# NUMERICAL OPTIMIZATION AND MODELING TECHNIQUES FOR POWER SYSTEM OPERATIONS AND PLANNING

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Tomás Andrés Tinoco De Rubira
March 2015



This dissertation is online at: http://purl.stanford.edu/js789bg0173

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

**Walter Murray, Primary Adviser**

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

**Peter Glynn**

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

**Ram Rajagopal**

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

**Robert Entriken**

Approved for the Stanford University Committee on Graduate Studies.

**Patricia J. Gumport, Vice Provost for Graduate Education**

*This signature page was generated electronically upon submission of this dissertation in electronic format. An original signed hard copy of the signature page is on file in University Archives.*

# Abstract

In modern societies, electricity is ubiquitous and its availability and reliability are often taken for granted. However, the reality is that these properties rely on the proper operation of one of man's most complex creations: electric power grids. These systems transport electric energy across wide geographical regions from places where it is produced to places where it is consumed. They do so in many cases with virtually no storage. To ensure an efficient and reliable operation of these complex machines, numerical optimization algorithms are applied at almost every level of the operations and planning spectrum. For example, they are used for planning network expansions, scheduling generators, adjusting control devices, determining system state, computing security margins, and for many other crucial tasks. However, many of the algorithms currently used by system operators and planners are based on heuristics and have severe limitations. For example, common power flow algorithms, which are used for scenario analysis, typically fail to provide useful system information in the absence of accurate estimates of system state. Optimal power flow algorithms, which are used for planning system adjustments, typically use inadequate techniques for handling discrete variables and identifying key control actions. Security assessment algorithms typically require expensive system simulations that prevent frequent security analyses. These deficiencies may compromise system performance as grids become more complex, variable and unpredictable, and are operated closer to their limits. In this work, modeling and numerical optimization techniques are explored for overcoming many of the limitations associated with the algorithms used for scenario analysis, planning and control, and online security assessment. The proposed techniques have been tested on mid to large-scale real power networks obtained from South American, North American, and European electric institutions. The results demonstrate the potential benefits of the proposed techniques for improving the tools used by operators and planners for maintaining system reliability and efficiency.

# Acknowledgments

It has been a long time since saying goodbye to my family, friends and beloved Ecuador, and getting on that "flying carrot" to pursue my dream of studying abroad, as my father had done. I was 18 years old and had a simple plan: go to California, stay with my relatives, learn English, and somewhere and somehow start studying. Being used to a comfortable life in Ecuador, surrounded by family, friends, hobbies, good music and great food, and never really having to step out of my comfort zone, I was set for a challenging experience. It did not take me long to realize that my determination for accomplishing what I had set out for had better be strong if I wanted to succeed. Looking back, I have also realized that the doors opened by the kind actions of many people were going to play an equal or perhaps greater role in my ability to accomplish my dream.

It has been a long time since first arriving in Southern California and staying with my uncle Pepe and his wife Teri. I am very thankful to them for opening the first door to me by letting me stay with them and helping me get settled. I will always remember the sacrifices they made to give me this opportunity. This was one of the most difficult times of my life because I was immature and my world changed from one day to the next, but without their help, I would have not become a California resident and hence not opened the possibility to start studying in this country.

It has been a long time since moving to Northern California to start attending community college while staying with my uncle Carlos and his wife Susan. I am very thankful to them for opening the second door to me by letting me stay with them and helping get started with community college. Their support was crucial in helping me gain the strength required to adapt again to a new environment. Without it, I would have not started one of the most rewarding periods of my life that lasted until I tried my luck on the PhD qualification exams.

Yes, the exams were difficult (in format) and left me a bit wounded for some time. But

they also had the positive effect of setting my path right towards an invaluable experience as a PhD student. I was lucky enough to meet Walter, my current research adviser, who not only is an expert in my field of academic interest, but is also a fun person, a great friend, a great mentor, and a great tennis player. I am very thankful to him for opening the third door to me by accepting me as his PhD student. Aside from helping me improve my numerical optimization, writing, and tennis skills, he also connected me with people and organizations that defined my PhD experience.

The first are Adam and Bob from the Electric Power Research Institute. Aside from being the project managers whose projects are allowing me to get a PhD today, they have provided me with a great and flexible working environment. Also, because of them, I have presented my work in several conferences and have acquired invaluable knowledge about power systems. For all of these, I am very thankful.

The second is the Aurora Solar team, in particular Chris and Dele. I am very thankful to them for allowing me to work in their company and for treating me as part of the Aurora family. By working with them I have been able to acquire invaluable knowledge about photovoltaic systems and about the exciting solar industry. Aurora's financial support and flexible working environment have been of great importance for completing my PhD.

Last but not least, I am very thankful to my family. In particular, I am very thankful to my father Jorge for the immense sacrifices he made to give me the opportunity to come to this country to study. I know it was also very hard for him to give up the many nice things we did together in Ecuador. I am very thankful to my mother Cecilia for her support during all these years. Seeing her overcoming many difficult challenges and doing well has given me strength to overcome my challenges. Finally, I am very thankful to my wife Lakkhana for trying hard to adjust to a new country and for bearing with my unusual lifestyle.

It is hard to believe that I am finally accomplishing my goal. As I mentioned before, it has been a long time, but also an amazing experience. Thank you all for helping me make this happen!

# Contents

# List of Tables

# List of Figures

# List of Acronyms and Abbreviations

| | |
|---|---|
| AC | Alternating Current |
| alOPF | Alternating Optimal Power Flow |
| alOPF-CD | alOPF with Coordinate Descent strategy |
| alOPF-DL | alOPF with Discrete Line-Search strategy |
| alOPF-OT | alOPF with Optimal Thresholding strategy |
| alOPF-RO | alOPF with Rounding strategy |
| alOPF-TH | alOPF with Thresholding strategy |
| ANN | Artificial Neural Network |
| API | Application Programming Interface |
| COB | Critical Operating Boundaries |
| CPU | Central Processing Unit |
| crOPF | Continuous Relaxation Optimal Power Flow |
| DC | Direct Current |
| FB | Fischer-Burmeister |
| GB | Gigabytes |
| GHz | Gigahertz |
| IEEE | Institute of Electrical and Electronics Engineers |
| IPCPM | Interior Point Cutting-Plane Method |
| LP | Linear Program |
| LTC | Load Tap-Changer |
| MILP | Mixed-Integer Linear Program |
| MPEC | Mathematical Program with Equilibrium Constraints |
| MUMPS | Multifrontal Massively Parallel Sparse Direct Solver |

| | |
|---|---|
| MVA | Mega Volt Amperes |
| MVAr | Mega Volt Amperes Reactive |
| MW | Mega Watts |
| NR | Newton-Raphson |
| OPF | Optimal Power Flow |
| PC | Personal Computer |
| PF | Power Flow |
| RAM | Random-Access Memory |
| rrOPF | Round and Resolve Optimal Power Flow |
| vPF | Voltage-Controlled Power Flow |

# List of Mathematical Symbols

| | |
|---|---|
| $\mathbb{C}$ | Set of complex numbers |
| $\mathbb{C}^n$ | Set of $n$-dimensional complex vectors |
| $\mathbb{C}^{m\times n}$ | Set of $m$-by-$n$ complex matrices |
| $\mathbb{N}$ | Set of natural numbers, *i.e.*, $\{1, 2, \ldots\}$ |
| $\mathbb{R}$ | Set of real numbers |
| $\mathbb{R}_+$ | Set of non-negative real numbers |
| $\mathbb{R}_{++}$ | Set of positive real numbers |
| $\mathbb{R}^n$ | Set of $n$-dimensional real vectors |
| $\mathbb{R}^{m\times n}$ | Set of $m$-by-$n$ real matrices |
| $\mathbb{Z}$ | Set of integers |
| $\mathbb{Z}_+$ | Set of non-negative integers |
| $\mathbb{Z}_{++}$ | Set of positive integers |
| $[a, b]$ | Set of real numbers between $a$ and $b$ |
| $[n]$ | Discrete set given by $\{1, \ldots, n\}$ |
| $e$ | Base of natural logarithm or vector of ones |
| $e_i$ | Elementary vector ($i$-th entry is one and the rest are zero) |
| $\mathcal{N}(\mu, \Sigma)$ | Gaussian random vector with mean $\mu$ and covariance $\Sigma$ |
| $\subset$ | Set containment relation |
| $\cap$ | Set intersection operator |
| $\cup$ | Set union operator |
| $\times$ | Set product or scalar multiplication |
| $\setminus$ | Set difference |
| $\in$ | Set membership |

| | |
|---|---|
| $\notin$ | Negation of set membership |
| $\{\|\}$ | Defines properties of elements of set |
| sup | Supremum of set |
| inf | Infimum of set |
| $\pm$ | Plus or minus |
| $\perp$ | Complementarity relation |
| $:=$ | Definition |
| $\neq$ | Negation of equality |
| $\sim$ | Indication of probability distribution |
| $\succ$ | Positive definite |
| $\succeq$ | Positive semi-definite |
| $\Longleftrightarrow$ | Equivalence |
| $\Longrightarrow$ | Implication |
| $\ll$ | Much less than |
| $\gg$ | Much greater than |
| $\square$ | Termination of proof |
| lim | Limit value |
| $\rightarrow$ | Convergence or function map |
| $\downarrow$ | Monotonically decreasing convergence |
| $\leftarrow$ | Assignment of variable |
| $\sum$ | Summation of elements of sequence |
| $\prod$ | Product of elements of sequence |
| $'$ | First derivative indicator |
| $\nabla$ | First derivative (gradient) operator |
| $\nabla^2$ | Second derivative operator |
| $\circ$ | Element-wise product |
| $T$ | Transpose operator |
| $*$ | Conjugate transpose operator or optimality indicator |
| $< a, b >$ | Inner product between $a$ and $b$ |
| $\cdot$ | Function argument |

| | |
|---|---|
| $\lvert \cdot \rvert$ | Magnitude of complex number or cardinality of set |
| $\lVert \cdot \rVert_2$ | L2 norm |
| $\lVert \cdot \rVert_1$ | L1 norm |
| $\lVert \cdot \rVert_\infty$ | Infinity norm |
| $\mathrm{sgn}(\cdot)$ | Sign function |
| $\log(\cdot)$ | Natural logarithm function |
| $\mathrm{E}(\cdot)$ | Expected value function |
| $\mathrm{Tr}(\cdot)$ | Trace function |
| $\lambda_{\max}(\cdot)$ | Maximum eigenvalue function |
| $\lambda_{\min}(\cdot)$ | Minimum eigenvalue function |
| $\kappa(\cdot)$ | Condition number function |
| $\Re(\cdot)$ | Real part function |
| $\Im(\cdot)$ | Imaginary part function |

# Chapter 1

# Introduction

## 1.1 Electric Power Networks

Electricity is a key commodity in modern societies. It not only powers our laptops and televisions, but it also provides us many of the basic items needed for living comfortably, *e.g.*, heating, cooling and lighting, and facilitates economic growth. In fact, it is so ingrained in our way of life that many of us take it for granted. The reality is that the wide-spread availability of this commodity and its accessible cost are the result of the workings of one of man's most complex creations: electric power grids. These systems, which are also known as electric power networks, are collections of power generating sources and power consuming elements, or loads, interconnected through transmission lines, transformers and ancillary equipment [1] [84]. They can cover wide geographical regions and include tens of thousands of components [72]. Electrically, they are typically balanced three-phase voltage systems. That is, each section of the network has three conductors, and each one carries a voltage sinusoid of equal magnitude and frequency, but having a phase difference of 120° [28]. These systems are expected to work around the clock transporting electricity from the various locations where it is generated, *e.g.*, power plants, to the locations where it is consumed, *e.g.*, houses, factories, and shopping malls. They must be both efficient and reliable. When a failure occurs, the consequences can be catastrophic. For example, the 31-hour blackout that occurred in North America in 2003 was estimated to have affected around 50 million people and cost between 4 to 6 billion dollars [72] [86]. Consequently, significant resources are invested by governments and other organizations in order to improve the hardware and software used by these complex systems as well as the technical skills of its workforce.

## 1.2 Role of Numerical Optimization

Numerical optimization plays a key role in almost every aspect of the operation and planning of electric power networks. Its applications cover times frames ranging from seconds to years. For example, optimal power flow problems are solved every few minutes or hours in order to determine how to adjust the system to minimize cost while maintaining security [26]. Unit commitment problems are solved many times a day in order to clear day-ahead power markets and determine generator schedules [82]. Power flow problems, which are also known as load flow problems, are solved regularly during operations planning in order to determine system performance under different loading conditions and network topology configurations [1]. Voltage stability margins are also obtained through numerical optimization for determining how far a system is from a possible collapse [8]. Network expansion problems are solved annually to determine how to upgrade a power network to meet future demands [10]. Similarly, optimal capacitor placement problems are solved to determine where capacitors should be located in order to improve system performance [7]. Improvements in the algorithms used for solving all of these different problems are highly desirable since the benefits can be substantial. For example, the probability of a blackout can be significantly reduced by improving the speed and robustness of algorithms that determine system security violations and provide corrective actions. Also, millions of dollars can be saved annually by improving the solution quality obtained from algorithms that clear power markets [16] [69].

## 1.3 Challenges

In general, optimization problems associated with electric power networks are challenging to solve. The reasons are numerous: First, power networks are characterized by non-convex equality constraints known as the "power flow equations" [4]. These constraints make it impossible for practical algorithms that consider these constraints to have global optimality or even feasibility guarantees. Second, many of the control devices used in power networks, such as switched shunt capacitors and tap-changing transformers, have a finite set of possible configurations, and hence accurate models require discrete variables [81]. Third, most electric power networks have no fast energy storage. Energy that is consumed is essentially produced at the same time by some generator. When a contingency occurs, *e.g.* a generator or transmission line outage, a power imbalance can cause cascading failures and

even a system-wide blackout in a matter of seconds. This is even more challenging when the economic forces are pushing systems to operate closer to their security limits [90]. Therefore, optimization algorithms used during operations need to be timely in detecting problems and suggesting corrective actions. Lastly, uncertainty is becoming an important factor to consider due to the planned increase of variable and distributed generation, in particular solar and wind [32]. Unfortunately, modeling this uncertainty within optimization problems significantly increases their size and complexity.

## 1.4 Contributions

This work focuses on developing and applying modeling and numerical optimization techniques for improving the way some of the most fundamental problems associated with power network operations and planning are solved. The ultimate goal is to provide system operators and planners with better tools to analyze and control their systems. With better tools, system reliability and efficiency can be improved, which is particularly crucial for the new more challenging scenario characterized by economic pressures, increased system complexity, and large-scale renewable energy penetration. More specifically, contributions are made in the following related areas:

- **Scenario Analysis:** During operations planning, scenario analysis needs to be performed regularly in order to determine whether an electric power network is adequately designed and configured to operate under various potential scenarios. Performing such an analysis requires solving power flow problems. Unfortunately, widely-used methods for solving such problems have serious robustness deficiencies, which complicate and possibly compromise system analysis. In this work, these deficiencies are addressed and alternatives that are more robust are described. The proposed techniques have theoretical guarantees and allow operators to obtain information about the state of power systems more reliably. Early ideas and results of this work have been included in technical reports published by the Electric Power Research Institute [36] [37], and published in the proceeding of the 2013 North American Power Symposium [62]. A more complete manuscript has been submitted for publication to the Journal of Computational Optimization and Applications [63].

- **Planning and Control**: Generator dispatch and other control device adjustments need to be planned regularly in operations planning to ensure that an electric power

network maintains system security and efficiency under expected operating conditions or potential scenarios. These system adjustments are obtained by solving optimal power flow problems, and require taking into consideration the discrete nature of certain system controls and identifying a practical set of control actions. However, current methods for solving these problems typically use inadequate rounding techniques for handling discrete variables, and utilize a number of control actions that is not practical for operations. In this work, techniques for handling discrete variables and for obtaining a practical number of control actions in optimal power flow problems are described. The proposed techniques are based on using a smooth sparsity-inducing penalty to obtain sparse controls and distributed exploration for finding promising discrete variable configurations. Early ideas and results of this work were presented at the 2015 Innovative Smart Grid Technologies Conference and will be published in the conference proceedings [64]. More recent results have been included in a technical report published by the Electric Power Research Institute [39].

