valley-filling is our intuitive notion of optimality. However, it is not always achievable. For example, the "valley" in the base load may be so deep that even if all EVs charge at their maximum rate, the valley still cannot be completely filled, e.g., at 4:00 in Figure 2.1 (right). Besides, EVs may have stringent deadlines such that the potential for shifting the load over time to yield valley-filling is limited. Defining optimal charging profiles as solving the ODLC problem (2.5) generalizes valley-filling according to Properties 2.2 and 2.3: valley-filling charging profiles are optimal, e.g., Figure

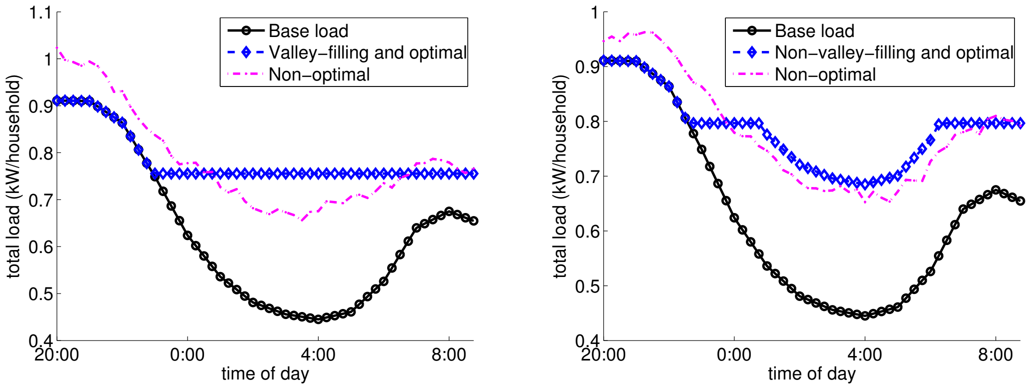


Figure 2.1: Base load profile is the average residential load in the service area of Southern California Edison from 20:00 on February 13, 2011 to 9:00 on February 14, 2011 [2]. Optimal total load profiles correspond to the solutions of the ODLC problem (2.5). With different EV specifications, optimal charging profiles can be valley-filling (left figure) or non-valley-filling (right figure). Hypothetical non-optimal total load profiles are shown in purple with dash-dot lines.

2.1 (left); while valley-filling charging profiles do not always exist, optimal charging profiles always exist, e.g., Figure 2.1 (right). We now define an equivalence relation between two charging profiles, and show that the set of optimal charging profiles is an equivalence class of this relation.

Definition 2.2. *Two feasible charging profiles $p = (p_0, \ldots, p_n)$ and $p' = (p'_0, \ldots, p'_n)$ are equivalent if $p_0 = p'_0$. We denote this relation by $p \sim p'$.*

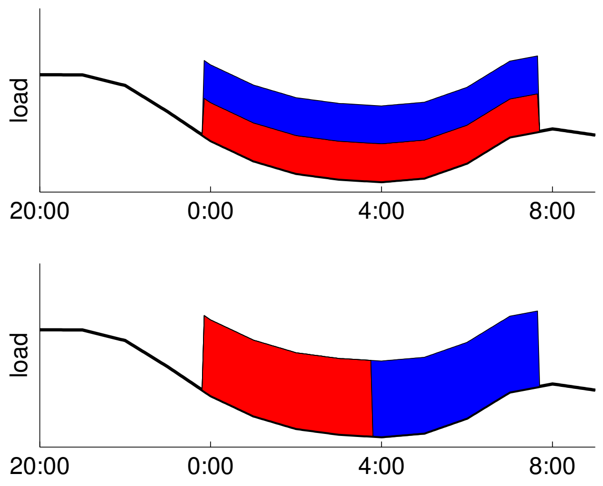


Figure 2.2: An example of equivalent charging profiles. In both top and bottom figures, the red region corresponds to the charging profile of one EV, and the blue region corresponds to the charging profile of another EV. The total load profiles in both figures equal, therefore the charging profiles in both figures are equivalent.

An example of equivalent charging profiles is given in Figure 2.2. It is not difficult to check that

$\sim$ is an equivalence relation (satisfies reflexivity, symmetry, and transitivity [103]), and therefore we can define an equivalence class $\{p' \in \mathcal{F} | \ p' \sim p\}$ for each $p \in \mathcal{F}$. Let

$$\mathbb{O} := \{p \in \mathcal{F} \mid p \text{ is optimal}\}$$

denote the set of optimal charging profiles.

Theorem 2.4. *Assume feasible charging profiles exist, i.e., $\mathcal{F} \neq \emptyset$. Then the set $\mathbb{O}$ of optimal charging profiles is non-empty, compact, convex, and an equivalence class.*

Proof. The set $\mathbb{O}$ is nonempty according to Property 2.3. Let $p^{\text{opt}} \in \mathbb{O}$, and define

$$\mathbb{O}' := \{p' \in \mathcal{F} \mid p' \sim p^{\text{opt}}\}.$$

It can be verified that $\mathbb{O}'$ is closed and convex. Since $\mathcal{F}$ is compact, the set $\mathbb{O}'$—a closed subset of $\mathcal{F}$—is also compact. Hence, $\mathbb{O}'$ is non-empty, compact, convex, and an equivalence class.

We are left to prove that $\mathbb{O} = \mathbb{O}'$. It is not difficult to check that $\mathbb{O}' \subseteq \mathbb{O}$, and we prove $\mathbb{O} \subseteq \mathbb{O}'$ as follows. For any $p \in \mathbb{O}$, it follows from the first order optimality condition [18, Ch. 4.2.3] that

$$\langle p_0^{\text{opt}}, p_0 - p_0^{\text{opt}} \rangle \geq 0, \quad \langle p_0, p_0^{\text{opt}} - p_0 \rangle \geq 0$$

since both p^{opt} and p minimize $L(p_0)$ over $\mathcal{F}$. It follows that

$$\langle p_0 - p_0^{\text{opt}}, p_0 - p_0^{\text{opt}} \rangle \leq 0$$

and therefore $p_0 = p_0^{\text{opt}}$. Hence, $p \in \mathbb{O}'$ and it follows that $\mathbb{O} \subseteq \mathbb{O}'$. $\square$

Corollary 2.5. *Optimal charging profile is in general not unique.*

To summarize, defining optimal charging profiles as the profiles that minimize the load variance generalizes valley-filling. With this definition, optimal charging profiles always exist and coincide with valley-filling charging profiles if valley-filling charging profiles exist.

2.3 Distributed Scheduling Algorithm

A distributed scheduling algorithm that solves the ODLC problem (2.5) is provided in this section. By distributed, we mean that EVs choose their own charging profiles, instead of being instructed by a centralized coordinator. The coordinator only uses control signals, e.g. electricity prices, to guide EVs in choosing their charging profiles. By scheduling, we mean that all EVs are available for negotiation at the beginning of the scheduling horizon (even though they are not necessarily available for charging as reflected by the time-varying $\overline{p}_1, \ldots, \overline{p}_n$). The EVs and the coordinator carry out an iterative procedure at the beginning of the scheduling horizon to determine the charging rates for each of the time slots $t \in \mathcal{T}$ in the future.

2.3.1 The Optimal Distributed Charging Algorithm

The distributed algorithm for solving (2.5) is presented in Algorithm 1, which we call the optimal distributed charging (ODC) algorithm. The superscripts denote iteration indices, e.g., $c^{(k)}$ denotes the control signal in iteration k.

Algorithm 1 Optimal distributed charging

```
Input: The coordinator knows the base load $b$, the number $n$ of EVs, and stopping criteria $\epsilon$. Each
    EV $i \in \mathcal{N}^+$ knows its charging requirement $P_i$ and bounds $(\underline{p}_i, \overline{p}_i)$ on its charging rates.
Output: Each EV $i \in \mathcal{N}^+$ computes its own vector $p_i$.
1: each EV $i \in \mathcal{N}^+$ initializes its charging profile as $p_i^{(0)} \leftarrow 0$ and sends $p_i^{(0)}$ to the coordinator;
2: $k \leftarrow 0$;
3: the coordinator calculates control signal $c^{(k)}$ as
       $$c^{(k)} \leftarrow \frac{1}{n} p_0^{(k)} = \frac{1}{n}\left(b + \sum_{i=1}^{n} p_i^{(k)}\right); \qquad (2.7)$$
4: if $k \geq 1$ and $\left\| c^{(k)} - c^{(k-1)} \right\| \leq \epsilon$, the coordinator broadcasts a stop signal for each EV to go to
   Step 7);
   otherwise the coordinator broadcasts $c^{(k)}$ to all EVs;
5: each EV $i \in \mathcal{N}^+$ calculates a new charging profile $p_i^{(k+1)}$ as
       $$p_i^{(k+1)} \leftarrow \text{argmin } \left\langle c^{(k)}, p_i \right\rangle + \frac{1}{2}\left\| p_i - p_i^{(k)} \right\|^2 \text{ s.t. } p_i \in \mathcal{F}_i; \qquad (2.8)$$
   and sends $p_i^{(k+1)}$ to the coordinator;
6: $k \leftarrow k + 1$; go to Step 3);
7: each EV $i \in \mathcal{N}^+$ sets final solution $p_i \leftarrow p_i^{(k)}$;
```

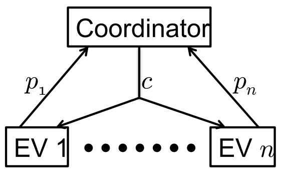


Figure 2.3: Schematic view of the information flow between the coordinator and the EVs. Given the control signal c, EVs update their charging profiles p_i independently. The coordinator guides their updates by altering the control signal c.

Figure 2.3 shows the information exchange between the coordinator and the EVs for Algorithm 1. Given the control signal c broadcast by the coordinator, EVs update their charging profiles p_i independently, and report the updated charging profiles to the coordinator. The coordinator alters the control signal c according to the received charging profiles. In practice, the control signal c can be, e.g., electricity prices.

There are two main steps (3) and (5) in Algorithm 1. In Step (3), the coordinator updates control signal c according to (2.7). Higher prices are set for slots with higher load, to incentivize the EVs to shift their electricity consumption to slots with lower load. In Step (5), EVs update their charging profiles to minimize the objective function in (2.8), which consist of two terms: the first term is the electricity cost and the second term penalizes the deviation of p_i from the charging profile $p_i^{(k)}$ calculated in the previous iteration. The penalty term ensures the convergence of Algorithm 1, and vanishes as $k \to \infty$ (see Theorem 2.10). Hence, (2.8) reduces to the electricity cost as $k \to \infty$.

We now prove that Algorithm 1 converges to optimal charging profiles. Let the superscript k for each variable denote its respective value in iteration k. For example, $p_0^{(k)} = b + \sum_{i \in \mathcal{N}} p_i^{(k)}$ denotes the total load profile in iteration k.

Lemma 2.6. *Let $i \in \mathcal{N}^+$ and $k \geq 1$. Then*

$$\left\langle c^{(k)}, p_i^{(k+1)} - p_i^{(k)} \right\rangle \leq - \left\| p_i^{(k+1)} - p_i^{(k)} \right\|^2 . \tag{2.9}$$

Proof. Fix an arbitrary $i \in \mathcal{N}^+$ and an arbitrary $k \geq 1$. The first order optimality condition for (2.8) implies that

$$\left\langle c^{(k)} + p_i^{(k+1)} - p_i^{(k)},\ p_i - p_i^{(k+1)} \right\rangle \geq 0 \tag{2.10}$$

for all $p_i \in \mathcal{F}_i$. Note that $p_i^{(k)} \in \mathcal{F}_i$. Hence,

$$\left\langle c^{(k)} + p_i^{(k+1)} - p_i^{(k)},\ p_i^{(k)} - p_i^{(k+1)} \right\rangle \geq 0,$$

which implies (2.9). $\square$

Lemma 2.7. *Let $i \in \mathcal{N}^+$ and $k \geq 1$. Then*

$$p_i^{(k+1)} = p_i^{(k)} \quad \Longleftrightarrow \quad \left\langle c^{(k)}, p_i - p_i^{(k)} \right\rangle \geq 0 \text{ for } p_i \in \mathcal{F}_i. \tag{2.11}$$

Lemma 2.7 follows from (2.10) and the strict convexity of (2.8).

Theorem 2.8. *Charging profile $p^{(k)}$ converges to the set $\mathbb{O}$ of optimal charging profiles, i.e.,*

$$p^{(k)} \to \mathbb{O}, \qquad k \to \infty.$$

Proof. Note that

$$\begin{aligned}
L\left(p^{(k+1)}\right) - L\left(p^{(k)}\right) &= \left\langle p_0^{(k+1)},\ p_0^{(k+1)} \right\rangle - \left\langle p_0^{(k)},\ p_0^{(k)} \right\rangle \\
&= 2\left\langle p_0^{(k)},\ p_0^{(k+1)} - p_0^{(k)} \right\rangle + \left\| p_0^{(k+1)} - p_0^{(k)} \right\|^2 \\
&= 2n\left\langle c^{(k)},\ \sum_{i=1}^{n} \left(p_i^{(k+1)} - p_i^{(k)}\right) \right\rangle + \left\| \sum_{i=1}^{n} \left(p_i^{(k+1)} - p_i^{(k)}\right) \right\|^2 \\
&\leq 2n\sum_{i=1}^{n} \left\langle c^{(k)},\ p_i^{(k+1)} - p_i^{(k)} \right\rangle + n\sum_{i=1}^{n} \left\| p_i^{(k+1)} - p_i^{(k)} \right\|^2 \\
&\leq -n\sum_{i=1}^{n} \left\| p_i^{(k+1)} - p_i^{(k)} \right\|^2 \ \leq\ 0 \qquad (2.12)
\end{aligned}$$

for $k \geq 1$. The first inequality is due to the Cauchy-Schwarz theorem, and the second inequality is due to Lemma 2.6. It is easy to check that $L(p^{(k+1)}) = L(p^{(k)})$ if and only if $p^{(k+1)} = p^{(k)}$.

If $p^{(k+1)} = p^{(k)}$, then it follows from Lemma 2.7 that $\langle c^{(k)}, p_i - p_i^{(k)} \rangle \geq 0$ for $i \in \mathcal{N}^+$ and $p_i \in \mathcal{F}_i$. Therefore,

$$\sum_{i=1}^{n} \left\langle 2p_0^{(k)},\ p_i - p_i^{(k)} \right\rangle = 2n\sum_{i=1}^{n} \left\langle c^{(k)},\ p_i - p_i^{(k)} \right\rangle \geq 0 \qquad (2.13)$$

for $p = (p_0, \ldots, p_n) \in \mathcal{F}$. This is the first order optimality condition for $p^{(k)}$ to solve (2.5). Hence, $p^{(k)} \in \mathbb{O}$. On the other hand, if $p^{(k)} \in \mathbb{O}$, then $L(p^{(k)}) \leq L(p^{(k+1)}) \leq L(p^{(k)})$. To summarize,

$$L\left(p^{(k+1)}\right) = L\left(p^{(k)}\right) \quad \Longleftrightarrow \quad p^{(k+1)} = p^{(k)} \quad \Longleftrightarrow \quad p^{(k)} \in \mathbb{O}.$$

Finally, the facts that $\mathcal{F}$ is compact, every $p \in \mathbb{O}$ minimizes L, and $L(p^{(k+1)}) < L(p^{(k)})$ if $p^{(k)} \notin \mathbb{O}$ imply that $p^{(k)} \to \mathbb{O}$ as $k \to \infty$. □

Definition 2.3. *A charging profile p is* stationary, *if $p^{(k)} = p$ for some $k \geq 0$ implies $p^{(m)} = p$ for all $m \geq k$.*

Corollary 2.9. *A charging profile p is stationary if and only if it is optimal.*

Corollary 2.9 follows from the fact that $p^{(k+1)} = p^{(k)}$ if and only if $p^{(k)} \in \mathbb{O}$, which is shown in the proof of Theorem 2.8.

Theorem 2.10. *Let p^{opt} be an optimal charging profile and $c^{\text{opt}} = p_0^{\text{opt}}/n$ be the corresponding control signal. Then*

- *total load profile converges to the optimal value, i.e., $p_0^{(k)} \to p_0^{\text{opt}}$ as $k \to \infty$;*
- *control signal converges to the optimal value, i.e., $c^{(k)} \to c^{\text{opt}}$ as $k \to \infty$;*
- *charging profile update vanishes, i.e., $\left\| p_i^{(k+1)} - p_i^{(k)} \right\| \to 0$ as $k \to \infty$ for $i \in \mathcal{N}^+$.*

Proof. According to Theorem 2.8, there exists a sequence $\{\hat{p}^{(k)}\}_{k\geq 1} \in \mathbb{O}$ of optimal charging profiles such that $\|p^{(k)} - \hat{p}^{(k)}\| \to 0$ as $k \to \infty$. Note that $\mathbb{O}$ is an equivalence class of p^{opt}. Hence, $\hat{p}_0^{(k)} = p_0^{\text{opt}}$ for $k \geq 1$. Therefore, $p_0^{(k)} \to p_0^{\text{opt}}$ as $k \to \infty$. It follows that $c^{(k)} \to c^{\text{opt}}$ as $k \to \infty$.

It follows from (2.12) that $L(p^{(k+1)}) - L(p^{(k)}) \leq -n \sum_{i=1}^{n} \|p_i^{(k+1)} - p_i^{(k)}\|^2$ for $k \geq 1$. Besides, $\lim_{k\to\infty} L(p^{(k+1)}) - L(p^{(k)}) = 0$. Hence, $\lim_{k\to\infty} \|p_i^{(k+1)} - p_i^{(k)}\| \to 0$ for $i \in \mathcal{N}^+$. $\square$

Theorem 2.8 shows that the sequence $p^{(0)}, p^{(1)}, \ldots, p^{(k)}, \ldots$ converges to the optimal set $\mathbb{O}$, while Theorem 2.10 shows that the sequences $p_0^{(0)}, p_0^{(1)}, \ldots, p_0^{(k)}, \ldots$ and $c^{(0)}, c^{(1)}, \ldots, c^{(k)}, \ldots$ converge to the optimal values p_0^{opt} and c^{opt}, respectively. Since charging profile updates $\left\| p_i^{(k+1)} - p_i^{(k)} \right\|$ vanish as $k \to \infty$ for all EVs, the objective function in (2.8) approximates the electricity cost after a certain number of iterations.

The EV update (2.8) is equivalent to $p_i^{(k+1)} = \operatorname{argmin}_{p_i \in \mathcal{F}_i} \left\| p_i - \left(p_i^{(k)} - c^{(k)} \right) \right\|^2$ for $i \in \mathcal{N}^+$ and $k \geq 0$. Hence, Algorithm 1 can be interpreted as a gradient projection method [15, Ch. 3.3.2].

Remark 2.2 (extensions). *Algorithm 1 converges in the presence of communication delay if the control signal c is scaled down [41]. Besides, Algorithm 1 can be modified to incorporate nonconvex load constraints [45].*

2.4 Case Studies

We verify the optimality of Algorithm 1 through case studies in this section. In particular, we compare Algorithm 1 with the algorithm proposed in [75], in *homogeneous* and *non-homogeneous* cases. Homogeneous cases refer to the cases where EVs have the same $\underline{p}_i$, $\overline{p}_i$, and P_i (all EVs plug in for charging at the same time, have the same deadline, need to charge the same amount of electricity, and have the same maximum charging rate); and non-homogeneous cases refer to cases where $\underline{p}_i$, $\overline{p}_i$, and P_i are not necessarily identical for all EVs. The algorithm proposed in [75] is referred to as MCH—name initials of the authors, and Algorithm 1 is referred to as ODC for convenience.

Uncertainty about the base load and EV arrivals are not considered in this chapter, i.e., it is assumed that all EVs are available for negotiation at the beginning of the scheduling horizon, and that the coordinator predicts the base load exactly. These assumptions are relaxed in Chapter 3. Power network constraints are not considered in this chapter, i.e., voltage and line constraints may be violated and power loss is ignored. This restriction will be addressed in Chapter 4–7.

We choose the average residential load profile in the service area of South California Edison from 20:00 on February 13, 2011 to 9:00 on February 14, 2011 as the base load profile per household. According to the typical EV charging characteristics [56], we assume that an EV can be charged at any rate from 0 to 3.3kW, after it plugs in and before its deadline. We consider the penetration level of 20 EVs in 100 households. The scheduling horizon is from 20:00 to 9:00 the next day, and is divided into 52 slots of 15 minutes. During each time slot, the charging rate of an EV is a constant. The parameters of Algorithm MCH are chosen as $p(x) = 0.15x^2$, $c = 1$, and $\delta = 0.15$. The parameter of Algorithm ODC is chosen as $\epsilon = 10^{-3}$.

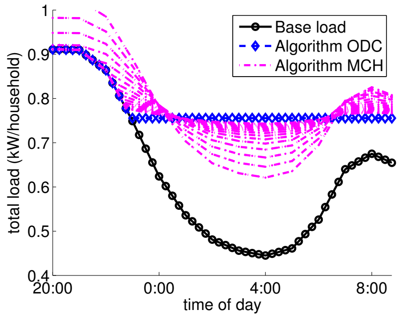


Figure 2.4: All EVs plug in at 20:00 with deadline at 9:00 on the next day, and need to charge 10kWh electricity. Multiple purple dash-dot curves correspond to the total load profiles in different iterations of Algorithm MCH.

Homogeneous case Although Algorithm ODC obtains optimal charging profiles irrespective of the EV specifications, we simulate the homogeneous case to compare it against Algorithm MCH. In this example, all EVs plug in at 20:00 with deadline at 9:00 the next morning, and need to charge 10kWh electricity. Figure 2.4 shows the average total load profiles per household in each iteration of Algorithm ODC and MCH. Both algorithms converge to a valley-filling profile. Moreover, Algorithm ODC converges with a single iteration.

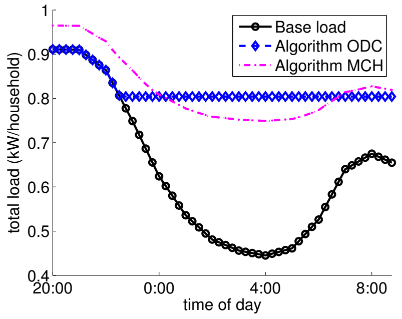


Figure 2.5: All EVs plug in at 20:00 with deadline at 9:00 on the next day, but need to charge different amounts of electricity that is uniformly distributed between 0 and 20kWh.

Different electricity consumption Figure 2.5 shows the average total load profile per household at convergence of Algorithm ODC and MCH in a non-homogeneous case, where EVs need to charge different amounts of electricity. Algorithm ODC still converges to a valley-filling profile in a few iterations, while Algorithm MCH does not. The optimality proof provided in [75] does not seem to extend straightforwardly to non-homogeneous cases.

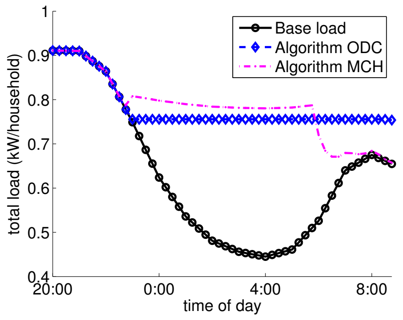


Figure 2.6: All EVs need to charge 10kWh electricity, but plug in at different times (uniformly distributed between 20:00 and 23:00) with different deadlines (uniformly distributed between 6:00 to 9:00 on the next day).

Different plug-in times and deadlines Figure 2.6 shows the average total load profiles per household at convergence of Algorithm ODC and MCH in another non-homogeneous case, where EVs plug in at different times with different deadlines. Algorithm ODC still converges to a valley-filling profile in a few iterations, while Algorithm MCH converges to a total load profile that is significantly lower around 6:00 to 9:00. This is because Algorithm MCH uses a penalty term to limit the deviation of the individual EV charging profiles from the average charging profile. EVs plug in at different times with different deadlines, but are forced to follow the same charging profile. Algorithm ODC changes the "deviation from the average penalty" to the "deviation from the previous iteration penalty". While preserving convergence, Algorithm ODC no longer imposes different EVs to follow a common average charging profile, therefore successfully deals with the heterogeneity in charging deadlines. In fact, Theorem 2.8 implies that Algorithm ODC always obtains optimal charging profiles, even if the EVs plug in at different times, have different deadlines, charge different amounts of electricity, and have different maximum charging rates.

2.5 Conclusions

We have proposed a distributed algorithm that schedules EV charging to optimally fill the valley in electricity demand. The EV charging scheduling problem has been formulated as an optimal control problem, and a gradient projection type distributed algorithm is proposed accordingly to solve the problem. The algorithm is iterative. In each iteration, each EV updates its own charging profile according to the control signal broadcast by a centralized coordinator, and the coordinator guides their updates by altering the control signal. We have proved that the algorithm converges to optimal charging profiles irrespective of the specifications of EVs.

Chapter 3

Real-Time Distributed Load Control

Real-time load control has the potential to compensate for the random fluctuations of renewable generation, by reducing or shifting the power consumption of electric loads in response to generation fluctuations. In this chapter, a real-time distributed algorithm that schedules deferrable loads to reduce the deviation of aggregate load (load minus renewable generation) from some externally specified target is proposed. At every time step, the algorithm minimizes the expected deviation to go with updated predictions on renewable generation and deferrable load arrivals. We prove that suboptimality of the algorithm vanishes quickly as prediction horizon expands. Further, we evaluate the algorithm via trace-based simulations.

Literature A number of distributed deferrable load control algorithms have been proposed in the literature. Some of these algorithms are evaluated based on simulations [4,55,77], while some others have theoretical performance guarantees [41, 75]. In particular, the algorithm proposed in [75] is optimal if electric vehicles are identical, and the algorithm proposed in [41] achieves optimality even if electric vehicles are not identical.

However, the algorithms proposed in [4,41,55,75,77] do not consider the uncertainties in renewable generation and deferrable load arrivals. In practice, only predictions of these quantities are known ahead of time, and the impact of prediction errors can be dramatic, e.g., see Figure 3.3.

Algorithms that consider the uncertainties in renewable generation or/and deferrable load arrivals have also been proposed in the literature. Most of these algorithms are evaluated with simulation-based studies, e.g., [22,31,34], while some are provided with analytic performance guarantees [28,72, 97]. For example, the algorithm proposed in [28] proposes an algorithm that achieves the optimal competitive ratio in the case where renewable generation is precisely known (and constant). The algorithm proposed in [72] also has certain worst-case performance guarantees.

While the algorithms proposed in [28, 72, 97] are analyzed with a “worst-case” perspective, this chapter focuses on the “average-case” perspective to highlight the value of prediction.

Summary *The goal of this chapter is to propose a real-time distributed deferrable load control algorithm that incorporates uncertain predictions on deferrable load arrivals and renewable generation.* In particular, contributions of the chapter are threefold.

First, we *model renewable generation prediction evolution as a Wiener filtering process* [106] (see Section 3.1.1), that is able to model any zero-mean and stationary prediction error.

Second, we *propose a real-time distributed algorithm (Algorithm 2) for deferrable load control in the presence of uncertainties (in Section 3.2), that reduces the deviation of aggregate load (load minus renewable generation) from some externally specified target profile.* At every time step, Algorithm 2 minimizes the expected deviation to go with up-to-date predictions on deferrable load arrivals and renewable generation. A key technique is the introduction of a *pseudo load*, that is simulated at the centralized coordinator to represent future deferrable load arrivals.

Third, we *analyze the expected deviation achieved by Algorithm 2 and provide trace-based simulations.* In particular, the theorems in Section 3.3 characterize the impact of prediction inaccuracies on the expected deviation. As time horizon expands, the expected deviation approaches the optimal value (Corollary 3.7), and the performance gain of Algorithm 2 increases over the optimal open-loop control (Corollary 3.10, 3.11). Trace-based simulations in real-world settings are provided in Section 3.4 to validate the analytic results, highlighting that Algorithm 2 obtains a small suboptimality even under high uncertainties, and improves significantly over the optimal open-loop control.

3.1 Model Overview and Notation

This chapter studies the design and analysis of real-time deferrable load control algorithms to compensate for the random fluctuations of renewable generation. In the following we present a model for this scenario that includes renewable generation, non-deferrable loads, and deferrable loads, which are described in turn.

Throughout, we consider a discrete-time model over a finite time horizon. The time horizon is divided into T time slots of equal length and indexed $1, \dots, T$. In practice, the time horizon could be one day and the length of a time slot could be 10 minutes.

3.1.1 Renewable Generation and Non-Deferrable Load

We aggregate renewable generation and non-deferrable load into a single process, termed the *base load* $b = \{b(\tau)\}_{\tau=1}^{T}$, that is defined as the difference between non-deferrable load and renewable generation. Renewable generation like wind and solar randomly fluctuates and is difficult to predict, therefore b is a stochastic process.

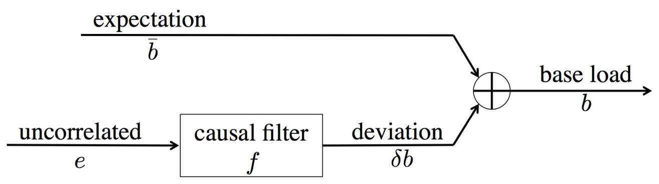


Figure 3.1: Diagram of the notation and structure of the model for base load, i.e., non-deferrable load minus renewable generation.

To model the uncertainty of base load, we use a causal filter based model described as follows, and illustrated in Figure 3.1. In particular, the base load at time τ is modeled as a random deviation $\delta b = \{\delta b(\tau)\}_{\tau=1}^{T}$ around its expectation $\bar{b} = \{\bar{b}(\tau)\}_{\tau=1}^{T}$. The process $\bar{b}$ is specified externally to the model, e.g., from historical data and weather report, and the process $\delta b(\tau)$ is further modeled as an uncorrelated sequence of identically distributed random variables $e = \{e(\tau)\}_{\tau=1}^{T}$ with mean 0 and variance σ^2, passing through a causal filter. Specifically, let $f = \{f(\tau)\}_{\tau=-\infty}^{\infty}$ denote the impulse response of this causal filter and assume that $f(0) = 1$, then $f(\tau) = 0$ for $\tau < 0$ and

$$\delta b(\tau) = \sum_{s=1}^{T} e(s) f(\tau - s), \quad \tau = 1, \ldots, T.$$

Given the model above, at time $t = 1, \ldots, T$, a prediction algorithm can estimate the sequence $e(s)$ for $s = 1, \ldots, t$, and predicts b as[1]

$$b_t(\tau) = \bar{b}(\tau) + \sum_{s=1}^{t} e(s) f(\tau - s), \quad \tau = 1, \ldots, T. \tag{3.1}$$

Note that $b_t(\tau) = b(\tau)$ for $\tau = 1, \ldots, t$ since f is causal.

This model allows for non-stationary base load through the specification of $\bar{b}$ and a broad class of models for uncertainty in the base load via f and e. In particular, two specific filters f that we consider in detail later in the paper are:

Example 3.1. *A filter with finite but flat impulse response, i.e., there exists $\Delta \in (0, T)$ such that*

$$f(t) = \begin{cases} 1 & \text{if } 0 \le t < \Delta \\ 0 & \text{otherwise}; \end{cases}$$

Example 3.2. *A filter with an infinite and exponentially decaying impulse response, i.e., there exists $a \in (0, 1)$ such that*

$$f(t) = \begin{cases} a^t & \text{if } t \ge 0 \\ 0 & \text{otherwise.} \end{cases}$$

[1]This prediction algorithm is a Wiener filter [106].

These two filters provide simple but informative examples for our discussion in Section 3.3.

3.1.2 Deferrable Load

To model deferrable loads we consider a setting where n deferrable loads arrive over the time horizon, each requiring a certain amount of electricity by a given deadline. Further, a real-time algorithm has imperfect information about the arrival times and sizes of these deferrable loads.

More specifically, we assume a total of n deferrable loads and label them in increasing order of their arrival times by $1, \ldots, n$, i.e., load i arrives no later than load $i+1$ for $i = 1, \ldots, n-1$. Further, let $n(t)$ denote the number of loads that arrive before (or at) time t for $t = 1, \ldots, T$ and fix $n(0) := 0$. Thus, load $1, \ldots, n(t)$ arrive before or at time t for $t = 1, \ldots, T$ and $n(T) = n$.

For each deferrable load, its arrival time and deadline, as well as other constraints on its power consumption, are captured via upper and lower bounds on its possible power consumption during each time. Specifically, the power consumption of deferrable load i at time t, $p_i(t)$, must be between given lower and upper bounds $\underline{p}_i(t)$ and $\overline{p}_i(t)$, i.e.,

$$\underline{p}_i(t) \leq p_i(t) \leq \overline{p}_i(t), \quad i = 1, \ldots, n, \; t = 1, \ldots, T. \tag{3.2}$$

These are specified externally to the model. For example, if an electric vehicle plugs in with Level II charging then its power consumption must be within $[0, 3.3]$kW. However, if it is not plugged in (has either not arrived yet or has already departed) then its power consumption is 0kW, i.e., within $[0, 0]$kW. Further, we assume that a deferrable load i must withdraw a fixed amount of energy P_i by its deadline, i.e.,

$$\sum_{t=1}^{T} p_i(t) = P_i, \quad i = 1, \ldots, n. \tag{3.3}$$

Finally, the n deferrable loads arrive randomly throughout the time horizon. Define

$$a(t) := \sum_{i=n(t-1)+1}^{n(t)} P_i \tag{3.4}$$

as the total energy request of all deferrable loads that arrive at time t for $t = 1, \ldots, T$. We assume that $\{a(t)\}_{t=1}^{T}$ is a sequence of independent identically distributed random variables with mean λ and variance s^2. Further, let

$$A(t) := \sum_{\tau=t+1}^{T} a(\tau) \tag{3.5}$$

denote the total energy requested after time t for $t = 1, \ldots, T$.

In summary, at time $t = 1, \ldots, T$, a real-time algorithm has full information about the deferrable loads that have arrived, i.e., $\underline{p}_i$, $\overline{p}_i$, and P_i for $i = 1, \ldots, n(t)$, and knows the expectation of future deferrable load total energy request $\mathbb{E}(A(t))$. However, a real-time algorithm has no other knowledge about deferrable loads that arrive after time t.

3.1.3 The Deferrable Load Control Problem

Recall that the objective of real-time deferrable load control is to compensate for the random fluctuations in renewable generation and non-deferrable load in order to "flatten" the *aggregate load* $p_0 = \{p_0(t)\}_{t=1}^T$, which is defined as

$$p_0(t) = b(t) + \sum_{i=1}^{n} p_i(t), \quad t = 1, \ldots, T. \tag{3.6}$$

In this chapter, we focus on minimizing the *variance* of the aggregate load p_0, $V(p_0)$, as a measure of "flatness", that is defined as

$$V(p_0) = \frac{1}{T}\sum_{t=1}^{T}\left(p_0(t) - \frac{1}{T}\sum_{\tau=1}^{T} p_0(\tau)\right)^2. \tag{3.7}$$

To summarize, the optimal deferrable load control (ODLC) problem is as follows. Let $\mathcal{T} := \{1, \ldots, T\}$, $\mathcal{N} := \{0, \ldots, n\}$, and define $\mathcal{N}^+ := \mathcal{N}\backslash\{0\}$.

$$\begin{aligned}
\textbf{ODLC:}\ \min\ & \sum_{t=1}^{T}\left(p_0(t) - \frac{1}{T}\sum_{\tau=1}^{T} p_0(\tau)\right)^2 \\
\text{over}\ & p_i(t), \qquad i \in \mathcal{N},\ t \in \mathcal{T} \\
\text{s.t.}\ & p_0(t) = b(t) + \sum_{i=1}^{n} p_i(t), \qquad t \in \mathcal{T}; && (3.8a) \\
& \underline{p}_i(t) \le p_i(t) \le \overline{p}_i(t), \qquad i \in \mathcal{N},\ t \in \mathcal{T}; && (3.8b) \\
& \sum_{t=1}^{T} p_i(t) = P_i, \qquad i \in \mathcal{N}. && (3.8c)
\end{aligned}$$

In the above ODLC problem (3.8), the objective is simply T times the variance of aggregate load, $V(p_0)$, and the constraints correspond to equations (3.6), (3.2), and (3.3), respectively. We chose $V(p_0)$ as the objective for ODLC because of its significance for microgrid operators [52]. However, additionally, it is proved in Chapter 2 that the optimal solution does not change if the objective function in (3.8) is replaced by $f(p_0) = \sum_{t\in\mathcal{T}} U(p_0(t))$ where $U : \mathbb{R} \to \mathbb{R}$ is strictly convex. Hence, we can use $V(p_0)$ without loss of generality.

3.2 Algorithm Design

Given the optimal deferrable load control (ODLC) problem defined in (3.8), the first contribution of this chapter is to design an algorithm that solves the ODLC problem in real-time, given uncertain predictions of base load and deferrable loads.

There are two key challenges for the algorithm design. First, the algorithm has access only to uncertain predictions at any given time, i.e., at time t the algorithm only knows deferrable loads 1 to $n(t)$ rather than 1 to n, and only knows the prediction b_t instead of b itself. Second, even if there were no uncertainty in predictions, solving the ODLC problem (3.8) requires significant computational effort when there are a large number of deferrable loads.

Motivated by these challenges, in this section we design a distributed algorithm that provides strong performance guarantees even when there are uncertainties in the predictions. The algorithm we propose is built on Algorithm 1 in Chapter 2, which is distributed but assumes exact knowledge (certainty) about base load and deferrable loads.

In this section, we adapt Algorithm 1 to the setting where there is uncertainty in base load and deferrable load predictions, while maintaining strong performance guarantees. In particular, in this section we assume that at time t, only the prediction b_t is known, not b itself, and only information about deferrable loads 1 to $n(t)$ and the expectation of future energy requests $\mathbb{E}(A(t))$ are known.

Algorithm statement: To adapt Algorithm 1 to deal with uncertainty, we replace the base load b by its prediction b_t in Algorithm 1.

However, dealing with the unavailability of future deferrable load information is trickier. To do this we use a pseudo deferrable load, which is simulated at the coordinator, to represent future deferrable loads. At each time t, we will predict the electricity demand $p_{n+1}(\tau)$ of future deferrable loads at times $\tau = t+1, \ldots, T$, and denote these forecasts by $p_{n+1} := \{p_{n+1}(\tau) \mid \tau = t, \ldots, T\}$ with $p_{n+1}(t) := 0$. As will be seen in Algorithm 2, these forecast are chosen at each time t to minimize the ℓ_2 norm of aggregate load, subject to the following (and other) constraints:

$$\sum_{\tau=t}^{T} p_{n+1}(\tau) = \mathbb{E}(A(t)). \tag{3.9}$$

We also assume that p_{n+1} is point-wise upper and lower bounded by some upper and lower bounds $\overline{p}_{n+1}$ and $\underline{p}_{n+1}$, i.e.,

$$\underline{p}_{n+1}(\tau) \le p_{n+1}(\tau) \le \overline{p}_{n+1}(\tau), \quad \tau = t, \ldots, T. \tag{3.10}$$

Note that $\underline{p}_{n+1}(t) = \overline{p}_{n+1}(t) = 0$. The bounds $\underline{p}_{n+1}$ and $\overline{p}_{n+1}$ should be set according to historical data. Here, for simplicity, we consider them to be $\underline{p}_{n+1}(\tau) = 0$ and $\overline{p}_{n+1}(\tau) = \infty$ for $\tau = t+1, \ldots, T$.

Given the above setup, the coordinator solves the following ODLC(t) problem at every time slot $t = 1, \ldots, T$ to accommodate the availability of only partial information. Let $\mathcal{N}(t) := \{0, 1, \ldots, n(t)\}$

and define $\mathcal{N}^+(t) := \mathcal{N}(t)\backslash\{0\}$ for $t = 1, 2, \ldots, T$. Let $\mathcal{T}(t) := \{t, t+1, \ldots, T\}$ for $t = 1, 2, \ldots, T$.

$$\begin{aligned}
\textbf{ODLC}(t):\ \ &\min \quad \sum_{\tau\in\mathcal{T}(t)} \left(p_0(\tau) - \frac{1}{T-t+1}\sum_{s=t}^{T} p_0(s)\right)^2 \\
&\text{over} \quad p_i(\tau), \qquad i \in \mathcal{N}(t)\cup\{n+1\},\ \tau\in\mathcal{T}(t); \\
&\text{s.t.} \quad p_0(\tau) = b_t(\tau) + \sum_{i=1}^{n(t)} p_i(\tau) + p_{n+1}(\tau), \qquad \tau\in\mathcal{T}(t); && \text{(3.11a)} \\
&\qquad\ \ \underline{p}_i(\tau) \le p_i(\tau) \le \overline{p}_i(\tau), \qquad i\in\mathcal{N}^+(t)\cup\{n+1\},\ \tau\in\mathcal{T}(t); && \text{(3.11b)} \\
&\qquad\ \ \sum_{\tau\in\mathcal{T}(t)} p_i(\tau) = P_i(t), \qquad i\in\mathcal{N}^+(t)\cup\{n+1\} && \text{(3.11c)}
\end{aligned}$$

where $P_i(t) = P_i - \sum_{\tau=1}^{t-1} p_i(\tau)$ is the energy to be consumed at or after time t for $i = 1, \ldots, n(t)$; and $P_{n+1}(t) = \mathbb{E}(A(t))$ is the expected future energy request.

Now, adjusting Algorithm 1 to solve ODLC(t) gives Algorithm 2, which is a real-time, shrinking-horizon control algorithm. Note that if base load prediction is exact (i.e., $b_t = b$ for $t = 1, \ldots, T$) and all deferrable loads arrive at the beginning of the time horizon (i.e., $n(t) = n$ for $t = 1, \ldots, T$), then ODLC(1) reduces to ODLC and Algorithm 2 reduces to Algorithm 1.

Algorithm 2 Deferrable load control with uncertainty

```
Input: At time $t$, the coordinator knows the prediction $b_t$ of base load and the number $n(t)$ of deferrable loads. Each deferrable load $i \in \mathcal{N}^+(t)$ knows its future energy request $P_i(t)$ and power consumption bounds $(\overline{p}_i, \underline{p}_i)$. The utility sets stopping criteria $\epsilon$.
Output: At time $t$, each deferrable load $i \in \mathcal{N}^+(t)$ computes its own $\{p_i(\tau)\}_{\tau=t}^T$.
    At time slot $t = 1, \ldots, T$:
1: each old deferrable load $i \in \mathcal{N}^+(t-1)$ initializes its schedule $\{p_i^{(0)}(\tau)\}_{\tau=t}^T$ as the schedule it computes at time slot $t-1$;
   each new deferrable load $i \in \mathcal{N}^+(t)\backslash\mathcal{N}^+(t-1)$ initializes its schedule $\{p_i^{(0)}(\tau)\}_{\tau=t}^T$ as
       $$p_i^{(0)}(\tau) \leftarrow 0, \quad \tau \in \mathcal{T}(t);$$
2: $k \leftarrow 0$;
3: the coordinator computes pseudo load $p_{n+1}^{(k)}$ as
       $$\begin{aligned} p_{n+1}^{(k)} \ \leftarrow \ \text{argmin} \ & \sum_{\tau \in \mathcal{T}(t)} \left( b_t(\tau) + \sum_{i=1}^{n(t)} p_i^{(k)}(\tau) + p_{n+1}(\tau) \right)^2 \\ \text{s.t.} \quad & \underline{p}_{n+1}(\tau) \le p_{n+1}(\tau) \le \overline{p}_{n+1}(\tau), \qquad \tau \in \mathcal{T}(t); \\ & \sum_{\tau \in \mathcal{T}(t)} p_{n+1}(\tau) = P_{n+1}(t); \end{aligned} \tag{3.12}$$
   the coordinator then calculates control signal $c^{(k)}$ as
       $$c^{(k)}(\tau) \leftarrow \frac{1}{n(t)} p_0^{(k)} = \frac{1}{n(t)} \left( b_t(\tau) + \sum_{i=1}^{n(t)} p_i^{(k)}(\tau) + p_{n+1}^{(k)}(\tau) \right), \qquad \tau \in \mathcal{T}(t);$$
4: if $k \ge 1$ and $\|c^{(k)} - c^{(k-1)}\| \le \epsilon$, the coordinator broadcasts a stopping signal to go to Step 7); otherwise the coordinator broadcasts $c^{(k)}$ to deferrable load $i \in \mathcal{N}^+(t)$;
5: each deferrable load $i \in \mathcal{N}^+(t)$ calculates a new charging profile $\{p_i^{(k+1)}(\tau)\}_{\tau=t}^T$ as
       $$\begin{aligned} p_i^{(k+1)} \ \leftarrow \ \text{argmin} \ & \sum_{\tau \in \mathcal{T}(t)} c^{(k)}(\tau) p_i(\tau) + \frac{1}{2} \sum_{\tau \in \mathcal{T}(t)} \left( p_i(\tau) - p_i^{(k)}(\tau) \right)^2 \\ \text{s.t.} \quad & \underline{p}_i(\tau) \le p_i(\tau) \le \overline{p}_i(\tau), \qquad \tau \in \mathcal{T}(t); \\ & \sum_{\tau \in \mathcal{T}(t)} p_i(\tau) = P_i(t) \end{aligned} \tag{3.13}$$
   and sends $\{p_i^{k+1}(\tau)\}_{\tau=t}^T$ to the coordinator;
6: $k \leftarrow k+1$; go to Step 3);
7: each deferrable load $i \in \mathcal{N}^+(t)$ sets final solution $p_i \leftarrow p_i^{(k)}$ and updates $P_i(t+1) \leftarrow P_i(t) - p_i(t)$;
```

Algorithm convergence results: We provide analytic guarantees on the convergence and optimality of Algorithm 2. In particular, we prove that Algorithm 2 solves ODLC(t) for every time slot $t \in \mathcal{T}$. Let $\mathbb{O}(t)$ denote the set of optimal solutions to ODLC(t), and let

$$\operatorname{dist}\left(\left(p_0, \ldots, p_{n(t)}\right), \mathbb{O}(t)\right) := \min_{(\hat{p}_0, \ldots, \hat{p}_{n(t)}, \hat{p}_{n+1}) \in \mathbb{O}(t)} \| \left(p_0, \ldots, p_{n(t)}\right) - \left(\hat{p}_0, \ldots, \hat{p}_{n(t)}\right) \|$$

denote the distance from a schedule $(p_0, \ldots, p_{n(t)})$ to optimal schedules $\mathbb{O}(t)$.

Theorem 3.1. *For each $t \in \mathcal{T}$, the sequence of deferrable load schedules $p^{(0)}, p^{(1)}, \ldots, p^{(k)}, \ldots$ computed in Algorithm 2 converges to optimal schedules of ODLC(t), i.e.,*

$$\lim_{k \to \infty} \operatorname{dist}\left(p^{(k)}, \mathbb{O}(t)\right) = 0.$$

Proof. Fix an arbitrary $t \in \mathcal{T}$. For any $x = (p_1, \ldots, p_{n(t)}, p_{n+1})$ satisfying (3.11b)–(3.11c), let

$$p_0(x) = b_t + \sum_{i=1}^{n(t)} p_i + p_{n+1}, \qquad L(x) = \langle p_0(x),\ p_0(x) \rangle .$$

As discussed in Section 2.1, x^* minimizes L if and only if $(p_0(x^*), x^*)$ solves ODLC(t).

Follow the proof of Theorem 2.8 to obtain

$$L\left(p_1^{(k+1)}, \ldots, p_{n(t)}^{(k+1)}, p_{n+1}^{(k)}\right) \le L\left(p_1^{(k)}, \ldots, p_{n(t)}^{(k)}, p_{n+1}^{(k)}\right) \tag{3.14}$$

for $k \ge 1$, and the equality is attained if and only if $p_i^{(k+1)} = p_i^{(k)}$ for $i \in \mathcal{N}^+(t)$. It follows from the first order optimality condition for optimizing (3.13) that the equality in (3.14) is attained if and only if

$$\left\langle b_t + \sum_{i=1}^{n(t)} p_i^{(k)} + p_{n+1}^{(k)},\ p_i' - p_i^{(k)} \right\rangle \ge 0$$

for all $i \in \mathcal{N}^+(t)$ and all feasible p_i'.

According to (3.12),

$$L\left(p_1^{(k+1)}, \ldots, p_{n(t)}^{(k+1)}, p_{n+1}^{(k+1)}\right) \le L\left(p_1^{(k+1)}, \ldots, p_{n(t)}^{(k+1)}, p_{n+1}^{(k)}\right) \tag{3.15}$$

for $k \ge 0$, and the equality is attained if and only if $p_{n+1}^{(k+1)} = p_{n+1}^{(k)}$. It follows from the first order optimality condition for optimizing (3.12) that the equality in (3.15) is attained if and only if

$$\left\langle b_t + \sum_{i=1}^{n(t)} p_i^{(k+1)} + p_{n+1}^{(k)},\ p_{n+1}' - p_{n+1}^{(k)} \right\rangle \ge 0$$

for all feasible p_{n+1}'.

It then follows that

$$L\left(p_1^{(k+1)},\ldots,p_{n(t)}^{(k+1)},p_{n+1}^{(k+1)}\right) \le L\left(p_1^{(k)},\ldots,p_{n(t)}^{(k)},p_{n+1}^{(k)}\right),$$

i.e., $L(x^{(k+1)}) \le L(x^{(k)})$, and that the equality is attained if and only if $x^{(k+1)} = x^{(k)}$, and

$$\left\langle b_t + \sum_{i=1}^{n(t)} p_i^{(k)} + p_{n+1}^{(k)},\ p_i' - p_i^{(k)} \right\rangle \ge 0$$

for all feasible $x' = (p_1', \ldots, p_{n(t)}', p_{n+1})$, i.e., $x^{(k)}$ minimizes $L(x)$. Then by Lasalle's Theorem [104], we have $\mathrm{dist}(p^{(k)}, \mathbb{O}(t)) \to 0$ as $k \to \infty$. □

Definition 3.1. *Let $t \in \mathcal{T}$. A feasible schedule $p = (p_0, \ldots, p_{n(t)}, p_{n+1})$ of ODLC(t) is* t-valley-filling, *if there exists $C(t) \in \mathbb{R}$ such that*

$$p_{n+1}(\tau) + \sum_{i=1}^{n(t)} p_i(\tau) = [C(t) - b_t(\tau)]^+, \qquad \tau \in \mathcal{T}(t). \tag{3.16}$$

"t-valley-filling" provides a simple characterization of the optimal solutions of ODLC(t). Specifically, the following corollary follows immediately from Theorem 2.2.

Corollary 3.2. *Let $t \in \mathcal{T}$. If a t-valley-filling schedule $p = (p_0, \ldots, p_{n(t)}, p_{n+1})$ exists, then p solves ODLC(t). Furthermore, all optimal schedules $p^* = (p_0^*, \ldots, p_{n(t)}^*, p_{n+1}^*)$ of ODLC(t) satisfy $p_0^* = p_0$.*

This corollary serves as the basis for the performance analysis we perform in Section 3.3. Note that t-valley-filling schedules tend to exist in cases where the number of deferrable loads is large.

3.3 Performance Evaluation

To this point, we have shown that Algorithm 2 makes optimal decisions with the information available at every time slot, i.e., it solves ODLC(t) at time t for $t = 1, \ldots, T$. However, these decisions are still suboptimal compared to what could be achieved if exact information were available. In this section, our goal is to understand the impact of uncertainty on the performance. In particular, we study 1) how the uncertainties about base load and deferrable loads impact the expected load variance obtained by Algorithm 2, and 2) what is the improvement of using the real-time control provided by Algorithm 2 over using the optimal static control.

Our answers to these questions are below. Throughout, we focus on the special, but practically relevant, case when a t-valley-filling schedule exists at every time $t = 1, \ldots, T$. As we have mentioned previously, when the number of deferrable loads is large this is a natural assumption that holds for practical load profiles. The reason for making this assumption is that it allows us to use the characterization of optimal schedules given in (3.16). In fact, without loss of generality, we further assume $C(t) \geq b_t(\tau)$ for $\tau \in \mathcal{T}(t)$, under which (3.16) implies

$$p_0(t) = C(t) = \frac{1}{T-t+1} \left(\sum_{\tau \in \mathcal{T}(t)} b_t(\tau) + \mathbb{E}(A(t)) + \sum_{i=1}^{n(t)} P_i(t) \right) \tag{3.17}$$

for $t = 1, \ldots, T$. To summarize, equation (3.17) defines the model we use for the performance analysis of Algorithm 2.

The expected load variance of Algorithm 2 We start by calculating the expected load variance, $\mathbb{E}(V)$, of Algorithm 2. The goal is to understand how uncertainty about base load and deferrable loads impacts the load variance. Note that, if there are no base load prediction errors and deferrable loads arrive at the beginning of the time horizon, then Algorithm 2 obtains optimal schedules that have zero load variance. In contrast, when there are base load prediction errors and stochastic deferrable load arrivals, the expected load variance is given by the following theorem.

To state the result, recall that $\{f(t)\}_{t=-\infty}^{\infty}$ is the causal filter modeling the correlation of base load and define $F(t) := \sum_{s=0}^{t} f(s)$ for $t = 0, \ldots, T$.

Theorem 3.3. *Consider an instance where ODLC(t) admits a t-valley-filling solution at every time $t = 1, \ldots, T$. Then, the expected load variance obtained by Algorithm 2 is*

$$\mathbb{E}(V) = \frac{s^2}{T} \sum_{t=2}^{T} \frac{1}{t} + \frac{\sigma^2}{T^2} \sum_{t=0}^{T-1} F^2(t) \frac{T-t-1}{t+1}. \tag{3.18}$$

Theorem 3.3 explicitly and precisely states the interaction of the variability of the predictions of base load (σ) and deferrable loads (s) with the horizon length T. Further, it highlights the role of the impulse response of the causal filter through F. It follows immediately that the expected load variance $\mathbb{E}(V)$ tends to 0 as the uncertainties in base load and deferrable load arrivals vanish, i.e., $\sigma \to 0$ and $s \to 0$. Theorem 3.3 is proved in Appendix 3.C.

Corollary 3.4. *Consider an instance where ODLC(t) admits a t-valley-filling solution at every time $t = 1, \ldots, T$. Then, $\mathbb{E}(V) \to 0$ as $\sigma \to 0$ and $s \to 0$.*

Another remark about Theorem 3.3 is that the two terms in the expression (3.18) for the expected load variance $\mathbb{E}(V)$ correspond to the impact of uncertainties in deferrable load prediction and base load prediction, respectively. In particular, Theorem 3.3 is proved in Appendix 3.C by analyzing these two cases separately and then combining the results. Specifically, the following two lemmas are the key pieces in the proof of Theorem 3.3, but are also of interest in their own right. The lemmas are proved in Appendix 3.A and 3.B, respectively.

Lemma 3.5. *Consider an instance where ODLC(t) admits a t-valley-filling solution at every time $t = 1, \ldots, T$. If there is no base load prediction error, i.e., $b_t = b$ for $t = 1, \ldots, T$, then the expected load variance obtained by Algorithm 2 is*

$$\mathbb{E}(V) = s^2 \frac{\sum_{t=2}^{T} \frac{1}{t}}{T} \approx s^2 \frac{\ln T}{T}.$$

Lemma 3.6. *Consider an instance where ODLC(t) admits a t-valley-filling solution at every time $t = 1, \ldots, T$. If there are no deferrable load arrivals after time 1, i.e., $n(t) = n$ for $t = 1, \ldots, T$, then the expected load variance obtained by Algorithm 2 is*

$$\mathbb{E}(V) = \frac{\sigma^2}{T^2} \sum_{t=0}^{T-1} F^2(t) \frac{T-t-1}{t+1}.$$

Lemma 3.5 highlights that the more uncertainty in deferrable load arrivals, i.e., the larger the s, the larger the expected load variance $\mathbb{E}(V)$. On the other hand, the longer the time horizon T, the smaller the expected load variance $\mathbb{E}(V)$.

Similarly, Lemma 3.6 highlights that a larger base load prediction error, i.e., a larger σ results in a larger expected load variance $\mathbb{E}(V)$. However, if the impulse response $\{f(t)\}_{t=-\infty}^{\infty}$ of the modeling filter of the base load decays fast enough with t, then the following corollary highlights that the expected load variance actually tends to 0 as time horizon T increases despite the uncertainty of base load predictions. The corollary is proved in Appendix 3.D.

Corollary 3.7. *Consider an instance where ODLC(t) admits a t-valley-filling solution at every time $t = 1, \ldots, T$. If there are no deferrable load arrivals after time 1, i.e., $n(t) = n$ for $t = 1, \ldots, T$, and $|f(t)| \sim O(t^{-1/2-\alpha})$ for some $\alpha > 0$, then the expected load variance obtained by Algorithm 2 satisfies $\mathbb{E}(V) \to 0$ as $T \to \infty$.*

The improvement of Algorithm 2 over static control The goal of this section is to quantify the improvement of real-time control via Algorithm 2 over the optimal static (open-loop) control. To be more specific, we compare the expected load variance $\mathbb{E}(V)$ obtained by the real-time control Algorithm 2, with the expected load variance $\mathbb{E}(V')$ obtained by the optimal static control, which only uses base load prediction at the beginning of the time horizon (i.e., $\bar{b}$) to compute deferrable load schedules. We assume $n(t) = n$ for $t = 1, \ldots, T$ in this section since otherwise any static control cannot obtain a schedule for all deferrable loads. Thus, the interpretation of the results that follow is as a quantification of the value of incorporating updated based load predictions into the deferrable load controller.

To begin the analysis, note that $\mathbb{E}(V)$ for this setting is given in Lemma 3.6. Further, it can be proved that the optimal static control is to solve the ODLC problem (3.8) with b replaced by $\bar{b}$ to obtain a deferrable load schedule, and the expected load variance $\mathbb{E}(V')$ it obtains is given by the following lemma, which is proved in Appendix 3.E.

Lemma 3.8. *Consider an instance where ODLC (with b replaced by $\bar{b}$) admits a valley-filling solution. If there is no stochastic load arrival, i.e., $n(t) = n$ for $t = 1, \dots, T$, then the expected load variance $\mathbb{E}(V')$ obtained by the optimal static control is*

$$\mathbb{E}(V') = \frac{\sigma^2}{T^2} \sum_{t=0}^{T-1} \left(T(T-t) f^2(t) - F^2(t) \right).$$

Next, comparing $\mathbb{E}(V)$ and $\mathbb{E}(V')$ given in Lemma 3.6 and 3.8 shows that Algorithm 2 always obtains a smaller expected load variance than the optimal static control. Specifically, we prove the following corollary in Appendix 3.F.

Corollary 3.9. *Consider an instance where ODLC (with b replaced by $\bar{b}$) admits a valley-filling solution and ODLC(t) admits a t-valley-filling solution at every time $t = 1, \dots, T$. If there is no deferrable load arrival after time 1, i.e., $n(t) = n$ for $t = 1, \dots, T$, then*

$$\mathbb{E}(V') - \mathbb{E}(V) = \frac{\sigma^2}{T} \sum_{t=1}^{T} \frac{1}{2t} \sum_{m=0}^{t-1} \sum_{n=0}^{t-1} \left(f(m) - f(n) \right)^2 \geq 0.$$

Corollary 3.9 highlights that Algorithm 2 is guaranteed to obtain a smaller expected load variance than the optimal static control. The next step is to quantify how much smaller $\mathbb{E}(V)$ is in comparison with $\mathbb{E}(V')$.

To do this we compute the ratio $\mathbb{E}(V')/\mathbb{E}(V)$. Unfortunately, the general expression for the ratio is too complex to provide insight, so we consider two representative cases for the impulse response $f(t)$ of the causal filter in order to obtain insights. Specifically, we consider Example 3.1 and 3.2 in Section 3.1.1. Briefly, in Example 3.1 $f(t)$ is finite and in Example 3.2 $f(t)$ is infinite but decays exponentially in t. For these two cases, the ratio $\mathbb{E}(V')/\mathbb{E}(V)$ is summarized in the following corollaries, which are proved in Appendix 3.G and 3.H.

Corollary 3.10. *Consider an instance where ODLC (with b replaced by $\bar{b}$) admits a valley-filling solution and ODLC(t) admits a t-valley-filling solution at every time $t = 1, \dots, T$. If there is no deferrable load arrival after time 1, i.e., $n(t) = n$ for $t = 1, \dots, T$, and there exists $\Delta > 0$ such that*

$$f(t) = \begin{cases} 1 & \text{if } 0 \leq t < \Delta \\ 0 & \text{otherwise,} \end{cases}$$

then

$$\frac{\mathbb{E}(V')}{\mathbb{E}(V)} = \frac{T/\Delta}{\ln(T/\Delta)} \left(1 + O\left(\frac{1}{\ln(T/\Delta)} \right) \right).$$

Corollary 3.11. *Consider an instance that ODLC (with b replaced by $\bar{b}$) admits a valley-filling solution and ODLC(t) admits a t-valley-filling solution at every time $t = 1, \ldots, T$. If there is no deferrable load arrival after time 1, i.e., $n(t) = n$ for $t = 1, \ldots, T$, and there exists $a \in (0, 1)$ such that*

$$f(t) = \begin{cases} a^t & \textit{if } t \geq 0 \\ 0 & \textit{otherwise,} \end{cases}$$

then

$$\frac{\mathbb{E}(V')}{\mathbb{E}(V)} = \frac{1-a}{1+a} \frac{T}{\ln T} \left(1 + O\left(\frac{\ln \ln T}{\ln T}\right)\right).$$

Corollary 3.10 highlights that, in the case where f is finite, if we define $\lambda = T/\Delta$ as the ratio of time horizon to filter length, then the load reduction roughly scales as $\lambda/\ln(\lambda)$. Thus, the longer the time horizon is in comparison to the filter length, the larger the expected load variance reduction we obtain from using Algorithm 2 as compared with the optimal static control.