- **Online Security Assessment:** System operators currently rely on offline simulation studies and experience for assessing system security in real time. However, as electric power networks become more variable and unpredictable and operate closer to their limits, security problems may arise much more suddenly and frequently, and may involve issues that have not been experienced before. In this work, the critical operating boundaries problem is defined, and an efficient method for performing security assessment in real time is described. The proposed techniques are based on a linear system model, allow finding the nearest thermal and voltage security boundaries in terms of changes in the loading conditions, and suggest actions for restoring security margins. Early ideas and results of this work were presented at the 2015 Innovative Smart Grid Technologies Conference and will be published in the conference proceedings [31]. More recent results have been included in a technical report published by the Electric Power Research Institute [38].

## 1.5 Thesis Outline

The contents are organized as follows: Chapter 2 introduces notation and describes the mathematical model used for representing an electric power network. Chapter 3 provides an overview of scenario analysis in operations planning and the power flow problem, and

describes the robustness issues of current methods as well as the proposed improvements. Chapter 4 provides an overview of system planning and control and the optimal power flow problem, and describes the deficiencies associated with current methods as well as the proposed techniques for overcoming them. Chapter 5 provides and overview of online security assessment and the issues associated with current practices, introduces the critical operating boundaries problem, and describes the proposed solution techniques. Lastly, Chapter 6 summarizes the work and suggests future research directions.

# Chapter 2

# Power Network Model

In this chapter, the mathematical model used for representing an electric power network is introduced. The model follows the common and simple way to represent one such a network, which assumes that perfect symmetry exists between all the three phases of the system, treats the system as single-phase, and represents each transmission element as a lumped-circuit line model ($\pi$-model) [4] [28].

## 2.1 Components

A power network is modeled by a graph of $n \in \mathbb{N}$ nodes, which are commonly referred to as *buses*, connected through a set of $\mathcal{B} \subset [n] \times [n]$ branches, where $[n] := \{1, \ldots, n\}$. Generators, loads, and shunt devices are connected to these buses. To keep the presentation simple, it is assumed that each bus has exactly one of each of these components connected to it.

### 2.1.1 Buses

Each bus $k \in [n]$ carries a voltage sinusoid of magnitude $v_k \in \mathbb{R}$ and phase angle $\theta_k \in \mathbb{R}$. Typically, one of the buses with a generator connected to it, say bus $s \in [n]$, is chosen as a "slack bus". The function of this bus is to provide sufficient power to account for any load or losses not known in advance as well as to provide a reference angle from which all other phase angles are measured [1] [4]. The rest of the buses are partitioned into the set $\mathcal{R} \subset [n]$ of voltage regulated buses and the set $\mathcal{U} \subset [n]$ of voltage unregulated buses. Regulated buses have generators connected to them that have the capability of adjusting

their reactive power to maintain their voltage magnitude fixed, or regulated, at some set point. For each $k \in \mathcal{R}$, $v_k^t \in \mathbb{R}$ denotes the set point for the voltage magnitude at bus $k$.

Typically, buses have desirable voltage magnitude ranges that prevent equipment damage [87]. For bus $k \in [n]$, this range is given by

$$v_k^{\min} \leq v_k \leq v_k^{\max}, \tag{2.1}$$

where $v_k^{\min},\ v_k^{\max} \in \mathbb{R}_{++}$. Typical ranges are from 0.9 to 1.1 per unit bus nominal voltage.

### 2.1.2 Branches

Each branch $(k, m) \in \mathcal{B}$ corresponds to a transmission line or transformer, including phase shifters, and is represented by the model given in Figure 2.1. A branch's series and shunt admittances are given by

$$y_{km} := g_{km} + jb_{km} \in \mathbb{C} \tag{2.2}$$
$$y_{km}^{sh} := g_{km}^{sh} + jb_{km}^{sh} \in \mathbb{C} \tag{2.3}$$
$$y_{mk}^{sh} := g_{mk}^{sh} + jb_{mk}^{sh} \in \mathbb{C}, \tag{2.4}$$

where $g_{km}$ and $b_{km} \in \mathbb{R}$ denote the series conductance and susceptance, respectively, $g_{km}^{sh}$ and $b_{km}^{sh} \in \mathbb{R}$ denote the shunt conductance and susceptance, respectively, and $j$ denotes the imaginary unit. Transformer turns ratios, which for non-transformer branches are unity, are given by

$$t_{km} := a_{km} e^{j\vartheta_{km}} \in \mathbb{C} \tag{2.5}$$
$$t_{mk} := a_{mk} e^{j\vartheta_{mk}} \in \mathbb{C}, \tag{2.6}$$

where $a_{km},\ a_{mk} \in \mathbb{R}_{++}$ and $\vartheta_{km},\ \vartheta_{mk} \in \mathbb{R}$. These turns ratios imply the voltage relations

$$v_p e^{j\theta_p} = t_{km} v_k e^{j\theta_k} = a_{km} v_k e^{j(\theta_k + \vartheta_{km})} \tag{2.7}$$
$$v_q e^{j\theta_q} = t_{mk} v_m e^{j\theta_m} = a_{mk} v_m e^{j(\theta_m + \vartheta_{mk})}, \tag{2.8}$$

where $p$ and $q$ denote the internal points of the branch element shown in Figure 2.1 [4]. For $(k, m) \notin \mathcal{B}$, *i.e.*, for buses $k$ and $m$ that are not connected by a branch, the conventions $y_{km} = y_{km}^{sh} = y_{mk}^{sh} = 0$ and $t_{km} = t_{mk} = 1$ are used.

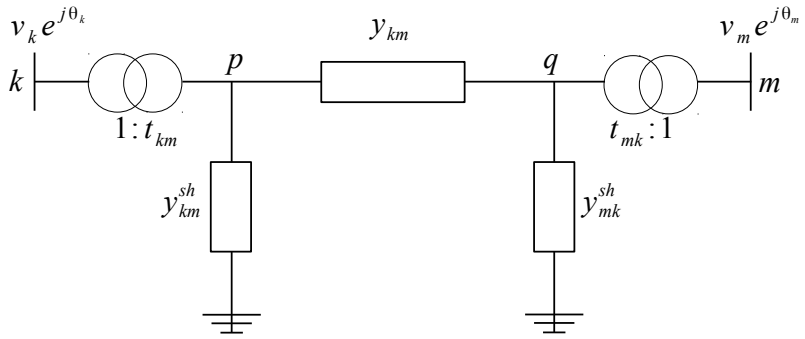


Figure 2.1: Model for branch element of power network [4].

The branch current $i_{km} \in \mathbb{C}$ leaving bus $k$ towards bus $m$ is given by the expression

$$i_{km} := t_{km}^{*}\big(y_{km}^{sh} v_p e^{j\theta_p} + y_{km}(v_p e^{j\theta_p} - v_q e^{j\theta_q})\big) \tag{2.9}$$

$$= a_{km}^2 (y_{km}^{sh} + y_{km}) v_k e^{j\theta_k} - t_{km}^{*} t_{mk} y_{km} v_m e^{j\theta_m}, \tag{2.10}$$

where $*$ denotes complex conjugate. The branch current $i_{mk} \in \mathbb{C}$ leaving bus $m$ towards bus $m$ is given by a similar expression that is obtained by interchanging $k$ and $m$ in (2.10).

The branch apparent power flow $S_{km} \in \mathbb{C}$ from bus $k$ to bus $m$ is given by

$$S_{km} := v_k e^{j\theta_k} i_{km}^{*} \tag{2.11}$$

$$= P_{km} + jQ_{km}, \tag{2.12}$$

where $P_{km},\ Q_{km} \in \mathbb{R}$ denote the branch active and reactive power flows, respectively, which are given by

$$P_{km} := a_{km}^2 (g_{km}^{sh} + g_{km}) v_k^2 - a_{km} a_{mk} v_k v_m (g_{km} \cos\theta_{km} + b_{km} \sin\theta_{km}) \tag{2.13}$$

$$Q_{km} := -a_{km}^2 (b_{km}^{sh} + b_{km}) v_k^2 - a_{km} a_{mk} v_k v_m (g_{km} \sin\theta_{km} - b_{km} \cos\theta_{km}). \tag{2.14}$$

The quantity $\theta_{km} \in \mathbb{R}$ represents the phase angle difference across the branch and is given by

$$\theta_{km} := \theta_k - \theta_m - \phi_{km}, \tag{2.15}$$

where

$$\phi_{km} := \begin{cases} 0, & \text{if } k = m \\ -\vartheta_{km} + \vartheta_{mk}, & \text{otherwise.} \end{cases} \tag{2.16}$$

The quantity $\phi_{km} \in \mathbb{R}$ represents the phase shift of a phase-shifting transformer, and is zero for other branch types. Again, analogous expressions for $S_{mk}$, $P_{mk}$, $Q_{mk}$, $\theta_{mk}$ and $\phi_{mk}$ are obtained by interchanging $k$ and $m$.

Transmission lines and transformers have thermal ratings. To prevent damage, thermal constraints are typically imposed on the magnitude of the current or apparent power flow through a branch. In terms of the quantities defined above, for a branch $(k, m) \in \mathcal{B}$, these constraints are given by

$$|i_{km}| \leq i_{km}^{\max} \quad \text{and} \quad |i_{mk}| \leq i_{km}^{\max}, \tag{2.17}$$

or

$$|S_{km}| \leq S_{km}^{\max} \quad \text{and} \quad |S_{mk}| \leq S_{km}^{\max}, \tag{2.18}$$

where $i_{km}^{\max},\ S_{km}^{\max} \in \mathbb{R}_{++}$, and $|\cdot|$ gives the magnitude of a complex number [16].

Phase-shifting transformers allow adjusting the phase angle $\phi_{km}$ for controlling the active power flow through the branch [34] [50]. Load Tap-Changing (LTC) transformers allow adjusting the turns ratio magnitude $a_{km}$ for improving voltage profiles and reducing active power transmission losses [34] [50]. Possible values for $a_{km}$ form a discrete set that typically consists of many, say 30, evenly spaced positions between 0.9 to 1.1 per unit [50] [70] [81]. Limits for these controls are given by the constraints

$$a_{km}^{\min} \leq a_{km} \leq a_{km}^{\max} \tag{2.19}$$

$$\phi_{km}^{\min} \leq \phi_{km} \leq \phi_{km}^{\max}, \tag{2.20}$$

where $a_{km}^{\min},\ a_{km}^{\max} \in \mathbb{R}_{++}$, and $\phi_{km}^{\min},\ \phi_{km}^{\max} \in \mathbb{R}$.

### 2.1.3 Shunt Devices

Shunt devices are reactive power compensation devices such as capacitor and reactor banks [81] that are typically used for improving the voltage profile of a power network [30]. They are modeled by an admittance between a given bus and ground. More specifically, the admittance of the shunt device connected at bus $k \in [n]$ is given by

$$y_k^{sh} := g_k^{sh} + j b_k^{sh}, \tag{2.21}$$

where $g_k^{sh},\ b_k^{sh} \in \mathbb{R}$ are the conductance and susceptance, respectively.

The current $i_k^{sh} \in \mathbb{C}$ leaving bus $k$ through the shunt device connected at that bus is given by

$$i_k^{sh} := y_k^{sh} v_k e^{j\theta_k}. \tag{2.22}$$

Hence, the apparent power flow $S_k^{sh}$ from bus $k$ to ground is given by

$$S_k^{sh} := v_k e^{j\theta_k} \left(i_k^{sh}\right)^* \tag{2.23}$$

$$= P_k^{sh} + jQ_k^{sh}, \tag{2.24}$$

where $P_k^{sh},\ Q_k^{sh} \in \mathbb{R}$ denote the shunt active and reactive power flows, respectively, which are given by

$$P_k^{sh} := g_k^{sh} v_k^2 \tag{2.25}$$

$$Q_k^{sh} := -b_k^{sh} v_k^2. \tag{2.26}$$

Switched shunt devices allow adjusting the susceptance $b_k^{sh}$ by closing or opening switches that connect or remove capacitors or reactors in parallel. Possible values for $b_k^{sh}$ form a discrete set that typically consists of values that are not necessarily closely nor evenly spaced [50] [81]. Limits for these controls are given by

$$b_k^{\min} \leq b_k^{sh} \leq b_k^{\max}, \tag{2.27}$$

where $b_k^{\min},\ b_k^{\max} \in \mathbb{R}$.

### 2.1.4 Generators and Loads

A generator connected to bus $k \in [n]$ injects active power $P_k^g \in \mathbb{R}$ and reactive power $Q_k^g \in \mathbb{R}$ at that bus. Similarly, a load connected to bus $k \in [n]$ consumes active power $P_k^l \in \mathbb{R}$ and reactive power $Q_k^l \in \mathbb{R}$.

A generator has limits on its active and reactive powers. These limits define a region where the generator can operate safely [6] [53]. For the generator connected to bus $k \in [n]$, these limits are given by

$$P_k^{\min} \leq P_k^g \leq P_k^{\max} \tag{2.28}$$

$$Q_k^{\min} \leq Q_k^g \leq Q_k^{\max}, \tag{2.29}$$

where $P_k^{\min},\ P_k^{\max},\ Q_k^{\min},\ Q_k^{\max} \in \mathbb{R}$.

Typically, a generator adjusts its reactive power $Q_k^g$ inside the range $[Q_k^{\min}, Q_k^{\max}]$ in order to regulate the voltage magnitude of the bus to which it is connected. When reactive power outside this range is needed to keep the bus voltage magnitude at its set point, this voltage magnitude may drift away, effectively turning the bus into an unregulated bus. It may be the case that a generator is configured to regulate the voltage magnitude of a remote bus, *i.e.*, a neighboring bus that is different from the one to which the generator is connected. Typically, only a few generators are configured in this way in a power network. Although the algorithms implemented in this work can handle this configuration, it is assumed here that all generators regulate the bus to which they are connected in order the keep the presentation simple.

## 2.2 Nodal Admittance Matrix

From (2.10) and (2.22), the total current $i_k \in \mathbb{C}$ leaving bus $k$ through the branches and shunt device connected at that bus is given by

$$i_k := i_k^{sh} + \sum_{m \neq k} i_{km} \tag{2.30}$$

$$= \big(y_k^{sh} + \sum_{m \neq k} a_{km}^2 (y_{km}^{sh} + y_{km})\big) v_k e^{j\theta_k} - \sum_{m \neq k} t_{km}^* t_{mk} y_{km} v_m e^{j\theta_m}. \tag{2.31}$$

Defining the matrix $Y \in \mathbb{C}^{n \times n}$ by

$$Y_{km} := \begin{cases} y_k^{sh} + \sum_{j \neq k} a_{kj}^2 (y_{kj}^{sh} + y_{kj}), & \text{if } k = m \\ -a_{km} a_{mk} y_{km}, & \text{otherwise,} \end{cases} \tag{2.32}$$

and using $\phi_{km}$ as defined in (2.16), a more compact expression for $i_k$ is obtained:

$$i_k = \sum_{m \in [n]} Y_{km} e^{j\phi_{km}} v_m e^{j\theta_m}. \tag{2.33}$$

The real and imaginary parts of $Y$ are denoted by $G$ and $B$, respectively. These matrices are symmetric. The matrix $\tilde{Y}$ defined by

$$\tilde{Y}_{km} := Y_{km} e^{j\phi_{km}}, \tag{2.34}$$

for each $k,\ m \in [n]$ is known as the *nodal admittance matrix*, and is commonly used in the literature for characterizing power networks [4]. Two useful matrices that are obtained in a similar way but from $G$ and $B$ are $\tilde{G}$ and $\tilde{B}$. Their elements are given by

$$\tilde{G}_{km} := G_{km} e^{j\phi_{km}} \tag{2.35}$$

$$\tilde{B}_{km} := B_{km} e^{j\phi_{km}}, \tag{2.36}$$

for each $k,\ m \in [n]$. Since $G$ and $B$ are symmetric and $\phi_{km} = -\phi_{mk}$, $\tilde{G}$ and $\tilde{B}$ are Hermitian.

## 2.3 Power Flow Constraints

Electric power networks obey the law of conservation of power: at every bus, the net sum of powers injected is zero. Using the notation introduced above, these constraints can be expressed as

$$(P_k^g + jQ_k^g) - (P_k^l + jQ_k^l) - S_k^{sh} - \sum_{m\in[n]} S_{km} = 0, \tag{2.37}$$

for each $k \in [n]$. In terms of the matrix $Y$, these equations can be expressed as

$$(P_k^g + jQ_k^g) - (P_k^l + jQ_k^l) - \sum_{m\in[n]} v_k v_m Y_{km}^* e^{j(\theta_k - \theta_m - \phi_{km})} = 0, \tag{2.38}$$

for each $k \in [n]$. Equivalently, by separating real and imaginary components, they can be expressed as

$$P_k^g - P_k^l - \sum_{m\in[n]} v_k v_m (G_{km}\cos\theta_{km} + B_{km}\sin\theta_{km}) = 0 \tag{2.39}$$

$$Q_k^g - Q_k^l - \sum_{m\in[n]} v_k v_m (G_{km}\sin\theta_{km} - B_{km}\cos\theta_{km}) = 0, \tag{2.40}$$

for each $k \in [n]$, where $\theta_{km}$ is as defined in (2.15) [1] [4]. The left-hand sides of equations (2.39) and (2.40) are commonly referred to as the active and reactive power mismatches at bus $k$, respectively.

## 2.4 Voltage Regulation Constraints

As already noted, a generator may adjust its reactive power in order to keep its bus voltage magnitude fixed at a given set point. When its reactive power reaches one of its limits, the voltage magnitude at the bus regulated by that generator may move away from its set point. When this occurs, the bus essentially behaves as one without voltage regulation. The constraints that capture this behavior are given by

$$v_k = v_k^t + y_k - z_k \tag{2.41}$$

$$0 \leq \left(Q_k^g - Q_k^{\min}\right) \perp y_k \geq 0 \tag{2.42}$$

$$0 \leq \left(Q_k^{\max} - Q_k^g\right) \perp z_k \geq 0, \tag{2.43}$$

for each $k \in \mathcal{R}$, where $y_k,\ z_k \in \mathbb{R}$ represent positive and negative deviations, respectively, of regulated voltage magnitudes from their set points [74]. These constraints restrict the bus voltage magnitude to be at its set point unless the generator's reactive power reaches one of its limits. When the reactive power reaches its upper limit, the voltage magnitude may be lower than the set point. On the other hand, when the reactive power reaches its lower limit, the voltage magnitude may be higher than the set point. The symbol $\perp$ denotes complementarity and is defined by $a \perp b \iff ab = 0$, for all $a,\ b \in \mathbb{R}$. Constraints of the form of (2.42) and (2.43) are known as *complementarity constraints* [47] [71] [73] [74].

## 2.5 Units

Unless otherwise stated, the following units are assumed throughout for the main power network quantities: MW for active power, MVAr for reactive power, MVA for apparent power, radians for phase angles, and per unit bus nominal voltage for voltage magnitudes. For normalizing power quantities, the system base power is used, which typically equals 100 MVA.

# Chapter 3

# Scenario Analysis

## 3.1 Background and Overview

During operations planning, scenario analysis is performed regularly in order to determine whether an electric power network is adequately designed and configured to operate under various potential scenarios. These scenarios may consider load changes, planned equipment outages, network topology changes, or contingencies. Security criteria for assessing adequate system operation may consider voltage limit violations, thermal overloading of transmission lines or transformers, and voltage stability margins. To analyze each scenario, the *power flow problem* must be solved. This problem, which is also known as the *load flow problem*, consists of determining the steady-state operating point of an electric power network. That is, to determine the steady-state voltage magnitudes and phase angles at every bus of the network and any unknown generator powers [1] [4]. The various security quantities, *e.g.*, thermal violations, can be easily obtained from the system state. Unfortunately, current power flow solvers have questionable robustness [36] [37]. In particular, they may fail to find a solution when one exists, or may fail to provide useful information about the potential physical causes of infeasibility, due to purely algorithmic issues [36] [37]. When a failure of this type occurs, system operators and planners are left with the difficult and time-consuming task of trying to obtain information about the system by other means, such as trying multiple methods and making system modifications. When this also fails, they are left with limited or incorrect knowledge about the system, which may jeopardize its reliability and efficiency.

This lack of robustness of current power flow solvers comes from the solution method

used, which typically consists of the Newton-Raphson (NR) method for solving systems of nonlinear equations combined with switching heuristics for enforcing generator reactive power limits and voltage regulation. This method is simple to implement, is fast, and can exhibit a quadratic rate of convergence, but it suffers from serious robustness issues. Its convergence, for example, is only local. That is, it requires a starting point sufficiently close to the solution, otherwise, the method may diverge or cycle [60]. Also, the quality of the search directions computed by this method depends on the condition number of the Jacobian of the power flow equations. In many cases this matrix may be singular or nearly singular, especially in large and heavily loaded networks, leading to convergence issues. Finally, the switching heuristics, often called "PV-PQ switching", may also conflict with the NR process causing jumps or cycles that affect its convergence [37].

A reader may wonder why, given the limitations, this method is still the most widely used method for solving power flow problems. The reason is that under normal operations, a good estimate of the solution is often known, and hence the method is usually very fast. In operations, power flow problems are solved repetitively as conditions of the system change. Typically, the solution of a recent problem is used as an initial point for a new problem. When the change is small, this solution is a very good estimate. However, under other circumstances such as during expansion planning, contingency analysis, or when merging adjacent systems, a good initial solution estimate may not be known and the limitations of the method become more apparent. Moreover, with current deregulated practices and the large-scale integration of renewable energy projected for the near future, systems are likely to operate closer to their limits, and be more unpredictable and variable. In this scenario, the limitations of the method are likely to become a major obstacle towards more efficient and reliable electric grids.

Many techniques have been proposed for improving the robustness of this widely-used NR-based power flow method. Among them are the use of an "optimal multiplier" and the use of integration or gradient flow techniques for improving convergence [12] [15] [35] [40] [59] [76]. These techniques, although very helpful for solving more ill-conditioned problems and preventing divergence, do not resolve the issue of requiring a good initial point, still have issues with near rank-deficient Jacobians, and require switching heuristics for handling reactive power limits and voltage regulation.

The issue of robustness to poor initial points has been addressed by several authors in the power flow literature. For example, some authors have suggested using the solution of

a linear approximation of the power flow problem as an initial point, and have tested this approach on power networks of a few hundred buses [79] [46]. A more elaborate idea based on homotopy has been proposed in [27], which consists of first solving the Direct Current (DC) power flow [91] and then gradually transforming this problem into the Alternating Current (AC) power flow. The authors test this idea on small cases that include the IEEE test problems, which have at most 300 buses, and one mid-size case of 2383 buses. They compare the number of function evaluations required by this method with those required by the NR method, but do not provide robustness results as all cases considered are solved by both methods. In [62], the authors also explore a homotopy approach for improving the robustness of NR-based power flow methods to poor initial point. The approach consists of formulating the power flow problem as a constrained optimization problem in which the objective function encourages desirable properties of bus voltage magnitudes. This approach can be interpreted as using *maximum a posteriori estimation* [13] [58] for determining the power flow solution given the network topology, configuration, and loading conditions. From this statistical perspective, the particular voltage properties encouraged in the objective function can be translated into a particular assumption about the prior distribution of a power flow solution before knowing the exact loading conditions.

The issues associated with the use of switching heuristics for enforcing generator reactive power limits and voltage regulation have received relatively less attention. In [89], the authors describe some of these issues, analyze several supplementary heuristics for overcoming them, but conclude that the original heuristics should be used as they correctly simulate the interactions between voltage regulation and generator reactive power limits. In our view, the weakness of this approach comes from trying to simulate these interactions during the NR process with infeasible points, since feasibility is generally achieved only in the limit. The authors also recommend using switching heuristics instead of optimization techniques for handling the complementarity constraints that model generator reactive power limits and voltage regulation. They argue that handling these constraints using such techniques may be problematic. Examples of optimization techniques for handling these constraints have been explored in [73] [74] [83] in the context of maximum loadability of power networks and the optimal power flow problem. In [83], the authors propose handling these constraints by means of an exact penalty function, but do not test this approach. On the other hand, the authors from [73] [74] use the interior point method described in [9] and test this approach on IEEE test cases.

In this chapter, a new power flow method referred to as *voltage-controlled Power Flow (vPF)* is presented that overcomes the limitations of the widely-used method consisting of NR and switching heuristics. The method formulates the power flow problem as a constrained optimization problem in which the objective function includes prior information about power flow solutions. The inclusion of prior information allows convexity to be added to the problem to favor physically meaningful iterates. It also makes the algorithm less sensitive to poor initial points and near rank-deficient Jacobians. It is shown that under reasonable assumptions, the proposed method is *globally convergent* [33] in the sense of always finding at least a stationary point of the sum of the squares of the infeasibilities. This work extends the work of [62], in which the benefits of including prior information about power flow solutions are explored, by including generator reactive power limits and voltage regulation, and by applying a more efficient optimization algorithm based on the *augmented Lagrangian method* [33] [67]. The interactions between generator reactive power limits and voltage regulation are modeled with complementarity constraints, and handled using smooth approximations of the Fischer-Burmeister function, as described in [71]. To improve the performance of the optimization algorithm, second-derivative information is exploited for computing primal search directions and updating Lagrange multiplier estimates. The performance of the proposed method on several power flow cases is presented. The test cases collected for this study come from real power networks obtained from several North American, South American, and European system operators. Some of the cases are large-scale cases having 45k and 58k buses. This contrasts to the common practice found in the power flow literature, which consists of testing algorithms only on IEEE test problems.

## 3.2 The Power Flow Problem

As mentioned before, the power flow or PF problem consists of determining all bus voltage magnitudes and angles as well as any unknown generator powers for specific loading conditions. The values obtained for these unknowns must make physical sense, *i.e.*, they must satisfy the power flow constraints described in Section 2.3 as well as the voltage regulation

constraints described in 2.4. Mathematically, this problem can be formulated as

$$\begin{aligned} &\text{find} && x \\ &\text{subject to} && f(x) = 0 \\ &&& v_k = v_k^t + y_k - z_k,\ k \in \mathcal{R} \\ &&& 0 \leq \left(Q_k^g - Q_k^{\min}\right) \perp y_k \geq 0,\ k \in \mathcal{R} \\ &&& 0 \leq \left(Q_k^{\max} - Q_k^g\right) \perp z_k \geq 0,\ k \in \mathcal{R}. \end{aligned} \tag{3.1}$$

Here, $f$ is the vector-valued function whose scalar function entries are the active and reactive power mismatches for each $k \in [n] \setminus \{s\}$, *i.e.*, the left-hand sides of equations (2.39) and (2.40). The vector $x$ is composed of voltage angles $\{\theta_i\}_{i\in[n]\setminus\{s\}}$, voltage magnitudes $\{v_i\}_{i\in[n]\setminus\{s\}}$, generator reactive powers $\{Q_i^g\}_{i\in\mathcal{R}}$, and regulated voltage magnitude deviations $\{y_i\}_{i\in\mathcal{R}}$ and $\{z_i\}_{i\in\mathcal{R}}$. The unknown active and reactive powers of the slack generator (bus $s$) are not included in the formulation of the problem. The reason is that they appear linearly in the power flow constraints and values for them can be obtained easily after values for the other variables are known. Additionally, for the slack bus, the known voltage magnitude and phase angle are treated as constants of the problem.

## 3.3 Newton-Raphson Power Flow

As already noted, the most widely used method for solving power flow problems is based on the NR method combined with switching heuristics for handling reactive power limits and voltage regulation. The NR method is a well known iterative algorithm for solving nonlinear systems of equations of the form $f(x) = 0$, where $f : \mathbb{R}^m \to \mathbb{R}^m$ is continuously differentiable and $m \in \mathbb{N}$. The method starts with an initial solution estimate $x_0$ and proceeds iteratively as follows: During iteration $k \in \mathbb{Z}_+$, it computes the next solution estimate $x_{k+1}$ by finding the root of the linearization of $f$ at $x_k$, *i.e.*, by solving

$$f_k + J_k(x_{k+1} - x_k) = 0, \tag{3.2}$$

where $J_k$ is the Jacobian of $f$ at $x_k$ and $f_k = f(x_k)$. This gives the update formula

$$x_{k+1} = x_k + p_k, \tag{3.3}$$

where $p_k$ satisfies the "Newton system" $J_k p_k = -f_k$. This procedure is repeated until $x_k$ satisfies $\|f(x_k)\|_\infty < \epsilon$ for some $k$ and some predefined $\epsilon > 0$.

This method is applied to the power flow problem by first assuming that generators do not reach their reactive power limits, and hence that voltage magnitudes at regulated buses are fixed at their set points. Under this assumption, the complementarity constraints of the power flow problem always hold and the power flow constraints have an equal number of equations and unknowns [1] [4] [75], which makes the NR method applicable. Table 3.1 shows what the known and unknown quantities are for each of the different bus types assuming that generators do not reach their reactive power limits. The nonlinear system of power flow equations is expressed as $f(x) = 0$, where $f$ consists of the active power mismatches at each bus $k \in [n] \setminus \{s\}$, and the reactive power mismatches at each bus $k \in \mathcal{U}$. The vector $x$ is composed of voltage angles $\{\theta_i\}_{i \in [n] \setminus \{s\}}$ and voltage magnitudes $\{v_i\}_{i \in \mathcal{U}}$. Reactive power mismatches at regulated buses $k \in \mathcal{R}$, as defined in Section 2.3, and reactive powers $\{Q_i^g\}_{i \in \mathcal{R}}$ of regulating generators need not be included in the NR process. The reason is that, since $\{Q_i^g\}_{i \in \mathcal{R}}$ appear linearly in the power flow equations, values for them are easily obtained from the values of the other variables such that the reactive power mismatches at their corresponding buses are kept at zero.

Table 3.1: Power flow known and unknown quantities.

| bus | type | known quantities | unknown quantities |
|---|---|---|---|
| $k = s$ | slack | $\theta_k$ and $v_k$ | $P_k^g$ and $Q_k^g$ |
| $k \in \mathcal{U}$ | unregulated | $P_k^g$ and $Q_k^g$ | $\theta_k$ and $v_k$ |
| $k \in \mathcal{R}$ | regulated | $P_k^g$ and $v_k$ | $\theta_k$ and $Q_k^g$ |

An improvement to the NR method is to include a line search procedure. This procedure consists of performing a one dimensional search along the computed search direction $p_k$ to determine a step length $\alpha_k$ such that the next iterate

$$x_{k+1} = x_k + \alpha_k p_k \tag{3.4}$$

is a "better" solution estimate. To define "better", a line search uses a *merit function* [33] [67], which for the NR method typically consists of $f^T f$. In the power flow context, this merit function is the sum of the squares of the bus power mismatches. The inclusion of a line search procedure has little or no negative impact on efficiency, but can improve robustness significantly. The line search procedure used here is based on bracketing and bisection and

enforces the *strong Wolfe conditions*

$$h(x_k + \alpha_k p_k) \leq h(x_k) + c_1 \alpha_k \nabla h(x_k)^T p_k \tag{3.5}$$

$$|\nabla h(x_k + \alpha_k p_k)^T p_k| \leq -c_2 \nabla h(x_k)^T p_k, \tag{3.6}$$

where $0 < c_1 < c_2 < 1$ and $h$ is the merit function $f^T f$ [67].