Similarly, Corollary 3.11 highlights that, in the case where f is infinite and exponentially decaying, the expected load variance reduction scales with T as $T/\ln T$ with coefficient $(1-a)/(1+a)$. Thus, the smaller a is, which means the faster f dies out, the more load variance reduction we obtain by using real-time control. This is similar to having a smaller Δ in the previous case.

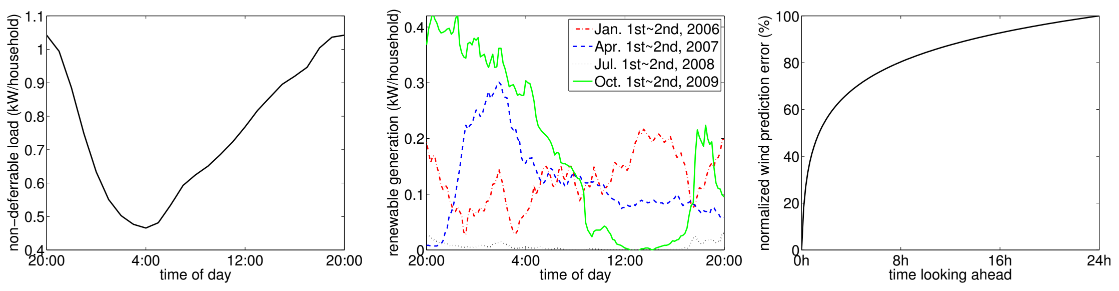
Remark 3.1. *It is shown in [27] that the typical performance of Algorithm 2 is similar to the average performance of Algorithm 2.*

3.4 Experimental results

In this section we use trace-based experiments in order to explore the generality of the analytic results in the previous section. In particular, the results in the previous section precisely characterize the expected load variance resulting from Algorithm 2 as a function of prediction uncertainties and quantify the improvement from the application of Algorithm 2 over the optimal static (open-loop) controller. However, the analytic results necessarily make assumptions on the statistics of the uncertainties. Therefore, it is important to assess the performance of the algorithm using data from real-world scenarios.

3.4.1 Experimental setup

The numerical experiments we perform use a time horizon of 24 hours, from 20:00 to 20:00 on the following day. The time slot length is 10 minutes, which is the granularity of the data we have obtained about renewable generation.



(a) non-deferrable load (b) wind generation (c) prediction error over time

Figure 3.2: Illustration of the traces used in the experiments. (a) shows the average residential load in the service area of Southern California Edison in 2012. (b) shows the total wind power generation of the Alberta Electric System Operator scaled to represent 20% penetration. (c) shows the normalized root-mean-square wind prediction error as a function of the time looking ahead for the model used in the experiments.

Base load Recall that base load is a combination of non-deferrable load and renewable generation. The non-deferrable load traces used in the experiments come from the average residential load in the service area of Southern California Edison in 2012 [93]. In the simulations, we assume that the non-deferrable load is precisely known so that uncertainties in the base load only come from renewable generation. In particular, non-deferrable load over the time horizon of a day is taken to be the average over the 366 days in 2012 as in Figure 3.2(a), and assumed to be known to the utility at the beginning of the time horizon. In practice, non-deferrable load at the substation feeder level can be predicted within 1–3% root-mean-square error looking 24 hours ahead [38].

The renewable generation traces we use come from the 10-minute historical data for total wind power generation of the Alberta Electric System Operator from 2004 to 2009 [6]. In the simulations, we scale the wind power generation so that its average over the 6 years corresponds to a number of penetration levels in the range between 5% and 30%, and pick the wind power generation of a randomly chosen day as the renewable generation during each run. Figure 3.2(b) shows the wind power generation for four representative days, one for each season, after scaling to 20% penetration.

We assume that the renewable generation is not precisely known until it is realized, but that a prediction of the generation, which improves over time, is available to the utility. The modeling of prediction evolution over time is according to a martingale forecasting process [50, 51], which is a standard model for an unbiased prediction process that improves over time.

Specifically, the prediction model is as follows: For wind generation $w(\tau)$ at time τ, the prediction error $w_t(\tau) - w(\tau)$ at time $t < \tau$ is the sum of a sequence of independent random variables $n_s(\tau)$ as

$$w_t(\tau) = w(\tau) + \sum_{s=t+1}^{\tau} n_s(\tau), \quad 0 \le t < \tau \le T.$$

Here $w_0(\tau)$ is the wind prediction without any observation, i.e., the expected wind generation $\bar{w}(\tau)$

at the beginning of the time horizon (used by static control).

The random variables $n_s(\tau)$ are assumed to be Gaussian with mean 0. Their variances are chosen as

$$\mathbb{E}(n_s^2(\tau)) = \frac{\sigma^2}{\tau - s + 1}, \quad 1 \le s \le \tau \le T$$

where $\sigma > 0$ is such that the root-mean-square prediction error $\sqrt{\mathbb{E}(w_0(T) - w(T))^2}$ looking T time slots (i.e., 24 hours) ahead is 0%–22.5% of the nameplate wind generation capacity.[2] According to this choice of the variances of $n_s(\tau)$, root-mean-square prediction error only depends on how far ahead the prediction is, as in Figure 3.2(c) in particular. This choice is motivated by [48].

Deferrable loads For simplicity, we consider the hypothetical case where all deferrable loads are electric vehicles. Since historical data for electric vehicle usage is not available, we are forced to use synthetic traces for this component of the experiments. Specifically, in the simulations the electric vehicles are considered to be identical, each requests 10kWh electricity by a deadline 8 hours after it arrives, and each must consume power at a rate within $[0, 3.3]$kW after it arrives and before its deadline.

In the simulations, the arrival process starts at 20:00 and ends at 12:00 the next day so that the deadlines of all electric vehicles lie within the time horizon of 24 hours. In each time slot during the arrival process, we assume that the number of arriving electric vehicles is uniformly distributed in $[0.8\lambda, 1.2\lambda]$, where λ is chosen so that electric vehicles (on average) account for 5%–30% of the non-deferrable loads.

Uncertainty about deferrable load arrivals is captured as follows. The prediction $\mathbb{E}(A(t))$ of future deferrable load total energy request is simply the arrival rate λ times the length of the rest of the arrival process $T' - t$ where T' is the end of the arrival process (12:00), i.e.,

$$\mathbb{E}(A(t)) = \lambda(T' - t), \quad t = 1, \ldots, T'.$$

If $t > T'$, i.e., the deferrable load arrival process has ended, then $\mathbb{E}(A(t)) = 0$.

Baselines for comparison Our goal in the simulations is to contrast the performance of Algorithm 2 with a number of common benchmarks to tease apart the impact of real-time control and the impact of different forms of uncertainty. To this end, we consider four controllers in our experiments:

(i) *Offline optimal control:* The controller has full knowledge about the base load and deferrable loads, and solves the ODLC problem offline. It is not realistic in practice, but serves as a benchmark for the other controllers since offline optimal control obtains the smallest possible load variance.

[2]Average wind generation is 15% of the nameplate capacity, so the root-mean-square prediction error looking T time slots ahead is 0%–150% the average wind generation.

(ii) *Static control with exact deferrable load arrival information*: The controller has full knowledge about deferrable loads (including those that have not arrived), but uses only the prediction of base load that is available at the beginning of the time horizon to compute a deferrable load schedule that minimizes the expected load variance. This static control is still unrealistic since a deferrable load is known only after it arrives. But, this controller corresponds to what is considered in prior works, e.g., [41, 45, 75].

(iii) *Real-time control with exact deferrable load arrival information.* The controller has full knowledge about deferrable loads (including those that have not arrived), and uses the prediction of base load that is available at the current time slot to update the deferrable load schedule by minimizing the expected load variance to go, i.e., Algorithm 2 with $n(t) = n$ for $t = 1, \ldots, T$. The control is unrealistic since a deferrable load is known only after it arrives; however, it provides the natural comparison for case (ii) above.

(iv) *Real-time control without exact deferrable load arrival information, i.e., Algorithm 2.* This corresponds to the realistic scenario where only predictions are available about future deferrable loads and base loads. The comparison with case (iii) highlights the impact of the uncertainty in deferrable load arrival.

The performance measure that we show in all plots is the "suboptimality" of the controllers, which we define as

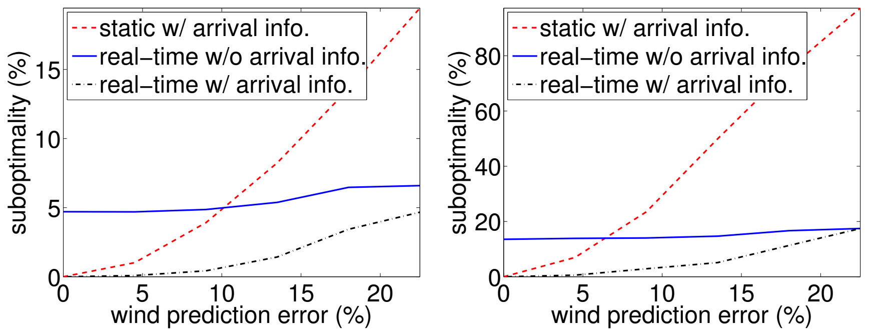
$$\eta := \frac{V - V^{\text{opt}}}{V^{\text{opt}}},$$

where V is the load variance obtained by the controller and V^{opt} is the load variance obtained by the offline optimal, i.e., case (i) above. Thus, the lines in the figures correspond to cases (ii)-(iv).

3.4.2 Experimental results

Our experimental results focus on two main goals: (i) understanding the impact of prediction accuracy on the expected load variance obtained by deferrable load control algorithms, and (ii) contrasting the real-time (closed-loop) control of Algorithm 2 with the optimal static (open-loop) controller. We focus on the impact of three key factors: wind prediction error, the penetration of deferrable load, and the penetration of renewable energy.

The impact of prediction error To study the impact of prediction error, we fix the penetration of both renewable generation (wind) and deferrable loads at 10% of non-deferrable load, and simulate the load variance obtained under different levels of root-mean-square wind prediction errors (0%–22.5% of the nameplate capacity looking 24 hours ahead). The results are summarized in Figure 3.3(a). It is not surprising that suboptimality of both the static and the real-time controllers that have exact information about deferrable load arrivals is zero when the wind prediction error is 0, since there is no uncertainty for these controllers in this case.



(a) Wind and deferrable load penetration are both 10%.
(b) Wind and deferrable load penetration are both 20%.

Figure 3.3: Illustration of the impact of wind prediction error on suboptimality of load variance.

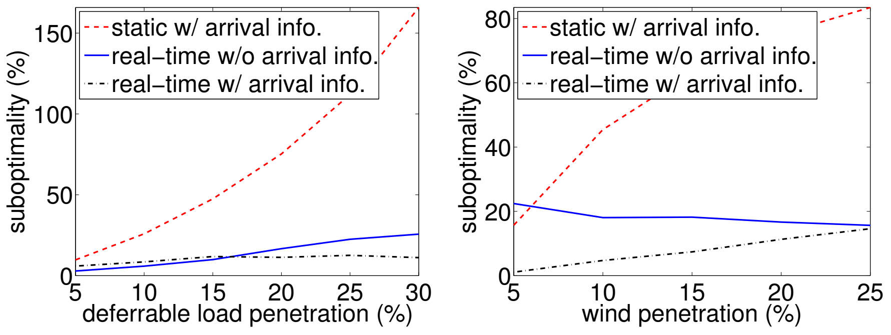
As prediction error increases, the suboptimality of both the static and the real-time control increases. However, notably, the suboptimality of real-time control grows much more slowly than that of static control, and remains small (<4.7%) if deferrable load arrivals are known, over the whole range 0%–22.5% of wind prediction error. At 22.5% prediction error, the suboptimality of static control is 4.2 times that of real-time control. This highlights that real-time control mitigates the influence of imprecise base load prediction over time.

Moving to the scenario where deferrable load arrivals are not known precisely, we see that the impact of this inexact information is less than 6.6% of the optimal variance. However, real-time control yields a load variance that is surprisingly resilient to the growth of wind prediction error, and eventually beats the optimal static control at around 10% wind prediction error, even though the optimal static control has exact knowledge of deferrable loads and the adaptive control does not.

As prediction error increases, the suboptimality of the real-time control with or without deferrable load arrival information gets close, i.e., the benefit of knowing additional information on future deferrable load arrivals vanishes as base load uncertainty increases. This is because the additional information is used to overfit the base load prediction error.

The same comparison is shown in Figure 3.3(b) for the case where renewable and deferrable load penetration are both 20%. Qualitatively the conclusions are the same, however at this higher penetration the contrast between the resilience of adaptive control and static control is magnified, while the benefit of knowing deferrable load arrival information is minified. In particular, real-time control without arrival information beats static control with arrival information, at a lower (around 7%) wind prediction error, and knowing deferrable load arrival information does not reduce suboptimality of real-time control with 22.5% wind prediction error.

The impact of deferrable load penetration Next, we look at the impact of deferrable load penetration on the performance of the various controllers. To do this, we fix the wind penetration level to be 20% and wind prediction error looking 24 hours ahead to be 18%, and simulate the load variance obtained under different deferrable load penetration levels (5%–30%). The results are summarized in Figure 3.4(a).



(a) Impact of deferrable load penetration (b) Impact of wind penetration

Figure 3.4: Suboptimality of load variance as a function of (a) deferrable load penetration and (b) wind penetration. In (a) the wind penetration is 20% and in (b) the deferrable load penetration is 20%. In both, the wind prediction error looking 24 hours ahead is 18%.

Not surprisingly, if future deferrable loads are known and uncertainty only comes from base load prediction error, then the suboptimality of real-time control is very small (<11.2%) over the whole range 5%–30% of deferrable load penetration, while the suboptimality of static control increases with deferrable load penetration, up to as high as 166% (14.9 times that of real-time control) at 30% deferrable load penetration.

However, without knowing future deferrable loads, the suboptimality of real-time control increases with the deferrable load penetration. This is because a larger amount of deferrable loads introduces larger uncertainties in deferrable load arrivals. But the suboptimality remains smaller than that of static control over the whole range of 5%–30% of deferrable load penetration. The highest suboptimality 25.7% occurs at 30% deferrable load penetration, and is less than 1/6 of the suboptimality of static control, which assumes exact deferrable load arrival information.

The impact of renewable penetration Finally, we study the impact of renewable penetration. To do this we fix the deferrable load penetration level to be 20% and the wind prediction error looking 24 hours ahead to be 18%, and simulate the load variance obtained by the four test cases under different wind penetration levels (5%–25%). The results are summarized in Figure 3.4(b).

A key observation is that if future deferrable loads are known and uncertainty only comes from base load prediction error, then the suboptimality of real-time control grows much slower than that of static control, as wind penetration level increases. As explained before, this highlights that real-time control mitigates the impact of base load prediction error over time. In fact, the suboptimality of real-time control is small (<15%) over the whole range of 5%–25% of wind penetration levels. Of course, without knowledge of future deferrable loads, the suboptimality of real-time control becomes bigger. However, it still eventually outperforms the optimal static controller at around 6% wind penetration, despite the fact that the optimal static controller is using exact information about deferrable loads.

3.5 Concluding remarks

We have proposed a real-time algorithm for distributed deferrable load control that can schedule a large number of deferrable loads to compensate for the random fluctuations in renewable generation. At any time, the algorithm incorporates updated predictions about deferrable loads and renewable generation to minimize the expected load variance to go. Further, we have derived an explicit expression for the expected aggregate load variance obtained by the algorithm by modeling the base load prediction updates as a Wiener filtering process. Additionally, we have highlighted the importance of the expression by using it to evaluate the improvement of real-time control over static control. Interestingly, the sub-optimality of static control is $O(T/\ln T)$ times that of real-time control in two representative cases of base load prediction updates. The qualitative insights from the analytic results were validated using trace-based simulations, which confirm that the algorithm has significantly smaller sub-optimality than the optimal static control.

Appendix

3.A Proof of Lemma 3.5

When $b_t = b$ and $\mathbb{E}(a(t)) = \lambda$ for $t = 1, \ldots, T$, the model (3.17) for Algorithm 2 reduces to

$$p_0(t) = \frac{1}{T-t+1}\left(\sum_{\tau=t}^{T} b(\tau) + \lambda(T-t) + \sum_{i=1}^{n(t)} P_i(t)\right) \tag{3.19}$$

for $t = 1, \ldots, T$. Then

$$\begin{aligned}
(T-t+1)p_0(t) &= \sum_{\tau=t}^{T} b(\tau) + \lambda(T-t) + \sum_{i=1}^{n(t)} P_i(t), \\
(T-t+2)p_0(t-1) &= \sum_{\tau=t-1}^{T} b(\tau) + \lambda(T-t+1) + \sum_{i=1}^{n(t-1)} P_i(t-1)
\end{aligned}$$

for $t = 2, \ldots, T$. Subtract the two equations and simplify using the fact that

$$b(t-1) + \sum_{i=1}^{n(t-1)} (P_i(t-1) - P_i(t)) = b(t-1) + \sum_{i=1}^{n(t-1)} p_i(t-1) = p_0(t-1)$$

and the definition of $a(t)$ to obtain

$$p_0(t) - p_0(t-1) = \frac{1}{T-t+1}(a(t) - \lambda)$$

for $t = 2, \ldots, T$. Substituting $t = 1$ into (3.19), it can be verified that $p_0(1) = \lambda + \sum_{\tau=1}^{T} b(\tau)/T + (a(1) - \lambda)/T$, therefore

$$p_0(t) = \lambda + \frac{1}{T}\sum_{\tau=1}^{T} b(\tau) + \sum_{\tau=1}^{t} \frac{1}{T-\tau+1}(a(\tau) - \lambda)$$

for $t = 1, \ldots, T$. The average aggregate load is

$$u = \frac{1}{T}\sum_{t=1}^{T} p_0(t) = \lambda + \frac{1}{T}\left(\sum_{\tau=1}^{T} b(\tau) + \sum_{\tau=1}^{T}(a(\tau)-\lambda)\right).$$

Hence,

$$\begin{aligned}
\mathbb{E}(p_0(t)-u)^2 &= \mathbb{E}\left(\sum_{\tau=1}^{t}\frac{1}{T-\tau+1}(a(\tau)-\lambda) - \frac{1}{T}\sum_{\tau=1}^{T}(a(\tau)-\lambda)\right)^2 \\
&= \mathbb{E}\left(\sum_{\tau=1}^{t}\frac{\tau-1}{T(T-\tau+1)}(a(\tau)-\lambda) - \frac{1}{T}\sum_{\tau=t+1}^{T}(a(\tau)-\lambda)\right)^2 \\
&= \frac{s^2}{T^2}\left(\sum_{\tau=1}^{t}\frac{(\tau-1)^2}{(T-\tau+1)^2} + T - t\right)
\end{aligned}$$

for $t = 1, \ldots, T$. The last equality holds because $(a(\tau)-\lambda)$ are independent for all τ and each of them have mean zero and variance s^2. It follows that

$$\begin{aligned}
\mathbb{E}(V) &= \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}(d(t)-u)^2 \\
&= \frac{s^2}{T^3}\left(\sum_{t=1}^{T}\sum_{\tau=1}^{t}\frac{(\tau-1)^2}{(T-\tau+1)^2} + \sum_{t=1}^{T}(T-t)\right) \\
&= \frac{s^2}{T^3}\left(\sum_{\tau=1}^{T}\frac{(\tau-1)^2}{T-\tau+1} + \sum_{t=1}^{T}(T-t)\right) \\
&= \frac{s^2}{T^3}\left(\sum_{t=1}^{T}\frac{(T-t)^2}{t} + \sum_{t=1}^{T}\frac{(T-t)t}{t}\right) \\
&= s^2\frac{\sum_{t=2}^{T}\frac{1}{t}}{T} \approx s^2\frac{\ln T}{T}.
\end{aligned}$$

3.B Proof of Lemma 3.6

In the case where no deferrable arrival after $t = 1$, i.e., $n(t) = n$ for $t = 1, \ldots, T$, the model (3.17) for Algorithm 2 reduces to

$$(T-t+1)p_0(t) = \sum_{\tau=t}^{T} b_t(\tau) + \sum_{i=1}^{n} P_i(t) \tag{3.20}$$

for $t = 1, \ldots, T$. Substitute t by $t-1$ to obtain

$$(T-t+2)p_0(t-1) = \sum_{\tau=t-1}^{T} b_{t-1}(\tau) + \sum_{i=1}^{n} P_i(t-1)$$

for $t = 2, \ldots, T$. Subtract the two equations to obtain

$$\begin{aligned}(T-t+1)p_0(t) - (T-t+2)p_0(t-1) &= \sum_{\tau=t}^{T} e(t)f(\tau - t) - b(t-1) - \sum_{i=1}^{n} p_i(t-1) \\ &= e(t)F(T-t) - p_0(t-1),\end{aligned}$$

which implies

$$p_0(t) - p_0(t-1) = \frac{1}{T-t+1} e(t)F(T-t)$$

for $t = 2, \ldots, T$. Substituting $t = 1$ into (3.20) and recalling the definition of b_t in (3.1), it can be verified that

$$p_0(1) = \frac{1}{T}\left(\sum_{i=1}^{n} P_i + \sum_{\tau=1}^{T} \bar{b}(\tau)\right) + \frac{1}{T} e(1)F(T-1).$$

Therefore,

$$p_0(t) = \frac{1}{T}\left(\sum_{i=1}^{n} P_i + \sum_{\tau=1}^{T} \bar{b}(\tau)\right) + \sum_{\tau=1}^{t} \frac{1}{T-\tau+1} e(\tau)F(T-\tau)$$

for $t = 1, \ldots, T$. The average aggregate load is

$$u = \frac{1}{T}\left(\sum_{i=1}^{n} P_i + \sum_{t=1}^{T} \bar{b}(t)\right) + \frac{1}{T}\sum_{\tau=1}^{T} e(\tau)F(T-\tau).$$

Hence,

$$\begin{aligned}\mathbb{E}(p_0(t) - u)^2 &= \mathbb{E}\left(\sum_{\tau=1}^{t} \frac{1}{T-\tau+1} e(\tau)F(T-\tau) - \sum_{\tau=1}^{T} \frac{1}{T} e(\tau)F(T-\tau)\right)^2 \\ &= \mathbb{E}\left(\sum_{\tau=1}^{t} \frac{\tau-1}{T(T-\tau+1)} e(\tau)F(T-\tau) - \sum_{\tau=t+1}^{T} \frac{1}{T} e(\tau)F(T-\tau)\right)^2 \\ &= \frac{\sigma^2}{T^2}\left(\sum_{\tau=1}^{t} \frac{(\tau-1)^2}{(T-\tau+1)^2} F^2(T-\tau) + \sum_{\tau=t+1}^{T} F^2(T-\tau)\right)\end{aligned}$$

for $t = 1, \ldots, T$. The last equality holds because $e(\tau)$ are uncorrelated random variables with mean zero and variance σ^2. It follows that

$$\begin{aligned}
\mathbb{E}(V) &= \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}(p_0(t)-u)^2 \\
&= \frac{\sigma^2}{T^3}\sum_{t=1}^{T}\left(\sum_{\tau=1}^{t}\frac{(\tau-1)^2}{(T-\tau+1)^2}F^2(T-\tau)+\sum_{\tau=t+1}^{T}F^2(T-\tau)\right) \\
&= \frac{\sigma^2}{T^3}\sum_{\tau=1}^{T}F^2(T-\tau)\frac{(\tau-1)^2}{T-\tau+1}+\frac{\sigma^2}{T^3}\sum_{\tau=2}^{T}(\tau-1)F^2(T-\tau) \\
&= \frac{\sigma^2}{T^2}\sum_{\tau=1}^{T}F^2(T-\tau)\frac{\tau-1}{T-\tau+1} = \frac{\sigma^2}{T^2}\sum_{t=0}^{T-1}F^2(t)\frac{T-t-1}{t+1}.
\end{aligned}$$

3.C Proof of Theorem 3.3

Similar to the proof of Lemma 3.5 and 3.6, use the model (3.17) to obtain

$$p_0(t) = \lambda + \frac{1}{T}\sum_{\tau=1}^{T}\bar{b}(\tau) + \sum_{\tau=1}^{t}\frac{1}{T-\tau+1}\left(e(\tau)F(T-\tau)+a(\tau)-\lambda\right)$$

for $t = 1, \ldots, T$ and

$$u = \lambda + \frac{1}{T}\sum_{\tau=1}^{T}\bar{b}(\tau) + \sum_{\tau=1}^{T}\frac{1}{T}\left(e(\tau)F(T-\tau)+a(\tau)-\lambda\right).$$

Hence,

$$\begin{aligned}
\mathbb{E}\left[p_0(t)-u\right]^2 &= \mathbb{E}\left[\sum_{\tau=1}^{t}\frac{1}{T-\tau+1}\left(e(\tau)F(T-\tau)+a(\tau)-\lambda\right)-\sum_{\tau=1}^{T}\frac{1}{T}\left(e(\tau)F(T-\tau)+a(\tau)-\lambda\right)\right]^2 \\
&= \mathbb{E}\left[\sum_{\tau=1}^{t}\frac{1}{T-\tau+1}e(\tau)F(T-\tau)-\sum_{\tau=1}^{T}\frac{1}{T}e(\tau)F(T-\tau)\right]^2 \\
&\quad +\mathbb{E}\left[\sum_{\tau=1}^{t}\frac{1}{T-\tau+1}\left(a(\tau)-\lambda\right)-\sum_{\tau=1}^{T}\frac{1}{T}\left(a(\tau)-\lambda\right)\right]^2.
\end{aligned}$$

The first term is exactly that in Lemma 3.6, and the second term is exactly that in Lemma 3.5. Hence, the expected load variance is

$$\mathbb{E}(V) = \frac{\sigma^2}{T^2}\sum_{t=0}^{T-1}F^2(t)\frac{T-t-1}{t+1} + \frac{s^2}{T}\sum_{t=2}^{T}\frac{1}{t}.$$

3.D Proof of Corollary 3.7

If $|f(t)| \sim O(t^{-1/2-\alpha})$ for some $\alpha > 0$, then $|f(t)| \leq Ct^{-1/2-\alpha}$ for some $C > 0$ and all $t \geq 1$. Without loss of generality, assume that $0 < \alpha < 1/2$ and $C \geq (1-2\alpha)/(1+2\alpha)$. Then $F(0) = 1$ and

$$|F(t)| = \left|\sum_{\tau=0}^{t} f(\tau)\right| \leq 1 + \sum_{\tau=1}^{t} C\tau^{-1/2-\alpha} \;\leq\; 1 + C + \int_1^t C\tau^{-1/2-\alpha} d\tau \leq \frac{2C}{1-2\alpha} t^{1/2-\alpha}$$

for $t = 1, \ldots, T$. The last inequality holds because $C \geq (1-2\alpha)/(1+2\alpha)$. Therefore it follows from Lemma 3.6 that

$$\begin{aligned}
\mathbb{E}(V) &\leq \frac{\sigma^2}{T} \sum_{s=0}^{T-1} F^2(s) \frac{1}{s+1} \\
&\leq \frac{\sigma^2}{T} + \frac{\sigma^2}{T} \sum_{s=1}^{T-1} \frac{4C^2}{(1-2\alpha)^2} s^{1-2\alpha} \frac{1}{s+1} \\
&\leq \frac{\sigma^2}{T} + \frac{\sigma^2}{T} \frac{4C^2}{(1-2\alpha)^2} \sum_{s=1}^{T-1} \frac{1}{s^{2\alpha}} \\
&\leq \frac{\sigma^2}{T} + \frac{\sigma^2}{T} \frac{4C^2}{(1-2\alpha)^2} + \frac{\sigma^2}{T} \frac{4C^2}{(1-2\alpha)^2} \int_1^{T-1} \frac{1}{s^{2\alpha}} ds \\
&\leq \frac{\sigma^2}{T} + \frac{4\sigma^2 C^2}{(1-2\alpha)^2 T} + \frac{4\sigma^2 C^2}{(1-2\alpha)^3 T^{2\alpha}}.
\end{aligned}$$

Hence, $\mathbb{E}(V) \to 0$ as $T \to \infty$.

3.E Proof of Lemma 3.8

The aggregate load d obtained by the optimal static algorithm is

$$\begin{aligned}
p_0(t) &= \frac{1}{T}\left(\sum_{i=1}^{n} P_i + \sum_{\tau=1}^{T} \bar{b}(\tau)\right) - \bar{b}(t) + b(t) \\
&= \frac{1}{T}\left(\sum_{i=1}^{n} P_i + \sum_{\tau=1}^{T} \bar{b}(\tau)\right) + \sum_{\tau=1}^{T} e(\tau) f(t-\tau)
\end{aligned}$$

for $t = 1, \ldots, T$. Hence,

$$\begin{aligned}
\mathbb{E}(p_0(t) - u)^2 &= \mathbb{E}\left(\sum_{\tau=1}^{T} e(\tau)\left(f(t-\tau) - \frac{1}{T} F(T-\tau)\right)\right)^2 \\
&= \frac{\sigma^2}{T^2} \sum_{\tau=1}^{T} T^2 f^2(t-\tau) - 2T f(t-\tau) F(T-\tau) + F^2(T-\tau)
\end{aligned}$$

for $t = 1, \dots, T$. It follows that

$$\begin{aligned}
\mathbb{E}(V') &= \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}(p_0(t)-u)^2 \\
&= \frac{\sigma^2}{T}\sum_{t=1}^{T}\sum_{\tau=1}^{T} f^2(t-\tau) - \frac{2\sigma^2}{T^2}\sum_{\tau=1}^{T}F(T-\tau)\sum_{t=1}^{T}f(t-\tau) + \frac{\sigma^2}{T^2}\sum_{\tau=1}^{T}F^2(T-\tau) \\
&= \frac{\sigma^2}{T}\sum_{t=1}^{T}\sum_{\tau=0}^{t-1} f^2(\tau) - \frac{\sigma^2}{T^2}\sum_{\tau=1}^{T}F^2(T-\tau) \\
&= \frac{\sigma^2}{T}\sum_{\tau=0}^{T-1}(T-\tau)f^2(\tau) - \frac{\sigma^2}{T^2}\sum_{\tau=0}^{T-1}F^2(\tau) \\
&= \frac{\sigma^2}{T^2}\sum_{t=0}^{T-1}\left(T(T-t)f^2(t) - F^2(t)\right).
\end{aligned}$$

3.F Proof of Corollary 3.9

Corollary 3.9 follows from Lemma 3.6 and Lemma 3.8 and the definition of F:

$$\begin{aligned}
\mathbb{E}(V') - \mathbb{E}(V) &= \frac{\sigma^2}{T}\sum_{t=0}^{T-1}(T-t)f^2(t) - \frac{\sigma^2}{T^2}\sum_{t=0}^{T-1}F^2(t)\left(1+\frac{T-t-1}{t+1}\right) \\
&= \frac{\sigma^2}{T}\sum_{n=0}^{T}(T-n)f^2(n) - \frac{\sigma^2}{T}\sum_{t=0}^{T-1}\frac{1}{t+1}F^2(t) \\
&= \frac{\sigma^2}{T}\sum_{n=0}^{T}\sum_{t=n+1}^{T} f^2(n) - \frac{\sigma^2}{T}\sum_{t=0}^{T-1}\frac{1}{t+1}F^2(t) \\
&= \frac{\sigma^2}{T}\sum_{t=1}^{T}\sum_{n=0}^{t-1} f^2(n) - \frac{\sigma^2}{T}\sum_{t=1}^{T}\frac{1}{t}\left(\sum_{n=0}^{t-1}f(n)\right)^2 \\
&= \frac{\sigma^2}{T}\sum_{t=1}^{T}\frac{1}{t}\left[t\sum_{n=0}^{t-1}f^2(n) - \left(\sum_{n=0}^{t-1}f(n)\right)^2\right] \\
&= \frac{\sigma^2}{T}\sum_{t=1}^{T}\frac{1}{2t}\sum_{m=0}^{t-1}\sum_{n=0}^{t-1}(f(m)-f(n))^2.
\end{aligned}$$

3.G Proof of Corollary 3.10

We have

$$F(t) = \begin{cases} t+1 & \text{if } 0 \le t < \Delta \\ \Delta & \text{if } t \ge \Delta \end{cases}$$

for $t = 0, \dots, T$. It follows that

$$\begin{aligned}
\mathbb{E}(V) &= \frac{\sigma^2}{T^2} \sum_{t=0}^{T-1} F^2(t) \frac{T-t-1}{t+1} \\
&= \frac{\sigma^2}{T^2} \sum_{t=0}^{\Delta-1} (t+1)^2 \frac{T-t-1}{t+1} + \frac{\sigma^2 \Delta^2}{T^2} \sum_{t=\Delta}^{T-1} \frac{T-t-1}{t+1} \\
&= \frac{\sigma^2}{T^2} \sum_{t=1}^{\Delta} t(T-t) + \frac{\sigma^2 \Delta^2}{T^2} \sum_{t=\Delta+1}^{T} \left(\frac{T}{t} - 1 \right) \\
&\in \frac{\sigma^2 \Delta^2}{T^2} \left[T \ln \frac{T+1}{\Delta+1} - T + \Delta,\ T \ln \frac{T}{\Delta} - T + \Delta + 1 \right] \\
&= \Delta \sigma^2 \frac{\ln(T/\Delta)}{T/\Delta} \left(1 + O\left(\frac{1}{\ln(T/\Delta)} \right) \right)
\end{aligned}$$

and

$$\begin{aligned}
\mathbb{E}(V') &= \frac{\sigma^2}{T} \sum_{t=0}^{T-1} (T-t) f^2(t) - \frac{\sigma^2}{T^2} \sum_{t=0}^{T-1} F^2(t) \\
&= \frac{\sigma^2}{T} \sum_{t=0}^{\Delta-1} (T-t) - \frac{\sigma^2}{T^2} \sum_{t=0}^{\Delta-1} (t+1)^2 - \frac{\sigma^2 \Delta^2}{T^2} (T - \Delta) \\
&= \Delta \sigma^2 \left(1 + O(\Delta/T) \right).
\end{aligned}$$

Hence, the expected load variance reduction is

$$\frac{\mathbb{E}(V')}{\mathbb{E}(V)} = \frac{T/\Delta}{\ln(T/\Delta)} \left(1 + O\left(\frac{1}{\ln(T/\Delta)} \right) \right).$$

3.H Proof of Corollary 3.11

We have $F(t) = (1 - a^{t+1})/(1-a)$ for $t = 0, \dots, T$. Note that for any $\Delta \in \{1, \dots, T\}$,

$$\mathbb{E}(V) = \frac{\sigma^2}{(1-a)^2 T^2} \sum_{t=1}^{T} \left(1 - a^t\right)^2 \frac{T-t}{t} = \frac{\sigma^2}{(1-a)^2 T^2} (A + B)$$

where

$$A := \sum_{t=1}^{\Delta} \frac{T-t}{t} \left(1 - a^t\right)^2, \quad B := \sum_{t=\Delta+1}^{T} \frac{T-t}{t} \left(1 - a^t\right)^2$$

by splitting the sum at Δ. We can bound the terms A and B separately, on the one hand,

$$0 \le A \le T \sum_{t=1}^{\Delta} \frac{1}{t} \le T(1 + \ln \Delta).$$

On the other hand,

$$\begin{aligned}
B &\geq \left(1-a^{\Delta+1}\right)^2 \sum_{t=\Delta+1}^{T} \frac{T-t}{t} \\
&\geq \left(1-a^{\Delta+1}\right)^2 \left(T \ln \frac{T+1}{\Delta+1} - T + \Delta\right) \\
&= \left(1-a^{\Delta+1}\right)^2 T \ln T \left(1 + O\left(\frac{\ln \Delta}{\ln T}\right)\right)
\end{aligned}$$

and

$$\begin{aligned}
B &\leq \sum_{t=\Delta+1}^{T} \frac{T-t}{t} \\
&\leq T \ln \frac{T}{\Delta} - T + \Delta \\
&= T \ln T \left(1 + O\left(\frac{\ln \Delta}{\ln T}\right)\right).
\end{aligned}$$

Select $\Delta = [\ln T]$ and let $T \to \infty$ to obtain

$$T \ln T \left(1 + O\left(\frac{\ln \ln T}{\ln T}\right)\right) \leq A + B \leq T \ln T \left(1 + O\left(\frac{\ln \ln T}{\ln T}\right)\right).$$

Therefore $A + B = T \ln T \left(1 + O\left(\ln \ln T / \ln T\right)\right)$ and it follows that

$$\mathbb{E}(V) = \frac{\sigma^2}{(1-a)^2} \frac{\ln T}{T} \left(1 + O\left(\frac{\ln \ln T}{\ln T}\right)\right).$$

We also have

$$\begin{aligned}
\mathbb{E}(V') &= \frac{\sigma^2}{T} \sum_{t=0}^{T-1} (T-t) a^{2t} - \frac{\sigma^2}{(1-a)^2 T^2} \sum_{t=1}^{T} \left(1-a^t\right)^2 \\
&= \frac{\sigma^2}{1-a^2} \left[1 - \frac{a^2(1-a^{2T})}{T(1-a^2)}\right] - \frac{\sigma^2}{(1-a)^2 T} \left[1 - \frac{a^2 + 2a - 2a^{T+1} - 2a^{T+2} + a^{2+2T}}{T(1-a^2)}\right] \\
&= \frac{\sigma^2}{1-a^2} \left(1 + O\left(\frac{1}{T}\right)\right).
\end{aligned}$$

Hence, the expected load variance reduction is

$$\frac{\mathbb{E}(V')}{\mathbb{E}(V)} = \frac{1-a}{1+a} \frac{T}{\ln T} \left(1 + O\left(\frac{\ln \ln T}{\ln T}\right)\right).$$

Chapter 4

Optimal Power Flow

Distribution networks are usually multiphase and radial. To facilitate power flow computation and optimization, two semidefinite programming (SDP) relaxations of the optimal power flow problem and a linear approximation of the power flow are proposed. We prove that the first SDP relaxation is exact if and only if the second one is exact. Case studies show that the second SDP relaxation is numerically exact and that the linear approximation obtains voltages within 0.0016 per unit of their true values for the IEEE 13, 34, 37, 123-bus networks and a real-world 2065-bus network.

Literature The optimal power flow (OPF) problem is nonconvex, and approximations and relaxations have been developed to solve it; see recent surveys in [20, 25, 39, 54, 81, 84]. For convex relaxations, it is first proposed in [57] to solve OPF as a second-order cone programming for single-phase radial networks and in [10] as a semidefinite programming (SDP) for single-phase mesh networks. While numerically illustrated in [57] and [10], whether or when the convex relaxations are exact is not studied until [66]; see [73, 74] for a survey and references to a growing literature on convex relaxations of OPF.

Most of these works assume a single-phase network, while distribution networks are typically multiphase and unbalanced [63]. It has been observed in [29, 60] that a multiphase network has an equivalent single-phase circuit model where each bus-phase pair in the multiphase network is identified with a single bus in the equivalent model. Hence methods for single-phase networks can be applied to the equivalent model of a multiphase unbalanced network. This approach is taken in [33] for solving optimal power flow problems. Additionally, [33] develops distributed solutions.

Summary This chapter develops convex relaxations of OPF and a linear approximation of power flow. Solving OPF through convex relaxation offers several advantages. It provides the ability to check if a solution is globally optimal. If it is not, the solution provides a lower bound on the minimum cost and hence a bound on how far any feasible solution is from optimality. Unlike approximations, if a relaxation is infeasible, it certifies that the original OPF is infeasible.

There are three questions on convex relaxations: 1) how to compute convex relaxations efficiently, 2) how to attain numerical stability, and 3) when can a globally optimum of OPF be obtained by solving its convex relaxation?

To address 1), the relaxation BIM-SDP is proposed in Section 4.2 to improve the computational efficiency of a standard SDP relaxation by exploiting the radial network topology. While the standard SDP relaxation declares $O(n^2)$ variables where $n+1$ is the number of buses in the network, BIM-SDP only declares $O(n)$ variables and is therefore more efficient.

To address 2), the relaxation BFM-SDP is proposed in Section 4.3 to improve the numerical stability of BIM-SDP by avoiding ill-conditioned operations. BIM-SDP is ill-conditioned due to subtractions of voltages that are close in value. Using alternative variables, BFM-SDP avoids these subtractions and is therefore numerically more stable.

To partially address 3), we prove in Section 4.5 that BIM-SDP is exact if and only if BFM-SDP is exact, and empirically show in Section 4.5 that BFM-SDP is numerically exact for the IEEE 13, 34, 37, 123-bus networks and a real-world 2065-bus network. Remarkably, BIM-SDP is numerically exact only for the IEEE 13 and 37-bus networks. This highlights the numerical stability of BFM-SDP.

Approximation LPF is proposed in Section 4.4 to estimate voltages and power flows. LPF is accurate when line loss is small compared with power flow and voltages are nearly balanced, i.e., the voltages of different phases have similar magnitudes and differ in angle by $\sim 120°$. Empirically, it is presented in Section 4.5 that LPF computes voltages within 0.0016 per unit of their true values for the IEEE 13, 34, 37, and 123-bus networks and a real-world 2065-bus network.

4.1 Optimal Power Flow Problem

OPF in multiphase radial networks is applicable for demand response and volt/var control.

4.1.1 A Standard Nonlinear Power Flow Model

A distribution network is composed of buses and lines connecting these buses. It is usually multiphase and radial. There is a substation bus in the network with a fixed voltage. Index the substation bus by 0 and the other buses by $1, 2, \ldots, n$. Let $\mathcal{N} = \{0, 1, \ldots, n\}$ denote the set of buses and define $\mathcal{N}^+ = \mathcal{N} \backslash \{0\}$. Each line connects an ordered pair (i, j) of buses where bus i lies between bus 0 and bus j. Let $\mathcal{E}$ denote the set of lines. Use $(i, j) \in \mathcal{E}$ and $i \to j$ interchangeably. If $i \to j$ or $j \to i$, denote $i \sim j$.

Let a, b, c denote the three phases of the network, let Φ_i denote the phases of bus $i \in \mathcal{N}$, and let Φ_{ij} denote the phases of line $i \sim j$. For each bus $i \in \mathcal{N}$, let V_i^ϕ denote its phase ϕ complex voltage for $\phi \in \Phi_i$ and define $V_i := [V_i^\phi]_{\phi \in \Phi_i}$; let I_i^ϕ denote its phase ϕ current injection for $\phi \in \Phi_i$

and define $I_i := [I_i^\phi]_{\phi \in \Phi_i}$; let s_i^ϕ denote its phase ϕ complex power injection for $\phi \in \Phi_i$ and define $s_i := [s_i^\phi]_{\phi \in \Phi_i}$. For each line $i \sim j$, let I_{ij}^ϕ denote the phase ϕ current from bus i to bus j for $\phi \in \Phi_{ij}$ and define $I_{ij} := [I_{ij}^\phi]_{\phi \in \Phi_{ij}}$; let z_{ij} denote the phase impedance matrix and define $y_{ij} := z_{ij}^{-1}$.

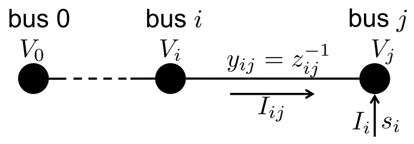


Figure 4.1: Summary of notations

Some notations are summarized in Figure 4.1. Further, let superscripts denote projection to specified phases, e.g., if $\Phi_i = abc$, then

$$V_i^{ab} = (V_i^a, V_i^b)^T.$$

Fill nonexisting phase entries by 0, e.g., if $\Phi_i = ab$, then

$$V_i^{abc} = (V_i^a, V_i^b, 0)^T.$$

Let a letter without subscripts denote a vector of the corresponding quantity, e.g., $z = [z_{ij}]_{i \sim j}$ and $s = [s_i]_{i \in \mathcal{N}}$.

Power flows are governed by [63]:

1) Ohm's law: $I_{ij} = y_{ij}(V_i^{\Phi_{ij}} - V_j^{\Phi_{ij}})$ for $i \sim j$.

2) Current balance: $I_i = \sum_{j:\, i \sim j} I_{ij}^{\Phi_i}$ for $i \in \mathcal{N}$.

3) Power balance: $s_i = \mathrm{diag}(V_i I_i^H)$ for $i \in \mathcal{N}$.

Eliminate current variables I_i and I_{ij} to obtain the following *bus injection model* (BIM):

$$s_i = \sum_{j:\, i \sim j} \mathrm{diag}\left[V_i^{\Phi_{ij}}(V_i^{\Phi_{ij}} - V_j^{\Phi_{ij}})^H y_{ij}^H\right]^{\Phi_i}, \quad i \in \mathcal{N}. \tag{4.1}$$

4.1.2 Optimal Power Flow

OPF determines the power injection that minimizes generation cost subject to physical and operational constraints. Generation cost is separable. In particular, let $C_i(s_i) : \mathbb{C}^{|\Phi_i|} \mapsto \mathbb{R}$ denote the generation cost at bus $i \in \mathcal{N}$, and

$$C(s) = \sum_{i \in \mathcal{N}} C_i(s_i)$$

is the generation cost of the network.

OPF has operational constraints on power injections and voltages besides physical constraints (4.1). First, while the substation power injection s_0 is unconstrained, a branch bus power injection s_i can only vary within some externally specified set $\mathcal{S}_i$, i.e.,

$$s_i \in \mathcal{S}_i, \qquad i \in \mathcal{N}^+. \tag{4.2}$$

For example, the sets $\mathcal{S}_i$ of two types of devices are illustrated in Figure 4.2. Note that $\mathcal{S}_i$ is usually not a box, and that $\mathcal{S}_i$ can be nonconvex or even disconnected.

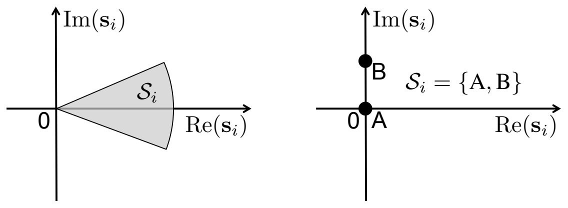


Figure 4.2: The left figure illustrates the set $\mathcal{S}_i$ of an inverter, and the right figure illustrates the set $\mathcal{S}_i$ of a shunt capacitor. Note that the set $\mathcal{S}_i$ is usually not a box, and that $\mathcal{S}_i$ can be nonconvex or even disconnected.

Second, while the substation voltage V_0 is fixed and given (denote by V_0^{ref} that is nonzero componentwise), a branch bus voltage can be regulated within a range, i.e., there exists $[\underline{V}_i^\phi, \overline{V}_i^\phi]_{i\in\mathcal{N}^+,\phi\in\Phi_i}$ such that

$$V_0 = V_0^{\text{ref}}; \tag{4.3a}$$

$$\underline{V}_i^\phi \le |V_i^\phi| \le \overline{V}_i^\phi, \quad i \in \mathcal{N}^+,\ \phi \in \Phi_i. \tag{4.3b}$$

For example, if voltages must stay within 5% from their nominal values, then $0.95 \le |V_i^\phi| \le 1.05$ per unit.

To summarize, OPF can be formulated as

$$\begin{aligned} \textbf{OPF:}\ \min \quad & \sum_{i\in\mathcal{N}} C_i(s_i) \\ \text{over} \quad & s, V \\ \text{s.t.} \quad & (4.1) - (4.3). \end{aligned}$$

The following assumptions are made throughout this paper.

1. The network $(\mathcal{N}, \mathcal{E})$ is connected.

2. Voltage lower bounds are strictly positive, i.e.,

$$\underline{V}_i^\phi > 0, \quad i \in \mathcal{N}^+,\ \phi \in \Phi_i.$$

3. Bus and line phases satisfy

$$\Phi_i \supseteq \Phi_{ij} = \Phi_j, \quad i \to j.$$

4.2 Bus Injection Model Semidefinite Programming

OPF is nonconvex due to (4.1), and a standard SDP relaxation has been developed to solve it [33]. In this section we propose a different SDP relaxation, called BIM-SDP, that exploits the radial network topology to reduce the computational complexity of the standard SDP.

BIM-SDP is derived by shifting the nonconvexity from (4.1) to some rank constraints and removing the rank constraints. Let $|A|$ denote the number of elements in a set A, and $\mathbb{H}^{k\times k}$ denote the set of $k \times k$ complex Hermitian matrices. Let $v_i \in \mathbb{H}^{|\Phi_i|\times|\Phi_i|}$ for $i \in \mathcal{N}$ and $W_{ij} \in \mathbb{C}^{|\Phi_{ij}|\times|\Phi_{ij}|}$ for $i \sim j$. If these matrices satisfy

$$\begin{bmatrix} v_i^{\Phi_{ij}} & W_{ij} \\ W_{ji} & v_j \end{bmatrix} = \begin{bmatrix} V_i^{\Phi_{ij}} \\ V_j \end{bmatrix} \begin{bmatrix} V_i^{\Phi_{ij}} \\ V_j \end{bmatrix}^H, \quad i \to j,$$

then (4.1) is equivalent to

$$s_i = \sum_{j:\, i\sim j} \operatorname{diag}\left[(v_i^{\Phi_{ij}} - W_{ij})y_{ij}^H\right]^{\Phi_i}, \quad i \in \mathcal{N}.$$

Lemma 4.1. *Let $v_i \in \mathbb{H}^{|\Phi_i|\times|\Phi_i|}$ for $i \in \mathcal{N}$ and $W_{ij} \in \mathbb{C}^{|\Phi_{ij}|\times|\Phi_{ij}|}$ for $i \sim j$. If*

- $v_0 = V_0^{\text{ref}}[V_0^{\text{ref}}]^H$ *for some* $V_0^{\text{ref}} \in \mathbb{C}^{|\Phi_0|}$*;*
- $\operatorname{diag}(v_i)$ *is nonzero componentwise for* $i \in \mathcal{N}$*;*
- $W_{ji} = W_{ij}^H$ *for* $i \to j$*;*
- $\begin{bmatrix} v_i^{\Phi_{ij}} & W_{ij} \\ W_{ji} & v_j \end{bmatrix}$ *is rank one for* $i \to j$*,*

then Algorithm 3 computes the unique V that satisfies $V_0 = V_0^{\text{ref}}$ and

$$v_i = V_i V_i^H, \qquad i \in \mathcal{N}; \tag{4.4a}$$

$$W_{ij} = V_i^{\Phi_{ij}}(V_j^{\Phi_{ij}})^H, \qquad i \sim j. \tag{4.4b}$$

Lemma 4.1 is proved in Appendix 4.A. It implies that OPF can be equivalently formulated as BIM-OPF. Let $A \succeq 0$ denote a hermitian matrix A being positive semidefinte.

Algorithm 3 Recover V from (v, W).

Input: (v, W) that satisfies the conditions in Lemma 4.1.
Output: V.

```
1: $V_0 \leftarrow V_0^{\text{ref}}$;
2: $\mathcal{N}_{\text{visit}} \leftarrow \{0\}$;
3: while $\mathcal{N}_{\text{visit}} \neq \mathcal{N}$ do
4:     find $i \to j$ such that $i \in \mathcal{N}_{\text{visit}}$ and $j \notin \mathcal{N}_{\text{visit}}$;
5:     compute
           $V_j \leftarrow \frac{1}{\operatorname{tr}\left(v_i^{\Phi_{ij}}\right)} W_{ji} V_i^{\Phi_{ij}}$;
           $\mathcal{N}_{\text{visit}} \leftarrow \mathcal{N}_{\text{visit}} \cup \{j\}$;
6: end while
```

$$
\begin{aligned}
\textbf{BIM-OPF:}\ \min \quad & \sum_{i \in \mathcal{N}} C_i(s_i) \\
\text{over} \quad & s_i \in \mathbb{C}^{|\Phi_i|} \text{ and } v_i \in \mathbb{H}^{|\Phi_i| \times |\Phi_i|} \text{ for } i \in \mathcal{N};\ W_{ij} \in \mathbb{C}^{|\Phi_{ij}| \times |\Phi_{ij}|} \text{ for } i \sim j, \\
\text{s.t.} \quad & s_i = \sum_{j:\, i \sim j} \operatorname{diag}\left[(v_i^{\Phi_{ij}} - W_{ij}) y_{ij}^H\right]^{\Phi_i}, \quad i \in \mathcal{N}; && \text{(4.5a)} \\
& s_i \in \mathcal{S}_i, \quad i \in \mathcal{N}^+; && \text{(4.5b)} \\
& v_0 = V_0^{\text{ref}} (V_0^{\text{ref}})^H; && \text{(4.5c)} \\
& \underline{v}_i \leq \operatorname{diag}(v_i) \leq \overline{v}_i, \quad i \in \mathcal{N}^+; && \text{(4.5d)} \\
& W_{ij} = W_{ji}^H, \quad i \to j; && \text{(4.5e)} \\
& \begin{bmatrix} v_i^{\Phi_{ij}} & W_{ij} \\ W_{ji} & v_j \end{bmatrix} \succeq 0, \quad i \to j; && \text{(4.5f)} \\
& \operatorname{rank} \begin{bmatrix} v_i^{\Phi_{ij}} & W_{ij} \\ W_{ji} & v_j \end{bmatrix} = 1, \quad i \to j && \text{(4.5g)}
\end{aligned}
$$

where the vectors $\underline{v}_i$ and $\overline{v}_i$ in (4.5d) are defined as

$$\underline{v}_i := [(\underline{V}_i^{\phi})^2]_{\phi \in \Phi_i},\ \overline{v}_i := [(\overline{V}_i^{\phi})^2]_{\phi \in \Phi_i}, \quad i \in \mathcal{N}^+.$$

If C_i (in the objective) and $\mathcal{S}_i$ [in (4.5g)] are convex, then BIM-OPF is convex except for (4.5g), and an SDP relaxation can be obtained by removing (4.5g) from BIM-OPF.

$$\begin{aligned} \textbf{BIM-SDP:}\ \min\quad & \sum_{i\in\mathcal{N}} C_i(s_i) \\ \text{over}\quad & s, v, W \\ \text{s.t.}\quad & (4.5\text{a}) - (4.5\text{f}). \end{aligned}$$

Note that BIM-SDP may be nonconvex due to C_i and $\mathcal{S}_i$.

If an optimal BIM-SDP solution (s, v, W) satisfies (4.5g), then (s, v, W) also solves BIM-OPF. Furthermore, a global optimum (s, V) of OPF can be recovered via Algorithm 3.

Theorem 4.2. *Given an optimal solution (s, v, W) of BIM-SDP that satisfies* (4.5g)*, Algorithm 3 computes a V such that (s, V) solves OPF.*

Theorem 4.2 follows directly from Lemma 4.1.

Definition 4.1. *BIM-SDP is* exact *if every optimal solution of BIM-SDP satisfies* (4.5g)*.*

If BIM-SDP is exact, then a global optimum of OPF can be obtained by solving BIM-SDP according to Theorem 4.2.

Comparison with a Standard SDP A standard SDP relaxation of OPF has been proposed in the literature [33]. It is derived by introducing

$$\tilde{W} = \begin{bmatrix} V_0 \\ \vdots \\ V_n \end{bmatrix} \begin{bmatrix} V_1^H & \cdots & V_n^H \end{bmatrix}$$

to shift the nonconvexity from (4.1) in BIM-OPF to rank$\tilde{W} = 1$, and removing the rank constraint. We call this relaxation standard-SDP for ease of reference.

BIM-SDP is computationally more efficient than standard-SDP since it has fewer variables. It is straightforward to verify that there are $O(n)$ variables in BIM-SDP and $O(n^2)$ variables in standard-SDP. Note that $n + 1$ is equal to the number of buses in the network.

Standard-SDP does not exploit the radial network topology. In $\tilde{W}$, only blocks corresponding to lines $i \sim j$ appear in other constraints than $\tilde{W} \succeq 0$, i.e., if bus i and bus j are not connected, then block (i, j) in $\tilde{W}$ only appears in $\tilde{W} \succeq 0$. Since the network is radial, $n^2 - n$ out of the $(n + 1)^2$ blocks in $\tilde{W}$ only appear in $\tilde{W} \succeq 0$, leaving significant potential for exploring sparsity.

Call these n^2 blocks that only appear in $\tilde{W} \succeq 0$ the $\tilde{W}$-only blocks and the other $2n + 1$ blocks the key-blocks. The purpose of having $\tilde{W}$-only blocks in the optimization is to make sure that the partial matrix specified by key-blocks can be completed to a positive semidefinite full matrix.

It is known that a partially positive semidefinite matrix has a positive semidefinite completion if and only if its underlying graph is chordal [40]. Essentially, BIM-SDP applies this technique to exploit the radial network topology.

4.3 BFM Semidefinite Programming

BIM-SDP is not numerically stable and a different relaxation is proposed in this section to improve the numerical stability of BIM-SDP.

4.3.1 Alternative Power Flow Model

We start with introducing a novel branch flow model (BFM) of power flow. BFM enhances the numerical stability of BIM (4.1). BIM (4.1) is ill-conditioned due to subtractions of $V_i^{\Phi_{ij}}$ and $V_j^{\Phi_{ij}}$ that are close in value. BFM obtains an improved numerical stability by avoiding such subtractions in the calculation of power flows (though such subtractions are still used in the calculation of voltages).

BFM is given by the following three equations.

1. Ohm's law:
$$V_i^{\Phi_{ij}} - V_j = z_{ij} I_{ij}, \quad i \to j. \tag{4.6}$$

2. Definition of auxiliary variables:
$$\ell_{ij} = I_{ij} I_{ij}^H, \; S_{ij} = V_i^{\Phi_{ij}} I_{ij}^H, \quad i \to j. \tag{4.7}$$

3. Power balance:
$$\sum_{i:\, i\to j} \operatorname{diag}(S_{ij} - z_{ij}\ell_{ij}) + s_j = \sum_{k:\, j\to k} \operatorname{diag}\left(S_{jk}\right)^{\Phi_j}, \quad j \in \mathcal{N}. \tag{4.8}$$

To interpret ℓ and S, note that $\operatorname{diag}(\ell_{ij})$ denotes the magnitude squares of current I_{ij}, and $\operatorname{diag}(S_{ij})$ denotes the sending-end power flow on line $i \to j$. To interpret (4.8), note that the receiving-end power flow on line $i \to j$ is

$$\operatorname{diag}(V_j I_{ij}^H) = \operatorname{diag}(S_{ij} - z_{ij}\ell_{ij}).$$

BIM and BFM are equivalent in the sense that they share the same solution set (s, V). More specifically, let

$$\begin{aligned}
\mathbb{F}_{\text{BIM}} &:= \{(s,V) \mid (s,V) \text{ satisfies } (4.1)\}, \\
\mathbb{F}_{\text{BFM}} &:= \left\{ (s,V) \;\middle|\; \begin{array}{l} \exists\ (I,\ell,S) \text{ such that} \\ (s,V,I,\ell,S) \text{ satisfies (4.6)–(4.8)} \end{array} \right\}
\end{aligned}$$

denote the sets of (s,V) that satisfy BIM or BFM.

Theorem 4.3. *The solution set* $\mathbb{F}_{\text{BIM}} = \mathbb{F}_{\text{BFM}}$.

Theorem 4.3 is proved in Appendix 4.B. It implies that OPF can be equivalently formulated as follows.

$$\begin{aligned}
\textbf{OPF':}\ \min\ & \sum_{i\in\mathcal{N}} C_i(s_i) \\
\text{over}\ & s,V,I,\ell,S \\
\text{s.t.}\ & (4.2)-(4.3),\ (4.6)-(4.8).
\end{aligned}$$

4.3.2 Branch Flow Model Semidefinite Programming

A numerically stable SDP that has a similar computational efficiency as BIM-SDP is proposed in this section. To motivate the SDP, assume (4.4a), (4.6), and (4.7) hold, then

$$V_j = V_i^{\Phi_{ij}} - z_{ij} I_{ij}, \quad i \to j.$$

Multiply both sides by their Hermitian transposes to obtain

$$v_j = v_i^{\Phi_{ij}} - (S_{ij} z_{ij}^H + z_{ij} S_{ij}^H) + z_{ij}\ell_{ij} z_{ij}^H, \quad i \to j. \tag{4.9}$$

Furthermore, the matrix

$$\begin{bmatrix} v_i^{\Phi_{ij}} & S_{ij} \\ S_{ij}^H & \ell_{ij} \end{bmatrix} = \begin{bmatrix} V_i^{\Phi_{ij}} \\ I_{ij} \end{bmatrix} \begin{bmatrix} V_i^{\Phi_{ij}} \\ I_{ij} \end{bmatrix}^H$$

is positive semidefinite and rank one for $i \to j$.

Lemma 4.4. *Let* $v_i \in \mathbb{H}^{|\Phi_i|\times|\Phi_i|}$ *for* $i \in \mathcal{N}$. *Let* $S_{ij} \in \mathbb{C}^{|\Phi_{ij}|\times|\Phi_{ij}|}$ *and* $\ell_{ij} \in \mathbb{H}^{|\Phi_{ij}|\times|\Phi_{ij}|}$ *for* $i \to j$. *If*

- $v_0 = V_0^{\text{ref}}[V_0^{\text{ref}}]^H$ *for some* $V_0^{\text{ref}} \in \mathbb{C}^{|\Phi_0|}$*;*
- $\text{diag}(v_i)$ *is nonzero componentwise for* $i \in \mathcal{N}$*;*
- (v,S,ℓ) *satisfies* (4.9)*;*

- $\begin{bmatrix} v_i^{\Phi_{ij}} & S_{ij} \\ S_{ij}^H & \ell_{ij} \end{bmatrix}$ *is rank one for* $i \to j$,

then Algorithm 4 computes the unique (V, I) *that satisfies* $V_0 = V_0^{\text{ref}}$, (4.4a), (4.6), *and* (4.7).