Unfortunately, a line search procedure alone is not enough to ensure convergence of the NR method. This is because the search directions may be poor in the sense of pointing along directions where the merit function decreases very little, leading to the method getting "stuck". This may happen when the search directions are not sufficient descent directions, which means that they get arbitrarily close to being orthogonal to the gradient, due to the Jacobian being nearly singular [60].

In power flow problems, near-singular Jacobians do arise. For example, it is known that the Jacobian of the power flow constraints is typically ill-conditioned for heavily loaded systems that are near their loadability limit [2] [17] [23] [24]. From our experience, the Jacobian is also ill-conditioned in regions where the voltage magnitudes have little physical meaning, *e.g.*, where some are much larger than the bus nominal voltage or close to zero. This latter case is closely related to poor initial points, since it has been observed that from poor initial points, the NR method may be attracted to points with such undesirable properties [36] [37] [62].

When the reactive power of a generator reaches its limit, voltage regulation may be lost. Switching heuristics, often called "PV-PQ switching", are commonly included in the NR method for simulating the inability of generators to keep their bus voltage magnitudes at their set points. These heuristics work by checking whether generator reactive power limit violations exist after each iteration of the NR method. If one exists, the reactive power of the corresponding generator is set to the violated limit and the corresponding bus is treated as an unregulated bus, allowing its voltage magnitude to deviate from its set point. Changes in bus types are done by modifying the sets $\mathcal{U}$ and $\mathcal{R}$. These heuristics essentially guess from a reactive power violation at iteration $k$ that the corresponding bus behaves as an unregulated bus at the solution. These guesses may be wrong and hence the heuristics also include other more complicated rules for reverting a bus that was previously changed to unregulated back to being regulated [89]. In practice, these heuristics are typically applied after the first or second iteration of the NR method in order to avoid making too many

poor initial guesses [1].

Algorithm 1 shows pseudocode for the widely-used NR-based power flow method equipped with the line search procedure and switching heuristics. This method is used as a benchmark method for evaluating the performance of the proposed power flow method described in the next section.

**Algorithm 1** NR power flow method

Given $x_0$ and feasibility tolerance $\epsilon_f > 0$
$k \leftarrow 0$
**while** True **do**
    **if** $k > 0$ **then**
        Perform PV-PQ switching heuristics
    **end if**
    **if** $\|f(x_k)\|_\infty < \epsilon_f$ **then**
        **return** $x_k$
    **end if**
    Compute search direction $p_k$ by solving $J_k p_k = -f_k$
    Perform line search with merit function $f^T f$ to obtain step length $\alpha_k$
    Update solution estimate $x_{k+1} \leftarrow x_k + \alpha_k p_k$
    $k \leftarrow k + 1$
**end while**

## 3.4 Modeling Approach

As noted in Section 3.1, the NR power flow method suffers from many limitations, two of which are its lack of robustness to poor initial points and to near rank-deficient Jacobians of the power flow constraints. In [62], it was found that by formulating the power flow problem as a constrained optimization problem in which the objective function encourages voltage magnitudes to remain near unity, robustness to poor initial points can be greatly improved. This idea can be extended to also encourage typical or desirable properties for all the other variables of the problem in such a way that the objective function is strongly convex and quadratic. With this, the dependency of the quality of the search directions on the rank of the Jacobian of the power flow constraints can be reduced, and global convergence to at least a stationary point of the sum of the squares of the infeasibilities can be obtained.

To eliminate the use of switching heuristics, the complementarity constraints that model

the interplay between voltage regulation and generator reactive power limits are approximated with "nice" smooth constraints that can be handled by a conventional nonlinear optimization algorithm. This is done by following the approach described and analyzed in detail in [71]. More specifically, a complementarity constraint

$$0 \leq a \perp b \geq 0, \tag{3.7}$$

where $a$ and $b$ are scalars, is first replaced by an equivalent constraint

$$a + b - \sqrt{a^2 + b^2} = 0. \tag{3.8}$$

Since the left-hand side of this constraint, which is called the *Fischer-Burmeister* (FB) function, is not differentiable at $(a, b) = (0, 0)$, the constraint is approximated with

$$a + b - \sqrt{a^2 + b^2 + 2\varsigma} = 0, \tag{3.9}$$

where $\varsigma$ is a small positive scalar. The resulting constraint is smooth and is equivalent to

$$a \geq 0,\ b \geq 0,\ ab = \varsigma. \tag{3.10}$$

Its feasible set is illustrated in Figure 3.1.

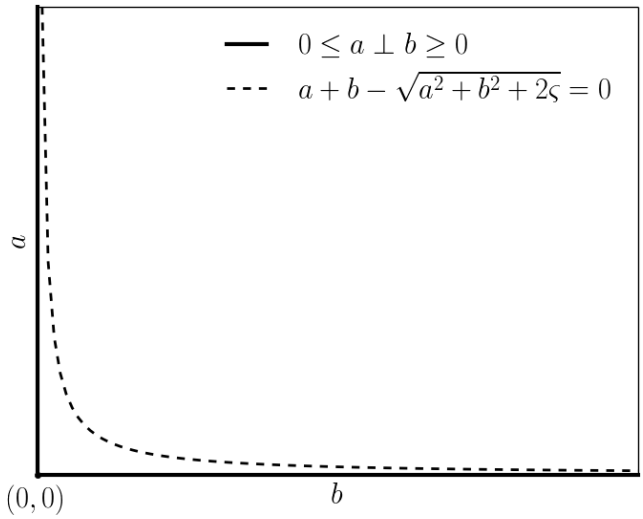


Figure 3.1: Smooth approximation of complementarity constraint.

The following lemma characterizes the error in (3.8) due to an error in (3.9).

**Lemma 3.4.1.** *For any positive $\epsilon$ and $\varsigma$,*

$$\left|a+b-\sqrt{a^2+b^2+2\varsigma}\right| \leq \epsilon \quad \Longrightarrow \quad \left|a+b-\sqrt{a^2+b^2}\right| \leq \epsilon+\sqrt{2\varsigma}. \tag{3.11}$$

*Proof.* Let $\epsilon$ and $\varsigma$ be positive scalars and suppose that

$$\left|a+b-\sqrt{a^2+b^2+2\varsigma}\right| \leq \epsilon. \tag{3.12}$$

Then, it follows that

$$\begin{aligned}
\left|a+b-\sqrt{a^2+b^2}\right| &\leq \left|a+b-\sqrt{a^2+b^2+2\varsigma}\right| + \left|\sqrt{a^2+b^2+2\varsigma}-\sqrt{a^2+b^2}\right| && (3.13)\\
&\leq \epsilon + \left|\sqrt{a^2+b^2+2\varsigma}-\sqrt{a^2+b^2}\right| && (3.14)\\
&= \epsilon + \sqrt{a^2+b^2+2\varsigma}-\sqrt{a^2+b^2} && (3.15)\\
&= \epsilon + h(a,b), && (3.16)
\end{aligned}$$

where

$$h(a,b) := \sqrt{a^2+b^2+2\varsigma}-\sqrt{a^2+b^2}. \tag{3.17}$$

Let $(\bar{a},\bar{b}) \neq (0,0)$ and suppose, without loss of generality, that $\bar{a} \neq 0$. Then, at $(\bar{a},\bar{b})$, $h$ is differentiable and

$$\frac{\delta h}{\delta a}(\bar{a},\bar{b}) = \bar{a}\left(\frac{1}{\sqrt{\bar{a}^2+\bar{b}^2+2\varsigma}}-\frac{1}{\sqrt{\bar{a}^2+\bar{b}^2}}\right) \neq 0. \tag{3.18}$$

Hence, for all $n \in \mathbb{N}$,

$$h(\bar{a},\bar{b}) \leq h(\bar{a}/n,\bar{b}). \tag{3.19}$$

By continuity, $h(\bar{a},\bar{b}) \leq h(0,\bar{b})$. It follows that $h(a,b) \leq h(0,0)$ for all $(a,b)$ and hence that

$$\begin{aligned}
\left|a+b-\sqrt{a^2+b^2}\right| &\leq \epsilon + h(0,0) && (3.20)\\
&= \epsilon + \sqrt{2\varsigma} && (3.21)
\end{aligned}$$

□

Combining the use of an objective function that includes prior knowledge about power

flow solutions, and smooth equality constraints that enforce power balance and the "switching" behavior of generators' voltage controllers, the power flow problem can be formulated as

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \varphi(x) \\ \text{subject to} \quad & f(x) = 0 \\ & \Phi(x) = 0 \\ & Ax = b, \end{aligned} \tag{3.22}$$

where $x$ is a vector composed of voltage magnitudes $\{v_i\}_{i\in[n]\setminus\{s\}}$, voltage angles $\{\theta_i\}_{i\in[n]\setminus\{s\}}$, reactive powers of generators $\{Q_i^g\}_{i\in\mathcal{R}}$, and positive and negative regulated voltage magnitude deviations $\{y_i\}_{i\in\mathcal{R}}$ and $\{z_i\}_{i\in\mathcal{R}}$, respectively. The function $\Phi$ is the smooth vector-valued function whose scalar function entries are the functions

$$\left(Q_k^g - Q_k^{\min}\right) + y_k(1+\varepsilon) - \sqrt{\left(Q_k^g - Q_k^{\min}\right)^2 + y_k^2 + 2\varsigma} \tag{3.23}$$

$$\left(Q_k^{\max} - Q_k^g\right) + z_k(1+\varepsilon) - \sqrt{\left(Q_k^{\max} - Q_k^g\right)^2 + z_k^2 + 2\varsigma}, \tag{3.24}$$

for each $k \in \mathcal{R}$, where $\varepsilon$ and $\varsigma$ are small positive scalars. The scaling factor $1+\varepsilon$ is only needed for guaranteeing that the variables $y$ and $z$ remain bounded and proving convergence of the applied algorithm, and may be omitted in practice. The function $\varphi$ is a strongly convex non-negative quadratic function, which is defined below, and $f$ is the vector-valued function of active and reactive power mismatches for each bus $k \in [n] \setminus \{s\}$. Lastly, the linear constraints $Ax = b$ represent the voltage regulation constraints (2.41).

The strongly convex non-negative quadratic function $\varphi$ is given by

$$\varphi := \alpha_q\varphi_q + \alpha_r\varphi_r + \alpha_u\varphi_u + \alpha_\theta\varphi_\theta + \alpha_d\varphi_d + \alpha_y\varphi_y + \alpha_z\varphi_z, \tag{3.25}$$

where

$$\varphi_q(x) := \frac{n}{|\mathcal{R}|} \sum_{i \in \mathcal{R}} \left( \frac{2Q_i^g - Q_i^{\max} - Q_i^{\min}}{Q_i^{\max} - Q_i^{\min}} \right)^2 \tag{3.26}$$

$$\varphi_r(x) := \frac{n}{|\mathcal{R}|} \sum_{i \in \mathcal{R}} \left( \frac{v_i - v_i^t}{\Delta v} \right)^2 \tag{3.27}$$

$$\varphi_u(x) := \frac{n}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \left( \frac{v_i - 1}{\Delta v} \right)^2 \tag{3.28}$$

$$\varphi_\theta(x) := \frac{n}{n-1} \sum_{i \in [n] \setminus \{s\}} \left( \frac{\theta_i}{\pi} \right)^2 \tag{3.29}$$

$$\varphi_d(x) := \frac{n}{|\mathcal{B}|} \sum_{(i,j) \in \mathcal{B}} \left( \frac{\theta_{ij} - \phi_{ij}}{\pi} \right)^2 \tag{3.30}$$

$$\varphi_y(x) := \frac{n}{|\mathcal{R}|} \sum_{i \in \mathcal{R}} \left( \frac{y_i}{\Delta v} \right)^2 \tag{3.31}$$

$$\varphi_z(x) := \frac{n}{|\mathcal{R}|} \sum_{i \in \mathcal{R}} \left( \frac{z_i}{\Delta v} \right)^2, \tag{3.32}$$

$\Delta v$ is a positive scalar, and the weights satisfy

$$\alpha_r \approx \alpha_u \gg \alpha_\theta \approx \alpha_d \approx \alpha_y \approx \alpha_z > \alpha_q > 0. \tag{3.33}$$

The penalty $\varphi_u$ has the effect of encouraging the algorithm applied to (3.22) to find feasible points that have unregulated voltage magnitudes near unity. The penalties $\varphi_r$, $\varphi_y$ and $\varphi_z$ encourage regulated voltage magnitudes to be near their set points. The penalty $\varphi_\theta$ encourages bus voltage angles to be small, which is justified since they can be naturally bounded between $-\pi$ and $\pi$. The penalty $\varphi_d$ encourages voltage angle differences between adjacent buses to be small, which is a common property of power networks under typical loading conditions that is often exploited by methods such as the *Fast Decoupled Newton-Raphson method* [1] to simplify the power flow equations. Lastly, the penalty $\varphi_q$ encourages generator reactive powers to be within their limits.

All the penalties that form the objective function $\varphi$ encourage desirable properties that are typical in well-designed systems that operate under manageable loading conditions. The

penalties $\varphi_r$ and $\varphi_u$ play the role of helping prevent the exploration of solutions in physically meaningless regions, which has been observed to be crucial for obtaining robustness to poor initial points [62]. The rest of the penalties serve mainly a regularization purpose and are assigned small weights relative to the those for $\varphi_r$ and $\varphi_u$. Hence, they do not necessarily affect the type of feasible points found by the algorithm, but help increase the robustness of the algorithm by allowing the linear algebra operations performed inside the algorithm to be stable. Lastly, it is noted that although the property of having generator reactive powers within limits is crucial for a solution to be meaningful, the weight of $\varphi_q$ need not be large since this property is already strongly encouraged by the approximate complementarity constraints $\Phi(x) = 0$.

To simplify the presentation, the new function $c$ is used to represent both the left-hand sides of the power flow constraints as well as the left-hand sides of the smooth approximations of the complementarity constraints. That is, $c$ is defined by

$$c(x) := \begin{bmatrix} f(x) \\ \Phi(x) \end{bmatrix}. \tag{3.34}$$

Hence, the problem formulation considered is given by

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \varphi(x) \\ \text{subject to} \quad & c(x) = 0 \\ & Ax = b. \end{aligned} \tag{3.35}$$

## 3.5 Solution Algorithm

The algorithm for solving (3.35) consists of approximately solving a sequence of linear equality constrained subproblems

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & L_{\mu_k}(x, \lambda_k) := \mu_k \varphi(x) - \mu_k \lambda_k^T c(x) + \frac{1}{2}\|c(x)\|_2^2 \\ \text{subject to} \quad & Ax = b, \end{aligned} \tag{3.36}$$

indexed by $k \in \mathbb{Z}_+$, where $\{\mu_k\}_{k\in\mathbb{Z}_+}$ and $\{\lambda_k\}_{k\in\mathbb{Z}_+}$ are sequences of positive penalty parameters and Lagrange multiplier estimates, respectively. Hence, the algorithm presents a

two-level structure: in the outer level, the sequences of penalty parameters and Lagrange multiplier estimates are constructed, while in the inner level, a reduced-space Newton's method is used to solve (3.36) with increasing accuracy.

The algorithm uses only first-order information of the nonlinear equality constraints to update the primal and dual variables, *i.e.*, $x$ and $\lambda$, respectively, during the early subproblems. A set of heuristics is used to determine when to incorporate second-order information in order to improve the convergence of the algorithm.

Table 3.2: Parameters of the vPF algorithm.

| name | default value | description |
|---|---|---|
| $\alpha_q \in \mathbb{R}_{++}$ | $1 \times 10^{-4}$ | weight for penalty function $\varphi_q$ |
| $\alpha_r \in \mathbb{R}_{++}$ | $1 \times 10^{-1}$ | weight for penalty function $\varphi_r$ |
| $\alpha_u \in \mathbb{R}_{++}$ | $1 \times 10^{-1}$ | weight for penalty function $\varphi_u$ |
| $\alpha_\theta \in \mathbb{R}_{++}$ | $1 \times 10^{-3}$ | weight for penalty function $\varphi_\theta$ |
| $\alpha_d \in \mathbb{R}_{++}$ | $1 \times 10^{-3}$ | weight for penalty function $\varphi_d$ |
| $\alpha_y \in \mathbb{R}_{++}$ | $1 \times 10^{-3}$ | weight for penalty function $\varphi_y$ |
| $\alpha_z \in \mathbb{R}_{++}$ | $1 \times 10^{-3}$ | weight for penalty function $\varphi_z$ |
| $\kappa \in \mathbb{R}_{++}$ | $1 \times 10^{-1}$ | coefficient for computing initial $\mu$ |
| $\beta_s \in (0, 1)$ | $1 \times 10^{-1}$ | factor for decreasing $\mu$ significantly |
| $\beta_l \in (0, 1)$ | $9 \times 10^{-1}$ | factor for decreasing $\mu$ slightly |
| $\tau_s \in (0, 1)$ | $1 \times 10^{-1}$ | factor that defines required reductions in $\|Z^T \nabla L_\mu\|_\infty$ |
| $\tau_f \in (0, 1)$ | $1 \times 10^{-1}$ | factor that defines desirable reductions in $\|c\|_\infty$ |
| $\epsilon_f \in (0, 1)$ | $1 \times 10^{-4}$ | feasibility tolerance for $c(x) = 0$ in per unit base power |
| $\zeta_s \in \mathbb{R}_{++}$ | $1 \times 10^{-2}$ | threshold for using second-order power flow information |
| $\rho_1 \in \mathbb{R}_{++}$ | $1 \times 10^{-4}$ | coefficient for regularization in first-order $\lambda$ update |
| $\rho_2 \in \mathbb{R}_{++}$ | $1 \times 10^{0}$ | coefficient for regularization in second-order $\lambda$ update |
| $\Delta v \in \mathbb{R}_{++}$ | $2 \times 10^{-1}$ | factor for normalizing voltage magnitude deviations |
| $\varsigma \in \mathbb{R}_{++}$ | $1 \times 10^{-8}$ | parameter for smooth approximation of FB function |

### 3.5.1 Outer Level

During each outer iteration $k \in \mathbb{Z}_+$, the algorithm first checks whether the current outer iterate $x_k$, which always satisfies $Ax_k = b$, is feasible with respect to the nonlinear equality constraints $c(x) = 0$ according to the specified tolerance. That is, it checks whether $x_k$ satisfies $\|c(x_k)\|_\infty < \epsilon_f$, where $\epsilon_f$ is a parameter of the algorithm (Table 3.2). If so, it terminates. Otherwise, it approximately solves subproblem (3.36) by keeping $\lambda_k$ and the

current penalty parameter $\mu_k$ fixed, as described in Section 3.5.2. Then, it performs either a regularized first-order or a regularized second-order dual variable update, depending on whether the exact Hessian of $L_{\mu_k}(\cdot, \lambda_k)$ is being used for updating the primal variables, as described in Section 3.5.4. Lastly, it updates the penalty parameter based on whether the primal infeasibilities are sufficiently reduced after approximately solving the last subproblem. Algorithm 2 shows pseudocode of this procedure.

The termination condition used by the algorithm consists only of feasibility and not of the first-order optimality conditions for (3.35), which if assuming some form of constraint qualification [33] [67] would include

$$Z^T\big(\nabla\varphi(x) - J(x)^T\lambda\big) = 0, \tag{3.37}$$

where $J$ is the Jacobian of $c$, and $Z$ is a matrix whose columns span the null space of $A$. The reason for not considering first-order optimality conditions is that the objective function is an artificial element added to the problem for helping the algorithm achieve feasibility.

**Algorithm 2** Outer level of vPF

Given $(x_0, \lambda_0)$ such that $Ax_0 = b$ and parameters from Table 3.2
$k \leftarrow 0$
Set initial penalty parameter $\mu_0 \leftarrow \max\{0.5\kappa\|c(x_0)\|_2^2/\varphi(x_0), \hat{\mu}\}$, where $\hat{\mu} > 0$
Set initial subproblem tolerance $\delta_0 \leftarrow \max\{\tau_s\|Z^T\nabla L_{\mu_0}(x_0, \lambda_0)\|_\infty, \hat{\delta}\}$, where $\hat{\delta} > 0$
**while** True **do**
    **if** $\|c(x_k)\|_\infty < \epsilon_f$ **then**
        **return** $(x_k, \lambda_k)$
    **end if**
    Find $x_{k+1}$ such that $\|Z^T\nabla L_{\mu_k}(x_{k+1}, \lambda_k)\|_\infty \leq \delta_k$ (Section 3.5.2)
    Update subproblem tolerance $\delta_{k+1} \leftarrow \tau_s\delta_k$
    Obtain new Lagrange multiplier estimates $\lambda_{k+1}$ (Section 3.5.4)
    **if** $\|c(x_{k+1})\|_\infty \leq \tau_f\|c(x_k)\|_\infty$ **then**
        Slightly decrease penalty parameter $\mu_{k+1} \leftarrow \beta_l\mu_k$
    **else**
        Significantly decrease penalty parameter $\mu_{k+1} \leftarrow \beta_s\mu_k$
    **end if**
    $k \leftarrow k + 1$
**end while**

### 3.5.2 Inner Level

During each inner iteration $j \in \mathbb{Z}_+$, the algorithm first checks whether the current inner iterate $x_j$ satisfies the first-order optimality condition of subproblem (3.36), which is

$$Z^T \nabla L_{\mu_k}(x, \lambda_k) = 0, \tag{3.38}$$

according to the specified tolerance. That is, it checks whether $x_j$ satisfies

$$\|Z^T \nabla L_{\mu_k}(x_j, \lambda_k)\|_\infty \leq \delta_k, \tag{3.39}$$

where $\delta_k$ is some positive scalar determined in the outer level. If so, it terminates. Otherwise, it computes a search direction using either the exact Hessian of $L_{\mu_k}(\cdot, \lambda_k)$ or an approximation of it, as described in Section 3.5.3. Then, a line search based on the strong Wolfe conditions (3.5) and (3.6) is performed with $L_{\mu_k}(\cdot, \lambda_k)$ as the merit function and $x_j$ is updated. Algorithm 3 shows pseudocode of this procedure.

**Algorithm 3** Inner level of vPF

Given $(x_k, \lambda_k, \mu_k, \delta_k)$ and parameters from Table 3.2
$j \leftarrow 0$
$x_j \leftarrow x_k$
**while** True **do**
    **if** $\|Z^T \nabla L_{\mu_k}(x_j, \lambda_k)\|_\infty \leq \delta_k$ **then**
        **return** $x_j$
    **end if**
    Compute search direction $p_j$ such that $Ap_j = 0$ (Section 3.5.3)
    Perform line search with merit function $L_{\mu_k}(\cdot, \lambda_k)$ to obtain step length $\alpha_j$
    Update solution estimate $x_{j+1} \leftarrow x_j + \alpha_j p_j$
    $j \leftarrow j + 1$
**end while**

### 3.5.3 Computation of Search Directions

Let $H_\mu(x, \lambda)$ be the Hessian of $L_\mu(\cdot, \lambda)$ at $x$. That is,

$$H_\mu(x, \lambda) = \mu \nabla^2 \varphi(x) - \Gamma(x, \lambda, \mu) + J(x)^T J(x), \tag{3.40}$$

where

$$\Gamma(x, \lambda, \mu) := \sum_{i=1}^{m} (\mu\lambda_i - c_i(x))\nabla^2 c_i(x) \tag{3.41}$$

and $m \in \mathbb{N}$ is the number of nonlinear constraints. By ignoring $\Gamma(x, \lambda, \mu)$, a positive definite approximation of $H_\mu(x, \lambda)$ is obtained, which is denoted by $\tilde{H}_\mu(x, \lambda)$. That is,

$$\tilde{H}_\mu(x, \lambda) := \mu\nabla^2\varphi(x) + J(x)^T J(x). \tag{3.42}$$

This approximation is sometimes called the "Gauss-Newton" Hessian approximation in the literature [22] [43]. However, the matrix $\Gamma(x, \lambda, \mu)$ contains valuable second-order or "curvature" information of the nonlinear equality constraints that should not be ignored. This information is especially useful near the solution for improving the convergence rate of the algorithm.

The search directions used at every inner iteration of the algorithm are of the form $p = Zq$, where $q$ solves the system

$$Z^T G_\mu(x, \lambda) Z q = -Z^T \nabla L_\mu(x, \lambda), \tag{3.43}$$

and $G_\mu(x, \lambda)$ is either the exact Hessian $H_\mu(x, \lambda)$ or the positive-definite approximation $\tilde{H}_\mu(x, \lambda)$.

To avoid the expensive operation of forming $J(x)^T J(x)$, the potential loss of sparsity and possible numerical problems due to the likelihood of large elements, a common property observed in the power flow cases tested, $q$ is obtained by solving the equivalent augmented system

$$\begin{bmatrix} \frac{1}{\mu} Z^T \hat{H}_\mu(x, \lambda) Z & Z^T J(x)^T \\ J(x)Z & -\mu I \end{bmatrix} \begin{bmatrix} q \\ y \end{bmatrix} = - \begin{bmatrix} \frac{1}{\mu} Z^T \nabla L_\mu(x, \lambda) \\ 0 \end{bmatrix}, \tag{3.44}$$

where

$$\hat{H}_\mu(x, \lambda) := \begin{cases} \mu\nabla^2\varphi(x) - \Gamma(x, \lambda, \mu), & \text{if } G_\mu(x, \lambda) = H_\mu(x, \lambda) \\ \mu\nabla^2\varphi(x), & \text{otherwise.} \end{cases} \tag{3.45}$$

The decision of whether to use the exact Hessian $H_\mu(x, \lambda)$ or the Hessian approximation $\tilde{H}_\mu(x, \lambda)$ for computing the search direction depends on a set of heuristics. These heuristics are based on the observation that, for power flow problems, the size of $\Gamma(x, \lambda, \mu)$ (in the norm sense) relative to that of $J(x)$ typically decreases several orders of magnitudes as the

number of iterations increases. The main idea is to check whether

$$\omega(x, \lambda, \mu) \leq \zeta_s, \tag{3.46}$$

where

$$\omega(x, \lambda, \mu) := \frac{\max\limits_{i,j}\{|\Gamma_{ij}(x, \lambda, \mu)|\}}{\max\limits_{i,j}\{|J_{ij}(x)|\}}, \tag{3.47}$$

and $\zeta_s$ is a parameter of the algorithm (Table 3.2). If so, the exact Hessian $H_\mu(x, \lambda)$ is used. Otherwise, the Hessian approximation $\tilde{H}_\mu(x, \lambda)$ is used for solving (3.44).

Using the exact Hessian $H_\mu(x, \lambda)$ may result in problems during the solution of the linear system (3.44) or in the search direction $p = Zq$ not satisfying the sufficient descent condition

$$-\frac{\nabla L_\mu(x, \lambda)^T p}{\|Z^T \nabla L_\mu(x, \lambda)\|_2 \|q\|_2} > \xi, \tag{3.48}$$

where $\xi$ is any predefined small positive scalar. If this happens, the threshold $\zeta_s$ is reduced and the process of computing the search direction is repeated.

Lastly, to ensure that the algorithm always tries to use the exact Hessian near the solution and takes Newton directions, the exact Hessian $H_\mu(x, \lambda)$ is tried periodically, say every 10 inner iterations, even if (3.46) does not hold. If after doing this a valid search direction is obtained, the threshold is updated by setting $\zeta_s = \omega(x, \lambda, \mu) + \delta$, for some small $\delta > 0$.

### 3.5.4 Regularized Dual Variable Update

When second derivatives are available, a known dual variable update that exploits this information consists of [11] [80]

$$\Delta\lambda = -B^{-1}\frac{c(x)}{\mu}, \tag{3.49}$$

where

$$B := J(x)Z\big(Z^T \nabla^2 L_\mu(x, \lambda) Z\big)^{-1} Z^T J(x)^T. \tag{3.50}$$

When subproblem (3.36) is solved exactly, *i.e.*, when $Z^T \nabla L_\mu(x, \lambda) = 0$, this update is equivalent to taking a Newton step for solving the augmented system

$$F_\mu(x, \lambda) = 0, \tag{3.51}$$

where

$$F_\mu(x,\lambda) := \begin{bmatrix} \frac{1}{\mu} Z^T \nabla L_\mu(x,\lambda) \\ Ax - b \\ c(x) \end{bmatrix}, \tag{3.52}$$

and to applying the NR method for maximizing the dual functional associated with $L_\mu$ [11] [80]. Properly taking into account $Z^T \nabla L_\mu(x,\lambda) \neq 0$ is achieved by working with system (3.51), but obtaining the Newton step for this system requires solving the system

$$\begin{bmatrix} \frac{1}{\mu} Z^T H_\mu(x,\lambda) & Z^T J(x)^T \\ A & 0 \\ J(x) & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ -\Delta\lambda \end{bmatrix} = - \begin{bmatrix} \frac{1}{\mu} Z^T \nabla L_\mu(x,\lambda) \\ 0 \\ c(x) \end{bmatrix}, \tag{3.53}$$

which is ill-conditioned when $J(x)$ is near rank-deficient. For this reason, the modified augmented system

$$(1-t) \begin{bmatrix} 0 \\ 0 \\ \Delta\lambda \end{bmatrix} + t F_\mu(x + \Delta x, \lambda + \Delta\lambda) = 0 \tag{3.54}$$

is used by the vPF algorithm when the exact Hessian is being used for computing primal search directions, where $t \in (0,1)$ is close to 1. Computing a Newton step for this modified augmented system requires solving the linear system

$$\begin{bmatrix} \frac{1}{\mu} Z^T H_\mu(x,\lambda) & Z^T J(x)^T \\ A & 0 \\ J(x) & -\eta I \end{bmatrix} \begin{bmatrix} \Delta x \\ -\Delta\lambda \end{bmatrix} = - \begin{bmatrix} \frac{1}{\mu} Z^T \nabla L_\mu(x,\lambda) \\ 0 \\ c(x) \end{bmatrix}, \tag{3.55}$$

where $\eta = (1-t)/t$. The term $-\eta I$ helps avoid numerical problems when $J(x)$ is near rank-deficient, which is common in power flow problems. This term comes from the first term of the left-hand side of (3.54), and hence explains the motivation behind the form of such expression.

The above system can be simplified by noticing that the condition $A\Delta x = 0$ is equivalent to $\Delta x = Z\Delta x_z$ for some $\Delta x_z$. Hence, system (3.55) can be written in terms of $\Delta x_z$ and

$\Delta\lambda$ as

$$\begin{bmatrix} \frac{1}{\mu}Z^T H_\mu(x,\lambda)Z & Z^T J(x)^T \\ J(x)Z & -\eta I \end{bmatrix} \begin{bmatrix} \Delta x_z \\ -\Delta\lambda \end{bmatrix} = - \begin{bmatrix} \frac{1}{\mu}Z^T \nabla L_\mu(x,\lambda) \\ c(x) \end{bmatrix}. \tag{3.56}$$

Again, to avoid forming $J(x)^T J(x)$ and $H_\mu(x,\lambda)$, the equivalent system

$$\begin{bmatrix} \frac{1}{\varsigma}Z^T \hat{H}_\mu(x,\lambda)Z & Z^T J(x)^T \\ J(x)Z & -\eta I \end{bmatrix} \begin{bmatrix} \Delta x_z \\ -\Delta\lambda \end{bmatrix} = - \begin{bmatrix} \frac{1}{\varsigma}Z^T (\mu\nabla\varphi(x) - \mu J(x)^T\lambda) \\ c(x) \end{bmatrix} \tag{3.57}$$

is solved using $\eta = \rho_2\mu$, where

$$\hat{H}_\mu(x,\lambda) = \mu\nabla^2\varphi(x) - \Gamma(x,\lambda,\mu), \tag{3.58}$$

$\varsigma = \mu + \eta$, and $\rho_2$ is a positive parameter (Table 3.2).