Algorithm 4 Recover (V, I) from (v, S, ℓ).

```
Input: $(v, S, \ell)$ that satisfies the conditions in Lemma 4.4.
Output: $(V, I)$.
1: $V_0 \leftarrow V_0^{\text{ref}}$;
2: $\mathcal{N}_{\text{visit}} \leftarrow \{0\}$;
3: while $\mathcal{N}_{\text{visit}} \neq \mathcal{N}$ do
4:     find $i \to j$ such that $i \in \mathcal{N}_{\text{visit}}$ and $j \notin \mathcal{N}_{\text{visit}}$;
5:     compute
           $I_{ij} \leftarrow \frac{1}{\text{tr}\left(v_i^{\Phi_{ij}}\right)} S_{ij}^H V_i^{\Phi_{ij}}$;
           $V_j \leftarrow V_i^{\Phi_{ij}} - z_{ij} I_{ij}$;
           $\mathcal{N}_{\text{visit}} \leftarrow \mathcal{N}_{\text{visit}} \cup \{j\}$;
6: end while
```

Lemma 4.4 is proved in Appendix 4.C. It implies that OPF' can be equivalently formulated as BFM-OPF.

$$
\begin{aligned}
\textbf{BFM-OPF:}\ \min\quad & \sum_{i \in \mathcal{N}} C_i(s_i) \\
\text{over}\quad & s_i \in \mathbb{C}^{|\Phi_i|}, v_i \in \mathbb{H}^{|\Phi_i| \times |\Phi_i|} \text{ for } i \in \mathcal{N}; \\
& S_{ij} \in \mathbb{C}^{|\Phi_{ij}| \times |\Phi_{ij}|}, \ell_{ij} \in \mathbb{H}^{|\Phi_{ij}| \times |\Phi_{ij}|} \text{ for } i \to j, \\
\text{s.t.}\quad & \sum_{i:\, i \to j} \text{diag}(S_{ij} - z_{ij}\ell_{ij}) + s_j = \sum_{k:\, j \to k} \text{diag}\left(S_{jk}\right)^{\Phi_j}, \quad j \in \mathcal{N}; && \text{(4.10a)} \\
& s_i \in \mathcal{S}_i, \quad i \in \mathcal{N}^+; && \text{(4.10b)} \\
& v_0 = V_0^{\text{ref}} (V_0^{\text{ref}})^H; && \text{(4.10c)} \\
& \underline{v}_i \le \text{diag}(v_i) \le \overline{v}_i, \quad i \in \mathcal{N}^+; && \text{(4.10d)} \\
& v_j = v_i^{\Phi_{ij}} - (S_{ij} z_{ij}^H + z_{ij} S_{ij}^H) + z_{ij} \ell_{ij} z_{ij}^H, \quad i \to j; && \text{(4.10e)} \\
& \begin{bmatrix} v_i^{\Phi_{ij}} & S_{ij} \\ S_{ij}^H & \ell_{ij} \end{bmatrix} \succeq 0, \quad i \to j; && \text{(4.10f)} \\
& \text{rank} \begin{bmatrix} v_i^{\Phi_{ij}} & S_{ij} \\ S_{ij}^H & \ell_{ij} \end{bmatrix} = 1, \quad i \to j. && \text{(4.10g)}
\end{aligned}
$$

If C_i and $\mathcal{S}_i$ are convex, then BFM-OPF is convex except for (4.10g), and an SDP relaxation can be obtained by removing (4.10g) from BFM-OPF.

$$\begin{aligned} \textbf{BFM-SDP:} \ \min \quad & \sum_{i\in\mathcal{N}} C_i(s_i) \\ \text{over} \quad & s, v, S, \ell \\ \text{s.t.} \quad & (4.10\text{a}) - (4.10\text{f}). \end{aligned}$$

Note that BFM-SDP may not be convex due to C_i and $\mathcal{S}_i$.

If an optimal BFM-SDP solution (s, v, S, ℓ) satisfies (4.10g), then (s, v, S, ℓ) also solves BFM-OPF. Moreover, Algorithm 4 produces a global optimum (s, V, I, ℓ, S) of OPF'.

Theorem 4.5. *Given an optimal solution (s, v, S, ℓ) of BFM-SDP that satisfies* (4.10g)*, compute (V, I) according to Algorithm 4. Then (s, V, I, ℓ, S) solves OPF'.*

Theorem 4.5 follows directly from Lemma 4.4.

Definition 4.2. *BFM-SDP is* exact *if every optimal solution of BFM-SDP satisfies* (4.10g).

If BFM-SDP is exact, then a global optimum of OPF' can be obtained by solving BFM-SDP according to Theorem 4.5.

4.3.3 Comparison with BIM-SDP

BFM-SDP is numerically more stability than BIM-SDP since it avoids subtractions of $v_i^{\Phi_{ij}}$ and W_{ij} that are close in value. Meanwhile, BFM-SDP has similar computational efficiency as BIM-SDP since they have the same number of variables and constraints.

There exists a bijective map between the feasible sets of BIM-SDP and BFM-SDP that preserves the objective value. Let $\mathbb{F}_{\text{BIM-SDP}}$ and $\mathbb{F}_{\text{BFM-SDP}}$ denote the feasible sets of BIM-SDP and BFM-SDP.

Theorem 4.6. *The map $f : \mathbb{F}_{BIM\text{-}SDP} \mapsto \mathbb{F}_{BFM\text{-}SDP}$ defined by $f(s, v, W) = (s, v, S, \ell)$ where*

$$\begin{aligned} S_{ij} &= (v_i^{\Phi_{ij}} - W_{ij})y_{ij}^H, && i \to j; \\ \ell_{ij} &= y_{ij}(v_i^{\Phi_{ij}} - W_{ji} - W_{ij} + v_j)y_{ij}^H, && i \to j \end{aligned}$$

is bijective, and its inverse $g : \mathbb{F}_{BFM\text{-}SDP} \mapsto \mathbb{F}_{BIM\text{-}SDP}$ is given by $g(s, v, S, \ell) = (s, v, W)$ where

$$W_{ij} = v_i^{\Phi_{ij}} - S_{ij}z_{ij}^H, \ W_{ji} = W_{ij}^H, \quad i \to j.$$

Theorem 4.6 is proved in Appendix 4.D. It implies that f is also bijective from the optimal solutions of BIM-SDP to the optimal solutions of BFM-SDP.

Corollary 4.7. *Let f be as in Theorem 4.6. A point (s, v, W) solves BIM-SDP if and only if $f(s, v, W)$ solves BFM-SDP.*

Theorem 4.8. *Let f be as in Theorem 4.6. A feasible solution (s, v, W) of BIM-SDP satisfies* (4.5g) *if and only if the feasible solution $f(s, v, W)$ of BFM-SDP satisfies* (4.10g).

Theorem 4.8 is proved in Appendix 4.E. It implies that BIM-SDP is exact if and only if BFM-SDP is exact.

Corollary 4.9. *BIM-SDP is exact if and only if BFM-SDP is exact.*

4.4 Linear approximation

A linear approximation of the power flow LPF is proposed in this section. It is obtained by assuming:

B1 Line losses are small, i.e., $z_{ij}\ell_{ij} \ll S_{ij}$ componentwise for $i \to j$.

B2 Voltages are nearly balanced, e.g., if $\Phi_i = abc$, then

$$\frac{V_i^a}{V_i^b} \approx \frac{V_i^b}{V_i^c} \approx \frac{V_i^c}{V_i^a} \approx e^{j2\pi/3}.$$

With B1, omit the $z_{ij}\ell_{ij}$ terms in (4.8) and (4.9) to obtain

$$\sum_{i:\, i\to j} \operatorname{diag}(S_{ij}) + s_j = \sum_{k:\, j\to k} \operatorname{diag}(S_{jk})^{\Phi_j}, \qquad j \in \mathcal{N}; \qquad (4.11a)$$

$$v_j = v_i^{\Phi_{ij}} - (S_{ij} z_{ij}^H + z_{ij} S_{ij}^H), \qquad i \to j. \qquad (4.11b)$$

Given s_j for $j \in \mathcal{N}^+$, (4.11a) determines uniquely s_0 and $\operatorname{diag}(S_{ij})$ for $i \to j$, but not the off-diagonal entries of S_{ij}. B2 is used to approximate the off-diagonal entries in S_{ij} with $\operatorname{diag}(S_{ij})$. Specifically, define

$$\alpha := e^{-j2\pi/3}, \quad \beta := \begin{bmatrix} 1 \\ \alpha \\ \alpha^2 \end{bmatrix}, \quad \gamma := \begin{bmatrix} 1 & \alpha^2 & \alpha \\ \alpha & 1 & \alpha^2 \\ \alpha^2 & \alpha & 1 \end{bmatrix},$$

and assume the voltages to be balanced. Then for each line $i \to j$, each column of S_{ij} is in the range space of $\beta^{\Phi_{ij}}$. It follows that if $\Lambda_{ij} = \operatorname{diag}(S_{ij})$, let $\operatorname{Diag}(\Lambda_{ij})$ denote a diagonal matrix with diagonal Λ_{ij}, then

$$S_{ij} = \gamma^{\Phi_{ij}} \operatorname{diag}(\Lambda_{ij}).$$

To summarize, (4.11) can be written as

$$\begin{aligned}
\textbf{LPF:} \quad & \sum_{i:\, i\to j} \Lambda_{ij} + s_j = \sum_{k:\, j\to k} \Lambda_{jk}^{\Phi_j}, && j \in \mathcal{N}; && \text{(4.12a)}\\
& S_{ij} = \gamma^{\Phi_{ij}} \mathrm{diag}(\Lambda_{ij}), && i \to j; && \text{(4.12b)}\\
& v_j = v_i^{\Phi_{ij}} - S_{ij} z_{ij}^H - z_{ij} S_{ij}^H, && i \to j. && \text{(4.12c)}
\end{aligned}$$

Given s_j for $j \in \mathcal{N}^+$ and v_0, (4.12) determines uniquely s_0, (Λ_{ij}, S_{ij}) for $i \to j$, and v_j for $j \in \mathcal{N}^+$ as

$$\begin{aligned}
s_0 &= -\sum_{k\in\mathcal{N}^+} s_k^{\Phi_0}; && \\
\Lambda_{ij} &= -\sum_{k\in \mathrm{Down}(j)} s_k^{\Phi_{ij}}, && i \to j;\\
S_{ij} &= \gamma^{\Phi_{ij}} \mathrm{diag}(\Lambda_{ij}), && i \to j;\\
v_j &= v_0^{\Phi_j} - \sum_{(k,l)\in\mathcal{P}_j} \left[S_{kl} z_{kl}^H + z_{kl} S_{kl}^H \right]^{\Phi_j}, && j \in \mathcal{N}^+,
\end{aligned}$$

where $\mathcal{P}_j$ denotes the path from bus 0 to bus j, and Down(j) denotes the downstream of j, i.e.,

$$\mathrm{Down}(j) := \{k \in \mathcal{N} \mid j \in \mathcal{P}_k\},$$

for $j \in \mathcal{N}^+$.

LPF generalizes the *Simplified DistFlow Equations* [12] from single-phase networks to multiphase networks. While DC approximation assumes a constant voltage magnitude, ignores reactive power, and assumes $r_{ij} = 0$, LPF does not.

4.5 Case studies

In this section, we 1) check if BIM-SDP (BFM-SDP) can be solved by the generic solver *sedumi* [96]; 2) compare the running times of BIM-SDP and BFM-SDP; 3) compute how close the BIM-SDP (BFM-SDP) solutions are to rank one; and 4) evaluate the accuracy of LPF for the IEEE 13, 34, 37, 123-bus networks [1] and a real-world 2065-bus network.

The test networks are modeled by BIM and BFM with the following simplifications: 1) transformers are modeled as lines with appropriate impedances; 2) circuit switches are modeled as open or short lines depending on the status of the switch; 3) regulators are modeled as having a fixed voltage (the same as the substation); 4) distributed load on a line is modeled as two identical loads located at two end buses of the line; and 5) line shunt is modeled using the π model, assuming a fixed impedance load at each end of the line with the impedance being half of the line shunt [63]. The real-world network locates in a residential/commercial area in Southern California. All simulations are done on a laptop with Intel Core 2 Duo CPU at 2.66GHz, 4G RAM, and MAC OS 10.9.2, MATLAB R_2013a.

4.5.1 BIM-SDP vs BFM-SDP

OPF is set up as follows. The objective is power loss, i.e.,

$$C(s) = \sum_{i\in\mathcal{N}} \sum_{\phi\in\Phi_i} \mathrm{Re}(s_i^\phi).$$

The power injection constraint (4.2) is set up such that

1. for a bus i representing a shunt capacitor with nameplate capacity $\overline{q}_i$,

$$\mathcal{S}_i = \{s \in \mathbb{C}^{|\Phi_i|} \mid \mathrm{Re}(s_i) = 0,\ 0 \le \mathrm{Im}(s_i) \le \overline{q}_i\};$$

2. for a solar photovoltaic bus i with real power generation p_i and nameplate rating $\overline{s}_i$,

$$\mathcal{S}_i = \{s \in \mathbb{C}^{|\Phi_i|} \mid \mathrm{Re}(s_i) = p_i,\ |s_i| \le \overline{s}_i\};$$

3. for a bus i with multiple devices, $\mathcal{S}_i$ is the summation of the above mentioned sets. Here, the summation $A + B$ of two sets A and B are defined as $A + B := \{a + b \mid a \in A, b \in B\}$.

Two choices of the voltage constraint (4.3) are considered:

1. $\underline{V}_i^\phi = 0.95$ and $\overline{V}_i^\phi = 1.05$ for $i \in \mathcal{N}^+$ and $\phi \in \Phi_i$;
2. $\underline{V}_i^\phi = 0.90$ and $\overline{V}_i^\phi = 1.10$ for $i \in \mathcal{N}^+$ and $\phi \in \Phi_i$.

BIM-SDP and BFM-SDP are applied to solve OPF. In particular, the generic optimization solver *sedumi* is used to solve them and the results are summarized in Table 4.1 and 4.2.

Table 4.1: Simulation results with 5% voltage flexibility.

network	BIM-SDP			BFM-SDP		
	value	time	ratio	value	time	ratio
IEEE 13-bus	152.7	1.08	9.5e-9	152.7	0.79	1.6e-10
IEEE 34-bus	-100.0	1.97	1.0	5.001e-5	3.00	0.712
IEEE 37-bus	212.3	2.32	1.1e-8	212.3	2.00	9.0e-11
IEEE 123-bus	-7140	6.02	2.2e-2	229.8	7.55	0.5e-11
Rossi 2065-bus	-100.0	111.56	1.0	19.15	90.32	4.8e-8

Table 4.2: Simulation results with 10% voltage flexibility.

network	BIM-SDP			BFM-SDP		
	value	time	ratio	value	time	ratio
IEEE 13-bus	152.7	1.05	8.2e-9	152.7	0.74	2.8e-10
IEEE 34-bus	-100.0	2.22	1.0	279.0	1.64	3.3e-11
IEEE 37-bus	212.3	2.66	1.5e-8	212.2	1.95	1.3e-10
IEEE 123-bus	-8917	7.21	3.2e-2	229.8	8.86	0.6e-11
Rossi 2065-bus	-100.0	115.50	1.0	19.15	96.98	4.3e-8

Table 4.1 summarizes the simulation results with $\underline{V} = 0.95$ and $\overline{V} = 1.05$, and Table 4.2 summarizes the simulations results with $\underline{V} = 0.9$ and $\overline{V} = 1.1$. Each table contains the (value, time, ratio) triple for each of the (network, relaxation) pairs. For example, in Table 4.1, the (value, time, ratio) triple for the (IEEE 13-bus, BIM-SDP) pair is (152.7, 1.08, 9.5e-9).

The entry "value" is the objective value in kW. In the above example, with 5% voltage flexibility, the minimum power loss of the IEEE 13-bus network computed using BIM-SDP is 152.7kW. The entry "time" is the running time in second. In the above example, with 5% voltage flexibility, it takes 1.08s to solve BIM-SDP for the IEEE 13-bus network.

The entry "ratio" quantifies how close an SDP solution is to rank one. Due to finite numerical precision, even if BIM-SDP (BFM-SDP) is exact, its numerical solution only approximately satisfies (4.5g) [(4.10g)], i.e., the matrices in (4.5g) [(4.10g)] are only approximately rank one. To quantify how close the matrices are to rank one, one can compute their largest two eigenvalues λ_1, λ_2 ($|\lambda_1| \geq |\lambda_2| \geq 0$) and look at their ratios $|\lambda_2/\lambda_1|$. The smaller the ratios, the closer the matrices are to rank one. The maximum ratio over all matrices in (4.5g) [(4.10g)] is the entry "ratio". In the above example, with 5% voltage flexibility, the solution of BIM-SDP for the IEEE 13-bus network satisfies $|\lambda_2/\lambda_1| \leq 9.5 \times 10^{-9}$ for all matrices in (4.5g). Hence, BIM-SDP is numerically exact.

With 10% voltage flexibility, BFM-SDP is numerically exact for all test networks while BIM-SDP is numerically exact for only two test networks. This highlights that BFM-SDP is numerically more stable than BIM-SDP, since both SDPs should be exact simultaneously if there are infinite digits of precision. When voltage flexibility reduces to 5%, the OPF problem for the IEEE 13-bus network becomes infeasible. Consequently, BFM-SDP is not numerically exact in this case.

To summarize, BFM-SDP is numerically exact for up to 2000-bus networks when OPF is feasible, while BIM-SDP gets into numerical difficulties for as few as 34-bus networks.

4.5.2 Accuracy of LPF

Now we evaluate the accuracy of LPF (4.12). In particular, given the optimal power injections computed by BFM-SDP in Section 4.5.1, we use the forward backward sweep algorithm (FBS) to obtain the real power flows and voltage magnitudes [63], use LPF to estimate the power flows and voltage magnitudes, and compare their differences. The results are summarized in Table 4.3.

Table 4.3: Accuracy of LPF.

network	time		error	
	FBS	LBF	V (p.u.)	S (%)
IEEE 13-bus	0.11s	0.03s	4.5e-4	3.1
IEEE 34-bus	0.16s	0.02s	1.0e-3	4.2
IEEE 37-bus	0.12s	0.02s	2.0e-4	1.5
IEEE 123-bus	0.37s	0.07s	5.5e-4	3.3
Rossi 2065-bus	4.73s	0.98s	1.6e-3	5.3

It can be seen that the voltages are within 0.0016 per unit and the power flows are within 5.3% of their true values for all test networks. This highlights the accuracy of LPF (4.12).

4.6 Conclusions

Two convex relaxations, BIM-SDP and BFM-SDP, have been presented to solve OPF in multiphase radial networks. BIM-SDP explores the radial network topology to improve the computational efficiency of a standard SDP relaxation, and BFM-SDP avoids ill-conditioned operations to enhance the numerical stability of BIM-SDP. We have proved that BIM-SDP is exact if and only if BFM-SDP is exact.

A linear approximation LPF has been proposed to estimate the power flows and voltages in multiphase radial networks. LPF is accurate when line loss is small and voltages are nearly balanced. Case studies show that BFM-SDP is numerically exact when OPF is feasible and LPF obtains voltages within 0.0016 per unit of their true values for the IEEE 13, 34, 37, 123-bus networks and a real-world 2065-bus network.

Appendix

4.A Proof of Lemma 4.1

We prove that Algorithm 3 computes a V that satisfies $V_0 = V_0^{\mathrm{ref}}$ and (4.4). The proof of uniqueness of such V is straightforward and omitted for brevity.

Let $\mathcal{N}^{(0)} := \{0\}$ and $\mathcal{N}^{(k)}$ denote the set $\mathcal{N}_{\mathrm{visit}}$ after iteration $k = 1, 2, \ldots, n$ of Algorithm 3. Let $\mathcal{E}^{(k)} := \mathcal{E} \cap \mathcal{N}^{(k)} \times \mathcal{N}^{(k)}$ denote the edges of the subgraph induced by $\mathcal{N}^{(k)}$ for $k = 0, 1, \ldots, n$.

After iteration $k \geq 0$, voltage V_i is recovered for $i \in \mathcal{N}^{(k)}$. In particular, after iteration n, V_i is recovered for $i \in \mathcal{N}^{(n)} = \mathcal{N}$. Hence, it suffices to prove

$$v_i = V_i V_i^H, \qquad i \in \mathcal{N}^{(k)}; \tag{4.13a}$$

$$W_{ij} = V_i^{\Phi_{ij}} V_j^H,\ W_{ji} = V_j (V_i^{\Phi_{ij}})^H, \qquad (i,j) \in \mathcal{E}^{(k)} \tag{4.13b}$$

for $k = 0, 1, \ldots, n$.

We prove (4.13) by induction. When $k = 0$, (4.13) holds trivially. Assume that (4.13) holds for $k = K$ $(0 \leq K \leq n-1)$, we prove that (4.13) holds for $k = K+1$ as follows.

Let $j = \mathcal{N}^{(k)} \backslash \mathcal{N}^{(k-1)}$. Since the network $(\mathcal{N}, \mathcal{E})$ is radial, there exists a unique i such that $i \to j$. Furthermore, $i \in \mathcal{N}^{(k-1)}$. It suffices to prove

$$v_j = V_j V_j^H, \quad W_{ij} = V_i^{\Phi_{ij}} V_j^H, \quad W_{ji} = V_j (V_i^{\Phi_{ij}})^H.$$

Since the matrix

$$\begin{bmatrix} v_i^{\Phi_{ij}} & W_{ij} \\ W_{ji} & v_j \end{bmatrix}$$

is hermitian and rank one, there exists $\alpha, \beta \in \mathbb{C}^{|\Phi_{ij}|}$ such that

$$\begin{bmatrix} v_i^{\Phi_{ij}} & W_{ij} \\ W_{ji} & v_j \end{bmatrix} = \eta \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \begin{bmatrix} \alpha^H & \beta^H \end{bmatrix}$$

where $\eta = \pm 1$. Since $v_i^{\Phi_{ij}} = V_i^{\Phi_{ij}}(V_i^{\Phi_{ij}})^H \succeq 0$ and $v_i^{\Phi_{ij}} \neq 0$, one has $\eta = 1$ and therefore

$$v_i^{\Phi_{ij}} = \alpha\alpha^H, \quad W_{ij} = \alpha\beta^H, \quad W_{ji} = \beta\alpha^H, \quad v_j = \beta\beta^H.$$

Furthermore, $V_i^{\Phi_{ij}} = \alpha\exp(\mathbf{i}\theta)$ for some $\theta \in \mathbb{R}$ since

$$V_i^{\Phi_{ij}}(V_i^{\Phi_{ij}})^H = v_i^{\Phi_{ij}} = \alpha\alpha^H.$$

It follows that

$$V_j \;=\; \frac{1}{\operatorname{tr}\left(v_i^{\Phi_{ij}}\right)} W_{ji} V_i^{\Phi_{ij}} \;=\; \frac{1}{\operatorname{tr}\left(\alpha\alpha^H\right)} \beta\alpha^H\alpha\exp(\mathbf{i}\theta) \;=\; \beta\exp(\mathbf{i}\theta).$$

Then, it is straightforward to verify that

$$\begin{aligned} V_j V_j^H &= \beta\beta^H = v_j, \\ V_i^{\Phi_{ij}} V_j^H &= \alpha\beta^H = W_{ij}, \\ V_j (V_i^{\Phi_{ij}})^H &= \beta\alpha^H = W_{ji}. \end{aligned}$$

This completes the proof that Algorithm 3 computes a V that satisfies $V_0 = V_0^{\text{ref}}$ and (4.4).

4.B Proof of Theorem 4.3

First prove $\mathbb{F}_{\text{BIM}} \subseteq \mathbb{F}_{\text{BFM}}$. Let $(s, V) \in \mathbb{F}_{\text{BIM}}$, and we want to prove $(s, V) \in \mathbb{F}_{\text{BFM}}$. Let $I_{ij} = y_{ij}(V_i^{\Phi_{ij}} - V_j^{\Phi_{ij}})$ for $i \sim j$, then (V, I) satisfies (4.6). Define (ℓ, S) according to (4.7). It suffices to prove (s, ℓ, S) satisfies (4.8). This is because

$$\begin{aligned} \sum_{k:\, j\to k} \operatorname{diag}(S_{jk})^{\Phi_j} - \sum_{i:\, i\to j} \operatorname{diag}\left(S_{ij} - z_{ij}\ell_{ij}\right) &= \sum_{k:\, j\to k} \operatorname{diag}\left(V_j^{\Phi_{jk}} I_{jk}^H\right)^{\Phi_j} - \sum_{i:\, i\to j} \operatorname{diag}\left(V_j I_{ij}^H\right) \\ &= \sum_{i:\, i\sim j} \operatorname{diag}\left(V_j^{\Phi_{ij}} I_{ji}^H\right)^{\Phi_j} \\ &= \sum_{i:\, i\sim j} \operatorname{diag}\left[V_j^{\Phi_{ij}} (V_j^{\Phi_{ij}} - V_i^{\Phi_{ij}})^H y_{ij}^H\right]^{\Phi_j} = s_j \end{aligned}$$

for $j \in \mathcal{N}$. This completes the proof of $\mathbb{F}_{\text{BIM}} \subseteq \mathbb{F}_{\text{BFM}}$.

Next prove $\mathbb{F}_{\text{BFM}} \subseteq \mathbb{F}_{\text{BIM}}$. Let $(s, V) \in \mathbb{F}_{\text{BFM}}$, and we want to prove $(s, V) \in \mathbb{F}_{\text{BIM}}$. Let (I, ℓ, S) be such that (s, V, I, ℓ, S) satisfies (4.6)–(4.8). It suffices to prove (s, V) satisfies (4.1). This is because

$$\begin{aligned}\sum_{i:\, i\sim j} \operatorname{diag}\left[V_j^{\Phi_{ij}}(V_j^{\Phi_{ij}} - V_i^{\Phi_{ij}})^H y_{ij}^H\right]^{\Phi_j} &= \sum_{k:\, j\to k} \operatorname{diag}\left(V_j^{\Phi_{jk}} I_{jk}^H\right)^{\Phi_j} - \sum_{i:\, i\to j} \operatorname{diag}\left(V_j^{\Phi_{ij}} I_{ij}^H\right)^{\Phi_j} \\ &= \sum_{k:\, j\to k} \operatorname{diag}(S_{jk})^{\Phi_j} - \sum_{i:\, i\to j} \operatorname{diag}\left[(V_i^{\Phi_{ij}} - z_{ij} I_{ij}) I_{ij}^H\right] \\ &= \sum_{k:\, j\to k} \operatorname{diag}(S_{jk})^{\Phi_j} - \sum_{i:\, i\to j} \operatorname{diag}\left(S_{ij} - z_{ij}\ell_{ij}\right) = s_j\end{aligned}$$

for $j \in \mathcal{N}$. This completes the proof of Theorem 4.3.

4.C Proof of Lemma 4.4

We prove that Algorithm 4 computes a (V, I) that satisfies $V_0 = V_0^{\text{ref}}$, (4.4a), (4.6), and (4.7). The proof of uniqueness of such (V, I) is straightforward and omitted for brevity.

Let $\mathcal{N}^{(0)} := \{0\}$ and $\mathcal{N}^{(k)}$ denote the set $\mathcal{N}_{\text{visit}}$ after iteration $k = 1, 2, \ldots, n$ of Algorithm 4. Let $\mathcal{E}^{(k)} := \mathcal{E} \cap \mathcal{N}^{(k)} \times \mathcal{N}^{(k)}$ denote the edges of the subgraph induced by $\mathcal{N}^{(k)}$ for $k = 0, 1, \ldots, n$.

After iteration $k \geq 0$, voltage V_i is recovered for $i \in \mathcal{N}^{(k)}$ and current I_{ij} is recovered for $(i, j) \in \mathcal{E}^{(k)}$. In particular, after iteration n, V_i is recovered for $i \in \mathcal{N}^{(n)} = \mathcal{N}$ and I_{ij} is recovered for $(i, j) \in \mathcal{E}^{(n)} = \mathcal{E}$. Hence, it suffices to prove

$$v_i = V_i V_i^H, \qquad i \in \mathcal{N}^{(k)}; \tag{4.14a}$$

$$V_i^{\Phi_{ij}} - V_j = z_{ij} I_{ij},\ \ell_{ij} = I_{ij} I_{ij}^H,\ S_{ij} = V_i^{\Phi_{ij}} I_{ij}^H, \qquad (i, j) \in \mathcal{E}^{(k)} \tag{4.14b}$$

for $k = 0, 1, \ldots, n$.

We prove (4.14) by induction. When $k = 0$, (4.14) holds trivially. Assume that (4.14) holds for $k = K$ $(0 \leq K \leq n - 1)$, we prove that (4.14) holds for $k = K + 1$ as follows.

Let $j = \mathcal{N}^{(k)} \backslash \mathcal{N}^{(k-1)}$. Since the network $(\mathcal{N}, \mathcal{E})$ is radial, there exists a unique i such that $i \to j$. Furthermore, $i \in \mathcal{N}^{(k-1)}$. It suffices to prove

$$v_j = V_j V_j^H,\ V_i^{\Phi_{ij}} - V_j = z_{ij} I_{ij},\ \ell_{ij} = I_{ij} I_{ij}^H,\ S_{ij} = V_i^{\Phi_{ij}} I_{ij}^H.$$

Since the matrix

$$\begin{bmatrix} v_i^{\Phi_{ij}} & S_{ij} \\ S_{ij}^H & \ell_{ij} \end{bmatrix}$$

is hermitian and rank one, there exists $\alpha, \beta \in \mathbb{C}^{|\Phi_{ij}|}$ such that

$$\begin{bmatrix} v_i^{\Phi_{ij}} & S_{ij} \\ S_{ij}^H & \ell_{ij} \end{bmatrix} = \eta \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \begin{bmatrix} \alpha^H & \beta^H \end{bmatrix}$$

where $\eta = \pm 1$. Since $v_i^{\Phi_{ij}} = V_i^{\Phi_{ij}} (V_i^{\Phi_{ij}})^H \succeq 0$ and $v_i^{\Phi_{ij}} \neq 0$, one has $\eta = 1$ and therefore

$$v_i^{\Phi_{ij}} = \alpha\alpha^H, \quad S_{ij} = \alpha\beta^H, \quad \ell_{ij} = \beta\beta^H.$$

Furthermore, $V_i^{\Phi_{ij}} = \alpha \exp(\mathbf{i}\theta)$ for some $\theta \in \mathbb{R}$ since

$$V_i^{\Phi_{ij}} (V_i^{\Phi_{ij}})^H = v_i^{\Phi_{ij}} = \alpha\alpha^H.$$

It follows that

$$I_{ij} \;=\; \frac{1}{\operatorname{tr}\left(v_i^{\Phi_{ij}}\right)} S_{ij}^H V_i^{\Phi_{ij}} \;=\; \frac{1}{\operatorname{tr}\left(\alpha\alpha^H\right)} \beta\alpha^H \alpha \exp(\mathbf{i}\theta) \;=\; \beta \exp(\mathbf{i}\theta)$$

and

$$V_j \;=\; V_i^{\Phi_{ij}} - z_{ij} I_{ij} \;=\; \alpha \exp(\mathbf{i}\theta) - z_{ij}\beta \exp(\mathbf{i}\theta) \;=\; (\alpha - z_{ij}\beta)\exp(\mathbf{i}\theta).$$

Then, it is straightforward to verify that

$$\begin{aligned} V_j V_j^H &\;=\; (\alpha - z_{ij}\beta)(\alpha - z_{ij}\beta)^H \;=\; v_i^{\Phi_{ij}} - (z_{ij} S_{ij}^H + S_{ij} z_{ij}^H) + z_{ij}\ell_{ij} z_{ij}^H \;=\; v_j, \\ I_{ij} I_{ij}^H &\;=\; \beta\beta^H \;=\; \ell_{ij}, \\ V_i^{\Phi_{ij}} I_{ij}^H &\;=\; \alpha\beta^H \;=\; S_{ij}. \end{aligned}$$

This completes the proof that Algorithm 4 computes a (V, I) that satisfies $V_0 = V_0^{\text{ref}}$, (4.4a), (4.6), and (4.7).

4.D Proof of Theorem 4.6

First prove that $f(s, v, W) \in \mathbb{F}_{\text{BFM-SDP}}$ for any $(s, v, W) \in \mathbb{F}_{\text{BIM-SDP}}$. Let $(s, v, W) \in \mathbb{F}_{\text{BIM-SDP}}$, let $(s, v, S, \ell) = f(s, v, W)$, and we want to prove $(s, v, S, \ell) \in \mathbb{F}_{\text{BFM-SDP}}$.

It is straightforward that (s, v, S, ℓ) satisfies (4.10b)–(4.10d). The point (s, v, S, ℓ) satisfies (4.10a) because

$$S_{ij} - z_{ij}\ell_{ij} = (v_i^{\Phi_{ij}} - W_{ij})y_{ij}^H - (v_i^{\Phi_{ij}} - W_{ij} - W_{ji} + v_j)y_{ij}^H = -(v_j - W_{ji})y_{ij}^H$$

for $i \to j$ and therefore

$$\begin{aligned}
&\sum_{k:\, j\to k} \mathrm{diag}(S_{jk})^{\Phi_j} - \sum_{i:\, i\to j} \mathrm{diag}\left(S_{ij} - z_{ij}\ell_{ij}\right) \\
&= \sum_{k:\, j\to k} \mathrm{diag}\left[(v_j^{\Phi_{jk}} - W_{jk})y_{jk}^H\right]^{\Phi_j} + \sum_{i:\, i\to j} \mathrm{diag}\left[(v_j - W_{ji})y_{ij}^H\right] \\
&= \sum_{i:\, i\sim j} \mathrm{diag}\left[(v_j^{\Phi_{ij}} - W_{ji})y_{ji}^H\right]^{\Phi_j} \;=\; s_j
\end{aligned}$$

for $j \in \mathcal{N}$. The point (s, v, S, ℓ) satisfies (4.10e) because

$$\begin{aligned}
&v_i^{\Phi_{ij}} - (S_{ij}z_{ij}^H + z_{ij}S_{ij}^H) + z_{ij}\ell_{ij}z_{ij}^H \\
&= v_i^{\Phi_{ij}} - (v_i^{\Phi_{ij}} - W_{ij} + v_i^{\Phi_{ij}} - W_{ji}) + v_i^{\Phi_{ij}} - W_{ij} - W_{ji} + v_j \\
&= v_j
\end{aligned}$$

for $i \to j$. The point (s, v, S, ℓ) satisfies (4.10f) because

$$\begin{aligned}
&\begin{bmatrix} v_i^{\Phi_{ij}} & S_{ij} \\ S_{ij}^H & \ell_{ij} \end{bmatrix} \succeq 0 \\
\Leftrightarrow\;& v_i^{\Phi_{ij}} \succeq 0,\; S_{ij} \in \mathrm{R}(v_i^{\Phi_{ij}}),\; \ell_{ij} \succeq S_{ij}^H (v_i^{\Phi_{ij}})^+ S_{ij} \\
\Leftrightarrow\;& v_i^{\Phi_{ij}} \succeq 0,\; W_{ij} \in \mathrm{R}(v_i^{\Phi_{ij}}),\; v_i^{\Phi_{ij}} - W_{ij} - W_{ji} + v_j \succeq (v_i^{\Phi_{ij}} - W_{ji})(v_i^{\Phi_{ij}})^+(v_i^{\Phi_{ij}} - W_{ij}) \\
\Leftrightarrow\;& v_i^{\Phi_{ij}} \succeq 0,\; W_{ij} \in \mathrm{R}(v_i^{\Phi_{ij}}),\; v_j \succeq W_{ji}(v_i^{\Phi_{ij}})^+ W_{ij} \\
\Leftrightarrow\;& \begin{bmatrix} v_i^{\Phi_{ij}} & W_{ij} \\ W_{ji} & v_j \end{bmatrix} \succeq 0
\end{aligned} \tag{4.15}$$

for $i \to j$. This completes the proof that $f(s, v, W) \in \mathbb{F}_{\text{BFM-SDP}}$ for any $(s, v, W) \in \mathbb{F}_{\text{BIM-SDP}}$.

Next show that $g(s, v, S, \ell) \in \mathbb{F}_{\text{BIM-SDP}}$ for any $(s, v, S, \ell) \in \mathbb{F}_{\text{BFM-SDP}}$. Let $(s, v, S, \ell) \in \mathbb{F}_{\text{BFM-SDP}}$, let $g(s, v, S, \ell) = (s, v, W)$, and we want to prove $(s, v, W) \in \mathbb{F}_{\text{BIM-SDP}}$.

It is straightforward that (s, v, W) satisfies (4.5b)–(4.5e). The point (s, v, W) satisfies (4.5a) because

$$(v_j - W_{ji})y_{ij}^H \;=\; -(v_i^{\Phi_{ij}} - z_{ij}S_{ij}^H - v_j)y_{ij}^H \;=\; -(S_{ij}z_{ij}^H - z_{ij}\ell_{ij}z_{ij}^H)y_{ij}^H \;=\; -(S_{ij} - z_{ij}\ell_{ij})$$

for $i \to j$ and therefore

$$\begin{aligned}
&\sum_{i:\, i\sim j} \operatorname{diag}\left[(v_j^{\Phi_{ij}} - W_{ji})y_{ji}^H\right]^{\Phi_j} \\
&= \sum_{i:\, i\to j} \operatorname{diag}\left[(v_j - W_{ji})y_{ji}^H\right] + \sum_{k:\, j\to k} \operatorname{diag}\left[(v_j^{\Phi_{jk}} - W_{jk})y_{jk}^H\right]^{\Phi_j} \\
&= -\sum_{i:\, i\to j} \operatorname{diag}(S_{ij} - z_{ij}\ell_{ij}) + \sum_{k:\, j\to k} \operatorname{diag}(S_{jk})^{\Phi_j} = s_j
\end{aligned}$$

for $j \in \mathcal{N}$. The point (s, v, W) satisfies (4.5f) due to (4.15). This completes the proof that $g(s, v, S, \ell) \in \mathbb{F}_{\text{BIM-SDP}}$ for any $(s, v, S, \ell) \in \mathbb{F}_{\text{BFM-SDP}}$.

It is straightforward to verify that $f \circ g$ and $g \circ f$ are both identity maps. This completes the proof of Theorem 4.6.

4.E Proof of Theorem 4.8

Let $(s, v, W) \in \mathbb{F}_{\text{BIM-SDP}}$ and $(s, v, S, \ell) = f(s, v, W)$. It suffices to prove that

$$\operatorname{rank}\begin{bmatrix} v_i^{\Phi_{ij}} & S_{ij} \\ S_{ij}^H & \ell_{ij} \end{bmatrix} = 1 \iff \operatorname{rank}\begin{bmatrix} v_i^{\Phi_{ij}} & W_{ij} \\ W_{ji} & v_j \end{bmatrix} = 1$$

for $i \to j$. Fix an arbitrary $i \to j$. Since $\operatorname{rank}(v_i^{\Phi_{ij}}) \ge 1$ by (4.5d),

$$\begin{aligned}
&\operatorname{rank}\begin{bmatrix} v_i^{\Phi_{ij}} & S_{ij} \\ S_{ij}^H & \ell_{ij} \end{bmatrix} = 1 \\
\Leftrightarrow\ &\operatorname{rank}(v_i^{\Phi_{ij}}) = 1,\ S_{ij} \in \mathrm{R}(v_i^{\Phi_{ij}}),\ \ell_{ij} = S_{ij}^H (v_i^{\Phi_{ij}})^+ S_{ij} \\
\Leftrightarrow\ &\operatorname{rank}(v_i^{\Phi_{ij}}) = 1,\ W_{ij} \in \mathrm{R}(v_i^{\Phi_{ij}}),\ v_i^{\Phi_{ij}} - W_{ji} - W_{ij} + v_j = (v_i^{\Phi_{ij}} - W_{ji})(v_i^{\Phi_{ij}})^+(v_i^{\Phi_{ij}} - W_{ij}) \\
\Leftrightarrow\ &\operatorname{rank}(v_i^{\Phi_{ij}}) = 1,\ W_{ij} \in \mathrm{R}(v_i^{\Phi_{ij}}),\ v_j = W_{ji}(v_i^{\Phi_{ij}})^+ W_{ij} \\
\Leftrightarrow\ &\operatorname{rank}\begin{bmatrix} v_i^{\Phi_{ij}} & W_{ij} \\ W_{ji} & v_j \end{bmatrix} = 1.
\end{aligned}$$

This completes the proof of Theorem 4.8.

Chapter 5

An OPF Solver

In this chapter, we derive a distributed algorithm for solving the OPF problem, and provide a suboptimality bound for the solution it obtains. Simulation results indicate that the suboptimality is within 2e-7, and that the distributed algorithm is much more efficient to compute than the convex relaxation method, for a series of real-world networks.

Literature The OPF problem is difficult to solve since power flow is governed by nonlinear physical laws. There are three ways to deal with this challenge: 1) approximate the power flow equations (by linear or easier nonlinear equations); 2) look for local optima of the OPF problem; and 3) convexify the constraints imposed by the nonlinear physical laws. The third approach has been explored in the previous chapter, and we focus on the first two approaches in this chapter.

Power flow equations can be approximated by some linear equations known as the DC power flow equations [7, 94, 95] if 1) power losses on the lines are small; 2) voltages are close to their nominal values; and 3) voltage angle differences between adjacent buses are small. With the DC power flow approximation, the OPF problem reduces to a linear programming. For transmission networks, the three assumptions are satisfied and the approach is widely used in practice. However, DC power flow equations do not consider voltages and reactive power flows, and therefore cannot be used for applications like power routing [108] and volt/var control [101]. Besides, the obtained solution may not be implementable since physical laws are not fully respected. Moreover, for distribution networks, power losses on the lines are not negligible and voltages can deviate significantly from their nominal values. Consequently, DC power flow equations are not accurate enough for distribution networks.

A number of algorithms look for a local optimum of the OPF problem. These algorithms use nonlinear power flow equations and therefore 1) can be used in applications like voltage regulation and volt/var control; 2) have physically implementable solutions; and 3) apply to both transmission and distribution networks. Representative algorithms of this kind include successive linear/quadratic programming [32], trust-region based methods [78,92], Lagrangian Newton method [11], and interior-point methods [21, 59, 100]. Some of these algorithms, especially those based on Newton-Ralphson, are quite successful empirically. However, these algorithms may not converge, and suboptimality of their solutions is rarely characterized.

Summary The distributed algorithm presented in this chapter also seeks a local optimum of OPF. It differs from other algorithms of this kind in its 1) guaranteed convergence, 2) high computational efficiency, and 3) well-characterizable suboptimality.

The algorithm implements gradient projection in a distributed manner. In each iteration of the algorithm, derivatives of the objective function with respect to controllable variables are estimated, and then used as the negative directions to update controllable variables. Derivatives are estimated using a linear approximation of the power flow equations, and line search is adopted to determine the step sizes of the updates.

The rest of this chapter is organized as follows. A simplified OPF that illustrates the main structure of the distributed algorithm is described in Section 5.1. The algorithm is provided in Section 5.2 and numerical results are presented in Section 5.5.

5.1 A Simplified OPF Problem

The OPF problem in distribution networks has been described in Section 4.1. To highlight the main components of the distributed algorithm without worrying about the complication of multiple phases, we assume that the network is single-phase and radial. In particular, the bus voltages V_i, bus power injections s_i, and branch power flows S_{ij} are complex scalars.

For each bus $i \in \mathcal{N}$, let $v_i = |V_i|^2$ denote the square of its complex voltage magnitude, e.g., if the voltage is $V_i = 1.05\angle 120^\circ$ per unit, then $v_i = 1.05^2$. Note that v_0 is fixed and given. Let $p_i = \mathrm{Re}(s_i)$ and $q_i = \mathrm{Im}(s_i)$ denote the real and reactive power injections, respectively. Let $\mathcal{P}_i$ denote the unique path from bus 0 to bus i. Since the network is radial, the path $\mathcal{P}_i$ is well-defined. For each line $(i, j) \in \mathcal{E}$, let $\ell_{ij} = |I_{ij}|^2$ denote the square of the complex current magnitude, e.g., if the current is $I_{ij} = 0.5\angle 10^\circ$, then $\ell_{ij} = 0.5^2$. Let $P_{ij} = \mathrm{Re}(S_{ij})$ and $Q_{ij} = \mathrm{Im}(S_{ij})$ denote the real and reactive power flows, respectively. Let $r_{ij} = \mathrm{Re}(z_{ij})$ and $x_{ij} = \mathrm{Im}(z_{ij})$ denote the resistance and reactance, respectively. Some of the notations are summarized in Figure 5.1.

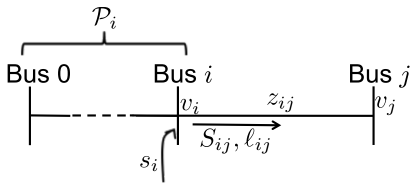


Figure 5.1: Some of the notations.

The simplified OPF problem that we seek to solve is as follows.

$$
\begin{aligned}
\min \quad & \sum_{i=0}^{n}(a_i p_i^2 + b_i p_i) && \text{(5.1a)}\\
\text{over} \quad & p_i \text{ and } q_i \text{ for } i \in \mathcal{N}^+; &&\\
& p_0, q_0,\ v_i \text{ for } i \in \mathcal{N}^+,\ P_{ij}, Q_{ij} \text{ and } \ell_{ij} \text{ for } i \to j; &&\\
\text{s.t.} \quad & \sum_{i:\, i\to j} (P_{ij} - r_{ij}\ell_{ij}) + p_j = \sum_{k:\, j\to k} P_{jk}, \qquad j \in \mathcal{N}; && \text{(5.1b)}\\
& \sum_{i:\, i\to j} (Q_{ij} - x_{ij}\ell_{ij}) + q_j = \sum_{k:\, j\to k} Q_{jk}, \qquad j \in \mathcal{N}; && \text{(5.1c)}\\
& v_i - v_j = 2(r_{ij}P_{ij} + x_{ij}Q_{ij}) - |z_{ij}|^2 \ell_{ij}, \qquad i \to j; && \text{(5.1d)}\\
& \ell_{ij} = \frac{P_{ij}^2 + Q_{ij}^2}{v_i}, \qquad i \to j; && \text{(5.1e)}\\
& \underline{v}_i \le v_i \le \overline{v}_i, \qquad i \in \mathcal{N}^+; && \text{(5.1f)}\\
& p_i \in \left[\underline{p}_i, \overline{p}_i\right],\ q_i \in \left[\underline{q}_i, \overline{q}_i\right], \qquad i \in \mathcal{N}^+. && \text{(5.1g)}
\end{aligned}
$$

The objective function (5.1a) is assumed to be separable, quadratic, and purely a function of p. Equations (5.1b)–(5.1e) are obtained from (4.10a), (4.10e), and (4.10g). They describe the physical laws that govern the power flow. Equation (5.1f) is the voltage regulation constraints, and equation (5.1g) is the power injection constraints. The results in this chapter generalize to arbitrary convex power injection constrains other than the form described by (5.1g).

5.1.1 Basic and Derived Variables

The following assumption is the foundation of the algorithm developed in this chapter:

Assumption 5.1. *Given the substation voltage v_0 and the branch bus power injections s_i for $i \in \mathcal{N}^+$, we assume there exists a unique "practical" power flow solution $([v_i]_{i\in\mathcal{N}^+}, [\ell_{ij}]_{i\to j}, [S_{ij}]_{i\to j}, s_0)$.*

Here "practical" means that the voltages v_i are close to their nominal values of 1.0 per unit. It is justified in [12] that all other power flow solutions have $v_i \approx 0$ for some $i \in \mathcal{N}^+$. In fact, Assumption 5.1 is commonly acknowledged by the industry. With this assumption, the optimization variables can be classified into two categories:

1. Basic variables that are controllable. These variables include the branch bus real and reactive power injections p_i and q_i for $i \in \mathcal{N}^+$.

2. Derived variables that are uniquely determined [by power flow equations (5.1b)–(5.1e)] after specifying the basic variables. These variables include the substation power injection p_0, q_0, the branch bus voltages v_i for $i \in \mathcal{N}^+$, and line flows P_{ij}, Q_{ij}, ℓ_{ij} for $i \to j$.

Let $x = (p_1, \ldots, p_n, q_1, \ldots, q_n)$ denote all basic variables, then derived variables are functions of x, i.e.,

$$\begin{aligned}
& p_0 = p_0(x),\ q_0 = q_0(x); && \\
& v_i = v_i(x), && i \in \mathcal{N}^+; \\
& P_{ij} = P_{ij}(x),\ Q_{ij} = Q_{ij}(x),\ \ell_{ij} = \ell_{ij}(x), && i \to j.
\end{aligned}$$

Remove derived variables to transform the OPF problem (5.1) into the following form:

$$\begin{aligned}
\min \quad & a_0 p_0^2(x) + b_0 p_0(x) + \sum_{i=1}^{n} (a_i p_i^2 + b_i p_i) \\
\text{over} \quad & x \\
\text{s.t.} \quad & \underline{v}_i \le v_i(x) \le \overline{v}_i, \qquad i \in \mathcal{N}^+; && (5.2a) \\
& \underline{p}_i \le p_i \le \overline{p}_i,\ \underline{q}_i \le q_i \le \overline{q}_i, \qquad i \in \mathcal{N}^+. && (5.2b)
\end{aligned}$$

While (5.2) is equivalent to (5.1) for radial networks under Assumption 5.1, (5.2) has much fewer optimization variables than (5.1) and is therefore potentially more efficient to compute. Furthermore, the derived variables $(p_0, [v_i]_{i \in \mathcal{N}^+})$ will be automatically computed by power flow physics once the basic variable x is computed. This motivates an iterative procedure for solving OPF: in each iteration, first update the basic variable x and then let derived variables be automatically computed by power flow physics.

5.2 A Gradient Projection Algorithm

The algorithm that we propose to solve the OPF problem (5.2) is presented in this section. It is a gradient projection algorithm. In each iteration of the algorithm, derivatives of the modified objective function with respect to the basic variables are estimated, and then used as the negative direction of updating the basic variables. Line search is implemented to determine the step sizes of basic variable updates so as to ensure the convergence of the algorithm.

5.2.1 Estimation of Derivatives

One of the key reasons that the gradient projection algorithm we propose in this chapter is efficient is that the derivatives can be estimated efficiently. In this section, we first compute the derivatives exactly, and then propose an estimation of the derivatives that can be done much more efficiently.

5.2.1.1 Exact Derivatives

We first compute the gradients exactly. Since the network is radial, (5.1b)–(5.1e) imply

$$\partial_x P_{ij} = r_{ij}\partial_x \ell_{ij} - \partial_x p_j + \sum_{k\colon j\to k} \partial_x P_{jk}, \tag{5.3a}$$

$$\partial_x Q_{ij} = x_{ij}\partial_x \ell_{ij} - \partial_x q_j + \sum_{k\colon j\to k} \partial_x Q_{jk}, \tag{5.3b}$$

$$\partial_x v_j = \partial_x v_i - 2\left(r_{ij}\partial_x P_{ij} + x_{ij}\partial_x Q_{ij}\right) + |z_{ij}|^2\partial_x \ell_{ij}, \tag{5.3c}$$

$$\partial_x \ell_{ij} = \frac{2P_{ij}}{v_i}\partial_x P_{ij} + \frac{2Q_{ij}}{v_i}\partial_x Q_{ij} - \frac{\ell_{ij}}{v_i}\partial_x v_i \tag{5.3d}$$

for $i \to j$. Let I denote the 2×2 identity matrix and remove $\partial_x \ell_{ij}$ to obtain

$$\left[I - \frac{2}{v_i}\begin{pmatrix} r_{ij} \\ x_{ij}\end{pmatrix}\begin{pmatrix} P_{ij} & Q_{ij}\end{pmatrix}\right]\begin{pmatrix}\partial_x P_{ij} \\ \partial_x Q_{ij}\end{pmatrix} = \sum_{k\colon j\to k}\begin{pmatrix}\partial_x P_{jk} \\ \partial_x Q_{jk}\end{pmatrix} - \begin{pmatrix}\partial_x p_j \\ \partial_x q_j\end{pmatrix} - \begin{pmatrix} r_{ij} \\ x_{ij}\end{pmatrix}\frac{\ell_{ij}}{v_i}\partial_x v_i,$$
$$\partial_x v_j = \left(1 - |z_{ij}|^2\frac{\ell_{ij}}{v_i}\right)\partial_x v_i - 2\left(r_{ij} - |z_{ij}|^2\frac{P_{ij}}{v_i}\right)\partial_x P_{ij} - 2\left(x_{ij} - |z_{ij}|^2\frac{Q_{ij}}{v_i}\right)\partial_x Q_{ij}$$

for $i \to j$. Hence, the gradients $\partial_x(P, Q, v, p_0, q_0)$ can be computed by Algorithm 5.

Algorithm 5 Compute gradients

```
Input: network graph $(\mathcal{N}, \mathcal{E})$, power flow solution $(p, q, P, Q, v, \ell)$, stopping criterion $\epsilon$.
Output: gradient $\partial_x(P, Q, v, p_0, q_0)$ where $x = (p_1, \ldots, p_n, q_1, \ldots, q_n)$.
1: Initialization.
       $\partial_x v_i \leftarrow 0$ for $i = 0, 1, \ldots, n$;
2: Backward sweep.
       From the leafs $j \to k$ to the roots $i \to j$, compute
       $\begin{pmatrix}\partial_x P_{ij} \\ \partial_x Q_{ij}\end{pmatrix} \leftarrow \left[I - \frac{2}{v_i}\begin{pmatrix} r_{ij} \\ x_{ij}\end{pmatrix}\begin{pmatrix} P_{ij} & Q_{ij}\end{pmatrix}\right]^{-1}\left[\sum_{k\colon j\to k}\begin{pmatrix}\partial_x P_{jk} \\ \partial_x Q_{jk}\end{pmatrix} - \begin{pmatrix}\partial_x p_j \\ \partial_x q_j\end{pmatrix} - \begin{pmatrix} r_{ij} \\ x_{ij}\end{pmatrix}\frac{\ell_{ij}}{v_i}\partial_x v_i\right]$;
3: Forward sweep.
       From the roots $i$ to the leafs $j$, compute
       $\partial_x v_j \leftarrow \left(1 - |z_{ij}|^2\frac{\ell_{ij}}{v_i}\right)\partial_x v_i - 2\left(r_{ij} - |z_{ij}|^2\frac{P_{ij}}{v_i}\right)\partial_x P_{ij} - 2\left(x_{ij} - z_{ij}^2\frac{Q_{ij}}{v_i}\right)\partial_x Q_{ij}$;
4: if update in $\partial_x(P, Q, v) > \epsilon$
       go to 2;
   end
5: Return value.
   $\begin{pmatrix}\partial_x p_0 \\ \partial_x q_0\end{pmatrix} \leftarrow \sum_{k\colon 0\to k}\left[I - \frac{2}{v_0}\begin{pmatrix} r_{0k} \\ x_{0k}\end{pmatrix}\begin{pmatrix} P_{0k} & Q_{0k}\end{pmatrix}\right]^{-1}\left[\sum_{l\colon k\to l}\begin{pmatrix}\partial_x P_{kl} \\ \partial_x Q_{kl}\end{pmatrix} - \begin{pmatrix}\partial_x p_k \\ \partial_x q_k\end{pmatrix} - \begin{pmatrix} r_{0k} \\ x_{0k}\end{pmatrix}\frac{\ell_{0k}}{v_0}\partial_x v_0\right]$;
```

Remark 5.1. *The calculation of gradients $\partial_x(P, Q, v, p_0, q_0)$ in Algorithm 5 depends on the power* *flow solution* (P, Q, v, p_0, q_0) *at which the gradients are evaluated. Since it is assumed in Assumption 5.1 that the practical* (P, Q, v, p_0, q_0) *is unique given* (p, q)*, the gradients* $\partial_x(P, Q, v, p_0, q_0)$ *is unique.*

Remark 5.2. *The implicit function theorem is used to derive* (5.3) *from* (5.1b)–(5.1e)*. To apply the theorem, we have made the mild assumption that the gradients* $\partial_x(P, Q, v, \ell)$ *exist.*

5.2.1.2 Approximated Derivatives

To avoid the iterative procedure described by Algorithm 5 to improve the computational efficiency, one can estimate the gradients as follows. Note that the current terms in (5.1b)–(5.1d) are much smaller than the other terms in practice, and one can estimate (P, Q, v) by $(\hat{P}, \hat{Q}, \hat{v})$ defined as

$$\begin{aligned} \sum_{i:\, i\to j} \hat{P}_{ij} + p_j &= \sum_{k:\, j\to k} \hat{P}_{jk}, && j \in \mathcal{N}; \\ \sum_{i:\, i\to j} \hat{Q}_{ij} + q_j &= \sum_{k:\, j\to k} \hat{Q}_{jk}, && j \in \mathcal{N}; \\ \hat{v}_i - \hat{v}_j &= 2(r_{ij}\hat{P}_{ij} + x_{ij}\hat{Q}_{ij}), && i \to j. \end{aligned}$$

Note that because the equations are linear, it is straightforward to obtain

$$\begin{aligned} \partial_x \hat{P}_{ij} &= \sum_{k:\, j\to k} \partial_x \hat{P}_{jk} - \partial_x p_j, && i \to j; \\ \partial_x \hat{Q}_{ij} &= \sum_{k:\, j\to k} \partial_x \hat{Q}_{jk} - \partial_x q_j, && i \to j; \\ \partial_x \hat{v}_j &= \partial_x \hat{v}_i - 2r_{ij}\partial_x \hat{P}_{ij} - 2x_{ij}\partial_x \hat{Q}_{ij}, && i \to j \end{aligned}$$

since the network is radial.

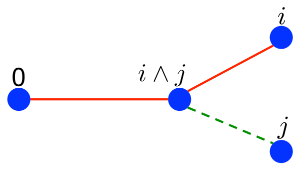


Figure 5.1: The bus $i \wedge j$ denotes the joint of bus i and bus j. The resistance R_i denotes the total resistance of the red solid line segment.

Let $i \wedge j$ denote the joint of bus i and j for $i, j \in \mathcal{N}$, and let R_i denote the total resistance from bus 0 to bus i for $i \in \mathcal{N}$. See Figure 5.1 for an example. Then $\partial_x(\hat{P}, \hat{Q}, \hat{v})$ has the following closed-form expression

$$\partial_{p_k} \hat{P}_{ij} = -\mathbf{1}_{j\in\mathcal{P}_k}, \qquad \partial_{q_k} \hat{P}_{ij} = 0, \qquad k = 1,2,\ldots,n,\ i \to j; \tag{5.4a}$$

$$\partial_{p_k} \hat{Q}_{ij} = 0, \qquad \partial_{q_k} \hat{Q}_{ij} = -\mathbf{1}_{j\in\mathcal{P}_k}, \qquad k = 1,2,\ldots,n,\ i \to j; \tag{5.4b}$$

$$\partial_{p_k} \hat{v}_i = 2R_{i\wedge k}, \qquad \partial_{q_k} \hat{v}_i = 2X_{i\wedge k}, \qquad k = 1,2,\ldots,n,\ i \in \mathcal{N}^+. \tag{5.4c}$$

We remark on several interesting properties of (5.4):

Remark 5.3. *The derivatives $\partial_{p_k} v_i$ and $\partial_{q_k} v_i$ are symmetric in (i,k), i.e., the impact of an injection on voltage is symmetric in their respective locations.*

Remark 5.4. *The impact on voltages of injections close to the substation is smaller; similarly, the voltages close to the substation are less sensitive to injections. This agrees with intuition.*

One can approximate $\partial_x(P,Q,v)$ by $\partial_x(\hat{P},\hat{Q},\hat{v})$, i.e.,

$$\partial_x(P,Q,v) \approx \partial_x(\hat{P},\hat{Q},\hat{v}).$$

Note that $\partial_x(\hat{P},\hat{Q},\hat{v})$ is a constant that does not depend on (P,Q,v), and therefore need to be computed only once ahead of time.