An alternative to the above dual variable update that does not use second derivatives consists of $\Delta\lambda = -c(x)/\mu$. When subproblem (3.36) is solved exactly, *i.e.*, when $Z^T\nabla L_\mu(x,\lambda) = 0$, this step can be shown to solve the least-squares problem

$$\underset{\Delta\lambda}{\text{minimize}} \quad \|Z^T J(x)^T\Delta\lambda - Z^T(\nabla\varphi(x) - J(x)^T\lambda)\|_2^2, \tag{3.59}$$

and be equivalent to applying steepest ascent for maximizing the dual functional associated with $L_\mu$ [11] [80]. Properly taking into account $Z^T\nabla L_\mu(x,\lambda) \neq 0$ is achieved by working with (3.59), but solving this problem requires solving the system

$$J(x)ZZ^T J(x)^T\Delta\lambda = J(x)ZZ^T\big(\nabla\varphi(x) - J(x)^T\lambda\big), \tag{3.60}$$

which is ill-conditioned when $J(x)$ is near rank-deficient. For this reason, the regularized least-squares problem

$$\underset{\Delta\lambda}{\text{minimize}} \quad \|Z^T J(x)^T\Delta\lambda - Z^T(\nabla\varphi(x) - J(x)^T\lambda)\|_2^2 + \eta\|\Delta\lambda\|_2^2 \tag{3.61}$$

is used by the vPF algorithm when the exact Hessian is not being used for computing primal search directions, where $\eta = \rho_1\mu$, and $\rho_1$ is a positive parameter (Table 3.2). Solving (3.61)

requires solving the normal equations

$$\big(J(x)ZZ^TJ(x)^T + \eta I\big)\Delta\lambda = J(x)ZZ^T\big(\nabla\varphi(x) - J(x)^T\lambda\big). \tag{3.62}$$

Again, to avoid forming $J(x)ZZ^TJ(x)^T$, the augmented system

$$\begin{bmatrix} -I & Z^TJ(x)^T \\ J(x)Z & \eta I \end{bmatrix}\begin{bmatrix} y \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} 0 \\ J(x)ZZ^T\big(\nabla\varphi(x) - J(x)^T\lambda\big) \end{bmatrix} \tag{3.63}$$

is solved. This system also has the term $\eta I$, which helps avoid problems with rank-deficient or near rank-deficient Jacobians. In this case, this term is a consequence of the regularization included in the least-squares formulation.

Once the dual step $\Delta\lambda$ is obtained, the new vector $\bar{\lambda}$ of Lagrange multiplier estimates is set to $\lambda + \Delta\lambda$. To ensure that $\bar{\lambda}$ remains bounded, entries larger than some predefined $\tilde{M} > 0$ in magnitude are set to $\tilde{M}$, ensuring that $\|\bar{\lambda}\|_\infty \leq \tilde{M}$ always holds.

### 3.5.5 Convergence Properties

It can be shown that, under the assumption that the power network has the property that bounded power mismatches implies bounded bus voltage magnitudes, the sequence of outer iterates $\{x_k\}$ produced by the vPF algorithm has a feasible point or a limit point that is a stationary point of the function $c^Tc$ (restricted to the set of $x$ such that $Ax = b$). This result can be obtained without assuming that the Jacobian of the power flow equations is full rank. The proof is presented in Appendix A.

An important observation is that the only property needed of the Lagrange multiplier estimates for the convergence proof is that they remain bounded. In an augmented Lagrangian algorithm, Lagrange multiplier estimates help encourage convergence of the algorithm to an optimal point without having to rely solely on adjusting the penalty parameter. However, when feasibility is not obtained, the penalty parameter needs to be driven to zero (or driven to infinity, depending on the implementation) in order to approach a stationary point of the sum of the squares of the infeasibilities. In this case, the method behaves essentially like a penalty function method [33], and multipliers do not play a role.

The property that bounded power mismatches implies bounded bus voltage magnitudes is needed to ensure that the voltage magnitudes of the outer iterates stay inside a compact set. Boundedness of the other variables, except the angles, is obtained from the boundedness

of the nonlinear constraint violations without additional assumptions. For the angles, this is not an issue since they appear inside (bounded) periodic functions and hence an equivalent sequence of outer iterates with bounded angles can be constructed easily. Power networks that have this property are referred here as *lossy networks*. For these networks, the total active power losses are positive for any set of complex bus voltages that are not all zero. A mathematical characterization for these networks is that the Hermitian matrix $\tilde{G}$ defined in (2.35) is positive definite. Proofs of this characterization of lossy networks and of other related properties are presented in Appendix B.

### 3.5.6 Sensitivity Information

It is well known that Lagrange multipliers provide sensitivity information about how the optimal objective value changes when the constraints are perturbed [33]. The objective value in (3.22) penalizes deviations from desirable properties that are typical in well-designed systems that operate under manageable conditions. Hence, it provides a measure of how well a system can handle particular loading conditions. Sensitivity information can be used for identifying key modifications to the system that are likely to result in positive and negative effects on the system in terms of handling particular loading conditions.

More specifically, let $x^*$ be the solution of (3.22), and $\lambda$ and $\pi$ be the Lagrange multipliers associated with the power flow and approximate complementarity constraints, respectively. Then, if the right-hand sides of the nonlinear constraints are perturbed by $\epsilon$ and $\delta$, and if $\bar{x}$ denotes the solution of

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \varphi(x) && (3.64) \\ \text{subject to} \quad & f(x) = \epsilon \\ & \Phi(x) = \delta && (3.65) \\ & Ax = b, \end{aligned}$$

it can be shown that, to first order,

$$\varphi(\bar{x}) \approx \varphi(x^*) + \lambda^T \epsilon + \pi^T \delta. \tag{3.66}$$

This suggest that small perturbations $\epsilon$ and $\delta$ such that

$$\text{sgn}(\epsilon_i) \neq \text{sgn}(\lambda_i) \tag{3.67}$$

$$\text{sgn}(\delta_i) \neq \text{sgn}(\pi_i), \tag{3.68}$$

are likely to have positive effects on the system.

From the specific form of the nonlinear constraints, perturbations of the form $f(x) = \epsilon$ and $\Phi(x) = \delta$ have specific physical interpretations in terms of operator actions or system design. In particular, a nonzero $\epsilon_i$ represents a small extra injection or consumption of active or reactive power at a specific bus. For example, this may represent the addition of a reactive power compensation device at a particular bus. Similarly, a nonzero $\delta_i$ represents a modification of either the reactive power limit of a generator or a change of its voltage magnitude set point. Figure 3.2 shows an example of the normalized sensitivities obtained for a power flow case after the reactive power consumption of loads in a region is increased. Positive sensitivities suggest that additional small reactive power sources are beneficial for the system under the given loading conditions.

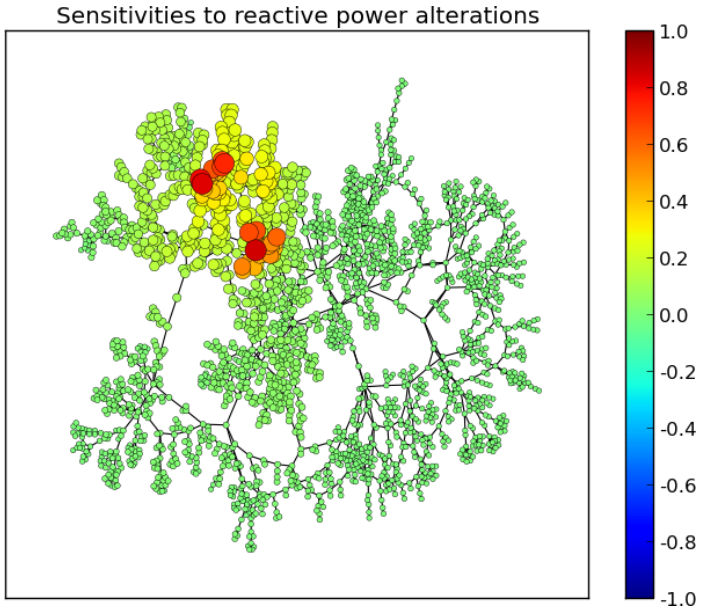


Figure 3.2: Sensitivity information provided by multiplier estimates.

The Lagrange multiplier estimates produced by the vPF algorithm can therefore be used for determining whether alterations such as adding small powers sinks or sources at

specific locations or changing generator voltage set points are likely to have positive effects on the system. The Lagrange multiplier estimates of largest magnitudes may be used for determining the most effective alterations. These can provide system operators and planners useful guidance for making system alterations, and in particular, can help them avoid making alterations that are *locally* beneficial but have an overall negative system effect.

## 3.6 Implementation

To assess the benefits and computational requirements of the techniques presented in this chapter, the NR and vPF methods were implemented and their performance compared through various experiments. The Python programming language was used for implementing a parsing tool, the testing environment, and the high-level parts of each of the power flow methods. The parsing tool allowed extracting power flow cases from PSS®E raw files, which was the key for testing the methods on real power flow networks. The linear algebra operations as well as the handling of sparse matrices was done through Python's scientific computing libraries `Numpy` and `Scipy`. For solving linear systems, the sequential version of the multifrontal sparse direct solver `MUMPS` [3] was used through the Python wrapper `pymumps`. The routines for performing function evaluations, *e.g.*, evaluation of objective function, constraint functions and their first and second derivatives, if done in Python, are typically the most time consuming tasks for a nonlinear programming solver. Hence, these routines were implemented in the C programming language and wrapped using Python's C API. Lastly, for manipulating, analyzing and visualizing power network graphs, the Python library `NetworkX` was used together with the powerful and open source graph visualization software `Graphviz` [29].

The implementations of the NR and vPF power flow methods include the following termination conditions, the first of which indicates that the method successfully solved a given case, while the rest indicate various forms of failure:

- The largest infeasibility of the nonlinear equality constraints becomes lower than $10^{-4}$ per unit system base MVA.
- A bus voltage magnitude becomes lower than 0.1 per unit bus nominal voltage.
- The line search procedure fails due to getting a search direction that is not a descent

direction, or due to not being able to find a suitable step length in a maximum of 40 iterations. For the vPF method, this termination condition is applied only if the positive definite Hessian approximation was used for computing the search direction.

- The penalty parameter $\mu$ becomes lower than $10^{-12}$. This is only applicable to the vPF method.

- The number of iterations exceeds 300. For the vPF method, the number of iterations is the combined number of inner iterations.

Instead of choosing a small positive parameter $\xi$ and testing condition (3.48), the vPF implementation checks whether the search direction results in a successful line search. This is done because, in practice, for $\xi$ small enough the line search typically fails when the search direction is too close to being orthogonal to the gradient, even when (3.48) holds. Lastly, initial Lagrange multiplier estimates equal to zero are used for the vPF method.

## 3.7 Experiments

The results of various numerical experiments performed to assess the benefits and computational requirements of the vPF method are presented here. The computer used for performing all the experiments was as a PC with Intel® Core™ i7 CPU, 2.80 GHz, 8 GB of RAM, and running the operating system Ubuntu 12.04. The properties of the power flow test cases collected are shown in Table 3.3. Missing information for Case A is due to a non-disclosure agreement.

Table 3.3: Properties of PF test cases.

| name | buses | branches | generation (MW) | load (MW) | region |
|---|---|---|---|---|---|
| Case A | — | — | — | — | South America |
| Case B | 2454 | 2781 | $7.5 \times 10^3$ | $7.3 \times 10^3$ | North America |
| Case C | 2468 | 3215 | $4.0 \times 10^4$ | $3.9 \times 10^4$ | North America |
| Case D | 3012 | 3572 | $2.8 \times 10^4$ | $2.7 \times 10^4$ | Eastern Europe |
| Case E | 45286 | 58994 | $7.1 \times 10^5$ | $6.9 \times 10^5$ | North America |
| Case F | 58513 | 73388 | $6.8 \times 10^5$ | $6.6 \times 10^5$ | North America |

### 3.7.1 Model Validation

In order to validate the power network model implemented, the solution of the power flow problem obtained with the implemented NR method was compared with that obtained with the commercial power flow software PSS®E and the open-source package MATPOWER 4.1 [92]. The test cases used for this validation were cases whose original format corresponded to the ones used by PSS®E and MATPOWER, and that were solvable by the NR method. Figures 3.3-3.5 show the results obtained. The left plots show the voltage magnitude errors while the right plots show the voltage angle errors for each bus at the computed operating point.

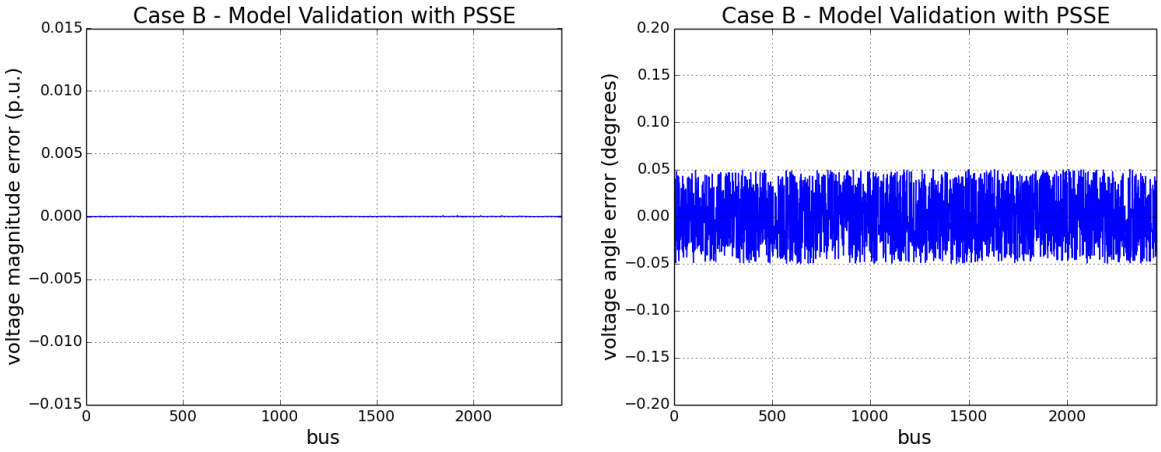


Figure 3.3: Power network model accuracy for PF Case B.

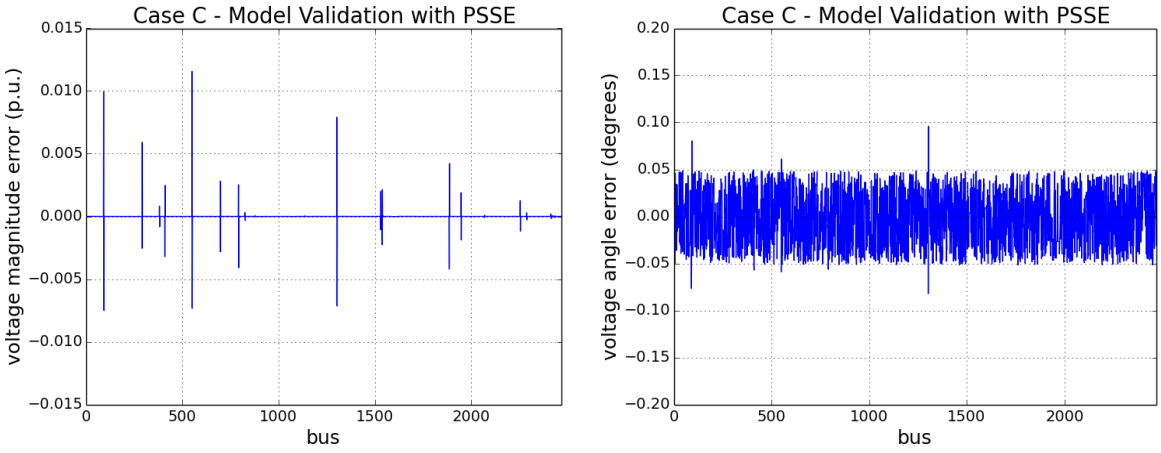


Figure 3.4: Power network model accuracy for PF Case C.