Finally, one can approximate $\partial_x(p_0,q_0)$ as follows. Sum up (5.1b) for $j\in\mathcal{N}$ to obtain

$$\sum_{i=0}^{n} p_i = \sum_{i\to j} r_{ij}\ell_{ij} = \sum_{i\to j} r_{ij}\frac{P_{ij}^2+Q_{ij}^2}{v_i}.$$

Hence,

$$\begin{aligned}
\partial_{p_i} p_0 &= -1 + \sum_{k\to l} r_{kl}\partial_{p_i}\left(\frac{P_{kl}^2+Q_{kl}^2}{v_k}\right)\\
&= -1 + \sum_{k\to l} r_{kl}\left(\frac{2P_{kl}}{v_k}\partial_{p_i}P_{kl} + \frac{2Q_{kl}}{v_k}\partial_{p_i}Q_{kl} - \frac{\ell_{kl}}{v_k}\partial_{p_i}v_k\right)\\
&\approx -1 + \sum_{k\to l} r_{kl}\left(\frac{2P_{kl}}{v_k}\partial_{p_i}\hat{P}_{kl} + \frac{2Q_{kl}}{v_k}\partial_{p_i}\hat{Q}_{kl} - \frac{\ell_{kl}}{v_k}\partial_{p_i}\hat{v}_k\right)\\
\partial_{q_i} p_0 &= \sum_{k\to l} r_{kl}\partial_{q_i}\left(\frac{P_{kl}^2+Q_{kl}^2}{v_k}\right)\\
&= \sum_{k\to l} r_{kl}\left(\frac{2P_{kl}}{v_k}\partial_{q_i}P_{kl} + \frac{2Q_{kl}}{v_k}\partial_{q_i}Q_{kl} - \frac{\ell_{kl}}{v_k}\partial_{q_i}v_k\right)\\
&\approx \sum_{k\to l} r_{kl}\left(\frac{2P_{kl}}{v_k}\partial_{q_i}\hat{P}_{kl} + \frac{2Q_{kl}}{v_k}\partial_{q_i}\hat{Q}_{kl} - \frac{\ell_{kl}}{v_k}\partial_{q_i}\hat{v}_k\right)
\end{aligned}$$

for $i = 1,2,\ldots,n$. The expressions for $\partial_x q_0$ are symmetric with r_{ij} replaced by x_{ij}.

5.2.2 Modified Objective Function

To enable a distributed algorithm, i.e., each bus i updates its own (p_i, q_i) locally, the constraints of OPF have better be decoupled, i.e., constraints of the form (5.2b) are easy to handle while constraints of the form (5.2a) should be avoided.

To avoid coupled constraints (5.2a), one can add a log-barrier function to the objective as

$$L(x;\mu) = a_0 p_0^2(x) + b_0 p_0(x) + \sum_{i=1}^{n}(a_i p_i^2 + b_i p_i) - \underline{\mu}\sum_{i=1}^{n}\ln(v_i(x) - \underline{v}_i) - \overline{\mu}\sum_{i=1}^{n}\ln(\overline{v}_i - v_i(x)),$$

where $\mu = (\underline{\mu}, \overline{\mu}) > 0$ componentwise. Note that $\underline{v}_i \le v_i(x) \le \overline{v}_i$ for $i \in \mathcal{N}^+$ is enforced since

$$\lim_{t \downarrow \underline{v}_i} -\underline{\mu}\ln(t - \underline{v}_i) = \infty, \quad \lim_{t \uparrow \overline{v}_i} -\overline{\mu}\ln(\overline{v}_i - t) = \infty, \qquad i \in \mathcal{N}^+$$

and OPF seeks to minimize L. Besides,

$$\lim_{\mu \downarrow 0} L(x;\mu) = a_0 p_0^2(x) + b_0 p_0(x) + \sum_{i=1}^{n}(a_i p_i^2 + b_i p_i)$$

and therefore solving OPF is similar to minimizing $L(x;\mu)$ with small enough μ.

To summarize, the distributed algorithm proposed in this chapter seeks to solve

$$\begin{aligned} \textbf{OPF}(\mu): \ \ \min \ \ & L(x;\mu) \\ \text{over} \ \ & p_1, \ldots, p_n, q_1, \ldots, q_n; \\ \text{s.t.} \ \ & \underline{p}_i \le p_i \le \overline{p}_i, \ \underline{q}_i \le q_i \le \overline{q}_i, \quad i = 1, 2, \ldots, n \end{aligned}$$

for a decreasing sequence of μ. In the rest of this section, we solve OPF(μ) for a specific μ.

5.2.3 Gradient Projection

There are two key steps in a gradient projection algorithm: 1) compute (or approximate) the gradients; and 2) choose a step size to update the optimization variables.

The gradients can be approximated using (5.4). In particular,

$$\begin{aligned}
\partial_{p_i} L &= (2a_0 p_0 + b_0)\partial_{p_i} p_0 + (2a_i p_i + b_i) - \sum_{k=1}^{n} \left[\frac{\underline{\mu}}{v_k - \underline{v}_k} + \frac{\overline{\mu}}{v_k - \overline{v}_k} \right] \partial_{p_i} v_k \\
&= (2a_0 p_0 + b_0) \left[-1 + \sum_{k \to l} r_{kl} \left(\frac{2P_{kl}}{v_k} \partial_{p_i} P_{kl} + \frac{2Q_{kl}}{v_k} \partial_{p_i} Q_{kl} - \frac{\ell_{kl}}{v_k} \partial_{p_i} v_k \right) \right] \\
&\quad + (2a_i p_i + b_i) - \sum_{k=1}^{n} \left[\frac{\underline{\mu}}{v_k - \underline{v}_k} + \frac{\overline{\mu}}{v_k - \overline{v}_k} \right] \partial_{p_i} v_k \\
&\approx (2a_0 p_0 + b_0) \left[-1 + \sum_{k \to l} r_{kl} \left(\frac{2P_{kl}}{v_k} \partial_{p_i} \hat{P}_{kl} + \frac{2Q_{kl}}{v_k} \partial_{p_i} \hat{Q}_{kl} - \frac{\ell_{kl}}{v_k} \partial_{p_i} \hat{v}_k \right) \right] \\
&\quad + (2a_i p_i + b_i) - \sum_{k=1}^{n} \left[\frac{\underline{\mu}}{v_k - \underline{v}_k} + \frac{\overline{\mu}}{v_k - \overline{v}_k} \right] \partial_{p_i} \hat{v}_k \\
&= -(2a_0 p_0 + b_0) \left[1 + \sum_{k \to l} r_{kl} \left(\frac{2P_{kl}}{v_k} \mathbf{1}_{l \in \mathcal{P}_i} + \frac{\ell_{kl}}{v_k} R_{i \wedge k} \right) \right] \\
&\quad + (2a_i p_i + b_i) - \sum_{k=1}^{n} \left[\frac{\underline{\mu}}{v_k - \underline{v}_k} + \frac{\overline{\mu}}{v_k - \overline{v}_k} \right] 2R_{i \wedge k} \qquad (5.5)
\end{aligned}$$

for $i = 1, 2, \ldots, n$. Similarly,

$$\partial_{q_i} L \approx -(2a_0 p_0 + b_0) \sum_{k \to l} r_{kl} \left(\frac{2Q_{kl}}{v_k} \mathbf{1}_{l \in \mathcal{P}_i} + \frac{\ell_{kl}}{v_k} X_{i \wedge k} \right) - \sum_{k=1}^{n} \left[\frac{\underline{\mu}}{v_k - \underline{v}_k} + \frac{\overline{\mu}}{v_k - \overline{v}_k} \right] 2X_{i \wedge k} \qquad (5.6)$$

for $i = 1, 2, \ldots, n$.

The step size can be determined by doing a line search along the direction of $-\partial_{(p,q)} L$ (in the software implementation, the estimation of $\partial_{(p,q)} L$ in (5.5)–(5.6) is used instead), i.e., back off the step size until the modified objective function can be well-approximated by its linearization around the current solution point. Three parameters α (determine the back off speed, set to 0.5 in the current implementation), β (criteria for the linearization of the objective to be accurate enough, set to 0.5 in the current implementation), and ϵ (criteria for the progress to be too slow, set to 1e-4 in the current implementation) are needed in the line search step. In particular, the line search is described in Algorithm 6. To state the algorithm, let

$$\prod_{C} x := \text{argmin}_{y \in C} \|y - x\|_2$$

denote the (unique) projection of a point x onto a non-empty compact convex set C.

Theorem 5.2 (well defined)**.** *Given specified input, Algorithm 6 always produces* $(p', q', \text{stopFlag})$.

Theorem 5.2 implies that Algorithm 6 is well defined.

Proof. Assume that Algorithm 6 fails to produce $(p', q', \text{stopFlag})$ for some instance. Consider this

Algorithm 6 Line search

```
Input: back-off parameter $\alpha \in (0,1)$, linearization parameter $\beta \in (0,1)$, progress parameter $\epsilon \ll 1$,
    current solution $(p,q)$, bounds $\left(\underline{p}, \overline{p}, \underline{q}, \overline{q}, \underline{v}, \overline{v}\right)$, gradient $\partial_{(p,q)} L$.
Output: update value $(p', q')$, stopping flag stopFlag.
1: $\eta = 1, \text{stopFlag} = 0$;
2: $(p', q') \leftarrow (p,q) - \eta \partial_{(p,q)} L$;
3: $p' \leftarrow \prod_{[\underline{p},\overline{p}]} p'$, $q' \leftarrow \prod_{[\underline{q},\overline{q}]} q'$;
4: run the backward-forward sweep algorithm to obtain the power flow solution $(v', p_0', q_0')$ with
   respect to $(p, q, v_0)$;
5: if $v' \notin [\underline{v}, \overline{v}]$
       $\eta \leftarrow \alpha\eta$, go to 2;
   end
6: $\Delta p \leftarrow p' - p$, $\Delta q \leftarrow q' - q$;
7: if $\|\Delta p\| + \|\Delta q\| \le \epsilon$
       stopFlag = 1;
   else if $L(p', q') > L(p,q) + \beta\left(\partial_p L \cdot \Delta p + \partial_q L \cdot \Delta q\right)$
       $\eta \leftarrow \alpha\eta$, go to 2;
   end
8: if $L(p', q') > L(p,q)$
       $p' \leftarrow p$, $q' \leftarrow q$;
   end
```

instance and derive a contradiction as follows. Let the superscript (k) denote the round of iteration for $k = 0, 1, 2, \ldots$ where iteration 0 refers to the initial value, e.g., $\Delta p^{(k)} = p^{(k)} - p$ for $k \ge 1$.

Let $m > 0$ denote the minimum positive number among $\{|\partial_{p_i} L|, |\partial_{q_i} L| \ : \ i = 1, 2, \ldots, n\}$. Note that $\partial_{p_i} L \cdot \Delta p_i^{(k)} \le 0$ and $\partial_{q_i} L \cdot \Delta q_i^{(k)} \le 0$ for $k \ge 1$ and $i \in \mathcal{N}^+$. Furthermore, $\partial_{p_i} L = 0$ implies $\Delta p_i^{(k)} = 0$ for $k \ge 1$ and $i \in \mathcal{N}^+$. Hence,

$$\partial_{p_i} L \cdot \Delta p_i^{(k)} \le -m |\Delta p_i^{(k)}|, \quad k \ge 1, \ i \in \mathcal{N}^+.$$

It follows that

$$\partial_p L \cdot \Delta p^{(k)} \ = \ \sum_{i \in \mathcal{N}^+} \partial_{p_i} L \cdot \Delta p_i^{(k)} \ \le \ \sum_{i \in \mathcal{N}^+} -m |\Delta p_i^{(k)}| \ = \ -m \left\| \Delta p^{(k)} \right\|_1$$

for $k \ge 1$, where $\| \cdot \|_1$ denotes the ℓ_1 norm of a vector, i.e., $\|x\|_1 = \sum_{i=1}^n |x_i|$ for $x \in \mathbb{R}^n$. Similarly,

$\partial_q L \cdot \Delta q^{(k)} \leq -m\|\Delta q^{(k)}\|_1$ for $k \geq 1$. It follows that

$$\begin{aligned} L(p^{(k)}, q^{(k)}) &= L(p + \Delta p^{(k)}, q + \Delta q^{(k)}) \\ &= L(p,q) + \partial_p L \cdot \Delta p^{(k)} + \partial_q L \cdot \Delta q^{(k)} + o(\Delta p^{(k)}, \Delta q^{(k)}) \\ &= L(p,q) + \beta \left(\partial_p L \cdot \Delta p^{(k)} + \partial_q L \cdot \Delta q^{(k)}\right) \\ &\quad + (1-\beta)\left(\partial_p L \cdot \Delta p^{(k)} + \partial_q L \cdot \Delta q^{(k)}\right) + o(\Delta p^{(k)}, \Delta q^{(k)}) \\ &\leq L(p,q) + \beta \left(\partial_p L \cdot \Delta p^{(k)} + \partial_q L \cdot \Delta q^{(k)}\right) \\ &\quad - m(1-\beta)\left(\left\|\Delta p^{(k)}\right\|_1 + \left\|\Delta q^{(k)}\right\|_1\right) + o(\Delta p^{(k)}, \Delta q^{(k)}) \end{aligned}$$

for $k \geq 1$. When k is sufficiently big, $\|\Delta p^{(k)}\|_1 + \|\Delta q^{(k)}\|_1$ is sufficiently small such that

$$o(\Delta p^{(k)}, \Delta q^{(k)}) \leq m(1-\beta)\left(\left\|\Delta p^{(k)}\right\|_1 + \left\|\Delta q^{(k)}\right\|_1\right).$$

Hence, eventually

$$L(p^{(k)}, q^{(k)}) \leq L(p,q) + \beta\left(\partial_p L \cdot \Delta p^{(k)} + \partial_q L \cdot \Delta q^{(k)}\right).$$

Then, the loop specified by Step 7) is exited and $(p', q', \text{stopFlag})$ is produced. This contradicts the assumption that Algorithm 6 fails to produce a $(p', q', \text{stopFlag})$ and completes the proof of Theorem 5.2. $\square$

Remark 5.5. *The introduction of ϵ in the "if" branch in Step 7) is to stop the iterations when progress gets too slow, i.e., $\|\Delta p\| + \|\Delta q\| \leq \epsilon$. When this happens,* stopFlag *is set to 1 and iterations are stopped. Otherwise, a large number of iterations will run without updating (p,q) significantly.*

With the "if" branch in Step 7), it is possible that $L(p', q') > L(p,q)$. In this case, (p', q') is set to (p,q) to ensure that new point (p', q') does not have a larger objective value than (p,q).

Definition 5.1 (local optimum)**.** *A point (p,q) is a local optimum for minimizing L if*

$$\begin{aligned} \langle \partial_{p_i} L, \tilde{p}_i - p_i \rangle \geq 0, \quad \forall \tilde{p}_i \in (\underline{p}_i, \overline{p}_i),\ \forall i \in \mathcal{N}^+; \\ \langle \partial_{q_i} L, \tilde{q}_i - q_i \rangle \geq 0, \quad \forall \tilde{q}_i \in (\underline{q}_i, \overline{q}_i),\ \forall i \in \mathcal{N}^+. \end{aligned}$$

Theorem 5.3 (stationary points)**.** *In Algorithm 6, if $\epsilon = 0$, then*

$$(p,q) = (p', q') \quad \Longleftrightarrow \quad (p,q) \text{ is a local optimum.}$$

Theorem 5.3 implies that a point (p,q) is stationary for Algorithm 6, i.e., the output $(p', q') = (p,q)$, if and only if (p,q) is a local optimum of minimizing L.

Proof. Either $\|\Delta p\| + \|\Delta q\| \le \epsilon = 0$ or $L(p', q') \le L(p, q) + \beta(\partial_p L \cdot \Delta p + \partial_q L \cdot \Delta q) \le L(p, q)$ when entering Step 8). In either case, $L(p', q') \le L(p, q)$ and therefore

$$p' = \prod_{[\underline{p}, \overline{p}]} (p - \eta \partial_p L), \quad q' = \prod_{[\underline{q}, \overline{q}]} (q - \eta \partial_q L)$$

after exiting Algorithm 6.

"$\Leftarrow$": Let (p, q) be a local optimum. Want to prove that $(p, q) = (p', q')$. For brevity, we present the proof of $p_i = p_i'$ for an arbitrarily chosen $i \in \mathcal{N}^+$ below. The proof of $q_i = q_i'$ for an arbitrarily chosen $i \in \mathcal{N}^+$ is similar.

Note that $p_i' = \prod_{[\underline{p}_i, \overline{p}_i]} (p_i - \eta \partial_{p_i} L)$. If $\partial_{p_i} L = 0$, then $p_i' = p_i$. If $\partial_{p_i} L > 0$, since

$$\langle \partial_{p_i} L, \tilde{p}_i - p_i \rangle \ge 0, \quad \underline{p}_i \le \tilde{p}_i \le \overline{p}_i,$$

let $\tilde{p}_i = \underline{p}_i$ to obtain $p_i = \underline{p}_i$; consequently $p_i' = p_i$. If $\partial_{p_i} L < 0$, let $\tilde{p}_i = \overline{p}_i$ to obtain $p_i = \overline{p}_i$; consequently $p_i' = p_i$. This completes the proof of "$\Leftarrow$".

"$\Rightarrow$": Assume $(p, q) = (p', q')$. Want to prove that (p, q) is a local optimum. For brevity, we present the proof of

$$\langle \partial_{p_i} L, \tilde{p}_i - p_i \rangle \ge 0, \quad \underline{p}_i \le \tilde{p}_i \le \overline{p}_i \tag{5.7}$$

for an arbitrary $i \in \mathcal{N}^+$. The proof of

$$\langle \partial_{q_i} L, \tilde{q}_i - q_i \rangle \ge 0, \quad \underline{q}_i \le \tilde{q}_i \le \overline{q}_i$$

for an arbitrary $i \in \mathcal{N}^+$ is similar.

Note that $p_i = p_i' = \prod_{[\underline{p}_i, \overline{p}_i]} (p_i - \eta \partial_{p_i} L)$. If $\partial_{p_i} L = 0$, then (5.7) holds. If $\partial_{p_i} L > 0$, then $p_i = \underline{p}_i$ and therefore (5.7) holds. If $\partial_{p_i} L < 0$, then $p_i = \overline{p}_i$ and therefore (5.7) holds. This completes the proof of "$\Rightarrow$".

Combining "$\Leftarrow$" and "$\Rightarrow$" completes the proof of Theorem 5.3. □

Theorem 5.4 (progress). *If $\epsilon = 0$, then the input (p, q) and output (p', q') of Algorithm 6 satisfy*

$$L(p', q') \le L(p, q).$$

The equality is attained if and only if $(p', q') = (p, q)$.

Theorems 5.3 and 5.4 imply that $L(p', q') < L(p, q)$ unless (p, q) is a local optimum, in which case $(p', q') = (p, q)$.

Proof. Either $\|\Delta p\| + \|\Delta q\| \le \epsilon = 0$ or $L(p', q') \le L(p, q) + \beta(\partial_p L \cdot \Delta p + \partial_q L \cdot \Delta q) \le L(p, q)$

when entering Step 8). In either case, $L(p',q') \leq L(p,q)$ and therefore $p' = \prod_{[\underline{p},\overline{p}]}(p-\eta\partial_p L)$, $q' = \prod_{[\underline{q},\overline{q}]}(q-\eta\partial_q L)$ after exiting Algorithm 6.

Let $m > 0$ denote the minimum positive number among $\{|\partial_{p_i}L|, |\partial_{q_i}L| : i \in \mathcal{N}^+\}$. Fix an arbitrary $i \in \mathcal{N}^+$. Note that $\partial_{p_i}L \cdot \Delta p_i \leq 0$ and that $\partial_{p_i}L = 0$ implies $\Delta p_i = 0$. Hence,

$$\partial_{p_i}L \cdot \Delta p_i \leq -m|\Delta p_i|.$$

Similarly, $\partial_{q_i}L \cdot \Delta q_i \leq -m|\Delta q_i|$ for $i \in \mathcal{N}^+$. It follows that either $\|\Delta p\| + \|\Delta q\| = 0$, or

$$\begin{aligned} L(p',q') &\leq L(p,q) + \beta\left(\partial_p L \cdot \Delta p + \partial_q L \cdot \Delta q\right) \\ &= L(p,q) + \beta\sum_{i=1}^{n}\left(\partial_{p_i}L \cdot \Delta p_i + \partial_{q_i}L \cdot \Delta q_i\right) \\ &\leq L(p,q) - m\beta\sum_{i=1}^{n}\left(|\Delta p_i| + |\Delta q_i|\right) \\ &= L(p,q) - m\beta\left(\|\Delta p\|_1 + \|\Delta q\|_1\right). \end{aligned}$$

Hence, $L(p',q') \leq L(p,q)$ and the equality is attained if and only if $(p,q) = (p',q')$. □

Remark 5.6. *The gradient $\partial_{(p,q)}L$ is only approximated by* (5.5)—(5.6) *in the software implementation. Hence, Theorems 5.2–5.4 may not hold. However, the following conclusions can be made:*

- *Theorem 5.2 holds if $\epsilon > 0$. This is because $\|\Delta p^{(k)}\| + \|\Delta q^{(k)}\|$ tends to 0 as k tends to infinity and therefore the "if" branch in Step 7) will be chosen for sufficiently big k. Hence, with gradient $\partial_{(p,q)}L$ approximated, Algorithm 6 is still well-defined as long as $\epsilon > 0$.*

- *Theorems 5.3 and 5.4 hold if $\partial_{(p,q)}L$ is replaced by its approximate in Definition 5.1. This is because the proofs of Theorems 5.3 and 5.4 do not rely on the fact that $\partial_{(p,q)}L$ is a gradient. However, since one has to set $\epsilon > 0$ to ensure that Algorithm 6 always produces a solution, the $\epsilon = 0$ condition in Theorems 5.3 and 5.4 does not hold and the conclusions do not apply.*

5.2.4 Distributed Gradient Projection Algorithm

The distributed gradient projection algorithm is obtained by putting all the pieces in Section 5.2.1–5.2.3 together. It is summarized in Algorithm 7, which solves OPF(μ) with different values of μ. In particular, let $\mu_1, \mu_2, \ldots, \mu_K$ denote a sequence of μ that approaches 0 and $x = (p,q)$. Given a feasible initial point $x^{(0)}$, Algorithm 7 solves OPF(μ_1) with initial point $x^{(0)}$ to obtain $x^{(1)}$, then solves OPF(μ_2) with initial point $x^{(1)}$ to obtain $x^{(2)}$, ..., and finally solves OPF(μ_K) with initial point $x^{(K-1)}$ to obtain the final output solution $x^{(K)}$.

5.2.4.1 Algorithm Statement

To solve each OPF(μ), Algorithm 7 repeatedly calculates the gradient $\partial_x L(x;\mu)$ and does a line search along the direction of $-\partial_x L$ to update x, until stopFlag $= 1$, which indicates that numerically no further improvements can be made. Algorithm 7 is stated below.

Algorithm 7 A gradient decent algorithm

Input: A feasible point (p, q).
Output: A numerical solution (p^*, q^*) of (5.2).

```
1: $p^* = p$, $q^* = q$;
2: for $\mu = \mu_1, \mu_2, \ldots, \mu_K$
3:     do
4:        run backward-forward sweep to obtain the power flow solution $(v^*, p_0^*, q_0^*)$ at $(p^*, q^*, v_0)$;
5:        compute gradient $\partial_{(p,q)} L(p^*, q^*; \mu)$;
6:        run Algorithm 6 to get an updated $(p^*, q^*, \text{stopFlag})$;
7:     while stopFlag != 1
8: end
```

Theorem 5.5 (inner convergence)**.** *Consider an inner loop of Algorithm 7 where μ is a fixed constant. Let $\mathbb{O}^*$ denote the set of local minima of $L(x;\mu)$. If $\epsilon = 0$, then the sequence $x^{(1)}, x^{(2)}, \ldots, x^{(k)}, \ldots$ of intermediate power injections computed in Algorithm 7 converges to $\mathbb{O}^*$, i.e.,*

$$x^{(k)} \to \mathbb{O}^*, \quad k \to \infty.$$

Furthermore, when the set $\mathbb{O}^*$ has finitely many elements, then

$$x^{(k)} \to x^*, \quad k \to \infty$$

for some $x^* \in \mathbb{O}^*$.

Theorem 5.5 implies a local minimum of $L(x;\mu)$ is obtained in each inner loop of Algorithm 7.

Proof. If $x^{(k)} \nrightarrow \mathbb{O}^*$, then there exists $\delta > 0$ such that $\text{dist}(x^{(k)}, \mathbb{O}^*) > \delta$ infinitely often. Let $\mathbb{X} := [\underline{p}, \overline{p}] \times [\underline{q}, \overline{q}]$ denote the compact feasible set of x, and define

$$A := \{x \in \mathbb{X} \mid \text{dist}(x, \mathbb{O}^*) \geq \delta\}.$$

Then the set A is closed and therefore compact. Also define

$$B := \{x \in \mathbb{X} \mid L(x;\mu) \leq L(x^{(0)};\mu)\}.$$

Then the set B is also closed and compact. The sequence $x^{(0)}, x^{(1)}, \ldots, x^{(k)}, \ldots$ lies in the set $C := A \cap B$ infinitely often. Note that C is compact and that $L(x;\mu)$ has uniformly bounded first and second derivatives on C.

If $x^{(k)} \in C$, then $x^{(k)} \notin \mathbb{O}^*$ and therefore $x^{(k+1)} \neq x^{(k)}$ by Theorem 5.3. Besides,

$$x^{(k+1)} = \prod_{\mathbb{X}} \left(x^{(k)} - \eta^{(k)} \partial_x L(x^{(k)}; \mu) \right)$$

where $\eta^{(k)}$ is the first η in the exponential backoff stage that satisfies

$$L(x^{(k+1)}; \mu) \leq L(x^{(k)}; \mu) + \beta \partial_x L(x^{(k)}; \mu) \cdot \left(x^{(k+1)} - x^{(k)} \right).$$

There exists an $\eta_1 > 0$ such that for any $x^{(k)} \in C$ and any $\eta \in (0, \eta_1]$, the value x' defined as

$$x' = \prod_{\mathbb{X}} \left(x^{(k)} - \eta \partial_x L(x^{(k)}; \mu) \right)$$

satisfies

$$L(x'; \mu) \leq L(x^{(k)}; \mu) + \beta \partial_x L(x^{(k)}; \mu) \cdot \left(x' - x^{(k)} \right)$$

since $\partial_{xx} L_\mu$ is uniformly upper bounded on C. Hence, if $x^{(k)} \in C$, then $\eta^{(k)}$ is lower bounded by $\eta_0 = \alpha \eta_1 > 0$. It follows that if $x^{(k)} \in C$, then

$$\begin{aligned} L(x^{(k+1)}; \mu) - L(x^{(k)}; \mu) &\leq \beta \partial_x L(x^{(k)}; \mu) \cdot \left(x^{(k+1)} - x^{(k)} \right) \\ &= \beta \sum_{i=1}^{2n} \partial_i L \cdot \left(\prod_{[\underline{x}_i, \overline{x}_i]} \left(x_i^{(k)} - \eta^{(k)} \partial_i L \right) - x_i^{(k)} \right) \\ &\leq \beta \sum_{i=1}^{2n} \partial_i L \cdot \left(\prod_{[\underline{x}_i, \overline{x}_i]} \left(x_i^{(k)} - \eta_0 \partial_i L \right) - x_i^{(k)} \right). \end{aligned} \tag{5.8}$$

The expression in (5.8) is continuous in $x^{(k)}$ and nonpositive. Furthermore, it is strictly smaller than 0 since otherwise $x^{(k)} \in \mathbb{O}^*$, which contradicts with $\operatorname{dist}(x^{(k)}, \mathbb{O}^*) \geq \delta > 0$. Since C is compact,

$$L(x^{(k+1)}; \mu) - L(x^{(k)}; \mu) \leq -\theta$$

for some $\theta > 0$.

Since the sequence $\{x^{(k)} : k \geq 1\}$ visits C infinitely often, the limit

$$\lim_{k \to \infty} L(x^{(k)}; \mu) = -\infty.$$

This contradicts with the fact that $L(x; \mu)$ is lower bounded on $x \in \mathbb{X}$. Hence, one must have

$$x^{(k)} \to \mathbb{O}^*, \quad k \to \infty.$$

When the set $\mathbb{O}^*$ is finite, let $x^{*1}, x^{*2}, \ldots, x^{*m}$ denote all its elements and d denote the minimum distance among them. There exists r such that whenever $\text{dist}(x^{(k)}, \mathbb{O}^*) < r$, $\|x^{(k+1)} - x^{(k)}\| < d/3$. There exists $k_1 \in \mathbb{N}$ such that whenever $k \geq k_1$, $\text{dist}(x^{(k)}, \mathbb{O}^*) < \min\{r, d/3\}$. It follows that $\|x^{(k_1)} - x^*\| < d/3$ for an $x^* \in \mathbb{O}^*$. Then, it is not difficult to verify that $\|x^{(k_1+1)} - x^{**}\| > d/3$ for any $x^{**} \in \mathbb{O}^*$ but $x^{**} \neq x^*$. Hence, $\|x^{(k_1+1)} - x^*\| < d/3$. By mathematical induction, one can show that

$$\|x^{(k)} - x^*\| < d/3, \ \text{dist}\left(x^{(k)}, \mathbb{O}^* \backslash \{x^*\}\right) > 2d/3, \quad k \geq k_1.$$

Then, since $\text{dist}\left(x^{(k)}, \mathbb{O}^*\right) \to 0$, one must have

$$x^{(k)} \to x^*.$$

This completes the proof of Theorem 5.5. □

5.2.4.2 Distributed Implementation

An important advantage of Algorithm 7 is that it can be implemented in a distributed way. The infrastructure required to implement Algorithm 7 in a distributed way is described as follows.

- There is an agent at each branch bus $i \in \mathcal{N}^+$, and a coordinator at the substation bus 0. Call the agent at bus i agent i for brevity.
- Agent i knows the impedance (r_{ij}, x_{ij}) for all j such that $i \sim j$; it can measure the voltage v_i, the power flow (P_{ij}, Q_{ij}), and the current ℓ_{ij} for all j such that $i \to j$; it can control the power consumption (p_i, q_i); and it can communicate with the coordinator and agent j if $i \sim j$.
- The coordinator knows the impedance (r_{0j}, x_{0j}) for all j such that $0 \to j$; it can measure the voltage v_0, the substation power injection (p_0, q_0), the power flow (P_{0j}, Q_{0j}), and the current ℓ_{0j} for all j such that $0 \to j$; and it can communicate with all agents in the network.

Remark 5.7. *To implement Algorithm 7 in a distributed way, only buses i whose power injections (p_i, q_i) can be controlled need to have agents. For example, if $\underline{p}_i = \overline{p}_i$ and $\underline{q}_i = \overline{q}_i$, then bus i does not need to have an agent.*

There are two key components in the distributed implementation of Algorithm 7: 1) compute gradient $\partial_{(p,q)} L(p^*, q^*; \mu)$ in a distributed way; and 2) run line search (Algorithm 6) in a distributed way. For clarity, we first present these two key components before describing the distributed implementation of Algorithm 7.

Distributed gradient computation. The approximate gradients (5.5)–(5.6) can be computed in a backward forward sweep as described below.

Let $\text{down}(i) := \{j \in \mathcal{N} \mid i \in \mathcal{P}_j\}$ denote the buses downstream of bus $i \in \mathcal{N}$ as illustrated in Figure 5.2. Define

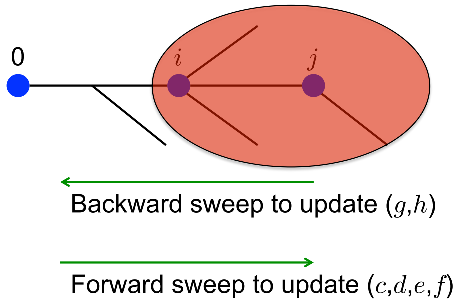

0
i
j
Backward sweep to update (g,h)
Forward sweep to update (c,d,e,f)

Figure 5.2: Buses down(i) downstream of bus i lies in the shaded region.

$$c_i = \sum_{k=1}^{n} 2R_{i\wedge k}\left(\frac{\underline{\mu}}{v_k - \underline{v}_k} + \frac{\overline{\mu}}{v_k - \overline{v}_k}\right), \qquad i \in \mathcal{N};$$
$$d_i = \sum_{k=1}^{n} 2X_{i\wedge k}\left(\frac{\underline{\mu}}{v_k - \underline{v}_k} + \frac{\overline{\mu}}{v_k - \overline{v}_k}\right), \qquad i \in \mathcal{N};$$
$$e_i = \sum_{k\to l} r_{kl}\left(\frac{2P_{kl}}{v_k}\mathbf{1}_{l\in\mathcal{P}_i} + \frac{\ell_{kl}}{v_k}R_{i\wedge k}\right), \qquad i \in \mathcal{N};$$
$$f_i = \sum_{k\to l} r_{kl}\left(\frac{2Q_{kl}}{v_k}\mathbf{1}_{l\in\mathcal{P}_i} + \frac{\ell_{kl}}{v_k}X_{i\wedge k}\right), \qquad i \in \mathcal{N}.$$

Then the approximate gradients (5.5)–(5.6) can be simplified as

$$\partial_{p_i} L = -(2a_0p_0 + b_0)(1 + e_i) + (2a_ip_i + b_i) - c_i, \qquad i \in \mathcal{N}^+; \tag{5.9a}$$
$$\partial_{q_i} L = -(2a_0p_0 + b_0)f_i - d_i, \qquad i \in \mathcal{N}^+. \tag{5.9b}$$

The centralized coordinator has access to p_0 and can broadcast $2a_0p_0 + b_0$ to all branch buses. Each agent i knows p_i and can easily compute $2a_ip_i + b_i$. Hence, the main challenge is to compute c_i, d_i, e_i, f_i in a distributed manner.

The quantities c_i, d_i, e_i, f_i can be computed recursively. To derive the recursive equations, note that for each $i \to j$, one has

$$R_{i\wedge k} - R_{j\wedge k} = -r_{ij}\mathbf{1}_{k\in\text{down}(j)}$$

and therefore

$$\begin{aligned} c_i - c_j &= \sum_{k=1}^{n} 2\left(R_{i\wedge k} - R_{j\wedge k}\right)\left(\frac{\underline{\mu}}{v_k - \underline{v}_k} + \frac{\overline{\mu}}{v_k - \overline{v}_k}\right) \\ &= -\sum_{k=1}^{n} 2r_{ij}\mathbf{1}_{k\in\mathrm{down}(j)}\left(\frac{\underline{\mu}}{v_k - \underline{v}_k} + \frac{\overline{\mu}}{v_k - \overline{v}_k}\right) \\ &= -2r_{ij}\sum_{k\in\mathrm{down}(j)}\left(\frac{\underline{\mu}}{v_k - \underline{v}_k} + \frac{\overline{\mu}}{v_k - \overline{v}_k}\right). \end{aligned}$$

Similarly,

$$d_i - d_j = -2x_{ij}\sum_{k\in\mathrm{down}(j)}\left(\frac{\underline{\mu}}{v_k - \underline{v}_k} + \frac{\overline{\mu}}{v_k - \overline{v}_k}\right).$$

Besides,

$$\mathbf{1}_{l\in\mathcal{P}_j} - \mathbf{1}_{l\in\mathcal{P}_i} = \mathbf{1}_{l=j}$$

and therefore

$$\begin{aligned} e_i - e_j &= \sum_{k\to l} r_{kl}\left[\frac{2P_{kl}}{v_k}\left(\mathbf{1}_{l\in\mathcal{P}_i} - \mathbf{1}_{l\in\mathcal{P}_j}\right) + \frac{\ell_{kl}}{v_k}\left(R_{i\wedge k} - R_{j\wedge k}\right)\right] \\ &= -\sum_{k\to l} r_{kl}\left[\frac{2P_{kl}}{v_k}\mathbf{1}_{l=j} + \frac{\ell_{kl}}{v_k} r_{ij}\mathbf{1}_{k\in\mathrm{down}(j)}\right] \\ &= -\frac{2r_{ij}P_{ij}}{v_i} - r_{ij}\sum_{k\in\mathrm{down}(j)}\frac{r_{kl}\ell_{kl}}{v_k}. \end{aligned}$$

Similarly,

$$f_i - f_j = -\frac{2r_{ij}Q_{ij}}{v_i} - x_{ij}\sum_{k\in\mathrm{down}(j)}\frac{r_{kl}\ell_{kl}}{v_k}.$$

Hence, if we define

$$\begin{aligned} g_i &= \sum_{k\in\mathrm{down}(i)}\left(\frac{\underline{\mu}}{v_k - \underline{v}_k} + \frac{\overline{\mu}}{v_k - \overline{v}_k}\right), & i \in \mathcal{N}^+; \\ h_i &= \sum_{k\in\mathrm{down}(i)}\frac{r_{kl}\ell_{kl}}{v_k}, & i \in \mathcal{N}^+, \end{aligned}$$

then c_i, d_i, e_i, f_i can be computed recursively as

$$c_j = c_i + 2r_{ij}g_j, \qquad i \to j; \tag{5.10a}$$

$$d_j = d_i + 2x_{ij}g_j, \qquad i \to j; \tag{5.10b}$$

$$e_j = e_i + \frac{2r_{ij}P_{ij}}{v_i} + r_{ij}h_j, \qquad i \to j; \tag{5.10c}$$

$$f_j = f_i + \frac{2r_{ij}Q_{ij}}{v_i} + x_{ij}h_j, \qquad i \to j. \tag{5.10d}$$

```
Algorithm 8 Distributed gradient computation
Input: the agent/coordinator at bus $i \in \mathcal{N}$ knows the parameter $(\underline{\mu}, \overline{\mu})$ and impedance $(r_{ij}, x_{ij})$ for
    all $j$ such that $i \sim j$; it can measure the voltage $v_i$, power flow $(P_{ij}, Q_{ij})$, and current $\ell_{ij}$ for all
    $j$ such that $i \to j$.
Output: each agent $i \in \mathcal{N}^+$ computes $(\partial_{p_i} L, \partial_{q_i} L)$ for $i \in \mathcal{N}^+$.
    Backward sweep to compute $(g, h)$.
 1: each agent $i \in \mathcal{N}^+$ measures $v_i$ and $\ell_{ij}$ for all $j$ such that $i \to j$. On receiving $(g_j, h_j)$ from all
    its downstream neighbors $j$, agent $i$ computes $(g_i, h_i)$ according to
        $$g_i = \frac{\underline{\mu}}{v_i - \underline{v}_i} + \frac{\overline{\mu}}{v_i - \overline{v}_i} + \sum_{j:\, i \to j} g_j; \qquad h_i = \sum_{j:\, i \to j} \left( \frac{r_{ij}\ell_{ij}}{v_i} + h_j \right), \tag{5.11}$$
    and sends $(g_i, h_i)$ to its (unique) upstream neighbor;
 2: backward sweep terminates when the coordinator receives $(g_j, h_j)$ from all its downstream neigh-
    bors $j$.
    Forward sweep to compute $(c, d, e, f)$.
 3: the coordinator measures $v_0$ and $(P_{0j}, Q_{0j})$ for all $j$ such that $0 \to j$, sets
        $$c_0 \leftarrow 0, \quad d_0 \leftarrow 0, \quad e_0 \leftarrow 0, \quad f_0 \leftarrow 0,$$
    and sends $(c_0, d_0, e_0 + 2r_{0j}P_{0j}/v_0, f_0 + 2r_{0j}Q_{0j}/v_0)$ to each downstream neighbor $j$;
 4: each agent $j \in \mathcal{N}^+$ measures $v_j$ and $(P_{jk}, Q_{jk})$ for all $k$ such that $j \to k$. On receiving $(c_i, d_i, e_i +
    2r_{ij}P_{ij}/v_i, f_i + 2r_{ij}Q_{ij}/v_i)$ from its (unique) upstream neighbor $i$, agent $j$ computes $c_j, d_j, e_j, f_j$
    according to (5.10), and sends $(c_j, d_j, e_j + 2r_{jk}P_{jk}/v_j, f_j + 2r_{jk}Q_{jk}/v_j)$ to each downstream
    neighbor $k$;
    Compute gradients.
 5: the coordinator broadcasts $2a_0p_0 + b_0$ to all agents $i \in \mathcal{N}^+$;
 6: once having received $2a_0p_0 + b_0$ from the substation and computed $(c_i, d_i, e_i, f_i)$, agent $i \in \mathcal{N}^+$
    computes $\partial_{p_i} L$ and $\partial_{q_i} L$ according to (5.9);
```

To summarize, the distributed gradient projection algorithm is given in Algorithm 8, where

$$\mathcal{L} := \{i \in \mathcal{N}^+ \mid \nexists j \text{ such that } i \to j\}$$

denotes the set of leaf buses. It consists of three main steps:

S1 Backward sweep to compute (g, h): for each agent $i \in \mathcal{N}^+$, when all its downstream neighbors j have computed (g_j, h_j), computes (g_i, h_i) according to (5.11).

S2 Forward sweep to compute (c, d, e, f): for each agent $j \in \mathcal{N}^+$, when its (unique) upstream neighbor i has computed $(c_i, d_i, e_i + 2r_{ij}P_{ij}/v_i, f_i + 2r_{ij}Q_{ij}/v_i)$, computes (c_j, d_j, e_j, f_j) as in (5.10).

S3 Compute $\partial_{(p,q)} L$: each agent $i \in \mathcal{N}^+$ computes $\partial_{p_i} L$ and $\partial_{q_i} L$ according to (5.9).

Distributed line search. Line search given in Algorithm 6 can also be implemented in a distributed manner, which gives rise to a distributed implementation of the inner loop of Algorithm 7. The distributed implementation is presented in Algorithm 9.

In Algorithm 9, each agent $i \in \mathcal{N}^+$ keeps track of its last approved power injection $(p_i^{\text{old}}, q_i^{\text{old}})$ and proposes tentative power injections $(p_i^{\text{new}}, q_i^{\text{new}})$ while computing some other quantities $(\Delta s_i, \Delta L_i, \text{value}_i^{\text{new}})$, with which the coordinator decides whether to approve the tentative power injections.

Tentative power injection $(p^{\text{new}}, q^{\text{new}})$ is computed by gradient projection (5.13a)–(5.13b), where the gradient is computed as in Algorithm 8 and the step size η is controlled by the coordinator. The coordinator initializes $\eta = 1$ and reduces η by a fraction of $1-\alpha$ until voltage constraints are satisfied, i.e., $\underline{v}_i \le v_i \le \overline{v}_i$ for $i \in \mathcal{N}^+$, and the modified objective function is well-approximated by its linearization, i.e., $L^{\text{new}} < L^{\text{thres}}$ (see step 11 in Algorithm 9).

The coordinator decides whether to approve the tentative power injection, i.e., set $(p^{\text{old}}, q^{\text{old}}) \leftarrow (p^{\text{new}}, q^{\text{new}})$, and when to terminate Algorithm 9, i.e., set (p', q'). In other cases, the coordinator reduces the step size η to $\alpha\eta$ to ask for the submission of new tentative power injection $(p^{\text{new}}, q^{\text{new}})$.

The coordinator makes these decisions based on $(\Delta s_i, \Delta L_i)$ computed by the agents $i \in \mathcal{N}^+$. The quantity Δs_i captures the update size of (p_i, q_i), and ΔL_i is the product of gradient $\partial_{(p_i,q_i)} L$ and power injection update. In particular, if $\sum_{i\in\mathcal{N}^+} \Delta s_i \le \epsilon$, the coordinator decides to stop the inner loop; or else if $L^{\text{new}} \le L^{\text{old}} + \beta \sum_{i\in\mathcal{N}^+} \Delta L_i$, the coordinator decides to approve the tentative power injection, i.e., $(p^{\text{old}}, q^{\text{old}}) \leftarrow (p^{\text{new}}, q^{\text{new}})$; or else the coordinator reduces the step size η by a fraction of $1-\alpha$, i.e., $\eta \leftarrow \alpha\eta$.

Distributed gradient decent. Algorithm 7 can be implemented in a distributed manner by calling Algorithm 9 for a sequence $(\mu_1, \mu_2, \ldots, \mu_K)$ of decreasing μ as illustrated in Figure 5.3.

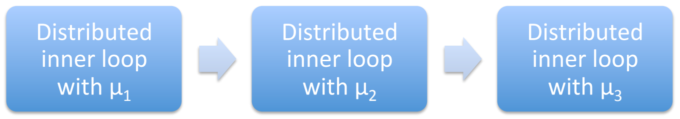

Distributed inner loop with μ1
Distributed inner loop with μ2
Distributed inner loop with μ3

Figure 5.3: Flow chart illustrating the distributed implementation of Algorithm 7.

Algorithm 9 Distributed inner loop

```
Input: each agent $i \in \mathcal{N}^+$ knows its original power injection $(p_i, q_i)$, voltage $v_i$, impedance $(r_{ij}, x_{ij})$
    for neighboring lines $i \sim j$, and power flow $(P_{ij}, Q_{ij})$ for downstream lines $i \to j$; the centralized
    coordinator at substation bus 0 knows algorithm parameters $\alpha$, $\beta$, $\mu$, and $\epsilon$.
Output: each agent $i \in \mathcal{N}^+$ computes a new $(p_i', q_i')$.
 1: the coordinator broadcasts $\mu$ to all agents $i \in \mathcal{N}^+$;
 2: each agent $i \in \mathcal{N}^+$ sets $(p_i^{\text{old}}, q_i^{\text{old}}) \leftarrow (p_i, q_i)$;
 3: each agent $i \in \mathcal{N}^+$ computes
        $$\text{value}_i^{\text{old}} = a_i \left(p_i^{\text{old}}\right)^2 + b_i p_i^{\text{old}} + \underline{\mu} \ln(v_i - \underline{v}_i) + \overline{\mu} \ln(\overline{v}_i - v_i) \qquad (5.12)$$
    and reports $\text{value}_i^{\text{old}}$ to the coordinator;
 4: the coordinator computes the original objective value
        $$L^{\text{old}} = a_0 p_0^2 + b_0 p_0 + \sum_{i \in \mathcal{N}^+} \text{value}_i^{\text{old}};$$
 5: run Algorithm 8 to obtain $(\partial_{p_i} L, \partial_{q_i} L)$ for each agent $i \in \mathcal{N}^+$;
 6: the coordinator initializes the step size $\eta \leftarrow 1$;
 7: the coordinator broadcasts $\eta$ to all agents $i \in \mathcal{N}^+$;
 8: each agent $i \in \mathcal{N}^+$ computes
        $$p_i^{\text{new}} \leftarrow \prod_{[\underline{p}_i, \overline{p}_i]} \left(p_i^{\text{old}} - \eta \partial_{p_i} L\right); \qquad (5.13a)$$
        $$q_i^{\text{new}} \leftarrow \prod_{[\underline{q}_i, \overline{q}_i]} \left(q_i^{\text{old}} - \eta \partial_{q_i} L\right); \qquad (5.13b)$$
        $$\Delta s_i \leftarrow \left|p_i^{\text{new}} - p_i^{\text{old}}\right| + \left|q_i^{\text{new}} - q_i^{\text{old}}\right|; \qquad (5.13c)$$
        $$\Delta L_i \leftarrow \partial_{p_i} L \cdot (p_i^{\text{new}} - p_i^{\text{old}}) + \partial_{q_i} L \cdot (q_i^{\text{new}} - q_i^{\text{old}}); \qquad (5.13d)$$
        $$\text{value}_i^{\text{new}} \leftarrow a_i \left(p_i^{\text{new}}\right)^2 + b_i p_i^{\text{new}} + \underline{\mu} \ln(v_i - \underline{v}_i) + \overline{\mu} \ln(\overline{v}_i - v_i), \qquad (5.13e)$$
    and reports $(p_i^{\text{new}}, q_i^{\text{new}}, \Delta s_i, \Delta L_i, \text{value}_i^{\text{new}})$ to the coordinator;
 9: network power flows stabilize to reach a steady state $(P, Q, v, p_0, q_0)$;
10: if an agent $i \in \mathcal{N}^+$ detects $v_i \notin [\underline{v}_i, \overline{v}_i]$, the agent sends out a signal to the coordinator;
    the coordinator sets $\eta \leftarrow \alpha\eta$, and returns to Step 7);
11: the coordinator computes
        $$\Delta s \leftarrow \sum_{i \in \mathcal{N}^+} \Delta s_i;$$
        $$L^{\text{new}} \leftarrow a_0 p_0^2 + b_0 p_0 + \sum_{i \in \mathcal{N}^+} \text{value}_i^{\text{new}};$$
        $$L^{\text{thres}} \leftarrow L^{\text{old}} + \beta \sum_{i \in \mathcal{N}^+} \Delta L_i;$$
    if $\Delta s \le \epsilon$, the coordinator sends out a signal to terminate the inner loop, i.e., go to Step 13);
    else if $L^{\text{new}} > L^{\text{thres}}$, the coordinator sets $\eta \leftarrow \alpha\eta$, and returns to Step 7);
    otherwise, the coordinator sends out a signal to update $(p^{\text{old}}, q^{\text{old}})$, i.e., go to Step 12);
12: each agent $i \in \mathcal{N}^+$ sets $p_i^{\text{old}} \leftarrow p_i^{\text{new}}$, $q_i^{\text{old}} \leftarrow q_i^{\text{new}}$, and returns to Step 3);
13: the coordinator sets
        $$\text{resetFlag} \leftarrow \begin{cases} 1 & \text{if } L^{\text{new}} > L^{\text{old}}, \\ 0 & \text{otherwise,} \end{cases}$$
    and broadcasts resetFlag to all agents $i \in \mathcal{N}^+$;
14: each agents sets
        $$(p_i', q_i') \leftarrow \begin{cases} (p_i^{\text{old}}, q_i^{\text{old}}) & \text{if resetFlag} = 1, \\ (p_i^{\text{new}}, q_i^{\text{new}}) & \text{otherwise;} \end{cases}$$
```

5.3 Performance analysis

Note that Algorithm 7 may converge to a local optimum of the OPF problem (5.2). Hence, we analyze the suboptimality of Algorithm 7 in this section.

5.3.1 Convexity

The OPF problem (5.1), or equivalently (5.2), is nonconvex. If they were convex, then the suboptimality gap of Algorithm 7 will be 0. We will show in this subsection that (5.2) is "nearly" convex. Hence, the suboptimality gap of Algorithm 7 should be small.

Compute the Hessian matrix. Let $H(x;\mu) := \partial_{xx}L(x;\mu)$ denote the Hessian matrix defined on $\mathbb{X} := [\underline{x}, \overline{x}]$. Let $\mathrm{diag}(a)$ denote the diagonal matrix with diagonal entries $(a_1, a_2, \ldots, a_n)$, then

$$\begin{aligned}
\frac{\partial^2 L}{\partial p_i \partial p_j} &= \frac{\partial}{\partial p_j}\left\{(2a_0p_0+b_0)\frac{\partial p_0}{\partial p_i} + (2a_ip_i+b_i) - \sum_{k=1}^n \left[\frac{\underline{\mu}}{v_k-\underline{v}_k} + \frac{\overline{\mu}}{v_k-\overline{v}_k}\right]\frac{\partial v_k}{\partial p_i}\right\} \\
&= \frac{\partial}{\partial p_j}\left\{(2a_0p_0+b_0)\frac{\partial p_0}{\partial p_i}\right\} + \frac{\partial}{\partial p_j}\{2a_ip_i+b_i\} - \frac{\partial}{\partial p_j}\left\{\sum_{k=1}^n \left[\frac{\underline{\mu}}{v_k-\underline{v}_k} + \frac{\overline{\mu}}{v_k-\overline{v}_k}\right]\frac{\partial v_k}{\partial p_i}\right\} \\
&= 2a_0\frac{\partial p_0}{\partial p_i}\frac{\partial p_0}{\partial p_j} + (2a_0p_0+b_0)\frac{\partial^2 p_0}{\partial p_i\partial p_j} + 2a_i\mathbf{1}_{i=j} \\
&\quad + \sum_{k=1}^n \left[\frac{\underline{\mu}}{(v_k-\underline{v}_k)^2} + \frac{\overline{\mu}}{(v_k-\overline{v}_k)^2}\right]\frac{\partial v_k}{\partial p_i}\frac{\partial v_k}{\partial p_j} - \sum_{k=1}^n \left[\frac{\underline{\mu}}{v_k-\underline{v}_k} + \frac{\overline{\mu}}{v_k-\overline{v}_k}\right]\frac{\partial^2 v_k}{\partial p_i\partial p_j} \\
&= 2a_0\left[\partial_p p_0 \partial_p^T p_0\right]_{ij} + (2a_0p_0+b_0)\left[\partial_{pp}p_0\right]_{ij} + 2\left[\mathrm{diag}(a)\right]_{ij} \\
&\quad + \sum_{k=1}^n \left[\frac{\underline{\mu}}{(v_k-\underline{v}_k)^2} + \frac{\overline{\mu}}{(v_k-\overline{v}_k)^2}\right]\left[\partial_p v_k \partial_p^T v_k\right]_{ij} - \sum_{k=1}^n \left[\frac{\underline{\mu}}{v_k-\underline{v}_k} + \frac{\overline{\mu}}{v_k-\overline{v}_k}\right]\left[\partial_{pp}v_k\right]_{ij}
\end{aligned}$$

for $i, j = 1, 2, \ldots, n$. Hence,

$$\begin{aligned}
\partial_{pp}L &= 2a_0\partial_p p_0 \partial_p^T p_0 + (2a_0p_0+b_0)\partial_{pp}p_0 + 2\mathrm{diag}(a) \\
&\quad + \sum_{k=1}^n \left[\frac{\underline{\mu}}{(v_k-\underline{v}_k)^2} + \frac{\overline{\mu}}{(v_k-\overline{v}_k)^2}\right]\partial_p v_k \partial_p^T v_k - \sum_{k=1}^n \left[\frac{\underline{\mu}}{v_k-\underline{v}_k} + \frac{\overline{\mu}}{v_k-\overline{v}_k}\right]\partial_{pp}v_k.
\end{aligned}$$

Similarly, one can compute

$$\begin{aligned}
\partial_{qq}L &= 2a_0\partial_q p_0 \partial_q^T p_0 + (2a_0p_0+b_0)\partial_{qq}p_0 \\
&\quad + \sum_{k=1}^n \left[\frac{\underline{\mu}}{(v_k-\underline{v}_k)^2} + \frac{\overline{\mu}}{(v_k-\overline{v}_k)^2}\right]\partial_q v_k \partial_q^T v_k - \sum_{k=1}^n \left[\frac{\underline{\mu}}{v_k-\underline{v}_k} + \frac{\overline{\mu}}{v_k-\overline{v}_k}\right]\partial_{qq}v_k
\end{aligned}$$

and

$$\begin{aligned}\partial_{qp}L &= 2a_0\partial_q p_0 \partial_p^T p_0 + (2a_0p_0+b_0)\partial_{qp}p_0 \\ &\quad + \sum_{k=1}^{n}\left[\frac{\underline{\mu}}{(v_k-\underline{v}_k)^2}+\frac{\overline{\mu}}{(v_k-\overline{v}_k)^2}\right]\partial_q v_k \partial_p^T v_k - \sum_{k=1}^{n}\left[\frac{\underline{\mu}}{v_k-\underline{v}_k}+\frac{\overline{\mu}}{v_k-\overline{v}_k}\right]\partial_{qp}v_k.\end{aligned}$$

Recall $x=(p,q)$. It follows that the Hessian matrix

$$\begin{aligned}H(x;\mu) &= \begin{bmatrix}\partial_{pp}L & \partial_{pq}L \\ \partial_{qp}L & \partial_{qq}L\end{bmatrix} \\ &= 2a_0\left[\partial_x p_0\right]\left[\partial_x p_0\right]^T + (2a_0p_0+b_0)\partial_{xx}p_0 + 2\begin{bmatrix}\mathrm{diag}(a) & \\ & 0\end{bmatrix} \\ &\quad + \sum_{k=1}^{n}\left(\frac{\underline{\mu}}{(v_k-\underline{v}_k)^2}+\frac{\overline{\mu}}{(v_k-\overline{v}_k)^2}\right)\left[\partial_x v_k\right]\left[\partial_x v_k\right]^T - \sum_{k=1}^{n}\left[\frac{\underline{\mu}}{v_k-\underline{v}_k}+\frac{\overline{\mu}}{v_k-\overline{v}_k}\right]\partial_{xx}v_k.\end{aligned} \tag{5.14}$$

Conditions for convexity. The following conditions will be encountered in asserting the positive semidefiniteness of $H(x;\mu)$.

C1) Quadratic coefficients are nonnegative, i.e., $a_i \geq 0$ for $i=0,1,\ldots,n$;

C2) Marginal cost at the substation is nonnegative, i.e., $2a_0p_0+b_0\geq 0$;

C3) $p_0(x)$ is convex on $x\in\mathbb{X}$;

C4) $v_k(x)$ is concave on $x\in\mathbb{X}$ for $k=1,2,\ldots,n$.

C1 and C2 are satisfied in practice. To justify C3, consider the following $\mathrm{SOCP}_0(x)$ problem

$$\begin{aligned}\mathrm{SOCP}_0(p,q):\ \ &\min\ \ p_0 \\ &\text{over}\ \ v,\ell,P,Q,p_0,q_0; \\ &\text{s.t.}\ \ \sum_{h:\,h\to i}(P_{hi}-r_{hi}\ell_{hi})+p_i = \sum_{j:\,i\to j}P_{ij}, \qquad i\in\mathcal{N}; \qquad (5.15a) \\ &\qquad \sum_{h:\,h\to i}(Q_{hi}-x_{hi}\ell_{hi})+q_i = \sum_{j:\,i\to j}Q_{ij}, \qquad i\in\mathcal{N}; \qquad (5.15b) \\ &\qquad v_i - 2(r_{ij}P_{ij}+x_{ij}Q_{ij}) + |z_{ij}|^2\ell_{ij} = v_j, \qquad i\to j; \qquad (5.15c) \\ &\qquad \ell_{ij}\geq \frac{P_{ij}^2+Q_{ij}^2}{v_i}, \qquad i\to j \qquad (5.15d)\end{aligned}$$

for each fixed $x=(p,q)$ in $\mathbb{X}$. $\mathrm{SOCP}_0(x)$ is a convex relaxation and will be explained in great detail in the next chapter. Here just note that (5.15d) is an inequality constraint while power flow constraint (5.1e) is an equality constraint. Also, we say that $\mathrm{SOCP}_0(x)$ is exact if every one of its optimal solutions satisfies (5.1e). The reason we emphasize SOCP_0 is that it has a close connection with C3, as formally stated in the following theorem.

Theorem 5.6. *Let C be convex. If $\text{SOCP}_0(x)$ is exact for $x \in C$, then $p_0(x)$ is convex on C.*

Theorem 5.6 implies that if $\text{SOCP}_0(x)$ is exact on $\mathbb{X}$, then $p_0(x)$ is convex on $\mathbb{X}$.

Proof. Let $\tilde{x} = (\tilde{p}, \tilde{q}) \in C$ and $\hat{x} = (\hat{p}, \hat{q}) \in C$ be distinct. It suffices to show that the point $x := \theta\tilde{x} + (1-\theta)\hat{x}$ satisfies $p_0(x) \le \theta p_0(\tilde{x}) + (1-\theta)p_0(\hat{x})$ for any $\theta \in (0,1)$.

Let $(\tilde{P}, \tilde{Q}, \tilde{v}, \tilde{\ell}, \tilde{p}_0, \tilde{q}_0)$ denote the power flow solution with respect to $\tilde{x}$, then $\tilde{p}_0 = p_0(\tilde{x})$. Let $(\hat{P}, \hat{Q}, \hat{v}, \hat{\ell}, \hat{p}_0, \hat{q}_0)$ denote the power flow solution with respect to $\hat{x}$, then $\hat{p}_0 = p_0(\hat{x})$. Since $p_0(x)$ is the optimal value of $\text{SOCP}_0(x)$, and the point

$$(P, Q, v, \ell, p_0, q_0) = \theta(\tilde{P}, \tilde{Q}, \tilde{v}, \tilde{\ell}, \tilde{p}_0, \tilde{q}_0) + (1-\theta)(\hat{P}, \hat{Q}, \hat{v}, \hat{\ell}, \hat{p}_0, \hat{q}_0)$$

is feasible for $\text{SOCP}_0(x)$, one must have

$$p_0(x) \le p_0 = \theta\tilde{p}_0 + (1-\theta)\hat{p}_0 = \theta p_0(\tilde{x}) + (1-\theta)p_0(\hat{x}).$$

This completes the proof of Theorem 5.6. □

Remarkably, $\text{SOCP}_0(x)$ is exact on $\mathbb{X}$ under mild conditions, as formalized in Theorem 5.7.

Theorem 5.7. *$\text{SOCP}_0(x)$ is exact for $x \in \mathbb{X}$ if Condition C1 in Theorem 6.2 holds.*

The Condition C1 in Theorem 6.2 is mild and always holds in practice with large margin, as will be elaborated in the next chapter. Hence, $\text{SOCP}_0(x)$ is exact on $\mathbb{X}$ under mild conditions. It follows that $p_0(x)$ is convex on $\mathbb{X}$. The proof of Theorem 5.7 is similar to that of Theorem 6.2 and omitted for brevity.

To justify C4, consider the following $\text{SOCP}_k(p, q)$ problem:

$$\begin{aligned}
\text{SOCP}_k(p,q): \quad \max \quad & v_k \\
\text{over} \quad & v, \ell, P, Q, p_0, q_0; \\
\text{s.t.} \quad & \sum_{h:\, h\to i} (P_{hi} - r_{hi}\ell_{hi}) + p_i = \sum_{j:\, i\to j} P_{ij}, && i \in \mathcal{N}; && (5.16a) \\
& \sum_{h:\, h\to i} (Q_{hi} - x_{hi}\ell_{hi}) + q_i = \sum_{j:\, i\to j} Q_{ij}, && i \in \mathcal{N}; && (5.16b) \\
& v_i - 2(r_{ij}P_{ij} + x_{ij}Q_{ij}) + (r_{ij}^2 + x_{ij}^2)\ell_{ij} = v_j, && i \to j; && (5.16c) \\
& \ell_{ij} \ge \frac{P_{ij}^2 + Q_{ij}^2}{v_i}, && i \to j && (5.16d)
\end{aligned}$$

for each $k \in \mathcal{N}^+$ and each $(p,q) \in \mathbb{X}$. $\mathrm{SOCP}_k(p,q)$ is a convex relaxation that will be explained in great detail in the next chapter. Here just note that (5.16d) is an inequality constraint while power flow constraint (5.1e) is an equality constraint. Also, we say that $\mathrm{SOCP}_k(p,q)$ is exact if every one of its optimal solutions satisfies (5.1e). The reason we emphasize SOCP_k is that it has a close connection with C4, as formally stated in the following theorem.

Theorem 5.8. *Let C be convex and $k \in \mathcal{N}^+$. If $\mathrm{SOCP}_k(x)$ is exact for $x \in C$, then $v_k(x)$ is concave on C.*

Theorem 5.8 implies that if $\mathrm{SOCP}_k(x)$ is exact on $\mathbb{X}$, then $v_k(x)$ is concave on $\mathbb{X}$. The proof of Theorem 5.8 is similar to that of Theorem 5.6 and omitted for brevity.

Similar to the case of SOCP_0, SOCP_k is also exact under mild conditions for $k \in \mathcal{N}^+$.

Theorem 5.9. $\mathrm{SOCP}_k(p,q)$ *is exact for $k \in \mathcal{N}^+$ and for $x \in \mathbb{X}$ if Condition C1 in Theorem 6.2 holds.*

The Condition C1 in Theorem 6.2 is mild and always holds in practice with large margin, as will be elaborated on in the next chapter. Hence, $\mathrm{SOCP}_k(x)$ is exact on $\mathbb{X}$ under mild conditions. It follows that $v_k(x)$ is concave on $\mathbb{X}$. The proof of Theorem 5.9 is similar to that of Theorem 6.2 and omitted for brevity.

Corollary 5.10. *Assume Condition C1 in Theorem 6.2 holds. Then $p_0(x)$ is convex on $\mathbb{X}$ and $v_k(x)$ is concave on $\mathbb{X}$ for $k \in \mathcal{N}^+$.*

Corollary 5.10 implies that C3 and C4 hold under mild conditions that are widely satisfied in practice. This completes the justification of C1–C4.

Convexity results. The following theorem studies the region A where $H(x;\mu)$ is positive semidefinite, i.e., $L(x;\mu)$ is locally convex.

Theorem 5.11. *Assume C1–C4 hold. Then $H(x;\mu) \succeq 0$ on*

$$A := \left\{ x \in \mathbb{X} \mid v(x) \le \frac{\underline{\mu}}{\overline{\mu}+\underline{\mu}}\overline{v} + \frac{\overline{\mu}}{\overline{\mu}+\underline{\mu}}\underline{v} \right\}.$$

In particular, $H(x;\mu) \succeq 0$ on $\mathbb{X}$ if $\mu = 0$ or $\overline{v} = \infty$.

Note that if $\overline{v}$ equals ∞, then $H(x;\mu) \succeq 0$ on $\mathbb{X}$. In general, $H(x;\mu) \succeq 0$ on a subset A of $\mathbb{X}$. The proof of Theorem 5.11 is a direct application of (5.14) and omitted for brevity.

Remark 5.8. *Let $F := \{x \in \mathbb{X} \mid \underline{v}_i \le v_i(x) \le \overline{v}_i \text{ for } i \in \mathcal{N}^+\}$ denote the feasible set of (5.2). If the sequence $\mu_1, \mu_2, \ldots, \mu_k, \ldots$ of μ is chosen according to*

$$\underline{\mu}_k = \delta_k,\ \overline{\mu}_k = \delta_k^2, \quad k = 1, 2, \ldots,$$

where $\lim_{k\to\infty} \delta_k = 0$, *then the difference set* $F \backslash A$ *vanishes as* $k \to \infty$.

We would like to emphasize that Theorem 5.11 does not guarantee $L(x;\mu)$ to be convex over F, though the set A can be arbitrarily close to F with carefully chosen μ. In fact, $L(x;\mu)$ cannot be convex over F since F is nonconvex.

5.3.2 Suboptimality

Unlike most nonlinear programming algorithms, the suboptimality of Algorithm 7 can be characterized as in the following theorem.

Theorem 5.12 (suboptimality). *Assume C1–C4 hold. Let* $x' = (p', q')$ *be feasible for* (5.2) *and* $x^* = (p^*, q^*)$ *be a local optimum of* $L(x;\mu)$. *Assume the matrix* $\partial_q v|_{x^*}$ *to be invertible, and define*

$$\partial_q v := \begin{bmatrix} \partial_q v_1 & \partial_q v_2 & \cdots & \partial_q v_n \end{bmatrix},$$

$$r(\theta) := \frac{1}{2}\left[\partial_q v|_{x^*}\right]^{-1} \begin{bmatrix} (x'-x^*)^T \cdot \partial_{xx} v_1|_{\theta x' + (1-\theta)x^*} \cdot (x'-x^*) \\ (x'-x^*)^T \cdot \partial_{xx} v_2|_{\theta x' + (1-\theta)x^*} \cdot (x'-x^*) \\ \vdots \\ (x'-x^*)^T \cdot \partial_{xx} v_n|_{\theta x' + (1-\theta)x^*} \cdot (x'-x^*) \end{bmatrix}, \quad \theta \in (0,1).$$

If $q' + r(\theta) \in (\underline{q}, \overline{q})$ *for all* $\theta \in (0,1)$, *then*

$$L(x^*;\mu) - L(x';\mu) \le (2a_0 p_0^* + b_0) \cdot \left(\partial_q p_0|_{x^*}\right)^T \cdot r(\eta) - a_0(p_0^* - p_0')^2$$

for some $\eta \in (0,1)$, *where* $p_0^* = p_0(x^*)$ *and* $p_0' = p_0(x')$.

Theorem 5.12 characterizes the suboptimality of an arbitrary local minimum x^* of $L(x;\mu)$. In particular, for any $x' = (p', q') \in F$ that satisfies $q' + r(\theta) \in (\underline{q}, \overline{q})$, the objective value $L(x^*;\mu)$ cannot exceed $L(x';\mu)$ by more than $(2a_0 p_0^* + b_0) \cdot \partial_q p_0|_{x^*} \cdot r(\eta) - a_0(p_0^* - p_0')^2$.

The term $r(\theta)$ is the deviation of $v(x)$ from its linear approximation, and is in practice small for all $\theta \in (0,1)$. Hence, x' can be nearly all points in $F \cap \mathbb{X}^\circ$ where $\mathbb{X}^\circ$ denotes the interior of $\mathbb{X}$. Additionally, the derivative $\partial_q p_0$ is also small and therefore the gap $(2a_0 p_0^* + b_0) \cdot \partial_q p_0|_{x^*} \cdot r(\eta) - a_0(p_0^* - p_0')^2 \approx -a_0(p_0^* - p_0')^2$. Hence, Theorem 5.12 roughly says that *a local optimum* x^* *is no worse than any strictly feasible point* x'.

Proof. The idea is to create a trajectory $x(\theta)$ of feasible solutions of (5.2) that approaches x^* as $\theta \to 0$, and make use of the fact that $L(x^*) \le L(x(\theta))$ for sufficiently small θ.

The trajectory $x(\theta)$ is constructed using the implicit function theorem. Let (v', p_0') and (v^*, p_0^*) denote the power flow solutions corresponding to x' and x^* respectively, i.e., $v' = v(x')$, $p_0' = p_0(x')$,

$v^* = v(x^*)$, $p_0^* = p_0(x^*)$. Consider the following function:

$$f(q,\theta) := (1-\theta)v^* + \theta v' - v\left[(1-\theta)p^* + \theta p', q\right].$$

Note that $f(q^*, 0) = v^* - v(p^*, q^*) = 0$ and that the partial derivative $\partial_q f = -\partial_q v$ is full rank. Hence, there exists $q(\theta)$ near a small neighborhood $(-\eta, \eta)$ where $\eta > 0$ of 0 that satisfies

$$q(0) = q^*, \quad f(q(\theta), \theta) = 0.$$

The equality $f(q(\theta), \theta) = 0$ is equivalent to

$$(1-\theta)v^* + \theta v' = v\left[(1-\theta)p^* + \theta p', q(\theta)\right].$$

Let $v(\theta) := (1-\theta)v^* + \theta v'$ and $p(\theta) := (1-\theta)p^* + \theta p'$ for $\theta \in (-\eta, \eta)$, then $v(\theta) = v(p(\theta), q(\theta))$. Let $p_0(\theta) := p_0(p(\theta), q(\theta))$ for $\theta \in (-\eta, \eta)$. To this point, the trajectory $x(\theta)$ has been constructed.

Now we show that $x(\theta) \in F$ for sufficiently small θ. In particular, it suffices to prove $q(\theta) \in [\underline{q}, \overline{q}]$ for sufficiently small θ. It follows from

$$0 \;=\; \partial_q f|_{(q^*,0)} \cdot \partial_\theta q|_0 + \partial_\theta f|_{(q^*,0)} \;=\; -\,\partial_q v|_{x^*} \cdot \partial_\theta q|_0 + v' - v^* - \partial_p v|_{x^*} \cdot (p' - p^*)$$

that

$$\begin{aligned}
\partial_\theta q|_0 &= \left[\partial_q v|_{x^*}\right]^{-1} \cdot \left[v' - v^* - \partial_p v|_{x^*} \cdot (p' - p^*)\right] \\
&= \left[\partial_q v|_{x^*}\right]^{-1} \cdot \left\{ \partial_q v|_{x^*} \cdot (q' - q^*) + \frac{1}{2} \begin{bmatrix} (x'-x^*)^T \cdot \partial_{xx} v_1|_{\nu x' + (1-\nu)x^*} \cdot (x'-x^*) \\ (x'-x^*)^T \cdot \partial_{xx} v_2|_{\nu x' + (1-\nu)x^*} \cdot (x'-x^*) \\ \vdots \\ (x'-x^*)^T \cdot \partial_{xx} v_n|_{\nu x' + (1-\nu)x^*} \cdot (x'-x^*) \end{bmatrix} \right\} \\
&= q' - q^* + r(\nu) \;\in\; \left(\underline{q}, \overline{q}\right) - q^*
\end{aligned}$$

for some $\nu \in (0,1)$. Therefore, by the hypothesis of the theorem,

$$q(\theta) = q^* + \theta \cdot \partial_\theta q|_0 + o(\theta) \in \left(\underline{q}, \overline{q}\right)$$

for sufficiently small θ.

Finally we make use of the local optimality of $L(x^*)$, which implies $L(x^*) \leq L(x(\theta))$ for sufficiently small $\theta > 0$. Substitute

$$\begin{aligned} L(x(\theta)) &= a_0 p_0^2(\theta) + b_0 p_0(\theta) + \sum_{i=1}^n a_i p_i^2(\theta) + b_i p_i(\theta) - \sum_{i=1}^n \left[\underline{\mu}\ln(v_i(\theta) - \underline{v}_i) + \overline{\mu}\ln(\overline{v}_i - v_i(\theta))\right] \\ &\le a_0 p_0^2(\theta) + b_0 p_0(\theta) + (1-\theta)\left(\sum_{i=1}^n a_i\left[p_i^*\right]^2 + b_i p_i^*\right) + \theta\left(\sum_{i=1}^n a_i\left[p_i'\right]^2 + b_i p_i'\right) \\ &\quad - (1-\theta)\sum_{i=1}^n \left[\underline{\mu}\ln(v_i^* - \underline{v}_i) + \overline{\mu}\ln(\overline{v}_i - v_i^*)\right] - \theta\sum_{i=1}^n \left[\underline{\mu}\ln(v_i' - \underline{v}_i) + \overline{\mu}\ln(\overline{v}_i - v_i')\right] \\ &= (1-\theta)L(x^*) + \theta L(x') + a_0 p_0^2(\theta) + b_0 p_0(\theta) - (1-\theta)\left[a_0\left(p_0^*\right)^2 + b_0 p_0^*\right] - \theta\left[a_0\left(p_0'\right)^2 + b_0 p_0'\right] \end{aligned}$$

to obtain

$$\theta\left[L(x^*) - L(x')\right] \le a_0 p_0^2(\theta) + b_0 p_0(\theta) - a_0\left(p_0^*\right)^2 - b_0 p_0^* + \theta\left[a_0\left(p_0^*\right)^2 + b_0 p_0^* - a_0\left(p_0'\right)^2 - b_0 p_0'\right]$$

for sufficiently small $\theta > 0$. Take the gradient with respect to θ at $\theta = 0$ to obtain that

$$\begin{aligned} L(x^*) - L(x') &\le (2a_0 p_0^* + b_0)\cdot\left[\partial_p p_0|_{x^*}\cdot(p' - p^*) + \partial_q p_0|_{x^*}\cdot(q' - q^* + r(\nu))\right] \\ &\quad + a_0\left(p_0^*\right)^2 + b_0 p_0^* - a_0\left(p_0'\right)^2 - b_0 p_0'. \end{aligned}$$

Due to the convexity of $p_0(x)$, one has

$$p_0' - p_0^* \ \ge\ \partial_p p_0|_{x^*}\cdot(p' - p^*) + \partial_q p_0|_{x^*}\cdot(q' - q^*)$$

and therefore

$$\begin{aligned} L(x^*) - L(x') &\le (2a_0 p_0^* + b_0)\cdot\left[p_0' - p_0^* + \partial_q p_0|_{x^*}\cdot r(\nu)\right] + a_0\left(p_0^*\right)^2 + b_0 p_0^* - a_0\left(p_0'\right)^2 - b_0 p_0' \\ &= (2a_0 p_0^* + b_0)\cdot\partial_q p_0|_{x^*}\cdot r(\nu) - a_0(p_0^* - p_0')^2. \end{aligned}$$

This completes the proof of Theorem 5.12. □

5.4 Multiphase Networks

The objective function to minimize is

$$L(x;\mu) = \sum_{\phi\in\Phi_0} a\left(p_0^\phi(x)\right)^2 + b p_0^\phi(x) - \sum_{k=1}^n \sum_{\phi\in\Phi_k}\left(\underline{\mu}\ln(v_k^\phi - \underline{v}_k) + \overline{\mu}\ln(\overline{v}_k - v_k^\phi)\right).$$

Hence, to compute $\partial_x L$, it suffices to compute $\partial_x p_0^\phi$ and $\partial_x v_k^\phi$ for each $k \in \mathcal{N}^+$ and each $\phi \in \{a, b, c\}$.

We estimate $\partial_x p_0^\phi$ and $\partial_x v_k^\phi$ for each $k \in \mathcal{N}^+$ and each $\phi \in \{a, b, c\}$ using the linearized power flow equations (4.12) repeated below:

$$\begin{aligned}
&\sum_{i:\, i\to j} \Lambda_{ij} + s_j = \sum_{k:\, j\to k} \Lambda_{jk}^{\Phi_j}, && j \in \mathcal{N};\\
&S_{ij} = \gamma^{\Phi_{ij}} \operatorname{diag}(\Lambda_{ij}), && i \to j;\\
&v_j = v_i^{\Phi_{ij}} - S_{ij} z_{ij}^H - z_{ij} S_{ij}^H, && i \to j.
\end{aligned}$$

In particular, let $\Lambda_{ij} = P_{ij} + \mathbf{i} Q_{ij}$ for $i \to j$, then (here we take the positive direction of s as being load)

$$\begin{aligned}
&\frac{\partial}{\partial p_i^\varphi} P_{kl}^\phi = \mathbf{1}_{\phi=\varphi} \mathbf{1}_{i\in \mathrm{down}(l)}, && \frac{\partial}{\partial q_i^\varphi} P_{kl}^\phi = 0, && i \in \mathcal{N}^+,\ k\to l,\ \varphi\in\Phi_i,\ \phi\in\Phi_{kl};\\
&\frac{\partial}{\partial p_i^\varphi} Q_{kl}^\phi = 0, && \frac{\partial}{\partial q_i^\varphi} Q_{kl}^\phi = \mathbf{1}_{\phi=\varphi} \mathbf{1}_{i\in \mathrm{down}(l)}, && i \in \mathcal{N}^+,\ k\to l,\ \varphi\in\Phi_i,\ \phi\in\Phi_{kl};\\
&\frac{\partial}{\partial p_i^\varphi} p_0^\phi = \mathbf{1}_{\phi=\varphi}, && \frac{\partial}{\partial q_i^\varphi} p_0^\phi = 0, && i \in \mathcal{N}^+,\ \varphi\in\Phi_i,\ \phi\in\Phi_{kl}.
\end{aligned}$$

Hence, we are left to compute $\partial_x v_k^\phi$ for $k \in \mathcal{N}^+$ and $\phi \in \Phi_k$.

To compute $\partial_x v_k^\phi$, noting that

$$v_j^\phi = v_i^\phi - \sum_{\varphi\in\Phi_{ij}} \left(S_{ij}^{\phi\varphi} \overline{z_{ij}^{\phi\varphi}} + z_{ij}^{\phi\varphi} \overline{S_{ij}^{\phi\varphi}} \right)$$

and that $S_{ij}^{\phi\varphi} = \Lambda_{ij}^\varphi \alpha^{\phi-\varphi}$ for $\phi, \varphi \in \Phi_{ij}$ (use $a = 0$, $b = 1$, and $c = 2$ when performing $\phi - \varphi$), one obtains

$$\begin{aligned}
\frac{\partial}{\partial p_k^\xi} v_j^\phi &= \frac{\partial}{\partial p_k^\xi} v_i^\phi - \sum_{\varphi\in\Phi_{ij}} \left(\overline{z_{ij}^{\phi\varphi}} \frac{\partial}{\partial p_k^\xi} S_{ij}^{\phi\varphi} + z_{ij}^{\phi\varphi} \frac{\partial}{\partial p_k^\xi} \overline{S_{ij}^{\phi\varphi}} \right)\\
&= \frac{\partial}{\partial p_k^\xi} v_i^\phi - \sum_{\varphi\in\Phi_{ij}} \left(\overline{z_{ij}^{\phi\varphi}} \alpha^{\phi-\varphi} \frac{\partial}{\partial p_k^\xi} \Lambda_{ij}^\varphi + z_{ij}^{\phi\varphi} \alpha^{\varphi-\phi} \frac{\partial}{\partial p_k^\xi} \overline{\Lambda_{ij}^\varphi} \right)\\
&= \frac{\partial}{\partial p_k^\xi} v_i^\phi - \sum_{\varphi\in\Phi_{ij}} \left(\overline{z_{ij}^{\phi\varphi}} \alpha^{\phi-\varphi} + z_{ij}^{\phi\varphi} \alpha^{\varphi-\phi} \right) \mathbf{1}_{\xi=\varphi} \mathbf{1}_{k\in\mathrm{down}(j)}\\
&= \frac{\partial}{\partial p_k^\xi} v_i^\phi - \left(\overline{z_{ij}^{\phi\xi}} \alpha^{\phi-\xi} + z_{ij}^{\phi\xi} \alpha^{\xi-\phi} \right) \mathbf{1}_{\xi\in\Phi_{ij}} \mathbf{1}_{k\in\mathrm{down}(j)}
\end{aligned}$$

for $i \to j$, $\phi \in \Phi_{ij}$, $k \in \mathcal{N}^+$, and $\xi \in \Phi_k$. Sum up over the path $\mathcal{P}_l$ to obtain that

$$\frac{\partial}{\partial p_k^\xi} v_l^\phi = - \sum_{(i,j)\in\mathcal{P}_l} \left(\overline{z_{ij}^{\phi\xi}} \alpha^{\phi-\xi} + z_{ij}^{\phi\xi} \alpha^{\xi-\phi} \right) \mathbf{1}_{\xi\in\Phi_{ij}} \mathbf{1}_{k\in\mathrm{down}(j)}$$

for $k \in \mathcal{N}^+$, $\xi \in \Phi_k$, $l \in \mathcal{N}^+$, and $\phi \in \Phi_l$. Substitute variables to obtain that

$$\frac{\partial}{\partial p_i^\xi} v_k^\phi = -\sum_{(s,t)\in\mathcal{P}_k} \left(\overline{z_{st}^{\phi\xi}}\alpha^{\phi-\xi} + z_{st}^{\phi\xi}\alpha^{\xi-\phi}\right)\mathbf{1}_{\xi\in\Phi_{st}}\mathbf{1}_{i\in\mathrm{down}(t)}$$

for $i \in \mathcal{N}^+$, $\xi \in \Phi_i$, $k \in \mathcal{N}^+$, and $\phi \in \Phi_k$. Then,

$$\begin{aligned}\frac{\partial}{\partial p_i^\xi} v_k^\phi - \frac{\partial}{\partial p_j^\xi} v_k^\phi &= \sum_{(s,t)\in\mathcal{P}_k} \left(\overline{z_{st}^{\phi\xi}}\alpha^{\phi-\xi} + z_{st}^{\phi\xi}\alpha^{\xi-\phi}\right)\mathbf{1}_{\xi\in\Phi_{st}}\left(\mathbf{1}_{j\in\mathrm{down}(t)} - \mathbf{1}_{i\in\mathrm{down}(t)}\right)\\ &= \sum_{(s,t)\in\mathcal{P}_k} \left(\overline{z_{st}^{\phi\xi}}\alpha^{\phi-\xi} + z_{st}^{\phi\xi}\alpha^{\xi-\phi}\right)\mathbf{1}_{\xi\in\Phi_{st}}\mathbf{1}_{j=t}\\ &= \left(\overline{z_{ij}^{\phi\xi}}\alpha^{\phi-\xi} + z_{ij}^{\phi\xi}\alpha^{\xi-\phi}\right)\mathbf{1}_{k\in\mathrm{down}(j)}\end{aligned}$$

for $i \to j$, $\xi \in \Phi_{ij}$, $k \in \mathcal{N}^+$, and $\phi \in \Phi_k$.

Similarly, one has

$$\begin{aligned}\frac{\partial}{\partial q_k^\xi} v_j^\phi &= \frac{\partial}{\partial q_k^\xi} v_i^\phi - \sum_{\varphi\in\Phi_{ij}} \left(\overline{z_{ij}^{\phi\varphi}}\frac{\partial}{\partial q_k^\xi} S_{ij}^{\phi\varphi} + z_{ij}^{\phi\varphi}\frac{\partial}{\partial q_k^\xi}\overline{S_{ij}^{\phi\varphi}}\right)\\ &= \frac{\partial}{\partial q_k^\xi} v_i^\phi - \sum_{\varphi\in\Phi_{ij}} \left(\overline{z_{ij}^{\phi\varphi}}\alpha^{\phi-\varphi}\frac{\partial}{\partial q_k^\xi}\Lambda_{ij}^\varphi + z_{ij}^{\phi\varphi}\alpha^{\varphi-\phi}\frac{\partial}{\partial q_k^\xi}\overline{\Lambda_{ij}^\varphi}\right)\\ &= \frac{\partial}{\partial q_k^\xi} v_i^\phi - \mathbf{i}\sum_{\varphi\in\Phi_{ij}} \left(\overline{z_{ij}^{\phi\varphi}}\alpha^{\phi-\varphi} - z_{ij}^{\phi\varphi}\alpha^{\varphi-\phi}\right)\mathbf{1}_{\xi=\varphi}\mathbf{1}_{k\in\mathrm{down}(j)}\\ &= \frac{\partial}{\partial q_k^\xi} v_i^\phi - \mathbf{i}\left(\overline{z_{ij}^{\phi\xi}}\alpha^{\phi-\xi} - z_{ij}^{\phi\xi}\alpha^{\xi-\phi}\right)\mathbf{1}_{\xi\in\Phi_{ij}}\mathbf{1}_{k\in\mathrm{down}(j)}\end{aligned}$$

for $i \to j$, $\phi \in \Phi_{ij}$, $k \in \mathcal{N}^+$, and $\xi \in \Phi_k$. Sum up over the path $\mathcal{P}_l$ to obtain that

$$\frac{\partial}{\partial q_k^\xi} v_l^\phi = -\mathbf{i}\sum_{(i,j)\in\mathcal{P}_l} \left(\overline{z_{ij}^{\phi\xi}}\alpha^{\phi-\xi} - z_{ij}^{\phi\xi}\alpha^{\xi-\phi}\right)\mathbf{1}_{\xi\in\Phi_{ij}}\mathbf{1}_{k\in\mathrm{down}(j)}$$

for $k \in \mathcal{N}^+$, $\xi \in \Phi_k$, $l \in \mathcal{N}^+$, and $\phi \in \Phi_l$. Substitute variables to obtain that

$$\frac{\partial}{\partial q_i^\xi} v_k^\phi = -\mathbf{i}\sum_{(s,t)\in\mathcal{P}_k} \left(\overline{z_{st}^{\phi\xi}}\alpha^{\phi-\xi} - z_{st}^{\phi\xi}\alpha^{\xi-\phi}\right)\mathbf{1}_{\xi\in\Phi_{st}}\mathbf{1}_{i\in\mathrm{down}(t)}$$

for $i \in \mathcal{N}^+$, $\xi \in \Phi_i$, $k \in \mathcal{N}^+$, and $\phi \in \Phi_k$. Then,

$$\begin{aligned}
\frac{\partial}{\partial q_i^\xi} v_k^\phi - \frac{\partial}{\partial q_j^\xi} v_k^\phi &= \mathbf{i} \sum_{(s,t)\in\mathcal{P}_k} \left(\overline{z_{st}^{\phi\xi}} \alpha^{\phi-\xi} - z_{st}^{\phi\xi} \alpha^{\xi-\phi} \right) \mathbf{1}_{\xi\in\Phi_{st}} \left(\mathbf{1}_{j\in\mathrm{down}(t)} - \mathbf{1}_{i\in\mathrm{down}(t)} \right) \\
&= \mathbf{i} \sum_{(s,t)\in\mathcal{P}_k} \left(\overline{z_{st}^{\phi\xi}} \alpha^{\phi-\xi} - z_{st}^{\phi\xi} \alpha^{\xi-\phi} \right) \mathbf{1}_{\xi\in\Phi_{st}} \mathbf{1}_{j=t} \\
&= \mathbf{i} \left(\overline{z_{ij}^{\phi\xi}} \alpha^{\phi-\xi} - z_{ij}^{\phi\xi} \alpha^{\xi-\phi} \right) \mathbf{1}_{k\in\mathrm{down}(j)}
\end{aligned}$$

for $i \to j$, $\xi \in \Phi_{ij}$, $k \in \mathcal{N}^+$, and $\phi \in \Phi_k$.

Now, the gradient $\partial_x L$ can be estimated as follows.

$$\begin{aligned}
\frac{\partial}{\partial p_i^\xi} L &= \sum_{\phi\in\Phi_0} \left(2ap_0^\phi + b \right) \frac{\partial}{\partial p_i^\xi} p_0^\phi - \sum_{k=1}^n \sum_{\phi\in\Phi_k} \left(\frac{\underline{\mu}}{v_k^\phi - \underline{v}_k} + \frac{\overline{\mu}}{v_k^\phi - \overline{v}_k} \right) \frac{\partial}{\partial p_i^\xi} v_k^\phi \\
&\approx 2ap_0^\xi + b - \sum_{k=1}^n \sum_{\phi\in\Phi_k} \left(\frac{\underline{\mu}}{v_k^\phi - \underline{v}_k} + \frac{\overline{\mu}}{v_k^\phi - \overline{v}_k} \right) \frac{\partial}{\partial p_i^\xi} v_k^\phi, \\
\frac{\partial}{\partial q_i^\xi} L &= \sum_{\phi\in\Phi_0} \left(2ap_0^\phi + b \right) \frac{\partial}{\partial q_i^\xi} p_0^\phi - \sum_{k=1}^n \sum_{\phi\in\Phi_k} \left(\frac{\underline{\mu}}{v_k^\phi - \underline{v}_k} + \frac{\overline{\mu}}{v_k^\phi - \overline{v}_k} \right) \frac{\partial}{\partial q_i^\xi} v_k^\phi \\
&\approx - \sum_{k=1}^n \sum_{\phi\in\Phi_k} \left(\frac{\underline{\mu}}{v_k^\phi - \underline{v}_k} + \frac{\overline{\mu}}{v_k^\phi - \overline{v}_k} \right) \frac{\partial}{\partial q_i^\xi} v_k^\phi
\end{aligned}$$

for $i \in \mathcal{N}^+$ and $\xi \in \Phi_i$. Then, the difference across lines are

$$\begin{aligned}
\frac{\partial}{\partial p_i^\xi} L - \frac{\partial}{\partial p_j^\xi} L &= \sum_{k=1}^n \sum_{\phi\in\Phi_k} \left(\frac{\underline{\mu}}{v_k^\phi - \underline{v}_k} + \frac{\overline{\mu}}{v_k^\phi - \overline{v}_k} \right) \left(\frac{\partial}{\partial p_j^\xi} v_k^\phi - \frac{\partial}{\partial p_i^\xi} v_k^\phi \right) \\
&= - \sum_{k=1}^n \sum_{\phi\in\Phi_k} \left(\frac{\underline{\mu}}{v_k^\phi - \underline{v}_k} + \frac{\overline{\mu}}{v_k^\phi - \overline{v}_k} \right) \left(\overline{z_{ij}^{\phi\xi}} \alpha^{\phi-\xi} + z_{ij}^{\phi\xi} \alpha^{\xi-\phi} \right) \mathbf{1}_{k\in\mathrm{down}(j)} \\
&= - \sum_{\phi\in\Phi_j} \left(\overline{z_{ij}^{\phi\xi}} \alpha^{\phi-\xi} + z_{ij}^{\phi\xi} \alpha^{\xi-\phi} \right) \sum_{k\in\mathrm{down}(j)} \mathbf{1}_{\phi\in\Phi_k} \left(\frac{\underline{\mu}}{v_k^\phi - \underline{v}_k} + \frac{\overline{\mu}}{v_k^\phi - \overline{v}_k} \right), \\
\frac{\partial}{\partial q_i^\xi} L - \frac{\partial}{\partial q_j^\xi} L &= \sum_{k=1}^n \sum_{\phi\in\Phi_k} \left(\frac{\underline{\mu}}{v_k^\phi - \underline{v}_k} + \frac{\overline{\mu}}{v_k^\phi - \overline{v}_k} \right) \left(\frac{\partial}{\partial q_j^\xi} v_k^\phi - \frac{\partial}{\partial q_i^\xi} v_k^\phi \right) \\
&= -\mathbf{i} \sum_{k=1}^n \sum_{\phi\in\Phi_k} \left(\frac{\underline{\mu}}{v_k^\phi - \underline{v}_k} + \frac{\overline{\mu}}{v_k^\phi - \overline{v}_k} \right) \left(\overline{z_{ij}^{\phi\xi}} \alpha^{\phi-\xi} - z_{ij}^{\phi\xi} \alpha^{\xi-\phi} \right) \mathbf{1}_{k\in\mathrm{down}(j)} \\
&= -\mathbf{i} \sum_{\phi\in\Phi_j} \left(\overline{z_{ij}^{\phi\xi}} \alpha^{\phi-\xi} - z_{ij}^{\phi\xi} \alpha^{\xi-\phi} \right) \sum_{k\in\mathrm{down}(j)} \mathbf{1}_{\phi\in\Phi_k} \left(\frac{\underline{\mu}}{v_k^\phi - \underline{v}_k} + \frac{\overline{\mu}}{v_k^\phi - \overline{v}_k} \right)
\end{aligned}$$

for $i \to j$ and $\xi \in \Phi_{ij}$. Define

$$g_j^\phi := \sum_{k\in\mathrm{down}(j)} \mathbf{1}_{\phi\in\Phi_k} \left(\frac{\underline{\mu}}{v_k^\phi - \underline{v}_k} + \frac{\overline{\mu}}{v_k^\phi - \overline{v}_k} \right)$$

for $j \in \mathcal{N}^+$ and $\phi \in \Phi_j$, then

$$\begin{aligned}
\frac{\partial}{\partial p_j^\xi} L &= \frac{\partial}{\partial p_i^\xi} L + \sum_{\phi \in \Phi_j} \left(\overline{z_{ij}^{\phi\xi}} \alpha^{\phi-\xi} + z_{ij}^{\phi\xi} \alpha^{\xi-\phi} \right) g_j^\phi, \\
\frac{\partial}{\partial q_j^\xi} L &= \frac{\partial}{\partial q_i^\xi} L + \mathbf{i} \sum_{\phi \in \Phi_j} \left(\overline{z_{ij}^{\phi\xi}} \alpha^{\phi-\xi} - z_{ij}^{\phi\xi} \alpha^{\xi-\phi} \right) g_j^\phi
\end{aligned}$$

for $i \to j$ and $\xi \in \Phi_{ij}$.

To summarize, the gradient can be estimated using the following backward-forward sweep method:

$$\begin{aligned}
g_i^\phi &= \frac{\underline{\mu}}{v_i^\phi - \underline{v}_i} + \frac{\overline{\mu}}{v_i^\phi - \overline{v}_i} + \sum_{j:\, i \to j} g_j^\phi, && i \to j,\ \phi \in \Phi_i; \\
\frac{\partial}{\partial p_j^\xi} L &= \frac{\partial}{\partial p_i^\xi} L + \sum_{\phi \in \Phi_j} \left(\overline{z_{ij}^{\phi\xi}} \alpha^{\phi-\xi} + z_{ij}^{\phi\xi} \alpha^{\xi-\phi} \right) g_j^\phi, && i \to j,\ \xi \in \Phi_j; \\
\frac{\partial}{\partial q_j^\xi} L &= \frac{\partial}{\partial q_i^\xi} L + \mathbf{i} \sum_{\phi \in \Phi_j} \left(\overline{z_{ij}^{\phi\xi}} \alpha^{\phi-\xi} - z_{ij}^{\phi\xi} \alpha^{\xi-\phi} \right) g_j^\phi, && i \to j,\ \xi \in \Phi_j.
\end{aligned}$$

5.5 Numerical Results

We evaluate the accuracy and efficiency of Algorithm 7 for a number of balanced test networks in this section. In particular, we use the convex relaxation approach to obtain the global optimal value of (5.1), and check by how much does the objective value obtained by Algorithm 7 deviate from the global optimal value. The convex relaxation is solved by CVX [49] and [96], and its execution time is used as a benchmark to investigate the efficiency of Algorithm 7. All simulations use matlab 7.9.0.529 (64-bit) with toolbox cvx 1.21 on Mac OS X 10.7.5 with 2.66GHz Intel Core 2 Due CPU and 4GB 1067MHz DDR3 memory.

5.5.1 Test Networks

The test networks include a 47-bus network [35], a 56-bus network [37], and subnetworks of a 2065-bus network. These networks are all in the service territory of Southern California Edison (SCE), a utility company in California, USA [2]. Topologies of the 47-bus network and the 56-bus network are given in Figure 5.1 with parameters provided in Table 5.1 and 5.2.

5.5.2 OPF Setup

The following OPF setup is used throughout this section.

1. The objective is to minimize power loss in the network.

2. The power injection constraints are as follows. For each bus $i \in \mathcal{N}^+$, there may be multiple devices including loads, capacitors, and PV panels. Assume that there is a total of D_i such

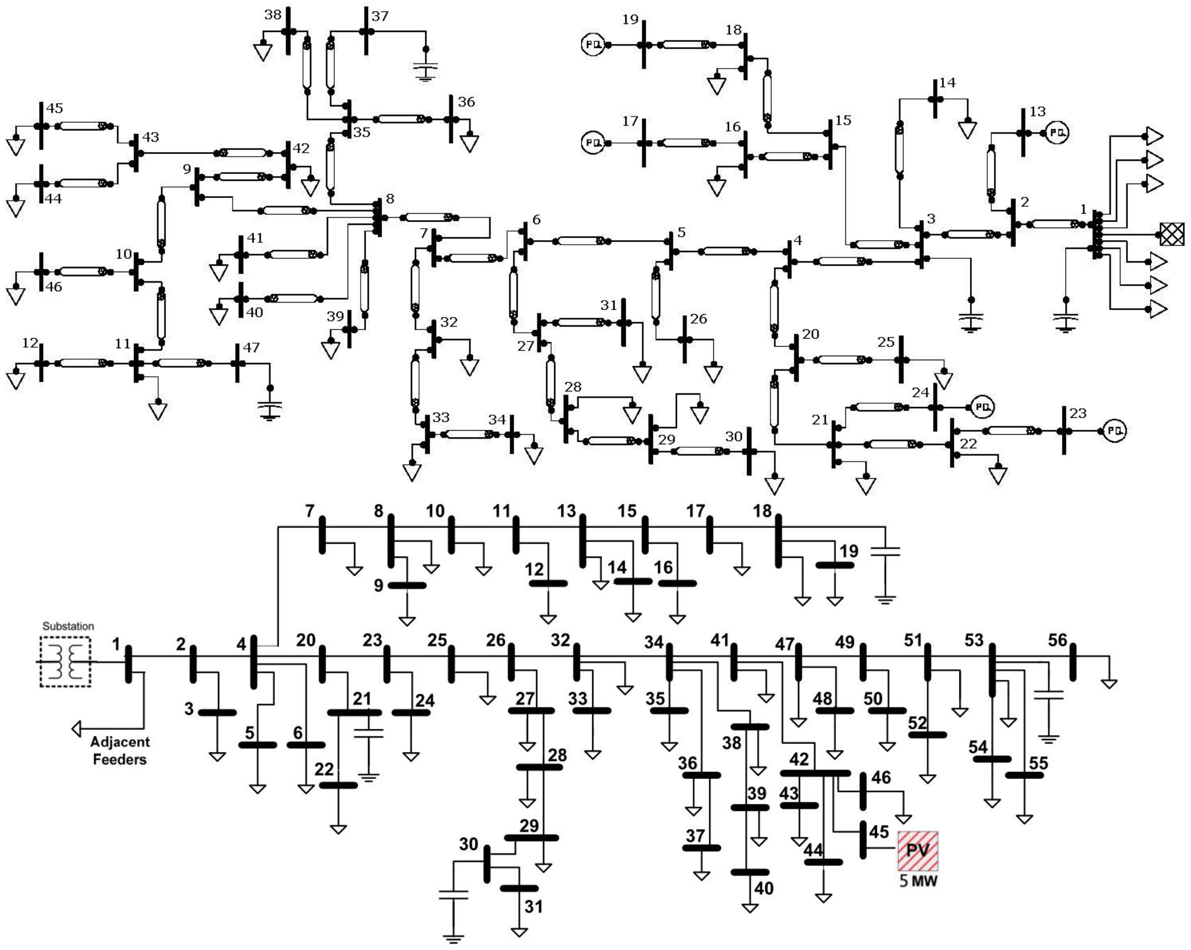


Figure 5.1: Topologies of the SCE 47-bus and 56-bus networks [35, 37].

devices and label them by $1, 2, \ldots, D_i$. Let $s_{i,d} = p_{i,d} + \mathbf{i}q_{i,d}$ denote the power injection of device d for $d = 1, 2, \ldots, D_i$. If device d is a load with given real and reactive power consumptions p and q, then we impose

$$s_{i,d} = -p - \mathbf{i}q. \tag{5.17}$$

If device d is a load with given peak apparent power s_{peak}, then we impose

$$s_{i,d} = -s_{\text{peak}} \exp(j\theta) \tag{5.18}$$

where $\theta = \cos^{-1}(0.9)$, i.e, power injection $s_{i,d}$ is considered to be a constant, obtained by assuming a power factor of 0.9 at peak apparent power. If device d is a capacitor with nameplate $\overline{q}$, then we impose

$$\text{Re}(s_{i,d}) = 0 \text{ and } 0 \leq \text{Im}(s_{i,d}) \leq \overline{q}. \tag{5.19}$$

Table 5.1: Line impedances, peak spot load, and nameplate ratings of capacitors and PV generators of the 47-bus network.

Network Data																	
Line Data				Line Data				Line Data				Load Data		Load Data		PV Generators	
From Bus	To Bus	R (Ω)	X (Ω)	From Bus	To Bus	R (Ω)	X (Ω)	From Bus	To Bus	R (Ω)	X (Ω)	Bus No	Peak MVA	Bus No	Peak MVA	Bus No	Nameplate Capacity
1	2	0.259	0.808	8	41	0.107	0.031	21	22	0.198	0.046	1	30	34	0.2		
2	13	0	0	8	35	0.076	0.015	22	23	0	0	11	0.67	36	0.27	13	1.5MW
2	3	0.031	0.092	8	9	0.031	0.031	27	31	0.046	0.015	12	0.45	38	0.45	17	0.4MW
3	4	0.046	0.092	9	10	0.015	0.015	27	28	0.107	0.031	14	0.89	39	1.34	19	1.5 MW
3	14	0.092	0.031	9	42	0.153	0.046	28	29	0.107	0.031	16	0.07	40	0.13	23	1 MW
3	15	0.214	0.046	10	11	0.107	0.076	29	30	0.061	0.015	18	0.67	41	0.67	24	2 MW
4	20	0.336	0.061	10	46	0.229	0.122	32	33	0.046	0.015	21	0.45	42	0.13		
4	5	0.107	0.183	11	47	0.031	0.015	33	34	0.031	0.010	22	2.23	44	0.45	Shunt Capacitors	
5	26	0.061	0.015	11	12	0.076	0.046	35	36	0.076	0.015	25	0.45	45	0.2	Bus	Nameplate
5	6	0.015	0.031	15	18	0.046	0.015	35	37	0.076	0.046	26	0.2	46	0.45	No.	Capacity
6	27	0.168	0.061	15	16	0.107	0.015	35	38	0.107	0.015	28	0.13				
6	7	0.031	0.046	16	17	0	0	42	43	0.061	0.015	29	0.13	$V_{\text{base}} = 12\text{kV}$		1	6.0 MVAR
7	32	0.076	0.015	18	19	0	0	43	44	0.061	0.015	30	0.2	$S_{\text{base}} = 1\text{MVA}$		3	1.2MVAR
7	8	0.015	0.015	20	21	0.122	0.092	43	45	0.061	0.015	31	0.07	$V_{\text{sub}} = 12.35\text{kV}$		37	1.8MVAR
8	40	0.046	0.015	20	25	0.214	0.046					32	0.13			47	1.8MVAR
8	39	0.244	0.046	21	24	0	0					33	0.27				

Table 5.2: Line impedances, peak spot load, and nameplate ratings of capacitors and PV generators of the 56-bus network.

Network Data																			
Line Data				Line Data				Line Data				Load Data		Load Data		Load Data	
From Bus.	To Bus.	R (Ω)	X (Ω)	From Bus.	To Bus.	R (Ω)	X (Ω)	From Bus.	To Bus.	R (Ω)	X (Ω)	Bus No.	Peak MVA	Bus No.	Peak MVA	Bus No.	Peak MVA
1	2	0.160	0.388	20	21	0.251	0.096	39	40	2.349	0.964	3	0.057	29	0.044	52	0.315
2	3	0.824	0.315	21	22	1.818	0.695	34	41	0.115	0.278	5	0.121	31	0.053	54	0.061
2	4	0.144	0.349	20	23	0.225	0.542	41	42	0.159	0.384	6	0.049	32	0.223	55	0.055
4	5	1.026	0.421	23	24	0.127	0.028	42	43	0.934	0.383	7	0.053	33	0.123	56	0.130
4	6	0.741	0.466	23	25	0.284	0.687	42	44	0.506	0.163	8	0.047	34	0.067	Shunt Cap	
4	7	0.528	0.468	25	26	0.171	0.414	42	45	0.095	0.195	9	0.068	35	0.094	Bus	Mvar
7	8	0.358	0.314	26	27	0.414	0.386	42	46	1.915	0.769	10	0.048	36	0.097	19	0.6
8	9	2.032	0.798	27	28	0.210	0.196	41	47	0.157	0.379	11	0.067	37	0.281	21	0.6
8	10	0.502	0.441	28	29	0.395	0.369	47	48	1.641	0.670	12	0.094	38	0.117	30	0.6
10	11	0.372	0.327	29	30	0.248	0.232	47	49	0.081	0.196	14	0.057	39	0.131	53	0.6
11	12	1.431	0.999	30	31	0.279	0.260	49	50	1.727	0.709	16	0.053	40	0.030	Photovoltaic	
11	13	0.429	0.377	26	32	0.205	0.495	49	51	0.112	0.270	17	0.057	41	0.046	Bus	Capacity
13	14	0.671	0.257	32	33	0.263	0.073	51	52	0.674	0.275	18	0.112	42	0.054		
13	15	0.457	0.401	32	34	0.071	0.171	51	53	0.070	0.170	19	0.087	43	0.083	45	5MW
15	16	1.008	0.385	34	35	0.625	0.273	53	54	2.041	0.780	22	0.063	44	0.057		
15	17	0.153	0.134	34	36	0.510	0.209	53	55	0.813	0.334	24	0.135	46	0.134	$V_{\text{base}} = 12\text{kV}$	
17	18	0.971	0.722	36	37	2.018	0.829	53	56	0.141	0.340	25	0.100	47	0.045	$S_{\text{base}} = 1\text{MVA}$	
18	19	1.885	0.721	34	38	1.062	0.406					27	0.048	48	0.196	$Z_{\text{base}} = 144\Omega$	
4	20	0.138	0.334	38	39	0.610	0.238					28	0.038	50	0.045		

If device d is a PV panel with nameplate $\overline{s}$ and real power generation p_i, then we impose

$$\text{Re}(s_{i,d}) = p_i \text{ and } |s_{i,d}| \leq \overline{s}. \tag{5.20}$$

The power injection at bus i is

$$s_i = \sum_{d=1}^{D_i} s_{i,d}$$

where $s_{i,d}$ satisfies one of (5.17)–(5.20).

3. The voltage regulation constraint is considered to be $0.95^2 \leq v_i \leq 1.05^2$ for $i \in \mathcal{N}^+$.

5.5.3 Results

Numerical results are summarized in Table 5.3. It can be seen that Algorithm 7 obtains 70× speed up over using the generic convex program solver CVX/sedumi, at the cost of a suboptimality gap within numerical precision, for large-scale networks. Note that Algorithm 7 is run in series rather than in parallel due to the limitation of our simulation platform. The speed up can be even more significant once parallel implementation is completed.

Table 5.3: Objective values and CPU times of CVX and IPM

# bus	CVX		IPM		error	speedup
	obj	time(s)	obj	time(s)		
42	10.4585	6.5267	10.4585	0.2679	-0.0e-7	24.36
56	34.8989	7.1077	34.8989	0.3924	+0.2e-7	18.11
111	0.0751	11.3793	0.0751	0.8529	+5.4e-6	13.34
190	0.1394	20.2745	0.1394	1.9968	+3.3e-6	10.15
290	0.2817	23.8817	0.2817	4.3564	+1.1e-7	5.48
390	0.4292	29.8620	0.4292	2.9405	+5.4e-7	10.16
490	0.5526	36.3591	0.5526	3.0072	+2.9e-7	12.09
590	0.7035	43.6932	0.7035	4.4655	+2.4e-7	9.78
690	0.8546	51.9830	0.8546	3.2247	+0.7e-7	16.12
790	0.9975	62.3654	0.9975	2.6228	+0.7e-7	23.78
890	1.1685	67.7256	1.1685	2.0507	+0.8e-7	33.03
990	1.3930	74.8522	1.3930	2.7747	+1.0e-7	26.98
1091	1.5869	83.2236	1.5869	1.0869	+1.2e-7	76.57
1190	1.8123	92.4484	1.8123	1.2121	+1.4e-7	76.27
1290	2.0134	101.0380	2.0134	1.3525	+1.6e-7	74.70
1390	2.2007	111.0839	2.2007	1.4883	+1.7e-7	74.64
1490	2.4523	122.1819	2.4523	1.6372	+1.9e-7	74.83
1590	2.6477	157.8238	2.6477	1.8021	+2.0e-7	87.58
1690	2.8441	147.6862	2.8441	1.9166	+2.1e-7	77.06
1790	3.0495	152.6081	3.0495	2.0603	+2.1e-7	74.07
1890	3.8555	160.4689	3.8555	2.1963	+1.9e-7	73.06
1990	4.1424	171.8137	4.1424	2.3586	+1.9e-7	72.84

5.6 Conclusions

A gradient projection type algorithm, Algorithm 7, has been derived for solving OPF. Algorithm 7 has a distributed implementation, and its suboptimality bound has been derived. Simulation results indicate that a serial implementation of Algorithm 7 obtains 70× speed up over the convex relaxation approach, at essentially no loss of optimality, for a number of real-world test networks.

Chapter 6

Exact Convex Relaxation for Single-Phase Radial Networks

We study sufficient conditions for a convex relaxation of the OPF problem to be exact in this chapter. In particular, we focus on single-phase radial networks, and prove that a second-order cone program is exact under a mild condition after shrinking the OPF feasible set slightly. The condition can be checked a priori, and holds for the IEEE 13, 34, 37, 123-bus networks and two real-world networks.

Literature Solving OPF through semidefinite relaxation is first proposed in [57] as a second-order cone program (SOCP) for radial networks and in [10] as a semidefinite program (SDP) for general networks in a bus injection model. It is first proposed in [35, 36] as an SOCP for radial networks in the branch flow model of [13, 14]. While these convex relaxations have been illustrated numerically in [57] and [10], whether or when they are exact is first studied in [66] (i.e., when an optimal solution of the original OPF problem can be recovered from every optimal solution of an SDP relaxation). Exploiting graph sparsity to simplify the SDP relaxation of OPF is first proposed in [9, 58] and analyzed in [17, 79]. These relaxations are equivalent for radial networks in the sense that there is a bijective map between their feasible sets [17]. The SOCP relaxation, however, has a much lower computational complexity. We will hence focus on the SOCP relaxation in this paper.

Convex relaxations may not be exact [19, 69, 80]. For radial networks, three types of sufficient conditions have been developed in the literature that guarantee their exactness. They are not necessary in general but have implications on allowable power injections, voltage magnitudes, or voltage angles:

1. *Power injections:* These conditions require that not both constraints on real and reactive power injections be binding at both ends of a line [16, 35, 36, 91, 109].

2. *Voltage angles:* These conditions require that the voltage angles across each line be sufficiently close [65, 68]. This is needed also for stability reasons.

3. *Voltages magnitudes:* These conditions require that the upper bounds on voltage magnitudes not be binding [41,42,71]. They can be enforced through affine constraints on power injections. This paper generalizes these results.

Summary The goal of this chapter is to show that after modifying the OPF problem for radial networks slightly, the corresponding SOCP relaxation is exact under a mild condition that can be checked a priori. In particular, contributions of this chapter are threefold.

First, we prove in Theorem 6.2 that *if voltage upper bounds do not bind at optimality, then the SOCP relaxation is exact under a mild condition.* The condition can be checked a priori and holds for the IEEE 13, 34, 37, 123-bus networks and two real-world networks. The condition has a physical interpretation that all upstream reverse power flows increase if the power loss on a line is reduced.

Second, in Section 6.3 we *modify the OPF problem by limiting power injections to a region where voltage upper bounds do not bind* so that the SOCP relaxation is exact under the aforementioned condition. We illustrate that this only eliminates power injections from the original feasible set that are close to voltage upper bounds. Examples exist where the SOCP relaxation is not exact without this modification.

Third, we prove in Theorem 6.8 that *the result in this paper unifies and generalizes the results in [41, 42].*

The rest of this chapter is organized as follows. The OPF problem and the SOCP relaxation are introduced in Section 6.1, and a sufficient condition for exactness is provided in Section 6.2. The condition consists of two parts, C1 and C2. Since C2 cannot be checked a priori, we propose in Section 6.3 a modified OPF problem that always satisfies C2 and therefore its SOCP relaxation is exact under C1. We compare C1 with prior works in Section 6.4 and show in Section 6.5 that C1 holds with large margin for a number of test networks.

6.1 The optimal power flow problem

6.1.1 Power flow model

A distribution network is composed of buses and lines connecting these buses, and is usually radial. The root of the network is a *substation bus* that connects to the transmission network. It has a fixed voltage and redistributes the bulk power it receives from the transmission network to other buses. Index the substation bus by 0 and the other buses by $1, \ldots, n$. Let $\mathcal{N} := \{0, \ldots, n\}$ denote the collection of all buses and define $\mathcal{N}^+ := \mathcal{N} \backslash \{0\}$. Each line connects an ordered pair (i, j) of buses where bus j lies on the unique path from bus i to bus 0. Let $\mathcal{E}$ denote the collection of all lines, and abbreviate $(i, j) \in \mathcal{E}$ by $i \to j$ whenever convenient. Note that the orientation of lines in this chapter is opposite to the orientation adopted in Chapter 4 and 5, to ease the proofs of theorems.

For each bus $i \in \mathcal{N}$, let v_i denote the square of the magnitude of its complex voltage, e.g., if the voltage is $1.05\angle 120^\circ$ per unit, then $v_i = 1.05^2$. The substation voltage v_0 is fixed and given. Let $s_i = p_i + \mathbf{i}q_i$ denote the power injection of bus i where p_i and q_i denote the real and reactive power injections, respectively. Let $\mathcal{P}_i$ denote the unique path from bus i to bus 0. Since the network is radial, the path $\mathcal{P}_i$ is well-defined. For each line $(i,j) \in \mathcal{E}$, let $z_{ij} = r_{ij} + \mathbf{i}x_{ij}$ denote its impedance. Let ℓ_{ij} denote the square of the magnitude of the complex current from bus i to bus j, e.g., if the current is $0.5\angle 10^\circ$, then $\ell_{ij} = 0.5^2$. Let $S_{ij} = P_{ij} + \mathbf{i}Q_{ij}$ denote the sending-end power flow from bus i to bus j where P_{ij} and Q_{ij} denote the real and reactive power flow, respectively. Some of the notations are summarized in Fig. 6.1. We use a letter without subscripts to denote a vector of the corresponding quantities, e.g., $v = (v_i)_{i\in\mathcal{N}^+}$, $\ell = (\ell_{ij})_{(i,j)\in\mathcal{E}}$. Note that subscript 0 is not included in nodal quantities such as v and s. For a complex number $a \in \mathbb{C}$, let $\bar{a}$ denote the conjugate of a.

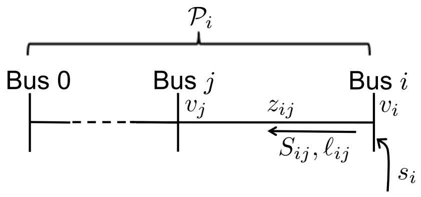


Figure 6.1: Some of the notations.

Given the network $(\mathcal{N}, \mathcal{E})$, the impedance z, and the substation voltage v_0, the other variables (s, S, v, ℓ, s_0) are described by the *branch flow model* for radial networks [13, 14]:

$$S_{ij} = s_i + \sum_{h:\, h\to i} (S_{hi} - z_{hi}\ell_{hi}), \qquad \forall (i,j) \in \mathcal{E}; \tag{6.1a}$$

$$0 = s_0 + \sum_{h:\, h\to 0} (S_{h0} - z_{h0}\ell_{h0}); \tag{6.1b}$$

$$v_i - v_j = 2\mathrm{Re}(\bar{z}_{ij}S_{ij}) - |z_{ij}|^2\ell_{ij}, \qquad \forall (i,j) \in \mathcal{E}; \tag{6.1c}$$

$$\ell_{ij} = \frac{|S_{ij}|^2}{v_i}, \qquad \forall (i,j) \in \mathcal{E}. \tag{6.1d}$$

6.1.2 The OPF problem

We consider the following controllable devices in a distribution network: distributed generators, inverters, controllable loads such as electric vehicles and smart appliances, and shunt capacitors. For application examples, in volt/var control, reactive power injection of inverters and shunt capacitors are controlled to regulate voltages; in demand response, real power consumption of controllable loads is reduced or shifted. Mathematically, power injection s is the control variable, and the other variables (S, v, ℓ, s_0) are determined by the power flow laws in (6.1) once s is specified.

The power injection s_i of a bus $i \in \mathcal{N}^+$ is constrained to be in a pre-specified set $\mathcal{S}_i$, i.e.,

$$s_i \in \mathcal{S}_i, \qquad i \in \mathcal{N}^+. \tag{6.2}$$

The set $\mathcal{S}_i$ for some controllable devices are:

- If s_i represents a shunt capacitor with nameplate capacity $\overline{q}_i > 0$, then

  $$\mathcal{S}_i = \{s \in \mathbb{C} \mid \mathrm{Re}(s) = 0,\ \mathrm{Im}(s) = 0 \text{ or } \overline{q}_i\}.$$

  Note that $\mathcal{S}_i$ is nonconvex and disconnected in this case.

- If s_i represents a solar panel with generation capacity $\overline{p}_i$, that is connected to the grid through an inverter with nameplate capacity $\overline{s}_i$, then

  $$\mathcal{S}_i = \{s \in \mathbb{C} \mid 0 \le \mathrm{Re}(s) \le \overline{p}_i,\ |s| \le \overline{s}_i\}.$$

- If s_i represents a controllable load with constant power factor η, whose real power consumption can vary continuously from $-\overline{p}_i$ to $-\underline{p}_i$ (here $\underline{p}_i \le \overline{p}_i \le 0$), then

  $$\mathcal{S}_i = \left\{s \in \mathbb{C} \mid \underline{p}_i \le \mathrm{Re}(s) \le \overline{p}_i,\ \mathrm{Im}(s) = \sqrt{1-\eta^2}\mathrm{Re}(s)/\eta\right\}.$$

Note that s_i can represent the aggregate power injection of multiple such devices with an appropriate $\mathcal{S}_i$, and that the set $\mathcal{S}_i$ is not necessarily convex or connected.

An important goal of control is to regulate the voltages to lie within pre-specified lower and upper bounds $\underline{v}_i$ and $\overline{v}_i$, i.e.,

$$\underline{v}_i \le v_i \le \overline{v}_i, \qquad i \in \mathcal{N}^+. \tag{6.3}$$

For example, if voltages must not deviate by more than 5% from their nominal values, then $0.95^2 \le v_i \le 1.05^2$ per unit. We consider the control objective

$$C(s, s_0) = \sum_{i \in \mathcal{N}} f_i(\mathrm{Re}(s_i)) \tag{6.4}$$

where $f_i : \mathbb{R} \to \mathbb{R}$ denotes the generation cost at bus i for $i \in \mathcal{N}$. If $f_i(x) = x$ for $i \in \mathcal{N}$, then C is the total power loss on the network.

The OPF problem seeks to minimize the generation cost (6.4), subject to power flow constraints

(6.1), power injection constraints (6.2), and voltage constraints (6.3):

$$
\begin{aligned}
\textbf{OPF:}\ \min\ & \sum_{i\in\mathcal{N}} f_i(\mathrm{Re}(s_i)) \\
\text{over}\ & s, S, v, \ell, s_0 \\
\text{s.t.}\ & S_{ij} = s_i + \sum_{h:\, h\to i} (S_{hi} - z_{hi}\ell_{hi}), \quad \forall (i,j)\in\mathcal{E}; && \text{(6.5a)} \\
& 0 = s_0 + \sum_{h:\, h\to 0} (S_{h0} - z_{h0}\ell_{h0}); && \text{(6.5b)} \\
& v_i - v_j = 2\mathrm{Re}(\bar{z}_{ij}S_{ij}) - |z_{ij}|^2\ell_{ij}, \quad \forall (i,j)\in\mathcal{E}; && \text{(6.5c)} \\
& \ell_{ij} = \frac{|S_{ij}|^2}{v_i}, \quad \forall (i,j)\in\mathcal{E}; && \text{(6.5d)} \\
& s_i \in \mathcal{S}_i, \quad i\in\mathcal{N}^+; && \text{(6.5e)} \\
& \underline{v}_i \le v_i \le \overline{v}_i, \quad i\in\mathcal{N}^+. && \text{(6.5f)}
\end{aligned}
$$

The following assumptions are made throughout this paper.

A1 The network $(\mathcal{N},\mathcal{E})$ is a tree. Distribution networks are usually radial.

A2 The substation voltage v_0 is fixed and given. In practice, v_0 can be modified several times a day, and therefore can be considered as a given constant at the timescale of OPF.

A3 Line resistances and reactances are strictly positive, i.e., $r_{ij} > 0$ and $x_{ij} > 0$ for $(i,j)\in\mathcal{E}$. This holds in practice because lines are passive (consume power) and inductive.

A4 Voltage lower bounds are strictly positive, i.e., $\underline{v}_i > 0$ for $i\in\mathcal{N}^+$. In practice, $\underline{v}_i$ is slightly below 1 per unit.

The equality constraint (6.5d) is nonconvex, and one can relax it to inequality constraints to obtain the following second-order cone programming (SOCP) relaxation [36]:

$$
\begin{aligned}
\textbf{SOCP:}\ \min\ & \sum_{i\in\mathcal{N}} f_i(\mathrm{Re}(s_i)) \\
\text{over}\ & s, S, v, \ell, s_0 \\
\text{s.t.}\ & \text{(6.5a)} - \text{(6.5c)},\ \text{(6.5e)} - \text{(6.5f)}; \\
& \ell_{ij} \ge \frac{|S_{ij}|^2}{v_i}, \quad \forall (i,j)\in\mathcal{E}. && \text{(6.6)}
\end{aligned}
$$

Note that SOCP is not necessarily convex, since we allow f_i to be nonconvex and $\mathcal{S}_i$ to be nonconvex. Nonetheless, we call it SOCP for brevity.