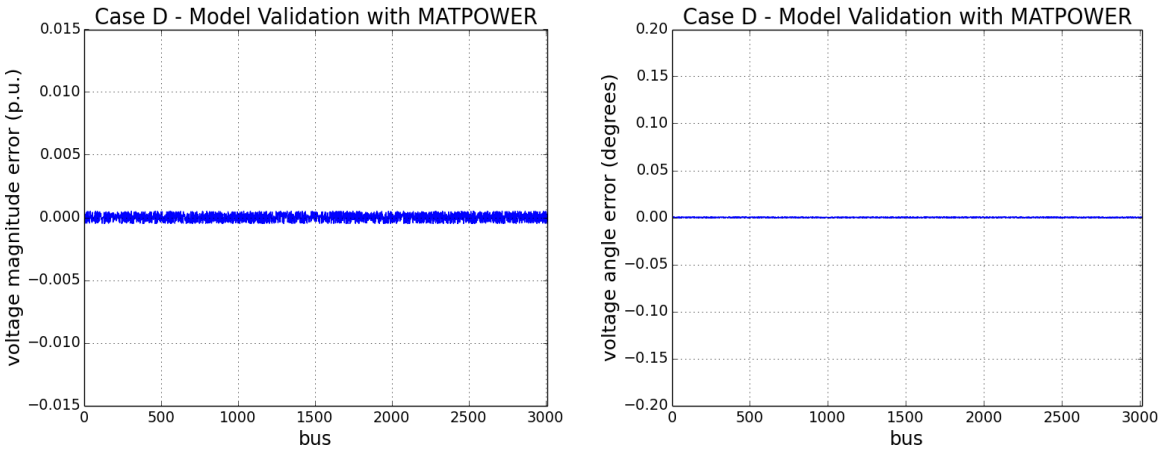


Figure 3.5: Power network model accuracy for PF Case D.

As the figures show, the model implemented for this study is accurate for Cases B and D, but slightly inaccurate for Case C. It was determined that the relatively large voltage magnitude errors obtained for a few buses for Case C are due to a limitation associated with the modeling of voltage regulation. Case C has a few groups of generators that regulate the same remote bus. However, in the model used for this study, only one generator was allowed to regulate a remote bus.

### 3.7.2 Performance of Methods on Test Cases

Tables 3.4-3.9 show the performance of the NR and vPF methods on each of the test cases of Table 3.3. The labels $k$ and $t$ denote the number of iterations and time in seconds, respectively, that a method took to find a feasible point or to terminate with an error. The label $\|f\|_\infty$ denotes the maximum power (active and reactive) mismatch in units of MVA. The label $\|h\|_\infty$ denotes the maximum violation of the (exact) complementarity constraints in units of MVA, where $h$ is the vector-valued function whose scalar function entries are given by the expressions

$$\left(Q_k^g - Q_k^{\min}\right) + \alpha y_k - \sqrt{\left(Q_k^g - Q_k^{\min}\right)^2 + \alpha^2 y_k^2} \tag{3.69}$$

$$\left(Q_k^{\max} - Q_k^g\right) + \alpha z_k - \sqrt{\left(Q_k^{\max} - Q_k^g\right)^2 + \alpha^2 z_k^2}, \tag{3.70}$$

for $k \in \mathcal{R}$, and $\alpha$ is the system base power, *i.e.*, 100 MVA. The labels $v^{\max}$ and $v^{\min}$ denote the maximum and minimum bus voltage magnitudes, respectively. The label $\max \Delta v$

denotes the maximum deviation of regulated voltage magnitudes from their set points. Lastly, the label $Q^+$ denotes the total absolute generator reactive power limit violations in units of MVAr associated with the last point found by the method before terminating.

Table 3.4: Performance of methods on PF Case A.

| method | $k$ | $t$ | $\|f\|_\infty$ | $\|h\|_\infty$ | $v^{\max}$ | $v^{\min}$ | $\max \Delta v$ | $Q^+$ |
|---|---|---|---|---|---|---|---|---|
| NR | 2 | 0.05 | $5.3 \times 10^{-4}$ | 0 | 1.11 | 0.86 | 0.00 | 0 |
| vPF | 21 | 0.97 | $4.8 \times 10^{-3}$ | $7.2 \times 10^{-3}$ | 1.11 | 0.86 | 0.00 | 0 |

Table 3.5: Performance of methods on PF Case B.

| method | $k$ | $t$ | $\|f\|_\infty$ | $\|h\|_\infty$ | $v^{\max}$ | $v^{\min}$ | $\max \Delta v$ | $Q^+$ |
|---|---|---|---|---|---|---|---|---|
| NR | 3 | 0.10 | $8.3 \times 10^{-7}$ | 0 | 1.11 | 0.95 | 0.05 | 0 |
| vPF | 23 | 1.55 | $1.0 \times 10^{-3}$ | $5.0 \times 10^{-3}$ | 1.11 | 0.95 | 0.05 | $2.7 \times 10^{-3}$ |

Table 3.6: Performance of methods on PF Case C.

| method | $k$ | $t$ | $\|f\|_\infty$ | $\|h\|_\infty$ | $v^{\max}$ | $v^{\min}$ | $\max \Delta v$ | $Q^+$ |
|---|---|---|---|---|---|---|---|---|
| NR | 3 | 0.11 | $6.7 \times 10^{-9}$ | 0 | 1.44 | 0.92 | 0.01 | 0 |
| vPF | 95 | 5.33 | $1.9 \times 10^{-3}$ | $9.9 \times 10^{-3}$ | 1.44 | 0.92 | 0.01 | 0 |

Table 3.7: Performance of methods on PF Case D.

| method | $k$ | $t$ | $\|f\|_\infty$ | $\|h\|_\infty$ | $v^{\max}$ | $v^{\min}$ | $\max \Delta v$ | $Q^+$ |
|---|---|---|---|---|---|---|---|---|
| NR | 3 | 0.15 | $1.1 \times 10^{-3}$ | 0 | 1.12 | 0.94 | 0.00 | 0 |
| vPF | 45 | 3.56 | $2.4 \times 10^{-3}$ | $7.4 \times 10^{-3}$ | 1.12 | 0.94 | 0.00 | 0 |

Table 3.8: Performance of methods on PF Case E.

| method | $k$ | $t$ | $\|f\|_\infty$ | $\|h\|_\infty$ | $v^{\max}$ | $v^{\min}$ | $\max \Delta v$ | $Q^+$ |
|---|---|---|---|---|---|---|---|---|
| NR | 6 | 4.54 | $5.4 \times 10^{-6}$ | 0 | 1.27 | 0.81 | 0.00 | 0 |
| vPF | 10 | 14.18 | $2.8 \times 10^{-3}$ | $7.7 \times 10^{-3}$ | 1.27 | 0.81 | 0.00 | 0 |

Table 3.9: Performance of methods on PF Case F.

| method | $k$ | $t$ | $\|f\|_\infty$ | $\|h\|_\infty$ | $v^{\max}$ | $v^{\min}$ | $\max \Delta v$ | $Q^+$ |
|---|---|---|---|---|---|---|---|---|
| NR | 6 | 7.44 | $2.5 \times 10^{3}$ | $4.9 \times 10^{3}$ | 1.45 | 0.10 | 0.25 | $3.23 \times 10^{3}$ |
| vPF | 241 | 403.77 | $7.8 \times 10^{-3}$ | $8.1 \times 10^{-3}$ | 1.45 | 0.82 | 0.25 | 0 |

The results show that for the cases that are solved by both methods, these methods find similar solutions in terms of $v^{\max}$, $v^{\min}$ and $\max \Delta v$. For these cases, the NR method takes much fewer iterations and time than the vPF method. In addition, it obtains exactly zero violations of the complementarity constraints ($\|h\|_\infty$) and generator reactive power limits ($Q^+$) due to the switching heuristics working successfully. The results obtained with the vPF method have small violations of the complementarity constraints ($\|h\|_\infty$), and these are all smaller than the theoretical bound of Lemma 3.4.1, which equals $2.4 \times 10^{-2}$ MVA using $\epsilon = 10^{-4}$, $\varsigma = 10^{-8}$, and 100 MVA for system base power. The small total absolute generator reactive power limit violations ($Q^+$) obtained with the vPF method on Case B are also the result of approximating the complementarity constraints and using the stated feasibility tolerance. For Case F, which is a hard case, the NR method also fails in a small number of iterations, and obtains large violations of complementarity constraints and generator reactive power limits. On the other hand, the vPF method succeeds in solving this hard case and requires a number of iterations and time that, due to the lack of performance reference, may or may not be considered large. By dividing the time required by the number of iterations, it can be concluded that the vPF method takes on average only about 1.7 times longer per iteration than the NR method. This is a small factor and suggest that the main focus for improving the speed competitiveness of the vPF method, on problems solved by both methods, should be to reduce the number of iterations.

In the next subsections, results of experiments that compare the performance of the NR and vPF methods in terms of handling poor initial points and heavily loaded and infeasible cases are shown. Case F was not used for these experiments since it is already a hard case that the NR method is not able to solve, and the perturbations done in the experiments only make it harder.

### 3.7.3 Handling of Poor Initial Points

To assess the robustness of the vPF method in handling poor initial points, the following experiment was performed: For each power flow case and for increasing values of standard deviation $\sigma$, 30 initial points were constructed by perturbing the initial point given in the power flow case data. The perturbations were done by adding Gaussian noise of zero mean and standard deviation $\sigma$ to the bus voltage magnitudes and phase angles in p.u. and radians, respectively. The NR and vPF methods were then executed with each of the points constructed and the percentage of the 30 cases that were solved as well as the average time

needed to solve them were recorded. Figures 3.6-3.10 show the results obtained.

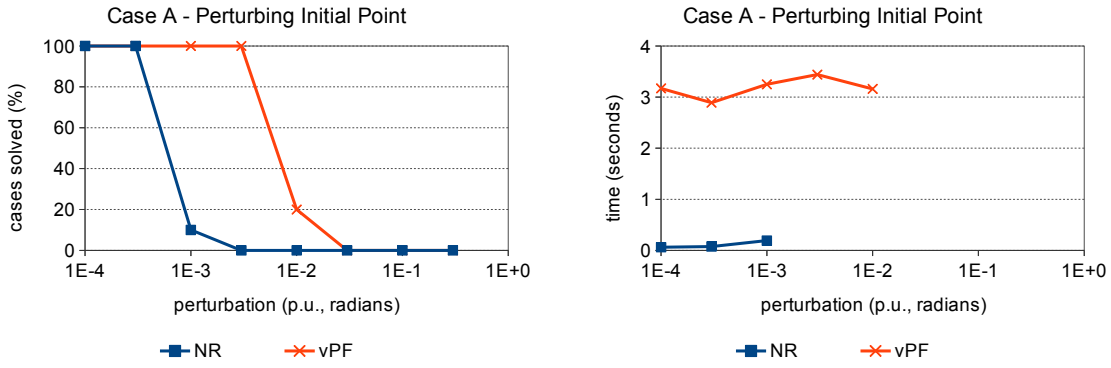


Figure 3.6: Perturbing initial point of PF Case A.

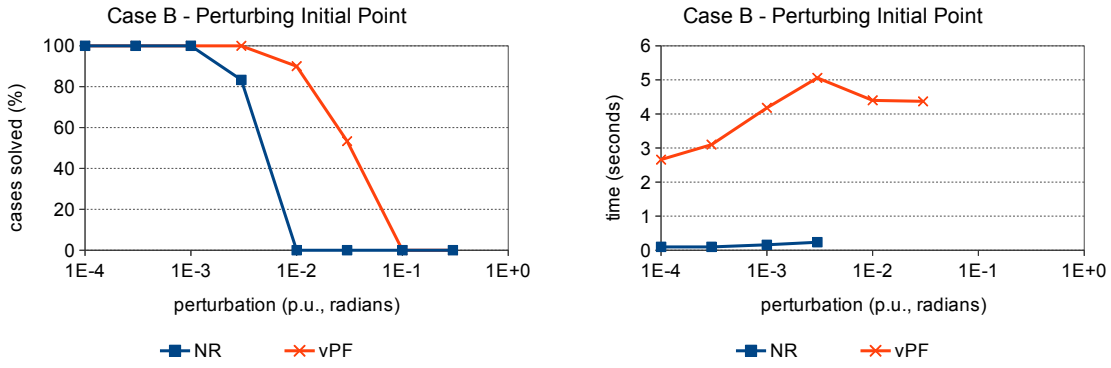


Figure 3.7: Perturbing initial point of PF Case B.

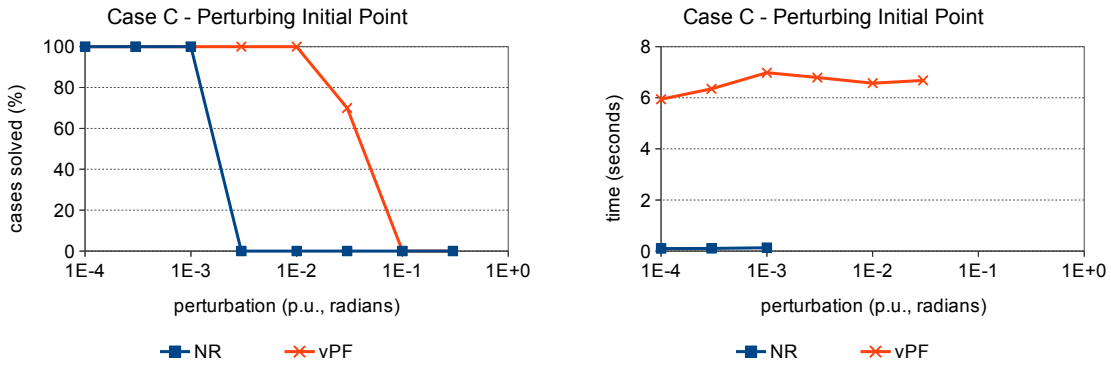


Figure 3.8: Perturbing initial point of PF Case C.

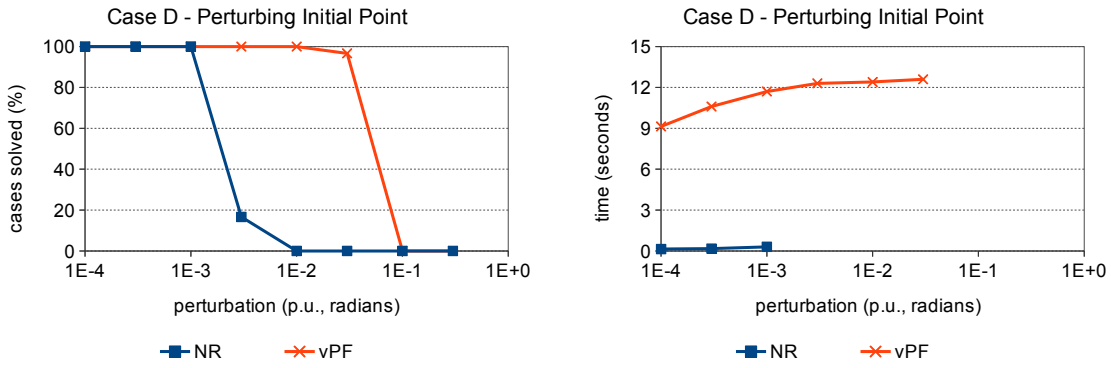


Figure 3.9: Perturbing initial point of PF Case D.