If an optimal SOCP solution $w = (s, S, v, \ell, s_0)$ is feasible for OPF, i.e., w satisfies (6.5d), then w is a global optimum of OPF. This motivates the following definition.

Definition 6.1. *SOCP is exact if every of its optimal solutions satisfies* (6.5d).

6.2 A sufficient condition

We now provide a sufficient condition that ensures SOCP is exact. It motivates a modified OPF problem in Section 6.3.

6.2.1 Statement of the condition

We start with introducing the notations that will be used in the statement of the condition. One can ignore the ℓ terms in (6.1a) and (6.1c) to obtain the *Linear DistFlow Model* [13, 14]:

$$\begin{aligned} S_{ij} &= s_i + \sum_{h:\, h\to i} S_{hi}, && \forall (i,j)\in\mathcal{E}; \\ v_i - v_j &= 2\mathrm{Re}(\bar{z}_{ij}S_{ij}), && \forall (i,j)\in\mathcal{E}. \end{aligned}$$

Let $(\hat{S},\hat{v})$ denote the solution of the Linear DistFlow model, then

$$\begin{aligned} \hat{S}_{ij}(s) &= \sum_{h:\, i\in\mathcal{P}_h} s_h, && \forall (i,j)\in\mathcal{E}; \\ \hat{v}_i(s) &:= v_0 + 2\sum_{(j,k)\in\mathcal{P}_i} \mathrm{Re}\left(\bar{z}_{jk}\hat{S}_{jk}(s)\right), && \forall i\in\mathcal{N} \end{aligned}$$

as in Fig. 6.1. Physically, $\hat{S}_{ij}(s)$ denotes the sum of power injections s_h towards bus 0 that go through line (i,j). Note that $(\hat{S}(s),\hat{v}(s))$ is affine in s, and equals (S,v) if and only if line loss $z_{ij}\ell_{ij}$ is 0 for $(i,j)\in\mathcal{E}$. For two complex numbers $a,b\in\mathbb{C}$, let $a\le b$ denote $\mathrm{Re}(a)\le\mathrm{Re}(b)$

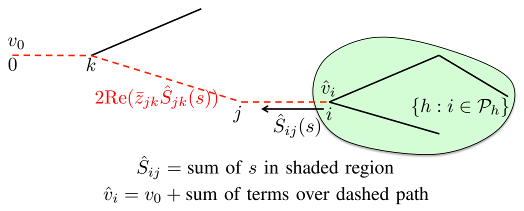

v_0
0
k
$2\mathrm{Re}(\bar{z}_{jk}\hat{S}_{jk}(s))$
j
$\hat{v}_i$
i
$\hat{S}_{ij}(s)$
$\{h : i\in\mathcal{P}_h\}$
$\hat{S}_{ij}$ = sum of s in shaded region
$\hat{v}_i = v_0$ + sum of terms over dashed path

Figure 6.1: Illustration of $\hat{S}_{ij}$ and $\hat{v}_i$. The shaded region is downstream of bus i, and contains the buses $\{h : i\in\mathcal{P}_h\}$. Quantity $\hat{S}_{ij}(s)$ is defined to be the sum of bus injections in the shaded region. The dashed lines constitute the path $\mathcal{P}_i$ from bus i to bus 0. Quantity $\hat{v}_i(s)$ is defined as v_0 plus the terms $2\mathrm{Re}(\bar{z}_{jk}\hat{S}_{jk}(s))$ over the dashed path.

and $\mathrm{Im}(a)\le\mathrm{Im}(b)$. For two vectors a,b of the same dimension, let $a\le b$ denote componentwise inequality. Define $<$, $>$, and $\ge$ similarly.

Lemma 6.1. *If* (s,S,v,ℓ,s_0) *satisfies* (6.1a)–(6.1c) *and* $\ell\ge 0$ *componentwise, then* $S\le\hat{S}(s)$ *and* $v\le\hat{v}(s)$.

Lemma 6.1 implies that $\hat{v}(s)$ and $\hat{S}(s)$ provide upper bounds on v and S. It is proved in Appendix 6.A. Let $\hat{P}(s)$ and $\hat{Q}(s)$ denote the real and imaginary parts of $\hat{S}(s)$, respectively. Then

$$\hat{P}_{ij}(s = p + \mathbf{i}q) = \hat{P}_{ij}(p) = \sum_{h:\, i \in \mathcal{P}_h} p_h, \qquad (i,j) \in \mathcal{E};$$
$$\hat{Q}_{ij}(s = p + \mathbf{i}q) = \hat{Q}_{ij}(q) = \sum_{h:\, i \in \mathcal{P}_h} q_h, \qquad (i,j) \in \mathcal{E}.$$

Assume that there exist $\overline{p}_i$ and $\overline{q}_i$ such that

$$\mathcal{S}_i \subseteq \{s \in \mathbb{C} \mid \text{Re}(s) \le \overline{p}_i,\ \text{Im}(s) \le \overline{q}_i\}$$

for $i \in \mathcal{N}^+$ as in Fig. 6.2, i.e., Re(s_i) and Im(s_i) are upper bounded by $\overline{p}_i$ and $\overline{q}_i$, respectively. Define $a^+ := \max\{a, 0\}$ for $a \in \mathbb{R}$. Let $I := \text{diag}(1,1)$ denote the 2×2 identity matrix, and define

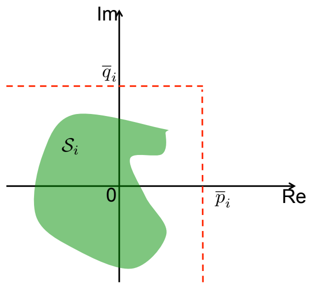


Figure 6.2: We assume that $\mathcal{S}_i$ lies in the left bottom corner of $(\overline{p}_i, \overline{q}_i)$, but do not assume that $\mathcal{S}_i$ is convex or connected.

$$u_{ij} := \begin{pmatrix} r_{ij} \\ x_{ij} \end{pmatrix}, \qquad \underline{A}_{ij} := I - \frac{2}{\underline{v}_i} \begin{pmatrix} r_{ij} \\ x_{ij} \end{pmatrix} \begin{pmatrix} \hat{P}^+_{ij}(\overline{p}) & \hat{Q}^+_{ij}(\overline{q}) \end{pmatrix}$$

for $(i,j) \in \mathcal{E}$. For each $i \in \mathcal{N}^+$, $(i,j_1) \in \mathcal{E}$ and $(i,j_2) \in \mathcal{E}$ implies $j_1 = j_2$, and therefore we can abbreviate u_{ij} and $\underline{A}_{ij}$ by u_i and $\underline{A}_i$, respectively, without ambiguity.

Further, let $\mathcal{L} := \{l \in \mathcal{N} \mid \nexists k \in \mathcal{N} \text{ such that } k \to l\}$ denote the collection of leaf buses in the network. For a leaf bus $l \in \mathcal{L}$, let $n_l + 1$ denote the number of buses on path $\mathcal{P}_l$, and suppose

$$\mathcal{P}_l = \{l_{n_l} \to l_{n_l - 1} \to \ldots \to l_1 \to l_0\}$$

with $l_{n_l} = l$ and $l_0 = 0$ as in Fig. 6.3. Let

$$\mathcal{S}_{\text{volt}} := \{s \in \mathbb{C}^n \mid \hat{v}_i(s) \le \overline{v}_i \text{ for } i \in \mathcal{N}^+\}$$

denote the power injection region where $\hat{v}(s)$ is upper bounded by $\overline{v}$. Since $v \le \hat{v}(s)$ (Lemma 6.1),

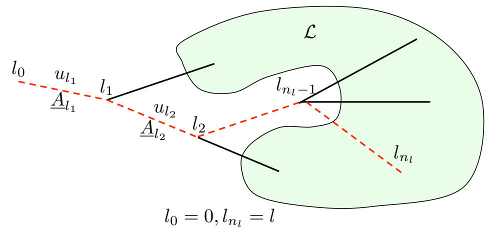


Figure 6.3: The shaded region denotes the collection $\mathcal{L}$ of leaf buses, and the path $\mathcal{P}_l$ of a leaf bus $l \in \mathcal{L}$ is illustrated by a dashed line.

the set $\mathcal{S}_{\text{volt}}$ is a power injection region where voltage upper bounds do not bind.

The following theorem provides a sufficient condition that guarantees the exactness of SOCP.

Theorem 6.2. *Assume that f_0 is strictly increasing, and that there exist $\overline{p}_i$ and $\overline{q}_i$ such that $\mathcal{S}_i \subseteq \{s \in \mathbb{C} \mid \text{Re}(s) \le \overline{p}_i,\ \text{Im}(s) \le \overline{q}_i\}$ for $i \in \mathcal{N}^+$. Then SOCP is exact if the following conditions hold:*

C1 $\underline{A}_{l_s}\underline{A}_{l_{s+1}} \cdots \underline{A}_{l_{t-1}} u_{l_t} > 0$ for any $l \in \mathcal{L}$ and any s, t such that $1 \le s \le t \le n_l$;

C2 every optimal SOCP solution $w = (s, S, v, \ell, s_0)$ satisfies $s \in \mathcal{S}_{\text{volt}}$.

Theorem 6.2 implies that if C2 holds, i.e., optimal power injections lie in the region $\mathcal{S}_{\text{volt}}$ where voltage upper bounds do not bind, then SOCP is exact under C1. C2 depends on SOCP solutions and cannot be checked a priori. This drawback motivates us to modify OPF such that C2 always holds and therefore the corresponding SOCP is exact under C1, as will be discussed in Section 6.3.

We illustrate the proof idea of Theorem 6.2 via a 3-bus linear network in Fig. 6.4. The proof

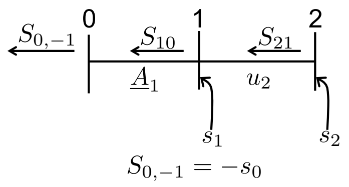


Figure 6.4: A 3-bus linear network.

for general radial networks is provided in Appendix 6.B. Assume C1 and C2 hold. If SOCP is not exact, then there exists an optimal SOCP solution $w = (s, S, v, \ell, s_0)$ that violates (6.5d). We will construct another feasible point $w' = (s', S', v', \ell', s_0')$ of SOCP that has a smaller objective value than w, contradicting the optimality of w and implying SOCP is exact.

There are two ways (6.5d) gets violated: 1) (6.5d) is violated on line $(1, 0)$; or 2) (6.5d) is satisfied on line $(1, 0)$ but violated on line $(2, 1)$. To illustrate the proof idea, we focus on the second case, i.e., the case where $\ell_{10} = |S_{10}|^2/v_1$ and $\ell_{21} > |S_{21}|^2/v_2$. In this case, the construction of w' is

$$\begin{aligned}
\textbf{Initialization:} \quad & s' = s,\ S'_{21} = S_{21};\\
\textbf{Forward sweep:} \quad & \ell'_{21} = |S'_{21}|^2/v_2,\\
& S'_{10} = S'_{21} - z_{21}\ell'_{21} + s'_1;\\
& \ell'_{10} = |S'_{10}|^2/v_1,\\
& S'_{0,-1} = S'_{10} - z_{10}\ell'_{10};\\
\textbf{Backward sweep:} \quad & v'_1 = v_0 + 2\text{Re}(\bar{z}_{10}S'_{10}) - |z_{10}|^2\ell'_{10};\\
& v'_2 = v'_1 + 2\text{Re}(\bar{z}_{21}S'_{21}) - |z_{21}|^2\ell'_{21}
\end{aligned}$$

where $S'_{0,-1} = -s'_0$. The construction consists of three steps:

S1 In the initialization step, s' and S'_{21} are initialized as the corresponding values in w.

S2 In the forward sweep step, $\ell'_{k,k-1}$ and $S'_{k-1,k-2}$ are recursively constructed for $k = 2, 1$ by alternatively applying (6.5d) (with v' replaced by v) and (6.5a)/(6.5b). This recursive construction updates ℓ' and S' alternatively along the path $\mathcal{P}_2$ from bus 2 to bus 0, and is therefore called a *forward sweep*.

S3 In the backward sweep step, v'_k is recursively constructed for $k = 1, 2$ by applying (6.5c). This recursive construction updates v' along the negative direction of $\mathcal{P}_2$ from bus 0 to bus 2, and is therefore called a *backward sweep*.

One can show that w' is feasible for SOCP and has a smaller objective value than w. This contradicts the optimality of w, and therefore SOCP is exact.

Remark 6.1. *Theorem 6.2 still holds if there is an additional power injection constraint $s \in \mathcal{S}$ in OPF, where $\mathcal{S}$ can be an arbitrary set. This is because we set $s' = s$ in the construction of w', and therefore $s \in \mathcal{S}$ implies $s' \in \mathcal{S}$. Hence, an additional constraint $s \in \mathcal{S}$ does not affect the fact that w' is feasible for SOCP and has a smaller objective value than w.*

6.2.2 Interpretation of C1

We illustrate C1 through a linear network as in Figure 6.5. The collection of leaf buses is a singleton $\mathcal{L} = \{n\}$, and the path from the only leaf bus n to bus 0 is $\mathcal{P}_n = \{n \to n-1 \to \cdots \to 1 \to 0\}$. Then, C1 takes the form

$$\underline{A}_s\underline{A}_{s+1}\cdots\underline{A}_{t-1}u_t > 0, \qquad 1 \le s \le t \le n.$$

That is, given any network segment $(s-1,t)$ where $1 \le s \le t \le n$, the multiplication $\underline{A}_s\underline{A}_{s+1}\cdots\underline{A}_{t-1}$ of $\underline{A}$ over the segment $(s-1,t-1)$ times u_t is strictly positive.

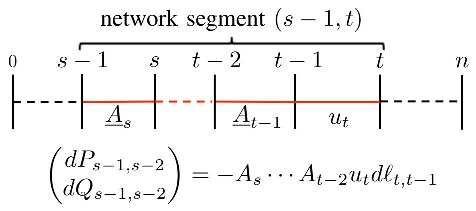

network segment $(s-1,t)$
0
$s-1$
s
$t-2$
$t-1$
t
n
$\underline{A}_s$
$\underline{A}_{t-1}$
u_t
$\begin{pmatrix} dP_{s-1,s-2} \\ dQ_{s-1,s-2} \end{pmatrix} = -A_s \cdots A_{t-2}u_t d\ell_{t,t-1}$

Figure 6.5: A linear network for the interpretation of Condition C1.

C1 only depends on SOCP parameters $(r, x, \overline{p}, \overline{q}, \underline{v})$. It can be checked a priori and efficiently since $\underline{A}$ and u are simple functions of $(r, x, \overline{p}, \overline{q}, \underline{v})$ that can be computed in $O(n)$ time and there are no more than $n(n+1)/2$ inequalities in C1.

Proposition 6.3. *If $(\overline{p}, \overline{q}) \le (\overline{p}', \overline{q}')$ and C1 holds for $(r, x, \overline{p}', \overline{q}', \underline{v})$, then C1 also holds for $(r, x, \overline{p}, \overline{q}, \underline{v})$.*

Proposition 6.3 implies that if C1 holds for a set of power injections, then C1 also holds for smaller power injections. It is proved in Appendix 6.C.

Proposition 6.4. *If $(\overline{p}, \overline{q}) \le 0$, then C1 holds.*

Proposition 6.4 implies that if every bus only consumes real and reactive power, then C1 holds. This is because when $(\overline{p}, \overline{q}) \le 0$, the quantities $\hat{P}_{ij}(\overline{p}) \le 0$, $\hat{Q}_{ij}(\overline{q}) \le 0$ for $(i,j) \in \mathcal{E}$. It follows that $\underline{A}_i = I$ for $i \in \mathcal{N}^+$. Hence, $\underline{A}_{l_s} \cdots \underline{A}_{l_{t-1}} u_{l_t} = u_{l_t} > 0$ for any $l \in \mathcal{L}$ and any s,t such that $1 \le s \le t \le n_l$.

For practical parameter ranges of $(r, x, \overline{p}, \overline{q}, \underline{v})$, line resistance and reactance $r_{ij}, x_{ij} \ll 1$ per unit for $(i,j) \in \mathcal{E}$, line flows $\hat{P}_{ij}(\overline{p}), \hat{Q}_{ij}(\overline{q})$ are on the order of 1 per unit for $(i,j) \in \mathcal{E}$, and voltage lower bound $\underline{v}_i \approx 1$ per unit for $i \in \mathcal{N}^+$. Hence, $\underline{A}_i$ is close to I for $i \in \mathcal{N}^+$, and therefore C1 is likely to hold. As will be seen in Section 6.5, C1 holds for several test networks, including those with big $(\overline{p}, \overline{q})$ (high penetration of distributed generation).

C1 has a physical interpretation. Recall that $S_{k,k-1}$ denotes the reverse power flow on line $(k, k-1)$ for $k = 1, \dots, n$ and introduce $S_{0,-1} := -s_0$ for convenience. If the power loss on a line is reduced, it is natural that all upstream reverse power flows will increase. More specifically, the power loss on line $(t, t-1)$ where $t \in \{1, 2, \dots, n\}$ is reduced if the current $\ell_{t,t-1}$ is reduced by $-d\ell_{t,t-1} > 0$. When power loss gets smaller, reverse power flow $S_{s-1,s-2}$ is likely to increase, i.e., $dS_{s-1,s-2} > 0$, for $s = 1, 2, \dots, t$.

Let $dS_{s-1,s-2} = dP_{s-1,s-2} + \mathbf{i}dQ_{s-1,s-2} > 0$ for $s = 1, \dots, t$. It can be verified that

$$(dP_{t-1,t-2} \;\; dQ_{t-1,t-2})^T = -u_t d\ell_{t,t-1},$$

and one can compute from (6.1) the Jacobian matrix

$$A_k \;:=\; \begin{pmatrix} \frac{\partial P_{k-1,k-2}}{\partial P_{k,k-1}} & \frac{\partial P_{k-1,k-2}}{\partial Q_{k,k-1}} \\ \frac{\partial Q_{k-1,k-2}}{\partial P_{k,k-1}} & \frac{\partial Q_{k-1,k-2}}{\partial Q_{k,k-1}} \end{pmatrix} \;=\; I - \frac{2}{v_k}\begin{pmatrix} r_{k,k-1} \\ x_{k,k-1} \end{pmatrix}(P_{k,k-1} \;\; Q_{k,k-1})$$

for $k = 1, \dots, n$. Therefore

$$(dP_{s-1,s-2} \;\; dQ_{s-1,s-2})^T = -A_s A_{s+1} \cdots A_{t-1} u_t d\ell_{t,t-1}$$

for $s = 1, \dots, t$. Then, $dS_{s-1,s-2} > 0$ implies

$$A_s A_{s+1} \cdots A_{t-1} u_t > 0 \tag{6.7}$$

for $s = 1, 2, \dots, t$. Note that $\underline{A}_k$ is obtained by replacing (P, Q, v) in A_k by $(\hat{P}^+(\overline{p}), \hat{Q}^+(\overline{q}), \underline{v})$ (so that $\underline{A}_k$ only depends on SOCP parameters), and then (6.7) becomes C1.

6.3 A modified OPF problem

The condition C2 in Theorem 6.2 depends on SOCP solutions and cannot be checked a priori. It can, however, be enforced by the additional constraint

$$s \in \mathcal{S}_{\text{volt}} \tag{6.8}$$

on OPF. Condition (6.8) is equivalent to n affine constraints on s, $\hat{v}_i(s) \le \overline{v}_i$ for $i \in \mathcal{N}^+$. Since $v_i \le \hat{v}_i(s)$ (Lemma 6.1), the constraints $v_i \le \overline{v}_i$ in (6.5f) become redundant after imposing (6.8). To summarize, the modified OPF problem is

$$\begin{aligned} \textbf{OPF-m:} \;\; \min \;\; & \sum_{i \in \mathcal{N}} f_i(\text{Re}(s_i)) \\ \text{over} \;\; & s, S, v, \ell, s_0 \\ \text{s.t.} \;\; & (6.5\text{a}) - (6.5\text{e}); \\ & \underline{v}_i \le v_i, \; \hat{v}_i(s) \le \overline{v}_i, \quad i \in \mathcal{N}^+. \end{aligned} \tag{6.9}$$

A modification to OPF is necessary to ensure an exact SOCP, since otherwise examples exist where SOCP is not exact. Remarkably, the feasible sets of OPF-m and OPF are similar since $\hat{v}_i(s)$ is close to v_i in practice [13, 14, 101].

One can relax (6.5d) to (6.6) to obtain the corresponding SOCP relaxation for OPF-m:

$$\begin{aligned} \textbf{SOCP-m:} \ \min \ & \sum_{i\in\mathcal{N}} f_i(\mathrm{Re}(s_i)) \\ \text{over} \ & s, S, v, \ell, s_0 \\ \text{s.t.} \ & (6.5\text{a}) - (6.5\text{c}), \ (6.6), \ (6.5\text{e}), \ (6.9). \end{aligned}$$

Note again that SOCP-m is not necessarily convex, since we allow f_i and $\mathcal{S}_i$ to be nonconvex.

Since OPF-m is obtained by imposing additional constraint (6.8) on OPF, it follows immediately from Remark 6.1 that SOCP-m relaxation is exact under C1—a mild condition that can be checked a priori.

Theorem 6.5. *Assume that f_0 is strictly increasing, and that there exist $\overline{p}_i$ and $\overline{q}_i$ such that $\mathcal{S}_i \subseteq \{s \in \mathbb{C} \mid \mathrm{Re}(s) \le \overline{p}_i, \ \mathrm{Im}(s) \le \overline{q}_i\}$ for $i \in \mathcal{N}^+$. Then SOCP-m is exact if C1 holds.*

The next result implies that SOCP (SOCP-m) has at most one optimal solution if it is convex and exact. The theorem is proved in Appendix 6.D.

Theorem 6.6. *If f_i is convex for $i \in \mathcal{N}$, $\mathcal{S}_i$ is convex for $i \in \mathcal{N}^+$, and SOCP (SOCP-m) is exact, then SOCP (SOCP-m) has at most one optimal solution.*

The proof of Theorem 6.6 also implies that the feasible set of OPF (OPF-m) is hollow, as stated in the following corollary.

Corollary 6.7. *Let $\tilde{x} = (\tilde{s}, \tilde{S}, \tilde{v}, \tilde{\ell}, \tilde{v}_0)$ and $\hat{x} = (\hat{s}, \hat{S}, \hat{v}, \hat{\ell}, \hat{v}_0)$ be two distinct feasible points of OPF (OPF-m), then any convex combination of $\tilde{x}$ and $\hat{x}$ cannot be feasible for OPF (OPF-m), i.e., the point $x = \theta\tilde{x} + (1-\theta)\hat{x}$ is infeasible for OPF (OPF-m) for any $\theta \in (0, 1)$.*

The proof of Corollary 6.7 is similar to that of Theorem 6.6 and omitted for brevity.

6.4 Connection with prior results

Theorem 6.2 unifies and generalizes the results in [41, 42] due to Theorem 6.8 proved in Appendix 6.E. Theorem 6.8 below says that C1 holds if at least one of the followings hold: 1) Every bus only consumes real and reactive power; 2) lines share the same resistance to reactance ratio; 3) The buses only consume real power and the resistance to reactance ratio increases as lines branch out from the substation; 4) The buses only consume reactive power and the resistance to reactance ratio decreases as lines branch out from the substation; 5) upper bounds $\hat{P}^+(\overline{p})$, $\hat{Q}^+(\overline{q})$ on reverse power flows are sufficiently small. Let

$$\mathcal{E}' := \{(i, j) \in \mathcal{E} \mid i \notin \mathcal{L}\}$$

denote the set of all non-leaf lines.

Theorem 6.8. *Assume that there exist $\overline{p}_i$ and $\overline{q}_i$ such that $\mathcal{S}_i \subseteq \{s \in \mathbb{C} \mid \mathrm{Re}(s) \le \overline{p}_i,\ \mathrm{Im}(s) \le \overline{q}_i\}$ for $i \in \mathcal{N}^+$. Then C1 holds if any one of the following statements is true:*

1) $\hat{S}_{ij}(\overline{p} + \boldsymbol{i}\overline{q}) \le 0$ for $(i,j) \in \mathcal{E}'$.

2) r_{ij}/x_{ij} is identical for $(i,j) \in \mathcal{E}$; and $\underline{v}_i - 2r_{ij}\hat{P}_{ij}^+(\overline{p}) - 2x_{ij}\hat{Q}_{ij}^+(\overline{q}) > 0$ for $(i,j) \in \mathcal{E}'$.

3) $r_{ij}/x_{ij} \ge r_{jk}/x_{jk}$ whenever $i \to j, j \to k$; and $\hat{P}_{ij}(\overline{p}) \le 0$, $\underline{v}_i - 2x_{ij}\hat{Q}_{ij}^+(\overline{q}) > 0$ for $(i,j) \in \mathcal{E}'$.

4) $r_{ij}/x_{ij} \le r_{jk}/x_{jk}$ whenever $i \to j, j \to k$; and $\hat{Q}_{ij}(\overline{q}) \le 0$, $\underline{v}_i - 2r_{ij}\hat{P}_{ij}^+(\overline{p}) > 0$ for $(i,j) \in \mathcal{E}'$.

5) $$\begin{bmatrix} \prod\limits_{(k,l)\in\mathcal{P}_j} 1 - \dfrac{2r_{kl}\hat{P}_{kl}^+(\bar{p})}{\underline{v}_k} & -\sum\limits_{(k,l)\in\mathcal{P}_j} \dfrac{2r_{kl}\hat{Q}_{kl}^+(\bar{q})}{\underline{v}_k} \\ -\sum\limits_{(k,l)\in\mathcal{P}_j} \dfrac{2x_{kl}\hat{P}_{kl}^+(\bar{p})}{\underline{v}_k} & \prod\limits_{(k,l)\in\mathcal{P}_j} 1 - \dfrac{2x_{kl}\hat{Q}_{kl}^+(\bar{q})}{\underline{v}_k} \end{bmatrix} \begin{bmatrix} r_{ij} \\ x_{ij} \end{bmatrix} > 0 \text{ for } (i,j) \in \mathcal{E}.$$

The results in [41, 42] say that, if there are no voltage upper bounds, i.e., $\overline{v} = \infty$, then SOCP is exact if any one of 1)–5) holds. Since C2 holds automatically when $\overline{v} = \infty$ and C1 holds if any one of 1)–5) holds (Theorem 6.8), the results in [41, 42] follow from Theorem 6.2. Besides, the following corollary follows immediately from Theorems 6.5 and 6.8.

Corollary 6.9. *Assume that f_0 is strictly increasing, and that there exist $\overline{p}_i$ and $\overline{q}_i$ such that $\mathcal{S}_i \subseteq \{s \in \mathbb{C} \mid \mathrm{Re}(s) \le \overline{p}_i,\ \mathrm{Im}(s) \le \overline{q}_i\}$ for $i \in \mathcal{N}^+$. Then SOCP-m is exact if any one of 1)–5) holds.*

6.5 Case Studies

In this section, we use six test networks to demonstrate that

1. SOCP is simpler computationally than SDP.
2. C1 holds. We define *C1 margin* that quantifies how well C1 is satisfied, and show that the margin is big.
3. The feasible sets of OPF and OPF-m are similar. We define *modification gap* that quantifies the difference between the feasible sets of OPF and OPF-m, and show that this gap is small.

6.5.1 Test networks

The test networks include IEEE 13, 34, 37, 123-bus networks [1] and two real-world networks [35,37] in the service territory of Southern California Edison (SCE), a utility company in California, USA [2].

Table 6.1: DG penetration, C1 margins, modification gaps, and computation times for different test networks.

	DG penetration	numerical precision	SOCP time	SDP time	C1 margin	estimated modification gap
IEEE 13-bus	0%	10^{-10}	0.5162s	0.3842s	27.6762	0.0362
IEEE 34-bus	0%	10^{-10}	0.5772s	0.5157s	20.8747	0.0232
IEEE 37-bus	0%	10^{-9}	0.5663s	1.6790s	$+\infty$	0.0002
IEEE 123-bus	0%	10^{-8}	2.9731s	32.6526s	52.9636	0.0157
SCE 47-bus	56.6%	10^{-8}	0.7265s	2.5932s	2.5416	0.0082
SCE 56-bus	130.4%	10^{-9}	1.0599s	6.0573s	1.2972	0.0053

The IEEE networks are unbalanced three-phase radial networks with some devices (regulators, circuit switches, transformers, and distributed loads) not modeled in (6.1). Therefore we modify the IEEE networks as follows.

1. Assume that each bus has three phases and split its load uniformly among the three phases.

2. Assume that the three phases are decoupled so that the network becomes three identical single phase networks.

3. Model closed circuit switches as shorted lines and ignore open circuit switches. Model regulators as multiplying the voltages by fixed constants (set to 1.08 in the simulations). Model transformers as lines with appropriate impedances. Model the distributed load on a line as two identical spot loads located at two ends of the line.

The SCE networks, a 47-bus network and a 56-bus network, are shown in Fig. 5.1 with parameters given in Tables 5.1 and 5.2.

These networks have increasing penetration of distributed generation (DG) as listed in Table 6.1. While the IEEE networks do not have any DG, the SCE 47-bus network has 56.6% DG penetration (6.4MW nameplate DG capacity against 11.3MVA peak spot load), and the SCE 56-bus network has 130.4% DG penetration (5MW nameplate DG capacity against 3.835MVA peak spot load).

6.5.2 SOCP is more efficient to compute than SDP

We compare the computation times of SOCP and SDP for the test networks, and summarize the results in Table 6.1. All simulations in this paper use matlab 7.9.0.529 (64-bit) with toolbox cvx 1.21 on Mac OS X 10.7.5 with 2.66GHz Intel Core 2 Due CPU and 4GB 1067MHz DDR3 memory.

We use the following OPF setup throughout the simulations.

1. The objective is to minimize power loss in the network.

2. The power injection constraints are as follows. For each bus $i \in \mathcal{N}^+$, there may be multiple devices including loads, capacitors, and PV panels. Assume that there is a total of D_i such devices and label them by $1, 2, \ldots, D_i$. Let $s_{i,d}$ denote the power injection of device d on bus

i for $d = 1, 2, \ldots, D_i$. If device d is a load with given real and reactive power consumptions p and q, then we impose

$$s_{i,d} = -p - \mathbf{i}q. \tag{6.10}$$

If device d is a load with given peak apparent power s_{peak}, then we impose

$$s_{i,d} = -s_{\text{peak}} \exp(j\theta) \tag{6.11}$$

where $\theta = \cos^{-1}(0.9)$, i.e, power injection $s_{i,d}$ is considered to be a constant, obtained by assuming a power factor of 0.9 at peak apparent power. If device d is a capacitor with nameplate $\overline{q}$, then we impose

$$\text{Re}(s_{i,d}) = 0 \text{ and } 0 \le \text{Im}(s_{i,d}) \le \overline{q}. \tag{6.12}$$

If device d is a PV panel with nameplate $\overline{s}$, then we impose

$$\text{Re}(s_{i,d}) \ge 0 \text{ and } |s_{i,d}| \le \overline{s}. \tag{6.13}$$

The power injection at bus i is

$$s_i = \sum_{d=1}^{D_i} s_{i,d}$$

where $s_{i,d}$ satisfies one of (6.10)–(6.13).

3. The voltage regulation constraint is considered to be $0.9^2 \le v_i \le 1.1^2$ for $i \in \mathcal{N}^+$. Note that we choose a small voltage lower bound 0.9 so that OPF is feasible for all test networks. We choose a big voltage upper bound 1.1 such that Condition C2 holds and therefore SDP/SOCP is exact under C1.

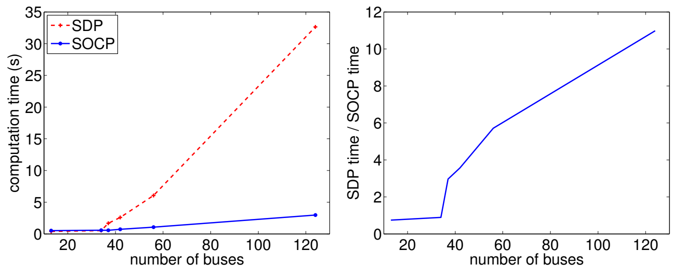


Figure 6.1: Comparison of the computation times for SOCP and SDP.

The computation times of SDP and SOCP for different test networks are summarized in Fig. 6.1. The number of buses determines the number of constraints and variables in the optimization, and therefore reflects the problem size. Network topology also affects the computation time. As the number of buses increases, the computation time of SOCP scales up much more slowly than that of SDP and their ratio increases dramatically. Hence SOCP is much more efficient than SDP for medium to large networks.

SOCP and SDP can only be solved to certain numerical precisions. The best numerical precision we obtain without applying pre-conditioning techniques are listed in Table 6.1.

6.5.3 C1 holds with a large margin

In this section, we show that C1 holds with a large margin for all test networks. Noting that C1 becomes more difficult to hold as $(\overline{p},\overline{q})$ increases (Proposition 6.3), one can increase $\overline{p}$, $\overline{q}$ until C1 fails. More specifically, let p_i^{fix} and q_i^{fix} denote the fixed real and reactive loads at bus $i \in \mathcal{N}^+$, let PV_i and Cap_i denote the nameplate capacities of the photovoltaic panels and the shunt capacitors at bus $i \in \mathcal{N}^+$, and define

$$\begin{aligned} \overline{p}_i(\eta) &:= p_i^{\text{fix}} + \eta \cdot \text{PV}_i, & i \in \mathcal{N}^+,\ \eta \geq 0; \\ \overline{q}_i(\eta) &:= q_i^{\text{fix}} + \eta \cdot (\text{PV}_i + \text{Cap}_i), & i \in \mathcal{N}^+,\ \eta \geq 0. \end{aligned}$$

When $\eta = 0$, one has $(\overline{p}(\eta), \overline{q}(\eta)) \leq 0$ and therefore C1 holds according to Proposition 6.4. According to Proposition 6.3, there exists a unique $\eta^* \in \mathbb{R}^+ \cup \{+\infty\}$ such that

$$\eta < \eta^* \Rightarrow \text{C1 holds for } (r, x, \overline{p}(\eta), \overline{q}(\eta), \underline{v}); \tag{6.14a}$$

$$\eta > \eta^* \Rightarrow \text{C1 does not hold for } (r, x, \overline{p}(\eta), \overline{q}(\eta), \underline{v}). \tag{6.14b}$$

Definition 6.2. *C1 margin is defined as the unique $\eta^* \geq 0$ that satisfies* (6.14).

Physically, η^* is the multiple by which one can scale up distributed generation (PVs) and shunt capacitors before C1 fails to hold. Noting that $\overline{p} = \overline{p}(1)$ and $\overline{q} = \overline{q}(1)$, C1 holds for $(r, x, \overline{p}, \overline{q}, \underline{v})$ if and only if $\eta^* > 1$ (ignore the corner case where $\eta^* = 1$). The larger η^* is, the "more easily" C1 holds.

The C1 margins of different test networks are summarized in Table 6.1. The minimum C1 margin is 1.30, meaning that one can scale up distributed generation and shunt capacitors by 1.30 times before C1 fails to hold. C1 margin of the IEEE 37-bus network is $+\infty$, and this is because there is neither distributed generation nor shunt capacitors in the network.

The C1 margin is above 20 for all IEEE networks, but much smaller for SCE networks. This is because SCE networks have big $\overline{p}$ and $\overline{q}$ (due to big PV_i and Cap_i) that make C1 more difficult to hold. However, note that the SCE 56-bus network already has a DG penetration of over 130%, and that one can still scale up its DG by a factor of 1.30 times before C1 breaks down. This highlights that C1 is a mild condition.

6.5.4 The feasible sets of OPF and OPF-m are similar

In this section, we show that OPF-m eliminates some feasible points of OPF that are close to the voltage upper bounds for all test networks. To present the result, let $\mathcal{F}_{\text{OPF}}$ denote the feasible set of OPF, let $\|\cdot\|_\infty$ denote the ℓ_∞ norm,[1] and let

$$\varepsilon \;:=\; \max\ \|\hat{v}(s)-v\|_\infty \ \text{ s.t. } \ (s,S,v,\ell,s_0)\in\mathcal{F}_{\text{OPF}} \tag{6.15}$$

denote the maximum deviation of v from its linear approximation $\hat{v}(s)$ over all OPF feasible points (s,S,v,ℓ,s_0).

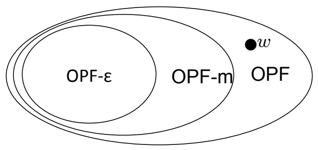


Figure 6.2: Feasible sets of OPF-ε, OPF-m, and OPF. The point w is feasible for OPF but not for OPF-m.

The value ε quantifies the difference between the feasible sets of OPF and OPF-m. Consider the OPF problem with a stricter voltage upper bound constraint:

$$\begin{aligned}
\textbf{OPF-}\boldsymbol{\varepsilon}\textbf{:}\ \min \quad & \sum_{i\in\mathcal{N}} f_i(\text{Re}(s_i))\\
\text{over} \quad & s,S,v,\ell,s_0\\
\text{s.t.} \quad & (6.5a) - (6.5e);\\
& \underline{v}_i \le v_i \le \overline{v}_i - \varepsilon, \quad i\in\mathcal{N}^+.
\end{aligned}$$

The feasible set $\mathcal{F}_{\text{OPF-}\varepsilon}$ of OPF-ε is contained in $\mathcal{F}_{\text{OPF}}$. Hence, for every $(s,S,\ell,v,s_0)\in\mathcal{F}_{\text{OPF-}\varepsilon}\subseteq\mathcal{F}_{\text{OPF}}$, one has

$$\hat{v}_i(s) \le v_i + \|\hat{v}(s)-v\|_\infty \le \overline{v}_i - \varepsilon + \varepsilon = \overline{v}_i, \quad i\in\mathcal{N}^+$$

by (6.15). It follows that $\mathcal{F}_{\text{OPF-}\varepsilon}\subseteq\mathcal{F}_{\text{OPF-m}}$ and therefore

$$\mathcal{F}_{\text{OPF-}\varepsilon}\subseteq\mathcal{F}_{\text{OPF-m}}\subseteq\mathcal{F}_{\text{OPF}}$$

as illustrated in Fig. 6.2.

If ε is small, then $\mathcal{F}_{\text{OPF-m}}$ is similar to $\mathcal{F}_{\text{OPF}}$. Any point w that is feasible for OPF but infeasible for OPF-m is close to the voltage upper bound since $v_i > \overline{v}_i - \varepsilon$ for some $i\in\mathcal{N}^+$. Such points are perhaps undesirable for robust operation.

[1] The ℓ_∞ norm of a vector $x=(x_1,\dots,x_n)\in\mathbb{R}^n$ is defined as $\|x\|_\infty := \max\{|x_1|,\dots,|x_n|\}$.

Definition 6.3. *The value* ε *defined in* (6.15) *is called the modification gap.*

We demonstrate that the modification gap ε is small for all test networks through Monte-Carlo simulations. Note that ε is difficult to compute since the objective function in (6.15) is not concave and the constraints in (6.15) are not convex. We choose 1000 samples of s, calculate the corresponding (S, v, ℓ, s_0) by solving the power flow equations (6.1a)–(6.1d) (using the *forward backward sweep* algorithm [63]) for each s, and compute $\varepsilon(s) := \|\hat{v}(s) - v\|_\infty$ if $(s, S, v, \ell, s_0) \in \mathcal{F}_{\text{OPF}}$. We use the maximum $\varepsilon(s)$ over the 1000 samples as an estimate for ε. The estimated modification gap ε^{set} we obtained for different test networks are listed in Table 6.1. For example, $\varepsilon^{\text{set}} = 0.0362$ for the IEEE 13-bus network, in which case the voltage constraints are $0.81 \le v_i \le 1.21$ for OPF and $0.81 \le v_i \le 1.1738$ for OPF-ε (assuming $\varepsilon = \varepsilon^{\text{set}}$).

6.6 Conclusion

We have proved that SOCP is exact if conditions C1 and C2 hold. C1 can be checked a priori, and has the physical interpretation that upstream power flows should increase if the power loss on a line is reduced. C2 requires that optimal power injections lie in a region $\mathcal{S}_{\text{volt}}$ where voltage upper bounds do not bind. We have proposed a modified OPF problem that includes the additional constraint that power injections lie in $\mathcal{S}_{\text{volt}}$, such that the corresponding SOCP relaxation is exact under C1. We have also proved that SOCP has at most one optimal solution if it is convex and exact. These results unify and generalize our prior works [41, 42]. Empirical studies show that C1 holds with a large margin and that the feasible sets of OPF and OPF-m are close, for the IEEE 13, 34, 37, 123-bus networks and two real-world networks.

Appendix

6.A Proof of Lemma 6.1

Let (s, S, v, ℓ, s_0) satisfy (6.1a)–(6.1c) and $\ell \geq 0$ componentwise. It follows from (6.1a) that

$$S_{ij} = s_i + \sum_{h:\, h\to i} (S_{hi} - z_{hi}\ell_{hi}) \leq s_i + \sum_{h:\, h\to i} S_{hi}$$

for $(i,j) \in \mathcal{E}$. On the other hand, $\hat{S}_{ij}(s)$ is the solution of

$$\hat{S}_{ij} = s_i + \sum_{h:\, h\to i} \hat{S}_{hi}, \quad (i,j) \in \mathcal{E}.$$

By induction from the leaf lines, one can show that

$$S_{ij} \leq \hat{S}_{ij}(s), \quad (i,j) \in \mathcal{E}.$$

It follows from (6.1c) that

$$v_i - v_j \;=\; 2\mathrm{Re}(\bar{z}_{ij}S_{ij}) - |z_{ij}|^2\ell_{ij} \;\leq\; 2\mathrm{Re}(\bar{z}_{ij}S_{ij}) \;\leq\; 2\mathrm{Re}(\bar{z}_{ij}\hat{S}_{ij}(s))$$

for $(i,j) \in \mathcal{E}$. Sum up the inequalities over $\mathcal{P}_i$ to obtain

$$v_i - v_0 \leq 2 \sum_{(j,k)\in\mathcal{P}_i} \mathrm{Re}(\bar{z}_{jk}\hat{S}_{jk}(s)),$$

i.e., $v_i \leq \hat{v}_i(s)$, for $i \in \mathcal{N}$.

6.B Proof of Theorem 6.2

The proof idea of Theorem 6.2 has been illustrated via a 3-bus linear network in Section 6.2.1. Now we present the proof of Theorem 6.2 for general radial networks. Assume that f_0 is strictly increasing, and that C1 and C2 hold. If SOCP is not exact, then there exists an optimal SOCP solution

$w = (s, S, v, \ell, s_0)$ that violates (6.5d). We will construct another feasible point $w' = (s', S', v', \ell', s_0')$ of SOCP that has a smaller objective value than w. This contradicts the optimality of w, and therefore SOCP is exact.

Construction of w'

The construction of w' is as follows. Since w violates (6.5d), there exists a leaf bus $l \in \mathcal{L}$ with $m \in \{1, \ldots, n_l\}$ such that w satisfies (6.5d) on $(l_1, l_0), \ldots, (l_{m-1}, l_{m-2})$ and violates (6.5d) on (l_m, l_{m-1}). Without loss of generality, assume $l_k = k$ for $k = 0, \ldots, m$ as in Fig. 6.B.1. Then

$$\ell_{m,m-1} > \frac{|S_{m,m-1}|^2}{v_m}, \tag{6.16a}$$

$$\ell_{k,k-1} = \frac{|S_{k,k-1}|^2}{v_k}, \quad k = 1, \ldots, m-1. \tag{6.16b}$$

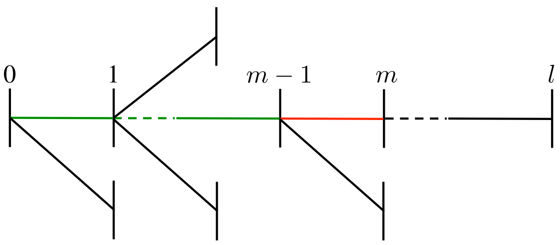


Figure 6.B.1: Bus l is a leaf bus, with $l_k = k$ for $k = 0, \ldots, m$. Equality (6.5d) is satisfied on $[0, m-1]$, but violated on $[m-1, m]$.

One can then construct $w' = (s', S', v', \ell', s_0')$ as in Algorithm 10. The construction consists of three steps:

S1 In the initialization step, s', ℓ' outside path $\mathcal{P}_m$, and S' outside path $\mathcal{P}_{m-1}$ are initialized as the corresponding values in w. Since $s' = s$, the point w' satisfies (6.5e). Furthermore, since $\ell'_{ij} = \ell_{ij}$ for $(i,j) \notin \mathcal{P}_m$ and $S'_{ij} = S_{ij}$ for $(i,j) \notin \mathcal{P}_{m-1}$, the point w' also satisfies (6.5a) for $(i,j) \notin \mathcal{P}_{m-1}$.

S2 In the forward sweep step, $\ell'_{k,k-1}$ and $S'_{k-1,k-2}$ are recursively constructed for $k = m, \ldots, 1$ by alternatively applying (6.5d) (with v' replaced by v) and (6.5a)/(6.5b). Hence, w' satisfies (6.5a) for $(i,j) \in \mathcal{P}_{m-1}$ and (6.5b).

S3 In the backward sweep step, v'_i is recursively constructed from bus 0 to leaf buses by applying (6.5c) consecutively. Hence, the point w' satisfies (6.5c).

The point w' satisfies another important property given below.

Lemma 6.10. *The point w' satisfies $\ell'_{ij} \geq |S'_{ij}|^2 / v_i$ for $(i,j) \in \mathcal{E}$.*

Algorithm 10 Construct a feasible point

```
Input: an optimal SOCP solution $w = (s, S, v, \ell, s_0)$ that violates (6.5d), a leaf bus $l \in \mathcal{L}$ with
    $1 \le m \le n_l$ such that (6.16) holds (assume $l_k = k$ for $k = 0, \ldots, m$ without loss of generality).
Output: $w' = (s', S', v', \ell', s_0')$.
    Initialization. Construct $s'$, $\ell'$ outside $\mathcal{P}_m$, and $S'$ outside $\mathcal{P}_{m-1}$.
 1: keep $s$: $s' \leftarrow s$;
 2: keep $\ell$ outside path $\mathcal{P}_m$: $\ell'_{ij} \leftarrow \ell_{ij}$ for $(i,j) \notin \mathcal{P}_m$;
 3: keep $S$ outside path $\mathcal{P}_{m-1}$: $S'_{ij} \leftarrow S_{ij}$ for $(i,j) \notin \mathcal{P}_{m-1}$;
    Forward sweep. Construct $\ell'$ on $\mathcal{P}_m$, $S'$ on $\mathcal{P}_{m-1}$, and $s_0'$.
 4: for $k = m, m-1, \ldots, 1$
 5:     $\ell'_{k,k-1} \leftarrow \frac{|S'_{k,k-1}|^2}{v_k}$;
 6:     $S'_{k-1,k-2} \leftarrow s_{k-1}\mathbf{1}_{k\neq 1} + \sum_{j:\, j\to k-1}(S'_{j,k-1} - z_{j,k-1}\ell'_{j,k-1})$;
 7: end
 8: $s_0' \leftarrow -S'_{0,-1}$;
    Backward sweep. Construct $v'$.
 9: $v_0' \leftarrow v_0$;
10: $\mathcal{N}_{\text{visit}} \leftarrow \{0\}$;
11: while $\mathcal{N}_{\text{visit}} \neq \mathcal{N}$
12:     find $i \notin \mathcal{N}_{\text{visit}}$ and $j \in \mathcal{N}_{\text{visit}}$ such that $i \to j$;
13:     $v_i' \leftarrow v_j' + 2\text{Re}(\bar{z}_{ij}S'_{ij}) - |z_{ij}|^2\ell'_{ij}$;
14:     $\mathcal{N}_{\text{visit}} \leftarrow \mathcal{N}_{\text{visit}} \cup \{i\}$;
15: end
```

Proof. When $(i,j) \notin \mathcal{P}_m$, it follows from Step S1 that $\ell'_{ij} = \ell_{ij} \ge |S_{ij}|^2/v_i = |S'_{ij}|^2/v_i$. When $(i,j) \in \mathcal{P}_m$, it follows from Step S2 that $\ell'_{ij} = |S'_{ij}|^2/v_i$. This completes the proof of Lemma 6.10. □

Lemma 6.10 implies that if $v' \ge v$, then w' satisfies (6.6).

Feasibility and Superiority of w'

We will show that w' is feasible for SOCP and has a smaller objective value than w. This result follows from Claims 6.11 and 6.12.

Claim 6.11. *If C1 holds, then $S'_{k,k-1} > S_{k,k-1}$ for $k = 0, \ldots, m-1$ and $v' \ge v$.*

Claim 6.11 is proved later in this appendix. Here we illustrate with Fig. 6.B.2 that $S'_{k,k-1} > S_{k,k-1}$ for $k = 0, \ldots, m-1$ seems natural to hold. Note that $S'_{m,m-1} = S_{m,m-1}$ and that $\ell'_{m,m-1} =$

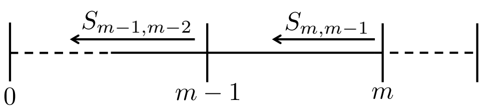


Figure 6.B.2: Illustration of $S'_{k,k-1} > S_{k,k-1}$ for $k = 0, \ldots, m-1$.

$|S'_{m,m-1}|^2/v_m = |S_{m,m-1}|^2/v_m < \ell_{m,m-1}$. Define $\Delta w = (\Delta s, \Delta S, \Delta v, \Delta \ell, \Delta s_0) = w' - w$, then

$\Delta\ell_{m,m-1} < 0$ and therefore

$$\Delta S_{m-1,m-2} \;=\; \Delta S_{m,m-1} - z_{m,m-1}\Delta\ell_{m,m-1} \;=\; -\,z_{m,m-1}\Delta\ell_{m,m-1} > 0. \tag{6.17}$$

Intuitively, after increasing $S_{m-1,m-2}$, upstream reverse power flow $S_{k,k-1}$ is likely to increase for $k = 0, \ldots, m-2$. C1 is a condition that ensures $S_{k,k-1}$ to increase for $k = 0, \ldots, m-1$.

Claim 6.12. *If C2 holds, then $v' \leq \overline{v}$.*

Proof. If C2 holds, then it follows from Lemma 6.1 that $v' \leq \hat{v}(s') = \hat{v}(s) \leq \overline{v}$. □

It follows from Claims 6.11 and 6.12 that $\underline{v} \leq v \leq v' \leq \overline{v}$, and therefore w' satisfies (6.5f). Besides, it follows from Lemma 6.10 that $\ell'_{ij} \geq |S'_{ij}|^2/v_i \geq |S'_{ij}|^2/v'_i$ for $(i,j) \in \mathcal{E}$, i.e., w' satisfies (6.6). Hence, w' is feasible for SOCP. Furthermore, w' has a smaller objective value than w because

$$\sum_{i\in\mathcal{N}} f_i(\mathrm{Re}(s'_i)) - \sum_{i\in\mathcal{N}} f_i(\mathrm{Re}(s_i)) \;=\; f_0(-\mathrm{Re}(S'_{0,-1})) - f_0(-\mathrm{Re}(S_{0,-1})) < 0.$$

This contradicts with the optimality of w, and therefore SOCP is exact. To complete the proof, we are left to prove Claim 6.11.

Proof of Claim 6.11

Assume C1 holds. First show that $\Delta S_{k,k-1} > 0$ for $k = 0, \ldots, m-1$. Recall that $S = P + \mathbf{i}Q$ and that $u_i = (r_{ij}\ x_{ij})^T$. It follows from (6.17) that

$$\begin{bmatrix} \Delta P_{m-1,m-2} \\ \Delta Q_{m-1,m-2} \end{bmatrix} = -u_m \Delta\ell_{m,m-1} > 0.$$

For any $k \in \{1, \ldots, m-1\}$, one has

$$\Delta S_{k-1,k-2} \;=\; \Delta S_{k,k-1} - z_{k,k-1}\Delta\ell_{k,k-1} \;=\; \Delta S_{k,k-1} - z_{k,k-1}\frac{|S'_{k,k-1}|^2 - |S_{k,k-1}|^2}{v_k},$$

which is equivalent to

$$\begin{bmatrix} \Delta P_{k-1,k-2} \\ \Delta Q_{k-1,k-2} \end{bmatrix} = B_k \begin{bmatrix} \Delta P_{k,k-1} \\ \Delta Q_{k,k-1} \end{bmatrix}$$

where

$$B_k = I - \frac{2}{v_k}\begin{bmatrix} r_{k,k-1} \\ x_{k,k-1} \end{bmatrix} \begin{bmatrix} \frac{P_{k,k-1}+P'_{k,k-1}}{2} & \frac{Q_{k,k-1}+Q'_{k,k-1}}{2} \end{bmatrix}.$$

Hence, one has

$$\begin{bmatrix} \Delta P_{k-1,k-2} \\ \Delta Q_{k-1,k-2} \end{bmatrix} = -B_k B_{k+1} \cdots B_{m-1} u_m \Delta \ell_{m,m-1}$$

for $k = 1, \ldots, m$. To prove $\Delta S_{k,k-1} > 0$ for $k = 0, \ldots, m-1$, it suffices to show that $B_k \cdots B_{m-1} u_m > 0$ for $k = 1, \ldots, m$.

C1 implies that $\underline{A}_s \cdots \underline{A}_{t-1} u_t > 0$ when $1 \leq s \leq t \leq m$. One also has $B_k - \underline{A}_k = u_k b_k^T$ where

$$b_k = \begin{bmatrix} \frac{2\hat{P}^+_{k,k-1}(\overline{p})}{\underline{v}_k} - \frac{P_{k,k-1}+P'_{k,k-1}}{v_k} \\ \frac{2\hat{Q}^+_{k,k-1}(\overline{q})}{\underline{v}_k} - \frac{Q_{k,k-1}+Q'_{k,k-1}}{v_k} \end{bmatrix} \geq 0$$

for $k = 1, \ldots, m-1$. To show that $B_k \cdots B_{m-1} u_m > 0$ for $k = 1, \ldots, m$, we prove the following lemma.

Lemma 6.13. *Given $m \geq 1$ and $d \geq 1$. Let $\underline{A}_1, \ldots, \underline{A}_{m-1}, A_1, \ldots, A_{m-1} \in \mathbb{R}^{d\times d}$ and $u_1, \ldots, u_m \in \mathbb{R}^d$ satisfy*

- *$\underline{A}_s \cdots \underline{A}_{t-1} u_t > 0$ when $1 \leq s \leq t \leq m$;*
- *there exists $b_k \in \mathbb{R}^d$ that satisfies $b_k \geq 0$ and $A_k - \underline{A}_k = u_k b_k^T$, for $k = 1, \ldots, m-1$.*

Then

$$A_s \cdots A_{t-1} u_t > 0 \tag{6.18}$$

when $1 \leq s \leq t \leq m$.

Proof. We prove that (6.18) holds when $1 \leq t \leq s \leq m$ by mathematical induction on $t - s$.

i) When $t - s = 0$, one has $A_s \cdots A_{t-1} u_t = u_t = \underline{A}_s \cdots \underline{A}_{t-1} u_t > 0$.

ii) Assume that (6.18) holds when $t - s = 0, 1, \ldots, K$ $(0 \leq K \leq m-2)$. When $t - s = K + 1$, one has

$$\begin{aligned} & A_s \cdots A_k \underline{A}_{k+1} \cdots \underline{A}_{t-1} u_t \\ = \; & A_s \cdots A_{k-1} \underline{A}_k \underline{A}_{k+1} \cdots \underline{A}_{t-1} u_t + A_s \cdots A_{k-1} (A_k - \underline{A}_k) \underline{A}_{k+1} \cdots \underline{A}_{t-1} u_t \\ = \; & A_s \cdots A_{k-1} \underline{A}_k \cdots \underline{A}_{t-1} u_t + A_s \cdots A_{k-1} u_k b_k^T \underline{A}_{k+1} \cdots \underline{A}_{t-1} u_t \\ = \; & A_s \cdots A_{k-1} \underline{A}_k \cdots \underline{A}_{t-1} u_t + \left(b_k^T \underline{A}_{k+1} \cdots \underline{A}_{t-1} u_t \right) A_s \cdots A_{k-1} u_k \end{aligned}$$

for $k = s, \ldots, t-1$. Since $b_k \geq 0$ and $\underline{A}_{k+1} \cdots \underline{A}_{t-1} u_t > 0$, the term $b_k^T \underline{A}_{k+1} \cdots \underline{A}_{t-1} u_t \geq 0$. According to induction hypothesis, $A_s \cdots A_{k-1} u_k > 0$. Hence,

$$A_s \cdots A_k \underline{A}_{k+1} \cdots \underline{A}_{t-1} u_t \geq A_s \cdots A_{k-1} \underline{A}_k \cdots \underline{A}_{t-1} u_t$$

for $k = s, \ldots, t-1$. By substituting $k = t-1, \ldots, s$ in turn, one obtains

$$A_s \cdots A_{t-1} u_t \;\geq\; A_s \cdots A_{t-2} \underline{A}_{t-1} u_t \;\geq\; \cdots \;\geq\; \underline{A}_s \cdots \underline{A}_{t-1} u_t > 0,$$

i.e., (6.18) holds when $t - s = K + 1$.

According to i) and ii), (6.18) holds when $t - s = 0, \ldots, m-1$. This completes the proof of Lemma 6.13. □

Lemma 6.13 implies that $B_s \cdots B_{t-1} u_t > 0$ when $1 \leq s \leq t \leq m$. In particular, $B_k \cdots B_{m-1} u_m > 0$ for $k = 1, \ldots, m$, and therefore $\Delta S_{k,k-1} > 0$ for $k = 0, \ldots, m-1$.

Next show that $v' \geq v$. Noting that $\Delta S_{ij} = 0$ for $(i,j) \notin \mathcal{P}_{m-1}$ and $\Delta \ell_{ij} = 0$ for $(i,j) \notin \mathcal{P}_m$, it follows from (6.5c) that

$$\Delta v_i - \Delta v_j = 2\mathrm{Re}(\bar{z}_{ij} \Delta S_{ij}) - |z_{ij}|^2 \Delta \ell_{ij} = 0$$

for $(i,j) \notin \mathcal{P}_m$. When $(i,j) \in \mathcal{P}_m$, one has $(i,j) = (k, k-1)$ for some $k \in \{1, \ldots, m\}$, and therefore

$$\begin{aligned}
\Delta v_i - \Delta v_j &= 2\mathrm{Re}(\bar{z}_{k,k-1} \Delta S_{k,k-1}) - |z_{k,k-1}|^2 \Delta \ell_{k,k-1} \\
&\geq \mathrm{Re}(\bar{z}_{k,k-1} \Delta S_{k,k-1}) - |z_{k,k-1}|^2 \Delta \ell_{k,k-1} \\
&= \mathrm{Re}(\bar{z}_{k,k-1} (\Delta S_{k,k-1} - z_{k,k-1} \Delta \ell_{k,k-1})) \\
&= \mathrm{Re}(\bar{z}_{k,k-1} \Delta S_{k-1,k-2}) > 0.
\end{aligned}$$

Hence, $\Delta v_i \geq \Delta v_j$ whenever $(i,j) \in \mathcal{E}$. Add the inequalities over path $\mathcal{P}_i$ to obtain $\Delta v_i \geq \Delta v_0 = 0$ for $i \in \mathcal{N}^+$, i.e., $v' \geq v$. This completes the proof of Claim 6.11.

6.C Proof of Proposition 6.3

Let $\underline{A}$ and $\underline{A}'$ denote the matrices with respect to $(\overline{p}, \overline{q})$ and $(\overline{p}', \overline{q}')$, respectively, i.e., let

$$\begin{aligned}
\underline{A}_i' &= I - \frac{2}{\underline{v}_i} u_i \left(\hat{P}_{ij}^+(\overline{p}') \; \hat{Q}_{ij}^+(\overline{q}') \right), \quad (i,j) \in \mathcal{E}; \\
\underline{A}_i &= I - \frac{2}{\underline{v}_i} u_i \left(\hat{P}_{ij}^+(\overline{p}) \; \hat{Q}_{ij}^+(\overline{q}) \right), \quad (i,j) \in \mathcal{E}.
\end{aligned}$$

When $(\overline{p}, \overline{q}) \leq (\overline{p}', \overline{q}')$, one has $\underline{A}_{l_k} - \underline{A}_{l_k}' = u_{l_k} b_{l_k}^T$ where

$$b_{l_k} = \frac{2}{\underline{v}_{l_k}} \begin{bmatrix} \hat{P}_{l_k l_{k-1}}^+(\overline{p}') - \hat{P}_{l_k l_{k-1}}^+(\overline{p}) \\ \hat{Q}_{l_k l_{k-1}}^+(\overline{q}') - \hat{Q}_{l_k l_{k-1}}^+(\overline{q}) \end{bmatrix} \geq 0$$

for any $l \in \mathcal{L}$ and any $k \in \{1 \ldots, n_l\}$.

If $\underline{A}'_{l_s} \cdots \underline{A}'_{l_{t-1}} u_{l_t} > 0$ for any $l \in \mathcal{L}$ and any s, t such that $1 \le s \le t \le n_l$, then it follows from Lemma 6.13 that $\underline{A}_{l_s} \cdots \underline{A}_{l_{t-1}} u_{l_t} > 0$ for any $l \in \mathcal{L}$ any s, t such that $1 \le s \le t \le n_l$. This completes the proof of Proposition 6.3.

6.D Proof of Theorem 6.6

In this appendix, we prove that SOCP has at most a unique solution under the conditions in Theorem 6.6. The proof for SOCP-m is similar and omitted for brevity.

Assume that f_i is convex for $i \in \mathcal{N}$, $\mathcal{S}_i$ is convex for $i \in \mathcal{N}^+$, SOCP is exact, and SOCP has at least one solution. Let $\tilde{w} = (\tilde{s}, \tilde{S}, \tilde{v}, \tilde{\ell}, \tilde{s}_0)$ and $\hat{w} = (\hat{s}, \hat{S}, \hat{v}, \hat{\ell}, \hat{s}_0)$ denote two arbitrary SOCP solutions. It suffices to show that $\tilde{w} = \hat{w}$.

Since SOCP is exact, $\tilde{v}_i \tilde{\ell}_{ij} = |\tilde{S}_{ij}|^2$ and $\hat{v}_i \hat{\ell}_{ij} = |\hat{S}_{ij}|^2$ for $(i, j) \in \mathcal{E}$. Define $w := (\tilde{w} + \hat{w})/2$. Since SOCP is convex, w also solves SOCP. Hence, $v_i \ell_{ij} = |S_{ij}|^2$ for $(i, j) \in \mathcal{E}$. Substitute $v_i = (\tilde{v}_i + \hat{v}_i)/2$, $\ell_{ij} = (\tilde{\ell}_{ij} + \hat{\ell}_{ij})/2$, and $S_{ij} = (\tilde{S}_{ij} + \hat{S}_{ij})/2$ to obtain

$$\hat{S}_{ij} \tilde{S}_{ij}^H + \tilde{S}_{ij} \hat{S}_{ij}^H = \hat{v}_i \tilde{\ell}_{ij} + \tilde{v}_i \hat{\ell}_{ij}$$

for $(i, j) \in \mathcal{E}$, where the superscript H stands for hermitian transpose. The right hand side

$$\hat{v}_i \tilde{\ell}_{ij} + \tilde{v}_i \hat{\ell}_{ij} = \hat{v}_i \frac{|\tilde{S}_{ij}|^2}{\tilde{v}_i} + \tilde{v}_i \frac{|\hat{S}_{ij}|^2}{\hat{v}_i} \ge 2|\tilde{S}_{ij}||\hat{S}_{ij}|,$$

and the equality is attained if and only if $|\tilde{S}_{ij}|/\tilde{v}_i = |\hat{S}_{ij}|/\hat{v}_i$. The left hand side

$$\hat{S}_{ij} \tilde{S}_{ij}^H + \tilde{S}_{ij} \hat{S}_{ij}^H \le 2|\tilde{S}_{ij}||\hat{S}_{ij}|,$$

and the equality is attained if and only if $\angle \hat{S}_{ij} = \angle \tilde{S}_{ij}$. Hence, $\tilde{S}_{ij}/\tilde{v}_i = \hat{S}_{ij}/\hat{v}_i$ for $(i, j) \in \mathcal{E}$.

Introduce $\hat{v}_0 := \tilde{v}_0 := v_0$ and define $\eta_i := \hat{v}_i/\tilde{v}_i$ for $i \in \mathcal{N}$, then $\eta_0 = 1$ and $\hat{S}_{ij} = \eta_i \tilde{S}_{ij}$ for $(i, j) \in \mathcal{E}$. Hence,

$$\hat{\ell}_{ij} = \frac{|\hat{S}_{ij}|^2}{\hat{v}_i} = \frac{|\eta_i \tilde{S}_{ij}|^2}{\eta_i \tilde{v}_i} = \eta_i \frac{|\tilde{S}_{ij}|^2}{\tilde{v}_i} = \eta_i \tilde{\ell}_{ij}$$

and therefore

$$\eta_j = \frac{\hat{v}_j}{\tilde{v}_j} = \frac{\hat{v}_i - 2\mathrm{Re}(z_{ij}^H \hat{S}_{ij}) + |z_{ij}|^2 \hat{\ell}_{ij}}{\tilde{v}_i - 2\mathrm{Re}(z_{ij}^H \tilde{S}_{ij}) + |z_{ij}|^2 \tilde{\ell}_{ij}} = \eta_i$$

for $(i, j) \in \mathcal{E}$. Since the network $(\mathcal{N}, \mathcal{E})$ is connected, $\eta_i = \eta_0 = 1$ for $i \in \mathcal{N}$. This implies $\hat{w} = \tilde{w}$ and completes the proof of Theorem 6.6.

6.E Proof of Theorem 6.8

Theorem 6.8 follows from Claims 6.14–6.18.

Claim 6.14. *Assume that there exist $\overline{p}_i$ and $\overline{q}_i$ such that $\mathcal{S}_i \subseteq \{s \in \mathbb{C} \mid \mathrm{Re}(s) \le \overline{p}_i,\ \mathrm{Im}(s) \le \overline{q}_i\}$ for $i \in \mathcal{N}^+$. Then C1 holds if $\hat{S}_{ij}(\overline{p} + \boldsymbol{i}\overline{q}) \le 0$ for $(i,j) \in \mathcal{E}'$.*

Proof. If $\hat{S}_{ij}(\overline{p} + \mathbf{i}\overline{q}) \le 0$ for $(i,j) \in \mathcal{E}'$, then $\underline{A}_{l_k} = I$ for $l \in \mathcal{L}$ and $k \in \{1 \ldots, n_l - 1\}$. It follows that $\underline{A}_{l_s} \cdots \underline{A}_{l_{t-1}} u_{l_t} = u_{l_t} > 0$ for $l \in \mathcal{L}$ and s, t such that $1 \le s \le t \le n_l$, i.e., C1 holds. □

Claim 6.15. *Assume that there exist $\overline{p}_i$ and $\overline{q}_i$ such that $\mathcal{S}_i \subseteq \{s \in \mathbb{C} \mid \mathrm{Re}(s) \le \overline{p}_i,\ \mathrm{Im}(s) \le \overline{q}_i\}$ for $i \in \mathcal{N}^+$. Then C1 holds if 1) r_{ij}/x_{ij} is identical for $(i,j) \in \mathcal{E}$; and 2) $\underline{v}_i - 2r_{ij}\hat{P}^+_{ij}(\overline{p}) - 2x_{ij}\hat{Q}^+_{ij}(\overline{q}) > 0$ for $(i,j) \in \mathcal{E}'$.*

Proof. Assume the conditions in Claim 6.15 hold. Fix an arbitrary $l \in \mathcal{L}$, and assume $l_k = k$ for $k = 0, \ldots, n_l$ without loss of generality. Fix an arbitrary $t \in \{1, \ldots, n_l\}$, and define $(\alpha_s\ \beta_s)^T := \underline{A}_s \cdots \underline{A}_{t-1} u_t$ for $s = 1, \ldots, t$. Then it suffices to prove that $\alpha_s > 0$ and $\beta_s > 0$ for $s = 1, \ldots, t$. In particular, we prove

$$\alpha_s > 0,\ \beta_s > 0,\ \alpha_s/\beta_s = r_{10}/x_{10} \tag{6.19}$$

inductively for $s = t, t-1, \ldots, 1$. Define $\eta := r_{10}/x_{10}$ and note that $r_{ij}/x_{ij} = \eta$ for all $(i,j) \in \mathcal{E}$.

i) When $s = t$, one has $\alpha_s = r_{t,t-1}$, $\beta_s = x_{t,t-1}$, and $\alpha_s/\beta_s = \eta$. Therefore (6.19) holds.

ii) Assume that (6.19) holds for $s = k$ ($2 \le k \le t$), then

$$\begin{bmatrix}\alpha_k & \beta_k\end{bmatrix}^T = c\begin{bmatrix}\eta & 1\end{bmatrix}^T$$

for some $c \in \{c \in \mathbb{R} \mid c > 0\}$. Abbreviate $r_{k-1,k-2}$ by r, $x_{k-1,k-2}$ by x, $\hat{P}^+_{k-1,k-2}(\bar{p})$ by P, and $\hat{Q}^+_{k-1,k-2}(\bar{q})$ by Q for convenience. Then

$$\underline{v}_{k-1} - 2rP - 2xQ > 0$$

and it follows that

$$\begin{aligned}
\begin{bmatrix}\alpha_{k-1}\\ \beta_{k-1}\end{bmatrix} &= \left(I - \frac{2}{\underline{v}_{k-1}}\begin{bmatrix}r\\ x\end{bmatrix}\begin{bmatrix}P & Q\end{bmatrix}\right)\begin{bmatrix}\alpha_k\\ \beta_k\end{bmatrix}\\
&= \left(I - \frac{2}{\underline{v}_{k-1}}x\begin{bmatrix}\eta\\ 1\end{bmatrix}\begin{bmatrix}P & Q\end{bmatrix}\right)c\begin{bmatrix}\eta\\ 1\end{bmatrix}\\
&= c\begin{bmatrix}\eta\\ 1\end{bmatrix} - \frac{2}{\underline{v}_{k-1}}c\begin{bmatrix}\eta\\ 1\end{bmatrix}\begin{bmatrix}P & Q\end{bmatrix}x\begin{bmatrix}\eta\\ 1\end{bmatrix}\\
&= \begin{bmatrix}\alpha_k\\ \beta_k\end{bmatrix} - \frac{2}{\underline{v}_{k-1}}\begin{bmatrix}\alpha_k\\ \beta_k\end{bmatrix}\begin{bmatrix}P & Q\end{bmatrix}\begin{bmatrix}r\\ x\end{bmatrix}\\
&= \left(1 - \frac{2}{\underline{v}_{k-1}}(rP + xQ)\right)\begin{bmatrix}\alpha_k\\ \beta_k\end{bmatrix}\\
&= \frac{1}{\underline{v}_{k-1}}\left(\underline{v}_{k-1} - 2rP - 2xQ\right)\begin{bmatrix}\alpha_k\\ \beta_k\end{bmatrix} > 0
\end{aligned}$$

and $\alpha_{k-1}/\beta_{k-1} = \alpha_k/\beta_k = \eta$. Hence, (6.19) holds for $s = k-1$.

According to i) and ii), (6.19) holds for $s = t, t-1 \ldots, 1$. This completes the proof of Claim 6.15. $\square$

Claim 6.16. *Assume that there exist $\overline{p}_i$ and $\overline{q}_i$ such that $\mathcal{S}_i \subseteq \{s \in \mathbb{C} \mid \mathrm{Re}(s) \le \overline{p}_i,\ \mathrm{Im}(s) \le \overline{q}_i\}$ for $i \in \mathcal{N}^+$. Then C1 holds if 1) $r_{ij}/x_{ij} \ge r_{jk}/x_{jk}$ whenever $(i,j), (j,k) \in \mathcal{E}$; and 2) $\hat{P}_{ij}(\overline{p}) \le 0$, $\underline{v}_i - 2x_{ij}\hat{Q}^+_{ij}(\overline{q}) > 0$ for all $(i,j) \in \mathcal{E}'$.*

Proof. Assume the conditions in Claim 6.16 hold. Fix an arbitrary $l \in \mathcal{L}$, and assume $l_k = k$ for $k = 0, \ldots, n_l$ without loss of generality. Fix an arbitrary $t \in \{1, \ldots, n_l\}$, and define $(\alpha_s\ \beta_s)^T := \underline{A}_s \cdots \underline{A}_{t-1} u_t$ for $s = 1, \ldots, t$. Then it suffices to prove that $\alpha_s > 0$ and $\beta_s > 0$ for $s = 1, \ldots, t$. In particular, we prove

$$\alpha_s > 0,\ \beta_s > 0,\ \alpha_s/\beta_s \ge r_{t,t-1}/x_{t,t-1} \qquad (6.20)$$

inductively for $s = t, t-1, \ldots, 1$. Define $\eta := r_{t,t-1}/x_{t,t-1}$ and note that $r_{s,s-1}/x_{s,s-1} \le \eta$ for $s = 1, 2, \ldots, t$.

i) When $s = t$, one has $\alpha_s = r_{t,t-1}$, $\beta_s = x_{t,t-1}$, and $\alpha_s/\beta_s = \eta$. Therefore (6.20) holds.

ii) Assume that (6.20) holds for $s = k$ $(2 \le k \le t)$, then

$$\alpha_k \ge \eta\beta_k > 0.$$

Abbreviate $r_{k-1,k-2}$ by r, $x_{k-1,k-2}$ by x, $\hat{P}^+_{k-1,k-2}(\bar{p})$ by P, and $\hat{Q}^+_{k-1,k-2}(\bar{q})$ by Q for convenience. Then

$$P = 0, \quad \underline{v}_k - 2xQ > 0$$

and it follows that

$$\begin{aligned}\begin{bmatrix}\alpha_{k-1}\\ \beta_{k-1}\end{bmatrix} &= \left(I - \frac{2}{\underline{v}_{k-1}}\begin{bmatrix}r\\ x\end{bmatrix}\begin{bmatrix}P & Q\end{bmatrix}\right)\begin{bmatrix}\alpha_k\\ \beta_k\end{bmatrix}\\ &= \begin{bmatrix}\alpha_k\\ \beta_k\end{bmatrix} - \frac{2}{\underline{v}_{k-1}}\begin{bmatrix}r\\ x\end{bmatrix}Q\beta_k.\end{aligned}$$

Hence,

$$\beta_{k-1} = \beta_k - \frac{2xQ}{\underline{v}_{k-1}}\beta_k = \frac{1}{\underline{v}_{k-1}}\left(\underline{v}_{k-1} - 2xQ\right)\beta_k > 0.$$

Then,

$$\alpha_{k-1} = \alpha_k - \frac{2rQ}{\underline{v}_{k-1}}\beta_k \geq \left(\eta - \frac{2rQ}{\underline{v}_{k-1}}\right)\beta_k \geq \eta\left(1 - \frac{2xQ}{\underline{v}_{k-1}}\right)\beta_k = \eta\beta_{k-1} > 0.$$

The second inequality is due to $r/x \leq \eta$. Hence, (6.20) holds for $s = k-1$.

According to i) and ii), (6.20) holds for $s = t, t-1, \ldots, 1$. This completes the proof of Claim 6.16. □

Claim 6.17. *Assume that there exist $\overline{p}_i$ and $\overline{q}_i$ such that $\mathcal{S}_i \subseteq \{s \in \mathbb{C} \mid \mathrm{Re}(s) \leq \overline{p}_i,\ \mathrm{Im}(s) \leq \overline{q}_i\}$ for $i \in \mathcal{N}^+$. Then C1 holds if 1) $r_{ij}/x_{ij} \leq r_{jk}/x_{jk}$ whenever $(i,j),(j,k) \in \mathcal{E}$; and 2) $\hat{Q}_{ij}(\overline{q}) \leq 0$, $\underline{v}_i - 2r_{ij}\hat{P}^+_{ij}(\overline{p}) > 0$ for all $(i,j) \in \mathcal{E}'$.*

Proof. The proof of Claim 6.17 is similar to that of Claim 6.16 and omitted for brevity. □

Claim 6.18. *Assume that there exist $\overline{p}_i$ and $\overline{q}_i$ such that $\mathcal{S}_i \subseteq \{s \in \mathbb{C} \mid \mathrm{Re}(s) \leq \overline{p}_i,\ \mathrm{Im}(s) \leq \overline{q}_i\}$ for $i \in \mathcal{N}^+$. Then C1 holds if*

$$\begin{bmatrix}\prod_{(k,l)\in\mathcal{P}_j} c_{kl} & -\sum_{(k,l)\in\mathcal{P}_j} d_{kl}\\ -\sum_{(k,l)\in\mathcal{P}_j} e_{kl} & \prod_{(k,l)\in\mathcal{P}_j} f_{kl}\end{bmatrix}\begin{bmatrix}r_{ij}\\ x_{ij}\end{bmatrix} > 0, \quad (i,j) \in \mathcal{E} \tag{6.21}$$

where $c_{kl} := 1 - 2r_{kl}\hat{P}^+_{kl}(\bar{p})/\underline{v}_k$, $d_{kl} := 2r_{kl}\hat{Q}^+_{kl}(\bar{q})/\underline{v}_k$, $e_{kl} := 2x_{kl}\hat{P}^+_{kl}(\bar{p})/\underline{v}_k$, and $f_{kl} := 1 - 2x_{kl}\hat{Q}^+_{kl}(\bar{q})/\underline{v}_k$.

The following lemma is used in the proof of Claim 6.18.

Lemma 6.19. *Given $i \geq 1$; c, d, e, $f \in \mathbb{R}^i$ such that $0 < c \leq 1$, $d \geq 0$, $e \geq 0$, and $0 < f \leq 1$*

componentwise; and $u \in \mathbb{R}^2$ that satisfies $u > 0$. If

$$\begin{bmatrix} \prod_{j=1}^{i} c_j & -\sum_{j=1}^{i} d_j \\ -\sum_{j=1}^{i} e_j & \prod_{j=1}^{i} f_j \end{bmatrix} u > 0, \tag{6.22}$$

then

$$\begin{bmatrix} c_j & -d_j \\ -e_j & f_j \end{bmatrix} \cdots \begin{bmatrix} c_i & -d_i \\ -e_i & f_i \end{bmatrix} u > 0 \tag{6.23}$$

for $j = 1, \ldots, i$.

Proof. Lemma 6.19 can be proved by mathematical induction on i.

i) When $i = 1$, Lemma 6.19 is trivial.

ii) Assume that Lemma 6.19 holds for $i = K$ $(K \geq 1)$. When $i = K + 1$, if

$$\begin{bmatrix} \prod_{j=1}^{i} c_j & -\sum_{j=1}^{i} d_j \\ -\sum_{j=1}^{i} e_j & \prod_{j=1}^{i} f_j \end{bmatrix} u > 0,$$

one can prove that (6.23) holds for $j = 1, \ldots, K + 1$ as follows.

First prove that (6.23) holds for $j = 2, \ldots, K + 1$. The idea is to construct some $c', d', e', f' \in \mathbb{R}^K$ and apply the induction hypothesis. The construction is

$$c' = (c_2,\ c_3,\ \ldots,\ c_{K+1}), \qquad d' = (d_2,\ d_3,\ \ldots,\ d_{K+1}),$$
$$e' = (e_2,\ e_3,\ \ldots,\ e_{K+1}), \qquad f' = (f_2,\ f_3,\ \ldots,\ f_{K+1}).$$

Clearly, c', d', e', f' satisfies $0 < c' \leq 1$, $d' \geq 0$, $e' \geq 0$, $0 < f' \leq 1$ componentwise and

$$\begin{bmatrix} \prod_{j=1}^{K} c'_j & -\sum_{j=1}^{K} d'_j \\ -\sum_{j=1}^{K} e'_j & \prod_{j=1}^{K} f'_j \end{bmatrix} u = \begin{bmatrix} \prod_{j=2}^{K+1} c_j & -\sum_{j=2}^{K+1} d_j \\ -\sum_{j=2}^{K+1} e_j & \prod_{j=2}^{K+1} f_j \end{bmatrix} u \geq \begin{bmatrix} \prod_{j=1}^{K+1} c_j & -\sum_{j=1}^{K+1} d_j \\ -\sum_{j=1}^{K+1} e_j & \prod_{j=1}^{K+1} f_j \end{bmatrix} u > 0.$$

Apply the induction hypothesis to obtain that

$$\begin{bmatrix} c'_j & -d'_j \\ -e'_j & f'_j \end{bmatrix} \cdots \begin{bmatrix} c'_K & -d'_K \\ -e'_K & f'_K \end{bmatrix} u > 0$$

for $j = 1, \ldots, K$, i.e., (6.23) holds for $j = 2, \ldots, K+1$.

Next prove that (6.23) holds for $j = 1$. The idea is still to construct some $c', d', e', f' \in \mathbb{R}^K$ and apply the induction hypothesis. The construction is

$$\begin{aligned} c' &= (c_1c_2,\ c_3,\ \ldots,\ c_{K+1}), & d' &= (d_1 + d_2,\ d_3,\ \ldots,\ d_{K+1}), \\ e' &= (e_1 + e_2,\ e_3,\ \ldots,\ e_{K+1}), & f' &= (f_1f_2,\ f_3,\ \ldots,\ f_{K+1}). \end{aligned}$$

Clearly, c', d', e', f' satisfies $0 < c' \le 1$, $d' \ge 0$, $e' \ge 0$, $0 < f' \le 1$ componentwise and

$$\begin{bmatrix} \prod_{j=1}^{K} c'_j & -\sum_{j=1}^{K} d'_j \\ -\sum_{j=1}^{K} e'_j & \prod_{j=1}^{K} f'_j \end{bmatrix} u = \begin{bmatrix} \prod_{j=1}^{K+1} c_j & -\sum_{j=1}^{K+1} d_j \\ -\sum_{j=1}^{K+1} e_j & \prod_{j=1}^{K+1} f_j \end{bmatrix} u > 0.$$

Apply the induction hypothesis to obtain

$$\begin{aligned} v'_2 &:= \begin{bmatrix} c'_2 & -d'_2 \\ -e'_2 & f'_2 \end{bmatrix} \cdots \begin{bmatrix} c'_K & -d'_K \\ -e'_K & f'_K \end{bmatrix} u > 0, \\ v'_1 &:= \begin{bmatrix} c'_1 & -d'_1 \\ -e'_1 & f'_1 \end{bmatrix} \cdots \begin{bmatrix} c'_K & -d'_K \\ -e'_K & f'_K \end{bmatrix} u > 0. \end{aligned}$$

It follows that

$$\begin{aligned} \begin{bmatrix} c_1 & -d_1 \\ -e_1 & f_1 \end{bmatrix} \cdots \begin{bmatrix} c_{K+1} & -d_{K+1} \\ -e_{K+1} & f_{K+1} \end{bmatrix} u &= \begin{bmatrix} c_1 & -d_1 \\ -e_1 & f_1 \end{bmatrix} \begin{bmatrix} c_2 & -d_2 \\ -e_2 & f_2 \end{bmatrix} v'_2 \\ &= \begin{bmatrix} c_1c_2 + d_1e_2 & -c_1d_2 - d_1f_2 \\ -e_1c_2 - f_1e_2 & f_1f_2 + e_1d_2 \end{bmatrix} v'_2 \\ &\ge \begin{bmatrix} c_1c_2 & -d_2 - d_1 \\ -e_1 - e_2 & f_1f_2 \end{bmatrix} v'_2 \\ &= \begin{bmatrix} c'_1 & -d'_1 \\ -e'_1 & f'_1 \end{bmatrix} v'_2 \\ &= v'_1 \ > \ 0, \end{aligned}$$

i.e., (6.23) holds for $j = 1$.

To this end, we have proved that (6.23) holds for $j = 1, \ldots, K+1$, i.e., Lemma 6.19 also holds for $i = K+1$.

According to i) and ii), Lemma 6.19 holds for $i \ge 1$. □

Proof of Claim 6.18. Fix an arbitrary $l \in \mathcal{L}$, and assume $l_k = k$ for $k = 0, \ldots, n_l$ without loss of generality. Fix an arbitrary $t \in \{1, \ldots, n_l\}$, then it suffices to prove that $\underline{A}_s \cdots \underline{A}_{t-1} u_t > 0$ for $s = 1, \ldots, t$. Denote $r_k := r_{k,k-1}$ and $S_k := S_{k,k-1}$ for $k = 1, \ldots, t$ for brevity.

Substitute $(i, j) = (k, k-1)$ in (6.21) to obtain

$$\begin{bmatrix} \prod_{s=1}^{k-1} \left(1 - \dfrac{2r_s \hat{P}_s^+}{\underline{v}_s}\right) & -\sum_{s=1}^{k-1} \dfrac{2r_s \hat{Q}_s^+}{\underline{v}_s} \\ -\sum_{s=1}^{k-1} \dfrac{2x_s \hat{P}_s^+}{\underline{v}_s} & \prod_{s=1}^{k-1} \left(1 - \dfrac{2x_s \hat{Q}_s^+}{\underline{v}_s}\right) \end{bmatrix} \begin{bmatrix} r_k \\ x_k \end{bmatrix} > 0 \tag{6.24}$$

for $k = 1, \ldots, t$. Hence,

$$\prod_{s=1}^{k-1} \left(1 - \frac{2r_s \hat{P}_s^+}{\underline{v}_s}\right) r_k > \sum_{s=1}^{k-1} \frac{2r_s \hat{Q}_s^+(\overline{q})}{\underline{v}_s} x_k \geq 0$$

for $k = 1, \ldots, t$. It follows that $1 - 2r_k \hat{P}_k^+ / \underline{v}_k > 0$ for $k = 1, \ldots, t-1$. Similarly, $1 - 2x_k \hat{Q}_k^+ / \underline{v}_k > 0$ for $k = 1, \ldots, t-1$. Then, substitute $k = t$ in (6.24) and apply Lemma 6.19 to obtain

$$\begin{bmatrix} 1 - \dfrac{2r_s \hat{P}_s^+}{\underline{v}_s} & -\dfrac{2r_s \hat{Q}_s^+}{\underline{v}_s} \\ -\dfrac{2x_s \hat{P}_s^+}{\underline{v}_s} & 1 - \dfrac{2x_s \hat{Q}_s^+}{\underline{v}_s} \end{bmatrix} \cdots \begin{bmatrix} 1 - \dfrac{2r_{t-1} \hat{P}_{t-1}^+(\overline{p})}{\underline{v}_{t-1}} & -\dfrac{2r_{t-1} \hat{Q}_{t-1}^+(\overline{q})}{\underline{v}_{t-1}} \\ -\dfrac{2x_{t-1} \hat{P}_{t-1}^+(\overline{p})}{\underline{v}_{t-1}} & 1 - \dfrac{2x_{t-1} \hat{Q}_{t-1}^+(\overline{p})}{\underline{v}_{t-1}} \end{bmatrix} \begin{bmatrix} r_t \\ x_t \end{bmatrix} > 0$$

for $s = 1, \ldots, t$, i.e., $\underline{A}_s \cdots \underline{A}_{t-1} u_t > 0$ for $s = 1, \ldots, t$. This completes the proof of Claim 6.18. $\square$

Chapter 7

Exact Convex Relaxation for Single-Phase Direct Current Networks

We study the OPF problem in direct current (DC) networks in this chapter. An SOCP relaxation is considered for solving the OPF problem. We prove that the SOCP relaxation is exact if either (1) voltage upper bounds do not bind; or (2) voltage upper bounds are uniform and power injection lower bounds are negative. Based on (1), a modified OPF problem is proposed, whose corresponding SOCP is guaranteed to be exact. We also prove that SOCP has at most one optimal solution if it is exact. Finally, we discuss how to improve numerical stability and how to include line constraints.

7.1 Introduction

Direct current (DC) networks (e.g., DC-microgrids) have the following advantages over alternative current (AC) networks [61,89,90]. 1) Some devices, e.g., photovoltaic panels, wind turbines, electric vehicles, electronic appliances, and fuel cells, are more easily integrated with DC networks than AC networks. These devices are either DC in nature or have a different frequency than the main grid. 2) DC microgrids are robust to voltage sags and frequency deviations in the main grid. This is because DC voltages are easy to stabilize and there is no frequency synchronization for DC networks. 3) System efficiency can be higher for DC networks because conversion losses of inverters can be avoided. This is why modern data centers use DC networks.

The optimal power flow (OPF) problem determines power generations/demands that minimize certain objectives such as generation cost or power loss [24]. It is one of the fundamental problems in power system operations. This paper focuses on the OPF problem in DC networks.

The OPF problem is difficult to solve since power flow is governed by nonlinear physical laws. There are three approaches to deal with this challenge: 1) approximate the power flow equations (by linear or easier nonlinear equations); 2) look for local optima of the OPF problem; and 3) convexify the constraints imposed by nonlinear power flow laws. After a brief introduction of the first two approaches, we will focus on the third approach.

Power flow equations can be approximated by some linear equations[1] if 1) power losses on the lines are small; 2) voltages are close to their nominal values; and 3) voltage angle differences between adjacent buses are small. With the linear power flow approximation, the OPF problem reduces to a linear programming [94]. For transmission networks, the three assumptions are satisfied and the approach is widely used in practice. However, the linear power flow approximation does not consider voltages and reactive power flows, and therefore cannot be used for applications like voltage regulation and volt/var control. Besides, the obtained solution may not be implementable since physical laws are not fully respected. Moreover, for distribution networks, power losses on the lines are not negligible and voltages can deviate significantly from their nominal values. Consequently, the linear power flow approximation is not accurate enough for distribution networks.

A number of algorithms look for local optima of the OPF problem. These algorithms use nonlinear power flow equations and therefore 1) can be used in applications like voltage regulation and volt/var control; 2) have physically implementable solutions; 3) apply to both transmission and distribution networks. Representative algorithms of this kind include successive linear/quadratic programming [32], trust-region based methods [78], Lagrangian Newton method [11], and interior-point methods [100]. Some of these algorithms, especially those based on Newton-Ralphson, are quite successful empirically. However, these algorithms may not converge to global optimal solutions.

The convexification approach is the focus of this paper. The idea is to optimize the OPF objective over a convex superset of the OPF feasible set (which is nonconvex). The resulting optimization problem, referred to as a convex relaxation, can be solved much more efficiently. Furthermore, if the optimal solution of a convex relaxation lies in the OPF feasible set, then it must solve the OPF problem. In such cases, the convex relaxation is called *exact*.

There are three central problems in pursuing the convexification approach:

1. *Exact relaxation:* When can a global optimum of the OPF problem be obtained by solving its relaxation?

2. *Efficient computation:* How to design computationally efficient algorithms that scale to large problem sizes?

3. *Numerical stability:* How to attain numerical stability especially for ill-conditioned problem instances?

Significant effort has been devoted in the literature to address Problem 1), and sufficient conditions have been derived to guarantee the exactness of the SDP relaxation for special networks such as radial

[1]Known as "DC" linear power flow equations. But this "DC" does not refer to direct current as described in this paper.

networks and DC networks; see [73, 74] for a tutorial with extensive references. To address Problem 2), [9,58] exploit graph sparsity to simplify the SDP relaxation based on the chordal extension theory [40, 82, 83]. One of the main difficulties is to choose a chordal extension of the network graph that minimizes the resulting problem size, and effective techniques have been proposed in [79]. Another direction to deal with computational complexity is through distributed algorithms. To address Problem 3), note that the SDP relaxation requires taking differences of voltages at neighboring buses. These voltages are close in practice and therefore taking their differences is numerically unstable. The SOCP relaxation proposed in [36] for radial networks avoids such subtractions and has an improved numerically stability.

Summary of contributions: The goal of this paper is to propose a convex relaxation of the OPF problem for DC networks, study its exactness, and improve its numerical stability. In particular, contributions of this paper are threefold.

First, we propose a second-order cone programming (SOCP) relaxation of the OPF problem for DC networks. The SOCP relaxation exploits network sparsity to improve computational efficiency of the standard SDP relaxation, but is less likely to be exact than the SDP relaxation for mesh networks [17].

Second, we prove that the SOCP relaxation is exact if either 1) voltage upper bounds do not bind; or 2) voltage upper bounds are uniform and power injection lower bounds are negative. In a DC microgrid, voltage upper bounds do not bind if there are no distributed generators, and are usually uniform. Besides, power injection lower bounds are nonpositive if generators are allowed to be turned off. Based on 1), we impose additional constraints on the OPF problem such that its SOCP relaxation is always exact. These constraints restrict power injections such that voltage upper bounds do not bind.

Third, we improve numerical stability of the SOCP relaxation by adopting alternative variables. The SOCP relaxation is ill-conditioned since it requires subtractions of numerically close voltages. By adopting different variables, such subtractions can be avoided and numerical stability can be improved.

The rest of the chapter is organized as follows. The OPF problem is formulated in Section 7.2 and an SOCP relaxation is introduced in Section 7.3. Section 7.4 provides sufficient conditions for the exactness of the SOCP relaxation, and Section 7.5 proposes a modified OPF problem that always has an exact SOCP relaxation. Section 7.6 describes how to improve numerical stability and how to include line constraints, and Section 7.7 provides numerical studies.

7.2 The Optimal Power Flow Problem

This paper studies the optimal power flow (OPF) problem in direct current (DC) networks, and is applicable for demand response and voltage regulation. In the following we present a model that incorporates nonlinear power flow.

7.2.1 Power Flow Model

A DC network is composed of buses and lines connecting these buses. It can be either radial or mesh. There is a swing bus in the network with a fixed voltage. Index the swing bus by 0 and the other buses by $1, \ldots, n$. Let $\mathcal{N} := \{0, \ldots, n\}$ denote the collection of all buses and define $\mathcal{N}^+ := \mathcal{N} \backslash \{0\}$. Each line connects a pair $\{i, j\}$ of buses. Let $\mathcal{E}$ denote the collection of all lines and abbreviate $\{i, j\} \in \mathcal{E}$ by $i \sim j$.

For each bus $i \in \mathcal{N}$, let V_i denote its voltage, I_i denote its current injection, and p_i denote its power injection. For each line $i \sim j$, let y_{ij} denote its admittance, I_{ij} denote the current from bus i to bus j, and define $z_{ij} := 1/y_{ij}$. In a DC network, V_i, I_i, p_i, y_{ij}, z_{ij}, and I_{ij} are all real numbers.

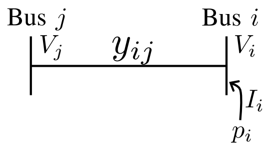


Figure 7.1: Summary of notations.

Some notations are summarized in Fig. 7.1. Further, we use a letter without subscripts to denote a vector of the corresponding quantities, e.g., $V = [V_i]_{i \in \mathcal{N}}$, $y = [y_{ij}]_{i \sim j}$.

Power flows are governed by the following physical laws:

- Ohm's Law: $I_{ij} = y_{ij}(V_i - V_j)$ for $\{i, j\} \in \mathcal{E}$;
- Current balance: $I_i = \sum_{j:\, j \sim i} I_{ij}$ for $i \in \mathcal{N}$;
- Power balance: $p_i = V_i I_i$ for $i \in \mathcal{N}$.

By eliminating current variables, one obtains

$$p_i = V_i \sum_{j:\, j \sim i} (V_i - V_j) y_{ij}, \quad i \in \mathcal{N}. \tag{7.1}$$

We use (7.1) to model the power flow in this paper.

7.2.2 The Optimal Power Flow Problem

The OPF problem determines power injection p that minimizes the total generation cost, subject to physical and operational constraints.

The total generation cost is assumed separable. In particular, let $f_i(p_i) : \mathbb{R} \to \mathbb{R}$ denote the generation cost of bus i for $i \in \mathcal{N}$. Then the total generation cost is

$$C(p) = \sum_{i\in\mathcal{N}} f_i(p_i). \tag{7.2}$$

Note that if $f_i(x) = x$ for $i \in \mathcal{N}$, then (7.2) reduces to the total power loss.

Besides the physical power flow constraint (7.1), the OPF problem has operational constraints on power injections and voltages.

First, while the substation power injection p_0 is unconstrained, the power injection p_i of a branch bus $i \in \mathcal{N}^+$ can only vary within some externally specified set $\mathcal{P}_i$:

$$p_i \in \mathcal{P}_i, \quad i \in \mathcal{N}^+. \tag{7.3}$$

For example, if bus i represents an inelastic load with power demand d_i, then $\mathcal{P}_i$ is a singleton

$$\mathcal{P}_i = \{-d_i\};$$

if bus i represents a controllable load that can be turned on and off (while it is turned on, it consumes d_i amount of power), then $\mathcal{P}_i$ contains two distinct points

$$\mathcal{P}_i = \{0, -d_i\};$$

if bus i represents a generator that can generate any amount of power between 0 and its capacity C_i, then $\mathcal{P}_i$ is an interval

$$\mathcal{P}_i = [0, C_i].$$

Note that the set $\mathcal{P}_i$ can be nonconvex.

Second, the substation voltage V_0 is fixed and given (denote by $V_0^{\text{ref}} > 0$), and the magnitudes of branch bus voltages need to be regulated within a narrow range, i.e., there exists $\underline{V}_i$ and $\overline{V}_i$ for $i \in \mathcal{N}^+$ such that

$$V_0 = V_0^{\text{ref}}; \tag{7.4a}$$

$$\underline{V}_i \le V_i \le \overline{V}_i, \quad i \in \mathcal{N}^+. \tag{7.4b}$$

For example, if voltages must not deviate by over 5% from their nominal values, then $0.95 \le V_i \le 1.05$ per unit [3].

There are other constraints in a real-world OPF problem, e.g., line constraints and security constraints. How to include line constraints will be discussed in Section 7.6.2. In DC microgrids, line constraints typically do not bind since distribution networks are often over-provisioned. Security constraints are ignored for simplicity.

To summarize, the OPF problem can be formulated as

$$\begin{aligned}
\textbf{OPF:}\ \min\ & \sum_{i\in\mathcal{N}} f_i(p_i) \\
\text{over}\ & p, V \\
\text{s.t.}\ & p_i = V_i \sum_{j:\, j\sim i} (V_i - V_j) y_{ij}, \quad i \in \mathcal{N}; \\
& p_i \in \mathcal{P}_i, \quad i \in \mathcal{N}^+; \\
& V_0 = V_0^{\text{ref}}; \\
& \underline{V}_i \le V_i \le \overline{V}_i, \quad i \in \mathcal{N}^+.
\end{aligned}$$

The following assumptions are made throughout this work.

A1) The network $(\mathcal{N}, \mathcal{E})$ is connected.

A2) Line admittance $y_{ij} > 0$ for $\{i,j\} \in \mathcal{E}$. In practice, $y > 0$ since lines are lossy.

A3) Voltage lower bound $\underline{V}_i > 0$ for $i \in \mathcal{N}^+$. In practice, $\underline{V}$ is slightly below 1.

7.3 An SOCP Relaxation

A second-order cone programming (SOCP) relaxation has been proposed to solve the OPF problem for radial networks [91]. We propose using it to solve the OPF problem for DC networks, which can be mesh.

The SOCP relaxation seeks to overcome the nonconvexity in (7.1). It is derived through two steps: (a) transform OPF to shift the nonconvexity in (7.1) to a rank constraint, and (b) remove the rank constraint.

Transform OPF: Introduce slack variables

$$v_i = V_i^2, \quad i \in \mathcal{N}; \tag{7.5a}$$

$$W_{ij} = V_i V_j, \quad i \sim j. \tag{7.5b}$$

Then, (7.1) is transformed to a linear equality constraint

$$p_i = \sum_{j:\, j \sim i} (v_i - W_{ij}) y_{ij}, \quad i \in \mathcal{N}$$

in (p, v, W). For each line $i \sim j$ where $i < j$, the 2×2 matrix

$$\begin{bmatrix} v_i & W_{ij} \\ W_{ji} & v_j \end{bmatrix} = \begin{bmatrix} V_i \\ V_j \end{bmatrix} \begin{bmatrix} V_i & V_j \end{bmatrix}$$

is rank one (assuming $V_i \neq 0$) and positive semidefinite.

The following lemma provides the theoretical foundation of transforming OPF. Let

$$\begin{aligned} A \succeq 0 \quad &\overset{\text{def}}{\Longleftrightarrow} \quad A \text{ is positive semidefinite,} \\ i \to j \quad &\overset{\text{def}}{\Longleftrightarrow} \quad i \sim j \ \&\ i < j, \\ x \in \mathbb{R}^+ \quad &\overset{\text{def}}{\Longleftrightarrow} \quad x \geq 0. \end{aligned}$$

Lemma 7.1. *Given $v_i > 0$ for $i \in \mathcal{N}$ and $W_{ij} \in \mathbb{R}^+$ for $i \to j$, let $W_{ji} = W_{ij}$ for $i \to j$. If*

$$\text{rank} \begin{bmatrix} v_i & W_{ij} \\ W_{ji} & v_j \end{bmatrix} \leq 1$$

for $i \to j$, then there exists a unique V that satisfies $V_0 = \sqrt{v_0}$ and (7.5). Furthermore, such V is given by

$$V_i = \sqrt{v_i}, \quad i \in \mathcal{N}.$$

The lemma is proved in Appendix 7.A.

Lemma 7.1 immediately implies that OPF is equivalent to

$$
\begin{aligned}
\textbf{OPF':}\ \min\quad & \sum_{i\in\mathcal{N}} f_i(p_i) \\
\text{over}\quad & p_i \in \mathbb{R},\ v_i \in \mathbb{R} \text{ for } i \in \mathcal{N}; \\
& W_{ij} \in \mathbb{R}^+ \text{ for } i \sim j, \\
\text{s.t.}\quad & p_i = \sum_{j:\, j\sim i} (v_i - W_{ij}) y_{ij}, \quad i \in \mathcal{N}; && (7.6a) \\
& p_i \in \mathcal{P}_i, \quad i \in \mathcal{N}^+; && (7.6b) \\
& v_0 = [V_0^{\text{ref}}]^2; && (7.6c) \\
& \underline{V}_i^2 \le v_i \le \overline{V}_i^2, \quad i \in \mathcal{N}^+; && (7.6d) \\
& W_{ij} = W_{ji}, \quad i \to j; && (7.6e) \\
& \begin{bmatrix} v_i & W_{ij} \\ W_{ji} & v_j \end{bmatrix} \succeq 0, \quad i \to j; && (7.6f) \\
& \operatorname{rank} \begin{bmatrix} v_i & W_{ij} \\ W_{ji} & v_j \end{bmatrix} = 1, \quad i \to j. && (7.6g)
\end{aligned}
$$

Note that the nonconvexity in (7.1) (in OPF) is transformed to the nonconvexity in (7.6g) (in OPF').

Remove rank constraint: The following SOCP relaxation can be obtained by removing the nonconvex rank constraint (7.6g) in OPF'.

$$
\begin{aligned}
\textbf{SOCP:}\ \min\quad & \sum_{i\in\mathcal{N}} f_i(p_i) \\
\text{over}\quad & p, v, W \\
\text{s.t.}\quad & (7.6a) - (7.6f).
\end{aligned}
$$

Note that SOCP may not be convex since f_i (in the objective) and $\mathcal{P}_i$ (in (7.6b)) may not be convex. Nonetheless, we call it second-order cone programming for convenience.

Exact SOCP relaxation: If an optimal SOCP solution (p, v, W) satisfies (7.6g), then (p, v, W) also solves OPF'. Furthermore, compute V as

$$V_i \leftarrow \sqrt{v_i}, \quad i \in \mathcal{N},$$

then it can be shown that (p, V) solves OPF. This motivates the definition of an *exact* SOCP relaxation as follows.

Definition 7.1. *SOCP is* exact*, provided that every optimal SOCP solution satisfies* (7.6g)*.*

When SOCP is exact, one can obtain a global optimum of the nonconvex OPF problem by solving a convex SOCP program (assuming f_i and $\mathcal{P}_i$ are convex).

Related work: An SDP relaxation has been proposed in literature via the same two steps: transformation and relaxation [10]. In the transformation step, slack variable

$$\tilde{W} := \begin{bmatrix} V_0 \\ \vdots \\ V_n \end{bmatrix} \begin{bmatrix} V_0 & \cdots & V_n \end{bmatrix}$$

is introduced and the nonconvexity in (7.1) (in OPF) is transformed to the nonconvexity in

$$\text{rank}\tilde{W} = 1.$$

In the relaxation step, the rank constraint $\text{rank}\tilde{W} = 1$ is removed, but a positive semidefinite constraint $\tilde{W} \succeq 0$ needs to be kept. We refer to this relaxation as SDP hereafter.

SDP enlarges the feasible set of OPF to a smaller convex superset than that of SOCP, and is therefore more likely to be exact [17]. However, we propose SOCP over SDP for DC networks for the following two reasons:

a) SOCP is much more efficient to compute than SDP;

b) SOCP is exact under existing conditions that guarantee the exactness of SDP.

To demonstrate (a), note that SDP introduces an $(n+1) \times (n+1)$ matrix $\tilde{W}$ and therefore the number of variables in SDP is $O(n^2)$. For a given set A, let

$$|A| \quad \overset{\text{def}}{\Longleftrightarrow} \quad \text{number of elements in } A.$$

SOCP introduces $|\mathcal{E}|$ 2×2 matrices and therefore the number of variables in SOCP is $O(|\mathcal{E}|)$. Power networks are usually sparse, i.e., $|\mathcal{E}| \ll n^2$. Hence, SOCP has fewer optimization variables than SDP and is therefore more efficient.

To demonstrate (b), we review existing conditions that guarantee the exactness of SDP/SOCP. The conditions are summarized in Propositions 7.2 and 7.3, and follow directly from a more general result in [64, Theorem 3.1].

Proposition 7.2 ([66])**.** *If there exists $\overline{p}_i$ such that $\mathcal{P}_i = (-\infty, \overline{p}_i]$ for $i \in \mathcal{N}^+$, and f_i is strictly increasing for $i \in \mathcal{N}$, then SDP is exact.*

Proposition 7.3 ([67])**.** *If there exists $\overline{p}_i$ such that $\mathcal{P}_i = (-\infty, \overline{p}_i]$ for $i \in \mathcal{N}^+$, and f_i is strictly increasing for $i \in \mathcal{N}$, then SOCP is exact.*

The conditions in Propositions 7.2 and 7.3 are the same, which completes the demonstration of (b).

7.4 Sufficient Conditions for Exact Relaxation

Two sufficient conditions that guarantee the exactness of SOCP are provided in this section. One condition (Theorem 7.4) requires nonbinding voltage upper bounds, and the other condition (Theorem 7.5) requires uniform voltage upper bound.

Theorem 7.4. *SOCP is exact provided that*

- $\overline{V}_i = \infty$ *for* $i \in \mathcal{N}^+$*;*
- f_0 *is strictly increasing.*

Theorem 7.4 is proved in Appendix 7.B. Note that voltage upper bounds do not bind if there are no distributed generators like photovoltaic panels.

Theorem 7.4 still holds if (7.6b) is generalized to

$$(p_1, \ldots, p_n) \in \mathcal{P} \tag{7.6b'}$$

where $\mathcal{P}$ can be arbitrary, since the proof of Theorem 7.4 does not require any structure on $\mathcal{P}$.

Theorem 7.5. *SOCP is exact provided that*

- $0 < V_0^{\text{ref}} \le \overline{V}_1 = \cdots = \overline{V}_n$*;*
- *there exists* $\underline{p}_i$, $\overline{p}_i$ *such that* $\underline{p}_i < 0$ *and* $\mathcal{P}_i = [\underline{p}_i, \overline{p}_i]$ *for* $i \in \mathcal{N}^+$*;*
- f_i *is strictly increasing for* $i \in \mathcal{N}$*.*

Theorem 7.5 implies that if voltage upper bounds are uniform and (7.6b) is a collection of box constraints with negative lower bounds, then SOCP is exact. It is proved in Appendix 7.C. Voltage upper bounds are usually uniform for distribution networks. If SOCP is convex with a closed feasible set, then there exists $\underline{p}_i, \overline{p}_i \in \mathbb{R} \cup \{\pm\infty\}$ such that $\mathcal{P}_i = [\underline{p}_i, \overline{p}_i]$ for $i \in \mathcal{N}^+$. Further, $\underline{p}_i \le 0$ if generators can be turned off.

Theorem 7.6. *If SOCP is convex and exact, then it has at most one optimal solution.*

Theorem 7.6 implies that if f_i and $\mathcal{P}_i$ are convex, and SOCP is exact, then SOCP has at most one optimal solution. It is proved in Appendix 7.D, and still holds if (7.6b) is generalized to (7.6b') with $\mathcal{P}$ being convex.

7.5 A Modified OPF Problem

A modified OPF problem that always has an exact SOCP relaxation is proposed in this section.

The modified OPF problem is motivated by Theorem 7.4. The idea is to impose additional constraints on $(p_1, \ldots, p_n)$ such that $v_i \leq \overline{V}_i^2$ in (7.6d) do not bind and therefore $\overline{V}$ is effectively ∞. More specifically, an affine function $\hat{v}_i(p_1, \ldots, p_n)$ that upper bounds v_i is derived. An additional constraint

$$\hat{v}_i(p_1, \ldots, p_n) \leq \overline{V}_i^2, \quad i \in \mathcal{N}^+ \tag{7.7}$$

is imposed on OPF' such that $v_i \leq \overline{V}_i^2$ does not bind.

7.5.1 Derive $\hat{v}_i(p_1, \ldots, p_n)$

First derive the affine functions $\hat{v}_i(p_1, \ldots, p_n)$. Let

$$P_{ij} := (v_i - W_{ij})y_{ij}, \qquad i \sim j \tag{7.8}$$

denote the sending-end power flow from bus i to bus j, and

$$\ell_{ij} := |I_{ij}|^2, \qquad i \to j \tag{7.9}$$

denote the magnitude square of the current on $i \sim j$, then

$$p_i = \sum_{j:\, j \sim i} P_{ij}, \quad i \in \mathcal{N}^+; \tag{7.10a}$$

$$P_{ij} + P_{ji} = z_{ij}\ell_{ij}, \quad i \to j; \tag{7.10b}$$

$$v_i - v_j = z_{ij}(P_{ij} - P_{ji}), \quad i \to j. \tag{7.10c}$$

Given the swing bus voltage v_0, branch bus power injection $(p_1, \ldots, p_n)$, and line current ℓ, then (7.10) is a collection of $n + 2|\mathcal{E}|$ linear equations on $n + 2|\mathcal{E}|$ variables $v_1, \ldots, v_n$ and P_{ij} for $i \sim j$.

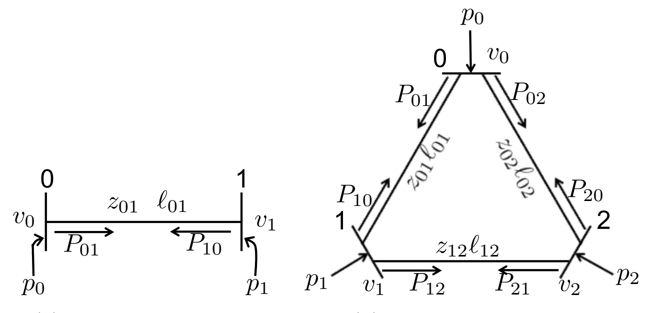
Lemma 7.7. *Given v_0, p_i for $i \in \mathcal{N}^+$, and ℓ_{ij} for $i \to j$. There exists a unique $([P_{ij}]_{i\sim j}, [v_i]_{i\in\mathcal{N}^+})$ that satisfies* (7.10a)–(7.10c)*.*

Lemma 7.7 implies that P_{ij} and v_i are linear functions in $(v_0, [p_i]_{i\in\mathcal{N}^+}, [\ell_{ij}]_{i\to j})$. It is proved in Appendix 7.E.

Definition 7.2. *Given v_0, denote the unique solution $([P_{ij}]_{i\sim j}, [v_i]_{i\in\mathcal{N}^+})$ to* (7.10a)–(7.10c) *as a* *function of* $([p_i]_{i\in\mathcal{N}^+}, [\ell_{ij}]_{i\to j})$ *by*

$$\hat{P}_{ij}\left([p_i]_{i\in\mathcal{N}^+}, [\ell_{ij}]_{i\to j}\right), \quad i\sim j;$$
$$\hat{v}_i\left([p_i]_{i\in\mathcal{N}^+}, [\ell_{ij}]_{i\to j}\right), \quad i\in\mathcal{N}^+.$$

Two examples, one for a two-bus network (in Fig. 7.1(a)) and one for a three-bus network (in Fig. 7.1(b)), are used to illustrate $\hat{P}_{ij}$ and $\hat{v}_i$.



(a) A two-bus network (b) A three-bus network

Figure 7.1: Two example networks.

Example 7.1. *For the two-bus network in Fig. 7.1(a),* (7.10) *is*

$$p_1 = P_{10},\ P_{01} + P_{10} = z_{01}\ell_{01},$$
$$v_0 - v_1 = z_{01}(P_{01} - P_{10}).$$

Given v_0*, the affine functions* $\hat{P}_{01}, \hat{P}_{10}, \hat{v}_1$ *are*

$$\begin{aligned}\hat{P}_{10}(p_1,\ell_{01}) &= p_1,\\ \hat{P}_{01}(p_1,\ell_{01}) &= z_{01}\ell_{01} - P_{10},\\ \hat{v}_1(p_1,\ell_{01}) &= v_0 + 2z_{01}p_1 - z_{01}^2\ell_{01}.\end{aligned}$$

Example 7.2. *For the three-bus network in Fig. 7.1(b),* (7.10) *is*

$$\begin{aligned}&p_1 = P_{10} + P_{12},\ p_2 = P_{20} + P_{21},\\ &P_{01} + P_{10} = z_{01}\ell_{01},\ v_0 - v_1 = z_{01}(P_{01} - P_{10}),\\ &P_{02} + P_{20} = z_{02}\ell_{02},\ v_0 - v_2 = z_{02}(P_{02} - P_{20}),\\ &P_{12} + P_{21} = z_{12}\ell_{12},\ v_1 - v_2 = z_{12}(P_{12} - P_{21}).\end{aligned}$$

Assume $z_{01} = z_{02} = z_{12} = 0.01$ *for brevity. Given* v_0*, abbreviate* (p_1, p_2) *by* p *and* $(\ell_{01}, \ell_{02}, \ell_{12})$ *by*

ℓ, then the affine functions $\hat{P}_{01}$, $\hat{P}_{10}$, $\hat{P}_{02}$, $\hat{P}_{20}$, $\hat{P}_{12}$, $\hat{P}_{21}$, $\hat{v}_1$, $\hat{v}_2$ are

$$\begin{aligned}
\hat{P}_{01}(p,\ell) &= -\frac{2}{3}p_1 - \frac{1}{3}p_2 + \frac{5}{600}\ell_{01} + \frac{1}{600}\ell_{02} + \frac{1}{200}\ell_{12},\\
\hat{P}_{10}(p,\ell) &= \frac{2}{3}p_1 + \frac{1}{3}p_2 + \frac{1}{600}\ell_{01} - \frac{1}{600}\ell_{02} - \frac{1}{200}\ell_{12},\\
\hat{P}_{02}(p,\ell) &= -\frac{1}{3}p_1 - \frac{2}{3}p_2 + \frac{1}{600}\ell_{01} + \frac{5}{600}\ell_{02} + \frac{1}{200}\ell_{12},\\
\hat{P}_{20}(p,\ell) &= \frac{1}{3}p_1 + \frac{2}{3}p_2 - \frac{1}{600}\ell_{01} + \frac{1}{600}\ell_{02} - \frac{1}{200}\ell_{12},\\
\hat{P}_{12}(p,\ell) &= \frac{1}{3}p_1 - \frac{1}{3}p_2 - \frac{1}{600}\ell_{01} + \frac{1}{600}\ell_{02} + \frac{1}{200}\ell_{12},\\
\hat{P}_{21}(p,\ell) &= -\frac{1}{3}p_1 + \frac{1}{3}p_2 + \frac{1}{600}\ell_{01} - \frac{1}{600}\ell_{02} + \frac{1}{200}\ell_{12},\\
\hat{v}_1(p,\ell) &= v_0 + \frac{1}{75}p_1 + \frac{1}{150}p_2 - \frac{1}{15000}\ell_{01} - \frac{1}{30000}\ell_{02} - \frac{1}{10000}\ell_{12},\\
\hat{v}_2(p,\ell) &= v_0 + \frac{1}{150}p_1 + \frac{1}{75}p_2 - \frac{1}{30000}\ell_{01} - \frac{1}{15000}\ell_{02} - \frac{1}{10000}\ell_{12}.
\end{aligned}$$

The following lemma shows that $\hat{v}$ is decreasing in ℓ. Let the operator $\geq$ denote componentwise.

Lemma 7.8. *If $\ell \geq \ell'$, then $\hat{v}_i(p,\ell) \leq \hat{v}_i(p,\ell')$ for $i \in \mathcal{N}^+$.*

Lemma 7.8 implies that $\hat{v}_i(p,\ell)$ is decreasing in ℓ. The lemma is proved in Appendix 7.F. In Examples 7.1 and 7.2, it can be seen that the coefficients of ℓ in $\hat{v}$ are negative.

Since current magnitude square $\ell \geq 0$, one obtains

$$\hat{v}_i(p,\ell) \leq \hat{v}_i(p,0), \quad i \in \mathcal{N}^+.$$

The left hand side is the real voltage v_i, and the right hand side is the affine function in p that we aim for.

Definition 7.3. *Define affine functions $\hat{v}_i\left([p_i]_{i\in\mathcal{N}^+}\right)$ as*

$$\hat{v}_i\left([p_i]_{i\in\mathcal{N}^+}\right) = \hat{v}_i\left([p_i]_{i\in\mathcal{N}^+}, [\ell_{ij}]_{i\to j}\right)\Big|_{\ell=0}, \quad i \in \mathcal{N}^+.$$

In Example 7.1,

$$\hat{v}_1(p_1) = v_0 + 2z_{01}p_1.$$

In Example 7.2,

$$\begin{aligned}
\hat{v}_1(p_1,p_2) &= v_0 + \frac{1}{75}p_1 + \frac{1}{150}p_2,\\
\hat{v}_2(p_1,p_2) &= v_0 + \frac{1}{150}p_1 + \frac{1}{75}p_2.
\end{aligned}$$

As having been discussed, $\hat{v}_i(p)$ upper bounds v_i.

Corollary 7.9. *Let (p, v, W) be feasible for SOCP, then $v_i \leq \hat{v}_i(p)$ for $i \in \mathcal{N}^+$.*

The corollary is proved in Appendix 7.G.

7.5.2 Impose Additional Constraint

If additional constraint (7.7) is imposed on SOCP, then it follows from Corollary 7.9 that the constraints $v_i \le \overline{V}_i^2$ in (7.6d) do not bind, and therefore $\overline{V}_i$ is effectively ∞. To summarize, the modified OPF problem is

$$\begin{aligned}
\textbf{OPF'-m:}\ \min\ & \sum_{i\in\mathcal{N}} f_i(p_i) \\
\text{over}\ & p, v, W \\
\text{s.t.}\ & (7.6\text{a}) - (7.6\text{c}); \\
& \underline{V}_i^2 \le v_i,\ \hat{v}_i(p) \le \overline{V}_i^2, \quad i \in \mathcal{N}^+; \qquad (7.11) \\
& (7.6\text{e}) - (7.6\text{g}).
\end{aligned}$$

Removing rank constraint (7.6g) gives the following relaxation

$$\begin{aligned}
\textbf{SOCP-m:}\ \min\ & \sum_{i\in\mathcal{N}} f_i(p_i) \\
\text{over}\ & p, v, W \\
\text{s.t.}\ & (7.6\text{a}) - (7.6\text{c}),\ (7.11),\ (7.6\text{e}) - (7.6\text{f}).
\end{aligned}$$

Note that SOCP-m may not be convex since f_i (in the objective) and $\mathcal{P}_i$ (in (7.6b)) may not be convex. Nonetheless, we call it second-order cone programming for convenience.

Recall that Theorem 7.4 holds for the more general power injection constraint (7.6b'), and note that (7.7) is a special case of (7.6b'). It follows that SOCP-m is always exact.

Theorem 7.10. *SOCP-m is exact if f_0 is strictly increasing.*

Theorem 7.10 still holds if (7.6b) is generalized to (7.6b') with $\mathcal{P}$ being arbitrary.

7.6 Extensions

7.6.1 Improve Numerical Stability

SOCP is ill-conditioned since (7.6a) requires subtractions of numerically close v_i and W_{ij}. One can avoid such subtractions by adopting alternative variables to improve the numerical stability of

SOCP. In particular, adopt variables p, v, P, ℓ as in the following convex relaxation:

$$\begin{aligned}
\textbf{stable-SOCP:}\ \min\ & \sum_{i\in\mathcal{N}} f_i(p_i) \\
\text{over}\ & p_i \in \mathbb{R},\ v_i \in \mathbb{R} \text{ for } i \in \mathcal{N}; \\
& P_{ij} \in \mathbb{R} \text{ for } i \sim j,\ \ell_{ij} \in \mathbb{R} \text{ for } i \to j; \\
\text{s.t.}\ & p_i = \sum_{j:\, j\sim i} P_{ij}, \quad i \in \mathcal{N}; \\
& p_i \in \mathcal{P}_i, \quad i \in \mathcal{N}^+; \\
& v_0 = [V_0^{\text{ref}}]^2; \\
& \underline{V}_i^2 \le v_i \le \overline{V}_i^2, \quad i \in \mathcal{N}^+; \\
& P_{ij} + P_{ji} = z_{ij}\ell_{ij}, \quad i \to j; \\
& v_i - v_j = z_{ij}(P_{ij} - P_{ji}), \quad i \to j; \\
& \ell_{ij} \ge \frac{P_{ij}^2}{v_i}, \quad i \to j.
\end{aligned}$$

Theorem 7.11. *SOCP and stable-SOCP are equivalent, i.e., there exists a one-to-one map between the feasible set of SOCP and the feasible set of stable-SOCP.*

Let $\mathcal{F}_{\text{SOCP}}$ and $\mathcal{F}_{\text{stable-SOCP}}$ denote the feasible sets of SOCP and stable-SOCP, respectively. Then the map

$$f : (p, v, W) \mapsto (p, v, P, \ell)$$

given by

$$\begin{aligned}
P_{ij} &= (v_i - W_{ij})y_{ij}, \quad i \to j; \\
\ell_{ij} &= y_{ij}^2(v_i - W_{ij} - W_{ji} + v_j), \quad i \to j
\end{aligned}$$

can be verified to be one-to-one from $\mathcal{F}_{\text{SOCP}}$ to $\mathcal{F}_{\text{stable-SOCP}}$.

7.6.2 Include Line Constraints

Noting that line constraints are not considered in the main text, we discuss how to include line constraints in this section.

Line constraints impose that line currents should not exceed certain thresholds, i.e., there exists $\overline{I}_{ij}$ for $i \to j$ such that

$$|I_{ij}| \le \overline{I}_{ij}.$$

It can be considered by adding constraints

$$y_{ij}^2(v_i - W_{ij} - W_{ji} + v_j) \le \overline{I}_{ij}^2, \quad i \to j$$

to SOCP/SOCP-m, or adding constraints

$$\ell_{ij} \le \overline{I}_{ij}^2, \quad i \to j$$

to stable-SOCP. But Theorems 7.4, 7.5, and 7.10 do not apply after adding these constraints.

One way to maintain some of the theoretical guarantees is to impose the line constraints in terms of power flows instead. In particular, $|I_{ij}| \le \overline{I}_{ij}$ is equivalent to $|P_{ij}| \le V_i\overline{I}_{ij}$. Assuming that V_i is close to its nominal value, $|P_{ij}| \le V_i\overline{I}_{ij}$ can be approximated by $|P_{ij}| \le \overline{P}_{ij}$ for some $\overline{P}_{ij} \in \mathbb{R}$. Since $\hat{P}_{ij}(p)$ provides an approximation of P_{ij}, $|P_{ij}| \le \overline{P}_{ij}$ can be further approximated by $|\hat{P}_{ij}(p)| \le \overline{P}_{ij}$.

Hence, one can impose

$$|\hat{P}_{ij}(p)| \le \overline{P}_{ij}, \quad i \to j \tag{7.12}$$

as an approximation of the line constraints to SOCP/SOCP-m/stable-SOCP. Since (7.12) is a constraint on p, Theorem 7.4 and 7.10 still hold after imposing the approximated line constraints (7.12).

7.7 Case Study

We empirically evaluate the exactness and computational efficiency of SOCP in this section. All simulations are done on a laptop with Intel Core 2 Duo CPU at 2.66GHz, 4G RAM, and Mac OS X 10.7.5.

More specifically, we check whether SOCP is exact, and compare its computation time with that of the SDP relaxation proposed in [10], for several test networks. SOCP and SDP are solved via *CVX* [49], and the test networks are modified from the matlab toolbox *matpower* by ignoring line reactances and reactive power flows. The results are summarized in Table 7.1.

Table 7.1: Exactness and computational efficiency of SOCP.

network	SDP time	SOCP time	ratio
case6ww	1.1s	1.1s	3.4e-13
case9	1.2s	1.4s	9.6e-10
case14	1.7s	1.3s	1.3e-9
case_ieee30	1.4s	1.1s	2.1e-8
case39	1.4s	1.2s	7.9e-12

The first column of Table 7.1 lists where the network data comes from. In particular, it provides the names of the ".m" files where the network data is stored (these files can be found in the folder of the matlab toolbox *matpower*). For example, the data for a 6-bus network is stored in file "case6ww.m", and the data for a 9-bus network is stored in file "case9.m".

For each network, the following numbers are presented:

1. SDP time: the computation time of SDP.

2. SOCP time: the computation time of SOCP.

3. ratio: used to quantify the exactness of SOCP.

The "ratio" column quantifies how numerically exact SOCP is. At a numerical SOCP solution (p, v, W), which can be slightly different from the real SOCP solution (p^*, v^*, W^*), a 2×2 matrix

$$W\{i,j\} = \begin{bmatrix} v_i & W_{ij} \\ W_{ji} & v_j \end{bmatrix}$$

can be obtained for each line $i \to j$. Let λ_{ij}^1, λ_{ij}^2 denote its two eigenvalues and assume $|\lambda_{ij}^1| \geq |\lambda_{ij}^2| \geq 0$.

Assume SOCP is exact, i.e., $\text{rank}(W^*\{i,j\}) = 1$ for $i \to j$. If there are infinite digits of precision, i.e., $(p, v, W) = (p^*, v^*, W^*)$, then $\text{rank}(W\{i,j\}) = \text{rank}(W^*\{i,j\}) = 1$ and therefore $\lambda_{ij}^2 = 0$ for $i \to j$. It follows that the ratio $|\lambda_{ij}^2|/|\lambda_{ij}^1| = 0$.

Due to finite digits of precision, the ratio $|\lambda_{ij}^2|/|\lambda_{ij}^1|$ is not exactly 0. The smaller ratio, the closer is $W\{i,j\}$ to rank one. And the column "ratio" lists upper bounds on the ratios $|\lambda_{ij}^2|/|\lambda_{ij}^1|$ over all $i \to j$. For example, for the 6-bus network specified in case6ww.m, the ratios $|\lambda_{ij}^2|/|\lambda_{ij}^1|$ are upper bounded by 3.4e-13.

It can be seen from the "ratio" column that SOCP is exact for all test networks. Furthermore, it can be seen from the "SDP time" and "SOCP time" columns that SOCP is more computationally efficient than SDP.

7.8 Conclusion

We have proposed an SOCP relaxation of the OPF problem for DC networks that is more computationally efficient than the standard SDP relaxation. We have proved that the SOCP relaxation is exact if either (1) voltage upper bounds do not bind, or (2) voltage upper bounds are uniform and power injections have box constraints with negative lower bounds. We have also proved that the SOCP relaxation has at most one optimal solution if it is convex and exact.

We have proposed a modified OPF problem that always has an exact SOCP relaxation. The modified OPF problem is motivated by (1) and obtained by imposing additional constraints on power injections such that voltage upper bounds do not bind.

We have discussed how to improve the numerical stability of SOCP—by adopting alternative variables to avoid ill-conditioned numerical operations. We have also discussed how to include line constraints: after adding some approximate line constraints, some of the theoretical guarantees remain unchanged.

Appendix

7.A Proof of Lemma 7.1

Existence Let $V_i = \sqrt{v_i}$ for $i \in \mathcal{N}$. It suffices to show that V satisfies $V_0 = \sqrt{v_0}$ and (7.5).

It is straightforward to check that V satisfies $V_0 = \sqrt{v_0}$ and (7.5a). The matrices

$$\begin{bmatrix} v_i & W_{ij} \\ W_{ji} & v_j \end{bmatrix}$$

are not full rank, and therefore

$$v_i v_j - W_{ij} W_{ji} = 0, \quad i \to j.$$

Since $W_{ij} \geq 0$, one has

$$\begin{aligned} W_{ij} &= \sqrt{W_{ij}^2} = \sqrt{W_{ij} W_{ji}} \\ &= \sqrt{v_i v_j} = V_i V_j \end{aligned}$$

for $i \sim j$, i.e., V satisfies (7.5b). This completes the proof of existence.

Uniqueness Let $\tilde{V}$ denote an arbitrary solution to $\tilde{V}_0 = \sqrt{v}_0$ and (7.5). It suffices to show that $\tilde{V}_i = \sqrt{v_i}$ for $i \in \mathcal{N}$.

Assume $\tilde{V}_i \neq \sqrt{v_i}$ for some $i \in \mathcal{N}$, then it follows from (7.5a) that $\tilde{V}_i = -\sqrt{v_i} < 0$. For any j such that $i \sim j$, one has $0 \leq W_{ij} = \tilde{V}_i \tilde{V}_j$ and therefore $\tilde{V}_j \leq 0$. It follows that $\tilde{V}_j < 0$ since $\tilde{V}_j^2 = v_j \neq 0$. Such propagation (when $\tilde{V}_i < 0$, one has $\tilde{V}_j < 0$ for all neighboring j) can continue and eventually one has $\tilde{V}_0 < 0$ since the network is connected. This contradicts with the assumption that $\tilde{V}_0 = \sqrt{v_0} \geq 0$. Hence, $\tilde{V}_i = \sqrt{v_i}$ for $i \in \mathcal{N}$, which completes the proof of uniqueness.

7.B Proof of Theorem 7.4

Assume the conditions in Theorem 7.4 hold. We will show that for any SOCP feasible point (p, v, W) that violates (7.6g), there exists another SOCP feasible point (p', v', W') that has a smaller objective value than (p, v, W). Hence, every SOCP solution must satisfy (7.6g), i.e., SOCP is exact.

Construction of (p', v', W') is based on Lemmas 7.12 and 7.13.

Lemma 7.12. *Let* (p, v, W) *be feasible for SOCP and violate* (7.6g) *on some* $i \to j$ *where* $i, j \neq 0$. *Then there exists* (p, v', W') *that*

- *satisfies* (7.6a), (7.6e), (7.6f);
- *satisfies*
$$v'_k \begin{cases} = v_k & k \neq i, j \\ > v_k & k = i, j; \end{cases}$$
- *violates* (7.6g) *for all* $k \to l$ *such that* $\{k, l\} \cap \{i, j\} \neq \emptyset$.

Furthermore, if $\overline{V}_k = \infty$ *for* $k \in \mathcal{N}^+$, *then* (p, v', W') *is feasible for SOCP.*

Lemma 7.12 implies that *violation of* (7.6g) *propagates to neighboring lines*: if there exists an SOCP solution (p, v, W) that violates (7.6g) on some line $i \to j$, then there exists an SOCP solution (p, v', W') that violates (7.6g) on all neighboring lines of $i \to j$.

Proof. Let (p, v, W) be feasible for SOCP and violate (7.6g) on $i \to j$ where $i, j \neq 0$. Since (p, v, W) satisfies (7.6f), one has $0 \leq W_{ij} \leq \sqrt{v_i v_j}$. Since (p, v, W) violates (7.6g) on $i \to j$, one has $W_{ij} \neq \sqrt{v_i v_j}$. Hence,
$$W_{ij} < \sqrt{v_i v_j}.$$

Pick $\epsilon \in (0, \sqrt{v_i v_j} - W_{ij})$, construct v' as

$$v'_k = \begin{cases} v_k + \frac{y_{ij}}{\sum_{l:\, l\sim k} y_{kl}} \epsilon & k = i, j \\ v_k & \text{otherwise}, \end{cases} \tag{7.13}$$

and construct W' as
$$W'_{kl} = \begin{cases} W_{kl} + \epsilon & \text{if } \{k, l\} = \{i, j\} \\ W_{kl} & \text{otherwise.} \end{cases}$$

We will show that (p, v', W') is as required in Lemma 7.12.

It follows immediately from (7.13) that $v'_k = v_k$ if $k \neq i, j$ and $v'_k > v_k$ if $k = i, j$. The point (p, v', W') satisfies (7.6a) because

$$\sum_{l:\, k\sim l} (v'_k - W'_{kl}) y_{kl} \;=\; \sum_{l:\, k\sim l} (v_k - W_{kl}) y_{kl} + \sum_{l:\, k\sim l} \frac{y_{ij}}{\sum_{l:\, l\sim k} y_{kl}} \epsilon y_{kl} - \epsilon y_{ij} \;=\; \sum_{l:\, k\sim l} (v_k - W_{kl}) y_{kl} \;=\; p_k$$

for $k = i, j$, and
$$\sum_{l:\, k\sim l} (v'_k - W'_{kl}) y_{kl} \;=\; \sum_{l:\, k\sim l} (v_k - W_{kl}) y_{kl} \;=\; p_k$$

for $k \neq i, j$. The point (p, v', W') satisfies (7.6e) because

$$W'_{kl} - W'_{lk} = W_{kl} - W_{lk} = 0$$

for $k \to l$. The point (p, v', W') satisfies (7.6f) because

$$W'_{kl} = W_{kl} \in [0, \sqrt{v_k v_l}] \subseteq \left[0, \sqrt{v'_k v'_l}\right]$$

for $\{k, l\} \neq \{i, j\}$, and

$$W'_{kl} = W_{kl} + \epsilon \in [\epsilon, \sqrt{v_k v_l}) \subseteq \left[0, \sqrt{v'_k v'_l}\right]$$

for $\{k, l\} = \{i, j\}$. It follows that $|W'_{kl}| \leq \sqrt{v_k v_l}$ for $k \to l$. In particular, for $k \to l$ such that $\{k, l\} \cap \{i, j\} \neq \emptyset$,

$$|W'_{kl}| \leq \sqrt{v_k v_l} < \sqrt{v'_k v'_l}.$$

We have shown that (p, v', W') is as required in Lemma 7.12. When $\overline{V}_k = \infty$ for $k \in \mathcal{N}^+$, it is straightforward that (p, v', W') is feasible for SOCP. This completes the proof of Lemma 7.12. $\square$

Lemma 7.13. *Let (p, v, W) be feasible for SOCP and violate* (7.6g) *on some $0 \to i$. Then there exists (p', v', W') that*

- *satisfies* (7.6a), (7.6e), (7.6f)*;*
- *satisfies*
$$v'_k \begin{cases} > v_k & k = i \\ = v_k & k \neq i; \end{cases}$$
- *satisfies $p'_0 < p_0$ and $p'_i = p_i$ for $i \in \mathcal{N}^+$.*

Furthermore, if $\overline{V}_k = \infty$ for $k \in \mathcal{N}^+$, then (p', v', W') is feasible for SOCP; if f_0 is strictly increasing, then (p', v', W') has a smaller objective value than (p, v, W).

Lemma 7.13 implies that *every SOCP solution satisfies* (7.6g) *on all neighboring lines of the swing bus:* for any SOCP feasible point (p, v, W) that violates (7.6g) on some neighboring line $0 \to i$ of the swing bus 0, there exists an SOCP feasible point (p', v', W') with a smaller objective value and therefore (p, v, W) cannot be optimal.

Proof. Let (p, v, W) be feasible for SOCP and violate (7.6g) on some $0 \to i$. Since (p, v, W) satisfies (7.6f), one has $0 \leq W_{0i} \leq \sqrt{v_0 v_i}$. Since (p, v, W) violates (7.6g) on $0 \to i$, one has $W_{0i} \neq \sqrt{v_0 v_i}$. Hence,

$$W_{0i} < \sqrt{v_0 v_i}.$$

Pick $\epsilon \in (0, \sqrt{v_0 v_i} - W_{0i})$, construct v' as

$$v'_k = \begin{cases} v_k + \frac{y_{0i}}{\sum_{h:\, h\sim k} y_{kh}}\epsilon & k = i \\ v_k & \text{otherwise,} \end{cases} \tag{7.14}$$

construct W' as

$$W'_{kl} = \begin{cases} W_{kl} + \epsilon & \text{if } \{k,l\} = \{0,i\} \\ W_{kl} & \text{otherwise,} \end{cases}$$

and construct p' as

$$p'_k = \sum_{l:\, l\sim k} (v'_k - W'_{kl}) y_{kl}, \quad k \in \mathcal{N}.$$

We will show that (p', v', W') is as required in Lemma 7.13.

It follows immediately from (7.14) that $v'_k = v_k$ if $k \neq i$ and $v'_k > v_k$ if $k = i$. The point (p', v', W') satisfies (7.6a) according to the construction of p'. The point (p', v', W') satisfies (7.6e) because

$$W'_{kl} - W'_{lk} = W_{kl} - W_{lk} = 0$$

for $k \to l$. The point (p, v', W') satisfies (7.6f) because

$$W'_{kl} = W_{kl} \in [0, \sqrt{v_k v_l}] \subseteq \left[0, \sqrt{v'_k v'_l}\right]$$

for $\{k,l\} \neq \{0,i\}$, and

$$W'_{kl} = W_{kl} + \epsilon \in [\epsilon, \sqrt{v_k v_l}) \subseteq \left[0, \sqrt{v'_k v'_l}\right]$$

for $\{k,l\} = \{0,i\}$. One can prove that $p'_i = p_i$ for $i \in \mathcal{N}^+$ as in Lemma 7.12, and

$$p'_0 \;=\; \sum_{k:\, k\sim 0} (v'_0 - W'_{0k}) y_{0k} \;=\; \sum_{k:\, k\sim 0} (v_0 - W_{0k}) y_{0k} - y_{0i}\epsilon \;<\; p_0.$$

We have shown that (p', v', W') is as required in Lemma 7.13. When $\overline{V}_k = \infty$ for $k \in \mathcal{N}^+$, it is straightforward that (p', v', W') is feasible for SOCP. When f_0 is strictly increasing, it is straightforward that (p', v', W') has a smaller objective value than (p, v, W). This completes the proof of Lemma 7.13. $\square$

Combining Lemmas 7.12 and 7.13 gives the proof of Theorem 7.4. Assume there exists an SOCP solution (p, v, W) that violates (7.6g). By repeating the construction described in Lemma 7.12, one can find an SOCP solution (p, v', W') that violates (7.6g) on some neighboring line $0 \to i$ of the swing bus 0 since the network is connected. By Lemma 7.13, this contradicts the optimality of (p, v, W). Hence, every SOCP solution must satisfy (7.6g), i.e., SOCP is exact. This completes the proof of Theorem 7.4.

7.C Proof of Theorem 7.5

Assume the conditions in Theorem 7.5 hold. We will show that for any SOCP feasible point (p, v, W) that violates (7.6g), there exists another SOCP feasible point (p', v', W') that has a smaller objective value than (p, v, W). Hence, every SOCP solution must satisfy (7.6g), i.e., SOCP is exact.

Construction of (p', v', W') is based on Lemmas 7.14–7.17.

Lemma 7.14. *Assume the conditions in Theorem 7.5 hold and let (p, v, W) be feasible for SOCP. If $p_i = \underline{p}_i$ for some $i \in \mathcal{N}^+$, then $v_i < \overline{V}_i^2$.*

Lemma 7.14 implies that *power injection lower bound $\underline{p}$ and voltage upper bounds cannot bind simultaneously*: if the constraint $p_i \geq \underline{p}_i$ is binding at some bus $i \in \mathcal{N}^+$, then $v_i \leq \overline{V}_i^2$ cannot bind at bus i.

Proof. When $p_i = \underline{p}_i < 0$, one has

$$0 > \underline{p}_i = p_i = \sum_{j:\, j\sim i} (v_i - W_{ij}) y_{ij}$$

and therefore $(v_i - W_{ij}) y_{ij} < 0$ for some $j \in \mathcal{N}$. It follows from (7.6g) that $W_{ij} \leq \sqrt{v_i v_j}$ and therefore

$$v_i < W_{ij} \leq \sqrt{v_i v_j} \quad \Rightarrow \quad v_i < v_j \leq \overline{V}_j^2 = \overline{V}_i^2.$$

This completes the proof of Lemma 7.14. □

Lemma 7.15. *Assume the conditions in Theorem 7.5 hold and let (p, v, W) be feasible for SOCP. If*

- (p, v, W) *violates* (7.6g) *on some $i \to j$;*
- $p_i > \underline{p}_i$, $p_j > \underline{p}_j$ *(introduce $\underline{p}_0 = -\infty$ since p_0 is unconstrained),*

then there exists (p', v, W') that

- *satisfies* (7.6a)–(7.6f)*;*
- *satisfies*

$$p'_k \begin{cases} < p_k & \text{if } k = i, j \\ = p_k & \text{otherwise.} \end{cases}$$

Lemma 7.15 implies that *if an SOCP solution violates* (7.6g) *on some* $i \to j$, *it must satisfy* $p_i = \underline{p}_i$ *or* $p_j = \underline{p}_j$.

Proof. Since (p, v, W) satisfies (7.6f), $0 \le W_{ij} \le \sqrt{v_i v_j}$. Since (p, v, W) violates (7.6g) on $i \to j$, $W_{ij} \ne \sqrt{v_i v_j}$. Hence, $W_{ij} < \sqrt{v_i v_j}$.

Pick an $\epsilon > 0$ such that

$$\epsilon < \min\left\{\frac{p_i - \underline{p}_i}{y_{ij}},\ \frac{p_j - \underline{p}_j}{y_{ij}},\ \sqrt{v_i v_j} - W_{ij}\right\},$$

construct W' as

$$W'_{kl} = \begin{cases} W_{kl} + \epsilon & \text{if } \{k, l\} = \{i, j\} \\ W_{kl} & \text{otherwise,} \end{cases}$$

and construct p' as

$$p'_k = \sum_{l:\, l \sim k} (v_k - W'_{kl}) y_{kl}, \quad k \in \mathcal{N}.$$

We will show that (p', v, W') is as required in Lemma 7.15.

The point (p', v, W') satisfies (7.6a) according to the construction of p'. When $k \ne i, j$, one has

$$p'_k = \sum_{l:\, l \sim k} (v_k - W'_{kl}) y_{kl} = \sum_{l:\, l \sim k} (v_k - W_{kl}) y_{kl} = p_k;$$

When $k = i, j$, one has

$$p'_k \;=\; \sum_{l:\, l \sim k} (v_k - W'_{kl}) y_{kl} \;=\; \sum_{l:\, l \sim k} (v_k - W_{kl}) y_{kl} - y_{ij}\epsilon \;=\; p_k - y_{ij}\epsilon \;\in\; (\underline{p}_k, \overline{p}_k).$$

Hence, (p', v, W') satisfies (7.6b) and

$$p'_k \begin{cases} < p_k & \text{if } k = i, j \\ = p_k & \text{otherwise.} \end{cases}$$

The point (p', v, W') satisfies (7.6e) because

$$W'_{kl} - W'_{lk} = W_{kl} - W_{lk} = 0$$

for $k \to l$. The point (p', v, W') satisfies (7.6f) because

$$W'_{kl} = W_{kl} \in [0, \sqrt{v_k v_l}]$$

when $\{k,l\} \neq \{i,j\}$ and

$$W'_{kl} = W_{kl} + \epsilon \in [0, \sqrt{v_k v_l})$$

when $\{k,l\} = \{i,j\}$. This completes the proof of Lemma 7.15. $\square$

Lemma 7.16. *Assume the conditions in Theorem 7.5 hold and let* (p, v, W) *be feasible for SOCP. If*

- (p, v, W) *violates* (7.6g) *on some* $i \to j$ *where* $i, j \neq 0$*;*
- $(p_i = \underline{p}_i \ \& \ p_j > \underline{p}_j)$ *or* $(p_i > \underline{p}_i \ \& \ p_j = \underline{p}_j)$,

then there exists (p', v', W') *that*

- *satisfies* (7.6a)–(7.6f)*;*
- *satisfies*

$$\sum_{i\in\mathcal{N}} f_i(p'_i) < \sum_{i\in\mathcal{N}} f_i(p_i).$$

Lemmas 7.15 and 7.16 imply that *every SOCP solution, if violating* (7.6g) *on some* $i \to j$ *where* $i, j \neq 0$*, must satisfy* $p_i = \underline{p}_i$ *and* $p_j = \underline{p}_j$.

Proof. We present the proof for the case where

$$p_i = \underline{p}_i \text{ and } p_j > \underline{p}_j.$$

The proof for the case where $p_i > \underline{p}_i$ and $p_j = \underline{p}_j$ is similar and omitted for brevity. As discussed in Lemma 7.15, one has

$$W_{ij} < \sqrt{v_i v_j}.$$

It follows from Lemma 7.14 that

$$v_i < \overline{V}_i^2.$$

Pick an $\epsilon > 0$ such that

$$\epsilon < \min\left\{\frac{p_j - \underline{p}_j}{y_{ij}},\ \sqrt{v_i v_j} - W_{ij},\ \frac{\sum_{k:\,k\sim i} y_{ki}}{y_{ij}}\left(\overline{V}_i^2 - v_i\right)\right\},$$

then

$$p_j - y_{ij}\epsilon \ > \ \underline{p}_j, \quad W_{ij} + \epsilon \ < \ \sqrt{v_i v_j}, \quad v_i + \frac{y_{ij}}{\sum_{k:\,k\sim i} y_{ki}}\epsilon \ < \ \overline{V}_i^2.$$

Construct W' as

$$W'_{kl} = \begin{cases} W_{kl} + \epsilon & \text{if } \{k,l\} = \{i,j\} \\ W_{kl} & \text{otherwise,} \end{cases}$$

construct v' as

$$v'_k = \begin{cases} v_k + \frac{y_{ij}}{\sum_{l:\, l\sim k} y_{kl}}\epsilon & \text{if } k = i \\ v_k & \text{otherwise,} \end{cases}$$

and construct p' as

$$p'_k = \sum_{l:\, l\sim k} (v'_k - W'_{kl}) y_{kl}, \quad k \in \mathcal{N}.$$

We will show that (p', v', W') is as required in Lemma 7.16.

The point (p', v', W') satisfies (7.6a) according to the construction of p'. When $k \neq i, j$, one has

$$p'_k = \sum_{l:\, l\sim k} (v'_k - W'_{kl}) y_{kl} = \sum_{l:\, l\sim k} (v_k - W_{kl}) y_{kl} = p_k;$$

Besides, one has

$$\begin{aligned} p'_i &= \sum_{k:\, k\sim i} (v'_i - W'_{ki}) y_{ki} \\ &= \sum_{k:\, k\sim i} (v_i - W_{ki}) y_{ki} + (v'_i - v_i) \sum_{k:\, k\sim i} y_{ki} - y_{ij}\epsilon \\ &= \sum_{k:\, k\sim i} (v_i - W_{ki}) y_{ki} \;=\; p_i, \\ p'_j &= \sum_{k:\, k\sim j} (v'_j - W'_{kj}) y_{kj} \\ &= \sum_{k:\, k\sim j} (v_j - W_{kj}) y_{kj} - y_{ij}\epsilon \\ &= p_j - y_{ij}\epsilon \;\in\; (\underline{p}_j, \overline{p}_j). \end{aligned}$$

Hence, (p', v', W') satisfies (7.6b) and

$$p'_k \begin{cases} < p_k & \text{if } k = j \\ = p_k & \text{otherwise.} \end{cases}$$

It follows that $\sum_{i\in\mathcal{N}} f_i(p'_i) < \sum_{i\in\mathcal{N}} f_i(p_i)$. Note that

$$v'_k \begin{cases} > v_k & \text{if } k = i \\ = v_k & \text{otherwise} \end{cases}$$

and $v_i' < \overline{V}_i^2$, the point (p', v', W') satisfies (7.6c) and (7.6d). The point (p', v, W') satisfies (7.6e) because

$$W_{kl}' - W_{lk}' = W_{kl} - W_{lk} = 0$$

for $k \to l$. The point (p', v, W') satisfies (7.6f) because

$$W_{kl}' = W_{kl} \in [0, \sqrt{v_k v_l}] \subseteq \left[0, \sqrt{v_k' v_l'}\right]$$

when $\{k, l\} \neq \{i, j\}$ and

$$W_{ij}' = W_{ij} + \epsilon \in [0, \sqrt{v_i v_j}) \subseteq \left[0, \sqrt{v_i' v_j'}\right].$$

This completes the proof of Lemma 7.16. $\square$

Lemma 7.17. *Assume the conditions in Theorem 7.5 hold and let (p, v, W) be feasible for SOCP. If*

- *(p, v, W) violates (7.6g) on some $i \to j$ where $i, j \neq 0$;*
- *$p_i = \underline{p}_i$ and $p_j = \underline{p}_j$,*

then there exists (p, v', W') that

- *satisfies (7.6a)–(7.6f);*
- *violates (7.6g) for $k \to l$ such that $\{k, l\} \cap \{i, j\} \neq \emptyset$.*

Lemmas 7.15–7.17 imply that *violation of (7.6g) propagates to neighboring lines:* if there exists an SOCP solution that violates (7.6g) on some $i \to j$ where $i, j \neq 0$, then there exists an SOCP solution that violates (7.6g) on all neighboring lines $k \to l$ of $i \to j$.

Proof. As discussed in Lemma 7.15, one has

$$W_{ij} < \sqrt{v_i v_j}.$$

It follows from Lemma 7.14 that

$$v_i < \overline{V}_i^2, \ v_j < \overline{V}_j^2.$$

Pick an $\epsilon > 0$ such that

$$\epsilon \ < \ \min\left\{\sqrt{v_i v_j} - W_{ij}, \ \frac{\sum_{k:\, k \sim i} y_{ki}}{y_{ij}} \left(\overline{V}_i^2 - v_i\right), \ \frac{\sum_{k:\, k \sim j} y_{kj}}{y_{ij}} \left(\overline{V}_j^2 - v_j\right)\right\},$$

then

$$W_{ij} + \epsilon \ < \ \sqrt{v_i v_j}, \quad v_i + \frac{y_{ij}}{\sum_{k:\, k \sim i} y_{ki}} \epsilon \ < \ \overline{V}_i^2, \quad v_j + \frac{y_{ij}}{\sum_{k:\, k \sim j} y_{kj}} \epsilon \ < \ \overline{V}_j^2.$$

Construct W' as

$$W'_{kl} = \begin{cases} W_{kl} + \epsilon & \text{if } \{k,l\} = \{i,j\} \\ W_{kl} & \text{otherwise,} \end{cases}$$

and construct v' as

$$v'_k = \begin{cases} v_k + \frac{y_{ij}}{\sum_{l:\, l\sim k} y_{kl}}\epsilon & \text{if } k = i, j \\ v_k & \text{otherwise.} \end{cases}$$

We will show that (p, v', W') is as required in Lemma 7.17.

It is straightforward to check that the point (p, v', W') satisfies (7.6c) and (7.6d). The point (p, v', W') satisfies (7.6a) because

$$\sum_{l:\, k\sim l} (v'_k - W'_{kl})y_{kl} \;=\; \sum_{l:\, k\sim l} (v_k - W_{kl})y_{kl} + \sum_{l:\, k\sim l} \frac{y_{ij}}{\sum_{l:\, l\sim k} y_{kl}}\epsilon y_{kl} - \epsilon y_{ij} \;=\; \sum_{l:\, k\sim l} (v_k - W_{kl})y_{kl} \;=\; p_k$$

for $k = i, j$, and

$$\sum_{l:\, k\sim l} (v'_k - W'_{kl})y_{kl} \;=\; \sum_{l:\, k\sim l} (v_k - W_{kl})y_{kl} \;=\; p_k$$

for $k \neq i, j$. The point (p, v', W') satisfies (7.6e) because

$$W'_{kl} - W'_{lk} = W_{kl} - W_{lk} = 0$$

for $k \to l$. The point (p, v', W') satisfies (7.6f) because

$$W'_{kl} = W_{kl} \in [0, \sqrt{v_k v_l}] \subseteq \left[0, \sqrt{v'_k v'_l}\right]$$

for $\{k,l\} \neq \{i,j\}$, and

$$W'_{kl} = W_{kl} + \epsilon \in [\epsilon, \sqrt{v_k v_l}) \subseteq \left[0, \sqrt{v'_k v'_l}\right]$$

for $\{k,l\} = \{i,j\}$. It follows that $|W'_{kl}| \leq \sqrt{v_k v_l}$ for $k \to l$. In particular, for $k \to l$ such that $\{k,l\} \cap \{i,j\} \neq \emptyset$,

$$|W'_{kl}| \leq \sqrt{v_k v_l} < \sqrt{v'_k v'_l}.$$

This completes the proof of Lemma 7.17. □

Combining Lemmas 7.14–7.17 gives the proof of Theorem 7.5. Assume the conditions in Theorem 7.5 hold. If SOCP is not exact, then there exists an SOCP solution (p, v, W) that violates (7.6g) on some $i \to j$.

According to Lemmas 7.15–7.16, one must have $p_i = \underline{p}_i$ and $p_j = \underline{p}_j$ since otherwise (p, v, W) cannot be optimal for SOCP (introduce $\underline{p}_0 = -\infty$ since p_0 is unconstrained).

According to Lemma 7.17, there exists an SOCP solution (p', v', W') that violates (7.6g) on all neighboring lines $k \to l$ of $i \to j$. Since the network $(\mathcal{N}, \mathcal{E})$ is connected, one can continue such propagation to obtain an SOCP solution that violates (7.6g) on some neighboring line $0 \to k$ of the swing bus 0. Then,

$$p_0 = \underline{p}_0 = -\infty \notin \mathbb{R}.$$

This contradicts with $p_0 \in \mathbb{R}$. Hence, SOCP is exact. This completes the proof of Theorem 7.5.

7.D Proof of Theorem 7.6

Assume that SOCP is convex, exact, and has at least one solution. Let $\tilde{w} = (\tilde{p}, \tilde{v}, \tilde{W})$ and $\hat{w} = (\hat{p}, \hat{v}, \hat{W})$ be two SOCP solutions. It suffices to prove $\tilde{w} = \hat{w}$.

Let $w := (p, v, W) := (\tilde{w} + \hat{w})/2$ be the average of $\tilde{w}$ and $\hat{w}$. Since SOCP is convex, w is optimal for SOCP. Since SOCP is exact, the points $\tilde{w}$, $\hat{w}$, and w all satisfy (7.6g), i.e.,

$$\tilde{v}_i\tilde{v}_j = \tilde{W}_{ij}^2, \tag{7.15a}$$

$$\hat{v}_i\hat{v}_j = \hat{W}_{ij}^2, \tag{7.15b}$$

$$v_i v_j = W_{ij}^2 \tag{7.15c}$$

for $i \to j$. Substitute $v = (\tilde{v} + \hat{v})/2$, $W = (\tilde{W} + \hat{W})/2$ in (7.15c), and simplify using (7.15a) and (7.15b) to obtain

$$\tilde{v}_i\hat{v}_j + \hat{v}_i\tilde{v}_j = 2\tilde{W}_{ij}\hat{W}_{ij}, \quad i \to j.$$

It follows that

$$\tilde{v}_i\hat{v}_j + \hat{v}_i\tilde{v}_j \;=\; 2\sqrt{\tilde{v}_i\tilde{v}_j\hat{v}_i\hat{v}_j} \;\le\; \tilde{v}_i\hat{v}_j + \hat{v}_i\tilde{v}_j$$

for $i \to j$. The inequality attains equality, and therefore

$$\frac{\hat{v}_i}{\tilde{v}_i} = \frac{\hat{v}_j}{\tilde{v}_j}, \quad i \to j.$$

Let $\eta_i := \hat{v}_i/\tilde{v}_i$ denote the ratio of $\hat{v}_i$ to $\tilde{v}_i$ for $i \in \mathcal{N}$, then $\eta_0 = 1$ and $\eta_i = \eta_j$ if $i \to j$. Since the network is connected, $\eta_i = 1$ for $i \in \mathcal{N}$. Hence, $\hat{v} = \tilde{v}$. Then, it follows from (7.15) that

$$\hat{W}_{ij} = \sqrt{\hat{v}_i\hat{v}_j} = \sqrt{\tilde{v}_i\tilde{v}_j} = \tilde{W}_{ij}, \quad i \to j.$$

We have shown that $(\hat{v}, \hat{W}) = (\tilde{v}, \tilde{W})$. It follows immediately that $\hat{p} = \tilde{p}$ and therefore $\hat{w} = \tilde{w}$.

7.E Proof of Lemma 7.7.

Fix (v_0, p, ℓ) and let $e := |\mathcal{E}|$ denote the number of lines. Then (7.10a)–(7.10c) collect $n + 2e$ linear equations in $n + 2e$ variables $(v_i)_{i\in\mathcal{N}^+}$ and $(P_{ij})_{i\sim j}$. To show the uniqueness of (v, P) satisfying (7.10a)–(7.10c), it suffices to prove that (7.10a)–(7.10c) are linearly independent, i.e., if the coefficients of v_i and P_{ij} for all i and all $i \sim j$ in

$$\sum_{i\in\mathcal{N}^+} a_i \left(\sum_{j:\, j\sim i} P_{ij}\right) + \sum_{i\to j} b_{ij}(P_{ij} + P_{ji}) + \sum_{i\to j} c_{ij}\left[v_i - v_j - z_{ij}(P_{ij} - P_{ji})\right]$$

are 0, then $(a, b, c) = 0$.

Introduce $a_0 := 0$ for convenience. For each $i \to j$, the coefficients of P_{ij} and P_{ji} being 0 implies

$$0 = a_i + b_{ij} - z_{ij}c_{ij}, \tag{7.16a}$$

$$0 = a_j + b_{ij} + z_{ij}c_{ij}. \tag{7.16b}$$

It follows that

$$c_{ij} = \frac{y_{ij}}{2}(a_i - a_j), \quad i \to j.$$

Hence, $a = 0$ implies $c = 0$, and it further follows from (7.16a) that $b = 0$. Therefore, it suffices to prove that $a = 0$.

Let $\mathcal{A} := \mathrm{argmax}_{i\in\mathcal{N}} a_i$ denote the set of buses i where a_i is maximized. Since $a_0 = 0$, $a = 0$ is equivalent to $\mathcal{A} = \mathcal{N}$. Since the network $(\mathcal{N}, \mathcal{E})$ is connected and $\mathcal{A} \neq \emptyset$, to prove $\mathcal{A} = \mathcal{N}$, it suffices to show

$$i \in \mathcal{A} \;\&\; i \sim j \quad \Rightarrow \quad j \in \mathcal{A}.$$

For $i \in \mathcal{N}^+$, the coefficient of v_i being 0 implies

$$\begin{aligned} & 0 = \sum_{j:\, i\to j} c_{ij} - \sum_{h:\, h\to i} c_{hi} \\ \Rightarrow\quad & 0 = \sum_{j:\, i\to j} \frac{y_{ij}}{2}(a_i - a_j) - \sum_{h:\, h\to i} \frac{y_{hi}}{2}(a_h - a_i) \\ \Rightarrow\quad & 0 = \sum_{j:\, i\sim j} y_{ij}(a_i - a_j). \end{aligned}$$

If $i \in \mathcal{A}$ and $i \sim j$, then

$$0 = \sum_{k:\, i\sim k} y_{ik}(a_i - a_k) \geq y_{ij}(a_i - a_j) \geq 0.$$

Hence, $a_j = a_i$ and therefore $j \in \mathcal{A}$. This completes the proof of Lemma 7.7.

7.F Proof of Lemma 7.8.

Let $\ell \geq \ell'$. Define $\Delta v_0 := 0$ and

$$\begin{aligned} \Delta v_i &:= \hat{v}_i(p,\ell') - \hat{v}_i(p,\ell), \quad i \in \mathcal{N}^+; \\ \Delta P_{ij} &:= \hat{P}_{ij}(p,\ell') - \hat{P}_{ij}(p,\ell), \quad i \sim j. \end{aligned}$$

Let $\mathcal{A} := \text{argmin}_{i\in\mathcal{N}} \Delta v_i$ denote the set of buses i where Δv_i is minimized. If $0 \in \mathcal{A}$, then $\Delta v_i \geq \Delta v_0 = 0$, i.e., $\hat{v}_i(p,\ell') \geq \hat{v}_i(p,\ell)$ for $i \in \mathcal{N}^+$. Hence, it suffices to prove that $0 \in \mathcal{A}$.

We prove $0 \in \mathcal{A}$ by contradiction. Assume $0 \notin \mathcal{A}$. Let $\mathcal{B}$ denotes a nonempty connected component of $\mathcal{A}$, then $0 \notin \mathcal{B}$. Whenever $i \sim j$, $i \sim \mathcal{B}$, and $j \notin \mathcal{B}$, one has $j \notin \mathcal{A}$ (otherwise $j \in \mathcal{B}$ by the definition of a connected component). Therefore $\Delta v_i < \Delta v_j$ (since Δv_i is minimized in $\mathcal{A}$) and it follows that

$$0 \;>\; \Delta v_i - \Delta v_j \;=\; z_{ij}\left(\Delta P_{ij} - \Delta P_{ji}\right) \;=\; 2z_{ij}\Delta P_{ij} - z_{ij}^2\left(\ell'_{ij} - \ell_{ij}\right) \;\geq\; 2z_{ij}\Delta P_{ij}.$$

Hence, $\Delta P_{ij} < 0$ for all $i \sim j$ such that $i \in \mathcal{B}$ and $j \notin \mathcal{B}$.

It follows that

$$\begin{aligned} 0 &= \sum_{i\in\mathcal{B}} p_i - \sum_{i\in\mathcal{B}} p_i \\ &= \sum_{i\in\mathcal{B}} \sum_{j:\, j\sim i} \hat{P}_{ij}(p,\ell') - \sum_{i\in\mathcal{B}} \sum_{j:\, j\sim i} \hat{P}_{ij}(p,\ell) \\ &= \sum_{i\in\mathcal{B}} \sum_{j:\, j\sim i} \Delta P_{ij} \\ &= \sum_{i\in\mathcal{B}} \left(\sum_{j\in\mathcal{B}:\, j\sim i} \Delta P_{ij} + \sum_{j\notin\mathcal{B}:\, j\sim i} \Delta P_{ij} \right) \\ &= \sum_{i\sim j, i\in\mathcal{B}, j\notin\mathcal{B}} \Delta P_{ij} + \sum_{i\sim j, i\in\mathcal{B}, j\in\mathcal{B}} \Delta P_{ij} \\ &< \sum_{i\sim j, i\in\mathcal{B}, j\in\mathcal{B}} \Delta P_{ij} \\ &= \frac{1}{2} \sum_{i\sim j, i\in\mathcal{B}, j\in\mathcal{B}} \left(\Delta P_{ij} + \Delta P_{ji}\right) \\ &= \frac{1}{2} \sum_{i\sim j, i\in\mathcal{B}, j\in\mathcal{B}} z_{ij}\left(\ell'_{ij} - \ell_{ij}\right) \leq 0, \end{aligned}$$

which is a contradiction. Hence, $0 \in \mathcal{A}$. This completes the proof of Lemma 7.8.

7.G Proof of Corollary 7.9

Let (p, v, W) be feasible for SOCP, and define P and ℓ according to (7.8) and (7.9). It is straightforward to check that the point (p, v, ℓ) satisfies (7.10a)–(7.10c) and therefore

$$v_i = \hat{v}_i(p, \ell), \quad i \in \mathcal{N}^+.$$

Since (v, W) satisfies (7.6f), one has $|W_{ij}| \leq \sqrt{v_i v_j}$ and therefore

$$\ell_{ij} \;=\; y_{ij}^2 \left(v_i - W_{ij} - W_{ji} + v_j\right) \;\geq\; y_{ij}^2 \left(\sqrt{v_i} - \sqrt{v_j}\right)^2 \;\geq\; 0$$

for $i \rightarrow j$, i.e., $\ell \geq 0$. It follows that $\hat{v}_i(p, 0) \geq \hat{v}_i(p, \ell)$, i.e., $\hat{v}_i(p) \geq v_i$, for $i \in \mathcal{N}^+$.

Bibliography

[1] Ieee distribution test feeders. online at available at http://ewh.ieee.org/soc/pes/dsacom/testfeeders/.

[2] Southern california edison. online at http://www.sce.com/.

[3] Electric power systems and equipment—voltage ratings (60 hertz). *ANSI Standard Publication*, (ANSI C84.1), 1995.

[4] S. Acha, T. C. Green, and N. Shah. Effects of optimised plug-in hybrid vehicle charging strategies on electric distribution network losses. In *IEEE PES Transmission and Distribution Conference and Exposition*, pages 1–6, 2010.

[5] D. J. Aigner and J. G. Hirschberg. Commercial/industrial customer response to time-of-use electricity prices: Some experimental results. *The RAND Journal of Economics*, 16(3):341–355, 1985.

[6] Alberta Electric System Operator. Wind power / ail data, 2009. http://www.aeso.ca/gridoperations/20544.html.

[7] O. Alsac, J. Bright, M. Prais, and B. Stott. Further developments in lp-based optimal power flow. *IEEE Transactions on Power Systems*, 5(3):697–711, 1990.

[8] Autobloggreen. Tesla investor says selling 500,000 evs in 2020 is totally doable. online at http://green.autoblog.com/2014/03/11/tesla-investor-says-selling-500000-evs-in-2020-possible/, 2014.

[9] X. Bai and H. Wei. A semidefinite programming method with graph partitioning technique for optimal power flow problems. *International Journal on Electrical Power and Energy Systems*, 33(7):1309–1314, 2011.

[10] X. Bai, H. Wei, K. Fujisawa, and Y. Wang. Semidefinite programming for optimal power flow problems. *International Journal of Electrical Power and Energy Systems*, 30(6-7):383–392, 2008.

[11] E. C. Baptista, E. A. Belati, and G. R. M. da Costa. Logarithmic barrier-augmented lagrangian function to the optimal power flow problem. *International Journal on Electrical Power and Energy Systems*, 27(7):528–532, 2005.

[12] M. E. Baran and F. F. Wu. Network reconfiguration in distribution systems for loss reduction and load balancing. *IEEE Transactions on Power Delivery*, 4(2):1401–1407, 1989.

[13] M. E. Baran and F. F. Wu. Optimal Capacitor Placement on radial distribution systems. *IEEE Transactions on Power Delivery*, 4(1):725–734, 1989.

[14] M. E. Baran and F. F. Wu. Optimal sizing of capacitors placed on a radial distribution systems. *IEEE Transactions on Power Delivery*, 4(1):735–743, 1989.

[15] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and distributed computation: numerical methods.* Prentice-Hall, Inc., 1989.

[16] S. Bose, D. F. Gayme, S. Low, and K. M. Chandy. Optimal power flow over tree networks. In *Allerton*, pages 1342–1348, 2011.

[17] S. Bose, S. Low, and K. Chandy. Equivalence of branch flow and bus injection models. In *Allerton*, pages 1893–1899, 2012.

[18] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[19] W. A. Bukhsh, A. Grothey, K. McKinnon, and P. Trodden. Local solutions of optimal power flow. *IEEE Transactions on Power Systems*, 28(4):4780–4788, 2013.

[20] M. B. Cain, R. P. O'Neill, and A. Castillo. History of optimal power flow and formulations. *Federal Energy Regulatory Commission*, 2012.

[21] F. Capitanescu, M. Glavic, D. Ernst, and L. Wehenkel. Interior-point based algorithms for the solution of optimal power flow problems. *Electric Power System Research*, 77(5-6):508–517, 2007.

[22] M. Caramanis and J. Foster. Management of electric vehicle charging to mitigate renewable generation intermittency and distribution network congestion. In *IEEE Conference on Decision and Control*, pages 4717–4722, 2009.

[23] M. Caramanis and J. M. Foster. Management of electric vehicle charging to mitigate renewable generation intermittency and distribution network congestion. In *IEEE Conference on Decision and Control*, pages 4717–4722, 2009.

[24] J. Carpentier. Contribution to the economic dispatch problem. *Bulletin de la Societe Francoise des Electriciens*, 3(8):431–447, 1962.

[25] A. Castillo and R. P. O'Neill. Survey of approaches to solving the ACOPF. Technical report, US FERC Technical report, 2013.

[26] L. Chen, N. Li, S. H. Low, and J. C. Doyle. Two market models for demand response in power networks. In *IEEE SmartGridComm*, pages 397–402, 2010.

[27] N. Chen, L. Gan, S. H. Low, and A. Wierman. Distributional analysis for model predictive deferrable load control. *arXiv:1403.6887*, 2014.

[28] S. Chen and L. Tong. iems for large scale charging of electric vehicles: architecture and optimal online scheduling. In *IEEE SmartGridComm*, pages 629–634, 2012.

[29] T.-H. Chen, M.-S. Chen, K.-J. Hwang, P. Kotas, and E. A. Chebli. Distribution system power flow analysis – a rigid approach. *IEEE Transactions on Power Delivery*, 6(3), 1991.

[30] K. Clement, E. Haesen, and J. Driesen. Coordinated charging of multiple plug-in hybrid electric vehicles in residential distribution grids. In *IEEE/PES Power Systems Conference and Exposition*, pages 1–7, 2009.

[31] A. Conejo, J. Morales, and L. Baringo. Real-time demand response model. *IEEE Transactions on Smart Grid*, 1(3):236–242, 2010.

[32] G. C. Contaxis, C. Delkis, and G. Korres. Decoupled optimal power flow using linear or quadratic programming. *IEEE Transactions on Power Systems*, 1(2):1–7, 1986.

[33] E. Dall'Anese, H. Zhu, and G. B. Giannakis. Distributed optimal power flow for smart microgrids. *IEEE Transactions on Smart Grid*, 4(3):464 – 1475, 2012.

[34] S. Deilami, A. Masoum, P. Moses, and M. Masoum. Real-time coordination of plug-in electric vehicle charging in smart grids to minimize power losses and improve voltage profile. *IEEE Transactions on Smart Grid*, 2(3):456–467, 2011.

[35] M. Farivar, C. R. Clarke, S. H. Low, and K. M. Chandy. Inverter VAR control for distribution systems with renewables. In *IEEE SmartGridComm*, pages 457–462, 2011.

[36] M. Farivar and S. H. Low. Branch flow model: relaxations and convexification (parts I, II). *IEEE Transactions on Power Systems*, 28(3):2554–2572, 2013.

[37] M. Farivar, R. Neal, C. Clarke, and S. Low. Optimal inverter var control in distribution systems with high pv penetration. In *PES General Meeting*, pages 1–7, 2012.

[38] E. A. Feinberg and D. Genethliou. Load forecasting. In *Applied Mathematics for Restructured Electric Power Systems*, Power Electronics and Power Systems, pages 269–285. Springer US, 2005.

[39] S. Frank, I. Steponavice, and S. Rebennack. Optimal power flow: a bibliographic survey i. *Energy Systems*, 3(3):221–258, 2012.

[40] M. Fukuda, M. Kojima, K. Murota, and K. Nakata. Exploiting sparsity in semidefinite programming via matrix completion i: General framework. *SIAM Journal on Optimization*, 11(3):647–674, 2001.

[41] L. Gan, N. Li, U. Topcu, and S. H. Low. On the exactness of convex relaxation for optimal power flow in tree networks. In *IEEE Conference on Decision and Control*, pages 465–471, 2012.

[42] L. Gan, N. Li, U. Topcu, and S. H. Low. Optimal power flow in tree networks. In *IEEE Conference on Decision and Control*, pages 2313– 2318, 2013.

[43] L. Gan and S. H. Low. Optimal power flow in multiphase radial networks. In *Power systems computation conference*, 2014.

[44] L. Gan, U. Topcu, and S. Low. Optimal decentralized protocol for electric vehicle charging. *IEEE Transactions on Power Systems*, 28(2):940–951, 2013.

[45] L. Gan, U. Topcu, and S. H. Low. Stochastic distributed protocol for electric vehicle charging with discrete charging rate. In *IEEE PES General Meeting*, pages 1–8, 2012.

[46] L. Gan, A. Wierman, U. Topcu, N. Chen, and S. H. Low. Real-time deferrable load control: handling the uncertainties of renewable generation. In *International Conference on Future Energy Systems*, pages 113–124, 2013.

[47] GAS2. Bmw wants to build 100000 evs annually by 2020. online at `http://gas2.org/2014/03/21/bmw-wants-build-100000-evs-annually-2020/`, 2014.

[48] G. Giebel, R. Brownsword, G. Kariniotakis, M. Denhard, and C. Draxl. *The State-Of-The-Art in Short-Term Prediction of Wind Power*. ANEMOS.plus, 2011.

[49] M. Grant, S. Boyd, and Y. Ye. Cvx: Matlab software for disciplined convex programming. online at `http://cvxr.com/cvx/`, 2008.

[50] S. C. Graves, D. B. Kletter, and W. B. Hetzel. A dynamic model for requirements planning with application to supply chain optimization. *Manufacturing & Service Operation Management*, 1(1):50–61, 1998.

[51] S. C. Graves, H. C. Meal, S. Dasu, and Y. Qiu. Two-stage production planning in a dynamic environment, 1986. `http://web.mit.edu/sgraves/www/papers/GravesMealDasuQiu.pdf`.

[52] N. Hatziargyriou, H. Asano, R. Iravani, and C. Marnay. Microgrids. *IEEE Power and Energy Magazine*, 5(4):78–94, 2007.

[53] Y.-Y. Hsu and C.-C. Su. Dispatch of direct load control using dynamic programming. *IEEE Transactions on Power Systems*, 6(3):1056–1061, 1991.

[54] M. Huneault and F. D. Galiana. A survey of the optimal power flow literature. *IEEE Transactions on Power Systems*, 6(2):762–770, 1991.

[55] M. Ilic, J. Black, and J. Watz. Potential benefits of implementing load control. In *IEEE PES Winter Meeting*, volume 1, pages 177–182, 2002.

[56] A. Ipakchi and F. Albuyeh. Grid of the future. *IEEE Power and Energy Magazine*, 7(2):52–62, 2009.

[57] R. Jabr. Radial distribution load flow using conic programming. *IEEE Transactions on Power Systems*, 21(3):1458–1459, 2006.

[58] R. Jabr. Exploiting sparsity in sdp relaxations of the opf problem. *IEEE Transactions on Power Systems*, 27(2):1138–1139, 2012.

[59] R. A. Jabr. A primal-dual interior-point method to solve the optimal power flow dispatching problem. *Optimization and Engineering*, 4(4):309–336, 2003.

[60] R. B. Jr, E. S. Hawkins, and W. W. Pleines. Mechanized calculation of unbalanced load flow on radial distribution circuits. *IEEE Transactions on Power Apparatus and Systems*, PAS–86(4):451–421, 1967.

[61] H. Kakigano, Y. Miura, and T. Ise. Low-voltage bipolar-type dc microgrid for super high quality distribution. *IEEE Transactions on Power Electronics*, 25(12):3066–3075, 2010.

[62] L. Kelly, A. Rowe, and P. Wild. Analyzing the impacts of plug-in electric vehicles on distribution networks in british columbia. In *Electrical Power & Energy Conference*, pages 1–6, 2009.

[63] W. H. Kersting. *Distribution System Modeling and Analysis*. CRC Press, 2006.

[64] S. Kim and M. Kojima. Exact solutions of some nonconvex quadratic optimization problems via sdp and socp relaxations. *Computational Optimization and Applications*, 26(2):143–154, 2003.

[65] A. Lam, B. Zhang, A. Dominguez-Garcia, and D. Tse. Optimal distributed voltage regulation in power distribution networks. *arXiv:1204.5226*, 2012.

[66] J. Lavaei and S. H. Low. Zero duality gap in optimal power flow problem. *IEEE Transactions on Power Systems*, 27(1):92–107, 2012.

[67] J. Lavaei, A. Rantzer, and S. Low. Power flow optimization using positive quadratic programming. In *IFAC World Congress*, 2011.

[68] J. Lavaei, D. Tse, and B. Zhang. Geometry of power flows in tree networks. In *IEEE PES General Meeting*, pages 1–8, 2012.

[69] B. Lesieutre, D. Molzahn, A. Borden, and C. L. DeMarco. Examining the limits of the application of semidefinite programming to power flow problems. In *Allerton*, pages 1492–1499, 2011.

[70] N. Li, L. Chen, and S. H. Low. Optimal demand response based on utility maximization in power networks. In *IEEE PES General Meeting*, pages 1–8, 2011.

[71] N. Li, L. Gan, S. Low, and L. Chen. Demand response in radial distribution networks. In *IEEE SmartGridComm*, pages 7–12, 2012.

[72] Q. Li, T. Cui, R. Negi, F. Franchetti, and M. D. Ilic. On-line decentralized charging of plug-in electric vehicles in power systems. *arXiv:1106.5063*, 2011.

[73] S. H. Low. Convex relaxation of optimal power flow, I: formulations and relaxations. *IEEE Transactions on Control of Network Systems*, 1(1):15–27, 2014.

[74] S. H. Low. Convex relaxation of optimal power flow, II: exactness. *IEEE Transactions on Control of Network Systems*, 2014. to appear.

[75] Z. Ma, D. Callaway, and I. Hiskens. Decentralized charging control for large populations of plug-in electric vehicles. In *IEEE CDC*, pages 206–212, 2010.

[76] Z. Ma, D. S. Callaway, and I. A. Hiskens. Decentralized charging control of large populations of plug-in electric vehicles. *IEEE Transactions on Control Systems Technology*, 21(1):67–78, 2013.

[77] K. Mets, T. Verschueren, W. Haerick, C. Develder, and F. De Turck. Optimizing smart energy control strategies for plug-in hybrid electric vehicle charging. In *IEEE/IFIP NOMS Wksps*, pages 293–299, 2010.

[78] W. Min and L. Shengsong. A trust region interior point algorithm for optimal power flow problems. *International Journal on Electrical Power and Energy Systems*, 27(4):293–300, 2005.

[79] D. Molzahn, J. Holzer, B. Lesieutre, and C. DeMarco. Implementation of a large-scale optimal power flow solver based on semidefinite programming. *IEEE Transactions on Power Systems*, 28(4):3987–3998, 2013.

[80] D. K. Molzahn, B. C. Lesieutre, and C. L. DeMarco. Investigation of non-zero duality gap solutions to a semidefinite relaxation of the optimal power flow problem. In *Hawaii International Conference on System Sciences*, January 6-9 2014.

[81] J. A. Momoh, M. E. El-Hawary, and R. Adapa. A review of selected optimal power flow literature to 1993. Part I: Nonlinear and quadratic programming approaches. *IEEE Transactions on Power Systems*, 14(1):96–104, 1999.

[82] K. Nakata, K. Fujisawa, M. Fukuda, M. Kojima, and K. Murota. Exploiting sparsity in semidefinite programming via matrix completion II: Implementation and numerical results. *Mathematical Programming*, 95(2):303–327, 2003.

[83] K. Nakata, M. Yamashita, K. Fujisawa, and M. Kojima. A parallel primal-dual interior-point method for semidefinite programs using positive definite matrix completion. *Parallel Computing*, 32(1):24–43, 2006.

[84] K. S. Pandya and S. K. Joshi. A survey of optimal power flow methods. *Journal of Theoretical and Applied Information Technology*, 4(5):450–458, 2008.

[85] K. Qian, C. Zhou, M. Allan, and Y. Yuan. Modeling of load demand due to ev battery charging in distribution systems. *IEEE Transactions on Power Systems*, 26(2):802–810, 2011.

[86] P. Richardson, D. Flynn, and A. Keane. Optimal charging of electric vehicles in low-voltage distribution systems. *IEEE Transactions on Power Systems*, 27(1):268–279, 2012.

[87] C. Roe, J. Meisel, A. Meliopoulos, F. Evangelos, and T. Overbye. Power system level impacts of phevs. In *Hawaii International Conference on System Sciences*, pages 1–10, 2009.

[88] N. Rotering and M. Ilic. Optimal charge control of plug-in hybrid electric vehicles in deregulated electricity markets. *IEEE Transactions on Power Systems*, 26(3):1021–1029, 2011.

[89] D. Salomonsson, L. Soder, and A. Sannino. An adaptive control system for a dc microgrid for data centers. In *IEEE Industry Applications Conference*, pages 2414–2421, 2007.

[90] D. Salomonsson, L. Soder, and A. Sannino. Protection of low-voltage dc microgrids. *IEEE Transactions on Power Delivery*, 24(3):1045–1053, 2009.

[91] S. Sojoudi and J. Lavaei. Physics of power networks makes hard optimization problems easy to solve. In *IEEE Power and Energy Society General Meeting*, pages 1–8, 2012.

[92] A. A. Sousa and G. L. Torres. Robust optimal power flow solution using trust region and interior methods. *IEEE Transactions on Power Systems*, 26(2):487–499, 2011.

[93] Southern California Edison. 2012 static load profiles, 2012. `http://www.sce.com/005_regul_info/eca/DOMSM12.DLP`.

[94] B. Stott and O. Alsac. Fast decoupled load flow. *IEEE Transactions on Power Apparatus and Systems*, PAS-93(3):859–869, 1974.

[95] B. Stott, J. Jardim, and O. Alsac. Dc power flow revisited. *IEEE Transactions on Power Systems*, 24(3):1290–1300, 2009.

[96] J. F. Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1-4):625–653, 1999.

[97] A. Subramanian, M. Garcia, A. Dominguez-Garcia, D. Callaway, K. Poolla, and P. Varaiya. Real-time scheduling of deferrable electric loads. In *ACC*, pages 3643–3650, 2012.

[98] O. Sundstrom and C. Binding. Planning electric-drive vehicle charging under constrained grid conditions. In *International Conference on Power System Technology*, pages 1–6, 2010.

[99] The Wall Street Journal. Nissan may hit electric-car sales target before 2020. online at `http://online.wsj.com/news/articles/SB10001424052702303546204579440652848502402`, 2014.

[100] G. L. Torres and V. H. Quintana. An interior-point method for nonlinear optimal power flow using voltage rectangular coordinates. *IEEE Transactions on Power Systems*, 13(4):1211–1218, 1998.

[101] K. Turitsyn, P. Sulc, S. Backhaus, and M. Chertkov. Local control of reactive power by distributed photovoltaic generators. In *IEEE SmartGridComm*, pages 79–84, 2010.

[102] R. A. Verzijlbergh, M. O. Grond, Z. Lukszo, J. G. Slootweg, and M. D. Ilic. Network impacts and cost savings of controlled ev charging. *IEEE Transactions on Smart Grid*, 3(3):1203–1212, 2012.

[103] Wikipedia. Equivalence relation. `http://en.wikipedia.org/wiki/Equivalence_relation`.

[104] Wikipedia. Krasovskii-lasalle principle. `http://en.wikipedia.org/wiki/Krasovskii-LaSalle_principle`.

[105] Wikipedia. Renewable energy. `http://en.wikipedia.org/wiki/Renewable_energy`.

[106] Wikipedia. Wiener filter. `http://en.wikipedia.org/wiki/Wiener_filter`.

[107] Wikipedia. Plug-in electric vehicles in the united states. online at `http://en.wikipedia.org/wiki/Plug-in_electric_vehicles_in_the_United_States`, 2014.

[108] Y. Xiao, Y. Song, and Y. Sun. Power flow control approach to power systems with embedded FACTS devices. *IEEE Transactions on Power Systems*, 17(4):943–950, 2002.

[109] B. Zhang and D. Tse. Geometry of injection regions of power networks. *IEEE Transactions on Power Systems*, 28(2):788–797, 2013.