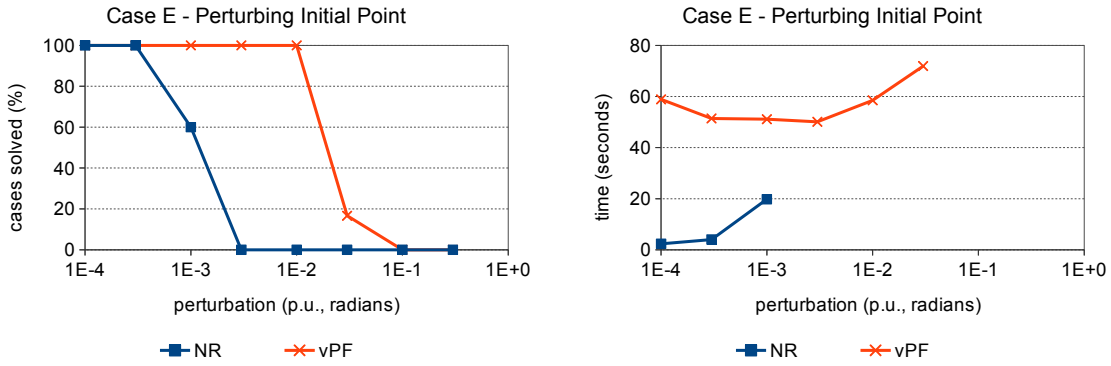


Figure 3.10: Perturbing initial point of PF Case E.

These results show that the vPF method is able to solve many more cases than the NR method for much larger perturbations of the initial point, and hence that it is much more robust in handling poor initial points. These results are similar to the ones obtained in our previous work [62]. Again, it can be seen that for cases that are solved by both methods, the NR method takes much shorter times than the vPF method.

### 3.7.4 Handling of Heavily Loaded and Infeasible Cases

To assess the performance of the vPF method in handling heavily loaded and infeasible cases, the following experiment was performed: For each power flow case and for increasing values of a scaling factor $\gamma$, the loads of the power flow case were scaled by $\gamma$ and the NR and vPF methods were executed. The minimum bus voltage magnitude $v^{\min}$ and a measure of the error associated with the last point found by the method before terminating were

recorded. This measure is given by $\max\{\|f\|_\infty, \|h\|_\infty\}$, where $f$ and $h$ are as defined in Section 3.7.2. In other words, the maximum of the violations of the power flow and (exact) complementarity constraints was used as a measure of error. Figures 3.11-3.15 show the results obtained. It is noted that in order to maintain the power factor of the loads unaltered, the same scaling factor was applied to both active and reactive power components.

In the left graph of the figures, the horizontal green line corresponds to the feasibility tolerance of $10^{-4}$ per unit system base MVA, which is equivalent to $10^{-2}$ MVA since the system base power for all test cases is 100 MVA. Any data point below this line implies that the method was able to solve the case to the required accuracy.

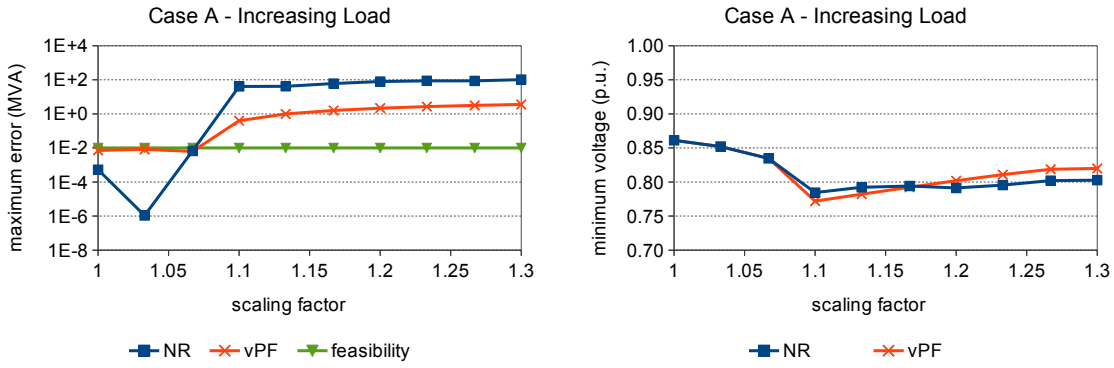


Figure 3.11: Increasing load of PF Case A.

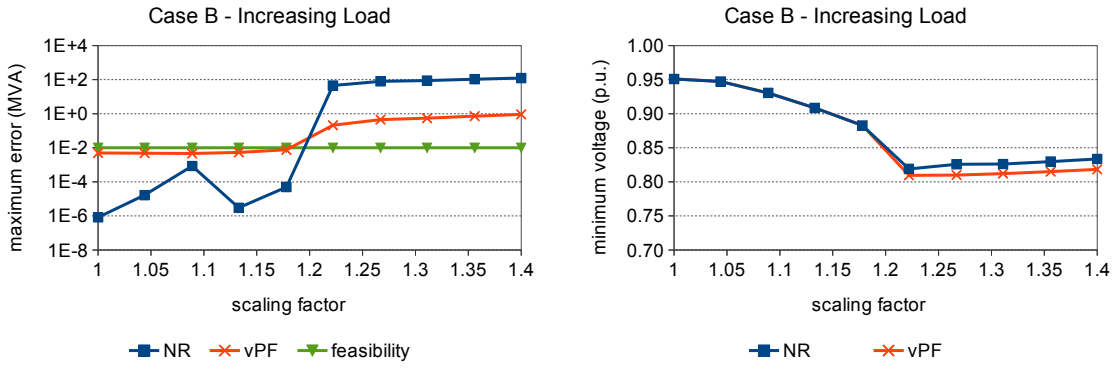


Figure 3.12: Increasing load of PF Case B.

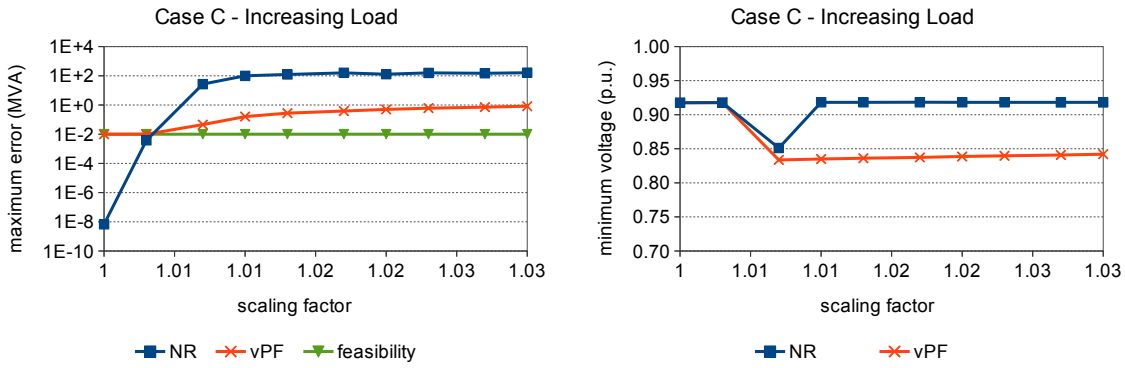


Figure 3.13: Increasing load of PF Case C.

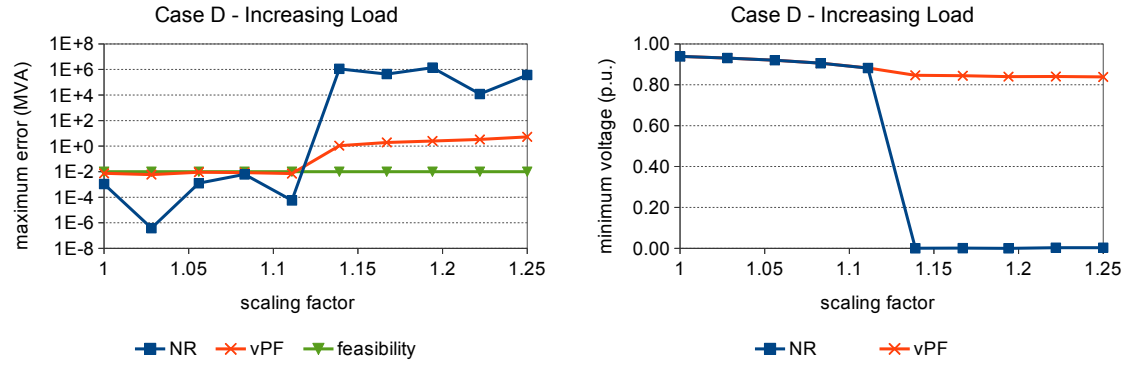


Figure 3.14: Increasing load of PF Case D.

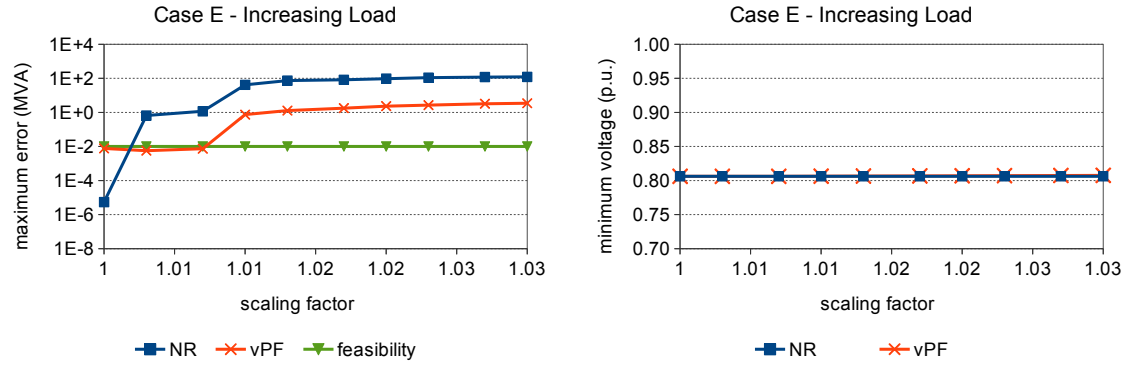


Figure 3.15: Increasing load of PF Case E.

The results show that the vPF method always finds a feasible point for cases for which the NR method finds a feasible point. As the load is increased beyond a certain point, both

methods fail to solve the case but the maximum error of the vPF method grows much less rapidly than the error of the NR method. For Case E, the vPF method finds feasible points for scaling factors for which the NR method fails with a large error. For this case, the vPF method had problems solving the linear systems but these were overcome by using a larger regularization factor $\rho_1$ ($10^{-2}$ instead of the default $10^{-4}$) during the first-order multiplier update (Section 3.5.4). The fact that the infeasible points found by the vPF method are better than those obtained by the NR method is likely due to the theoretical properties of the vPF method, which guarantee that it finds stationary points of the sum of the squares of the infeasibilities. In practice, stationary points are often local minimizers for problems for which negative curvature is not a significant property.

The behavior of the root mean square error of the NR and vPF methods as $\gamma$ increases was also analyzed in addition to the maximum error. The plots obtained had similar properties as those of Figures 3.11-3.15, and hence were not include here.

## 3.8 Conclusions

In this chapter, the limitations of the most widely used method for scenario analysis have been addressed. This method, which consists of a combination of Newton-Raphson and switching heuristics, lacks robustness to poor initial solution estimates and near rank-deficient Jacobian matrices, and its switching heuristics may negatively affect its convergence. To address all of these, prior knowledge about power flow solutions was incorporated to the problem via an objective function, complementarity constraints were used to model generator reactive power limits and voltage regulation, and a robust optimization algorithm that exploits second derivatives was applied. The proposed method was tested on several real power flow networks obtained from various North American, South American, and European electric power institutions, and the results were compared against those obtained with the traditional NR-based method. The experiments performed compared the properties of the solution, number of iterations, solution time, robustness to poor initial points, and handling of heavily loaded and infeasible cases. The results showed that the method equipped with the proposed techniques, which is referred to as *voltage-controlled Power Flow* (vPF), was much more robust to poor initial points. Also, it had a superior performance when handling heavily loaded and infeasible cases, since its error was observed to grow less rapidly than that of the traditional method after the loadability limit. However,

for cases that were solved by both methods, the vPF method was observed to be slower than the NR method, which was mainly due to requiring more iterations to solve a problem to the required accuracy. Hence, the vPF method may not be used as a complete replacement of the NR method, since the latter works well for simple cases. Instead, the vPF method should be used when the NR method fails to find a solution quickly or when it is known a priori that sensitivity information is needed for determining potential system problems and making alterations.

# Chapter 4

# Planning and Control

## 4.1 Background and Overview

During operations planning, adjustments to generator dispatches and control devices need to be determined to ensure that an electric power network operates securely and efficiently under expected or potential scenarios. To obtain these adjustments, the *optimal power flow problem* must be solved. This problem is similar to the *power flow problem* discussed in Chapter 3 in the sense that it enforces the power flow equations. However, instead of merely providing the system state, it allows changing generator dispatches and other system controls in order to improve specific system properties. More specifically, the optimal power flow or OPF problem consists of determining the "best" set of generator dispatches and control device settings that result in the system supplying the load without violating any security or device limits. The notion of "best" may be based on cost of generation, total system losses, market surplus, or other measures [16]. Security constraints may include voltage magnitude and branch thermal overload limits [16], while control devices may include tap changing transformers, phase shifters, and switched shunts [18].

Two important challenges of solving OPF problems in operations planning are that certain control devices have discrete settings, and that the number of control adjustments obtained must be practical for system operators to execute. Since the late 80s, deficiencies of OPF software have been identified with regards to these two important problem aspects [81]. The first deficiency is that the common two-stage rounding technique for handling discrete variables, in particular switched shunt admittances, typically results in sub-optimal

solutions and has been reported in the literature to even cause feasibility issues. This technique consists of first solving the problem by treating all variables as continuous, rounding discrete variables to their nearest discrete values, and then solving the problem again with all discrete variables fixed [81] [50] [70]. The second deficiency is that OPF methods typically utilize a number of control device adjustments that are difficult for operators to execute in practice. One difficulty associated with overcoming this deficiency comes from having to specify costs of making control adjustments, or hard limits for the number of these that are practical to adjust [81].

Several authors have proposed alternative approaches to the common two-stage rounding approach for handling discrete variables in order to obtain better solutions and avoid potential feasibility issues. These may be roughly categorized as being based on penalty functions, sequential linearization, sensitivities, and more elaborate rounding techniques. Penalty function approaches treat all variables as continuous and gradually drive the discrete ones to valid discrete values with penalty functions added to the objective. This approach is explored in [50] and [49]. However, it is known that this technique in general is likely to introduce many undesirable stationary points and local minimizers [5] [61]. Approaches based on sequential linearization typically involve handling the discrete variables within a linear subproblem using a Mixed-Integer Linear Programming (MILP) solver [5]. The authors in [48] explore this technique and use an Interior Point Cutting-Plane Method (IPCPM) for solving the MILP subproblems. Sensitivity-based approaches use sensitivities of the objective function and constraints with respect to the discrete variables as a heuristic for determining how the discrete variables should be modified. The authors in [21] explore two approaches based on this idea, the first of which involves solving a MILP problem while the other involves the use of a merit function. Lastly, more elaborate rounding techniques typically consist of performing more gradual or selective rounding of the discrete variables compared to the two-stage approach. The authors in [52] explore two approaches of this type based on probabilistic and thresholding ideas. In particular, it is shown that the probabilistic technique performs much better on average than the two-stage rounding technique in terms of solution quality.

Similarly, several authors have proposed techniques for obtaining solutions of OPF problems that utilize a practical number of control device adjustments. For example, the authors in [20] and [78] explore using sensitivities of the objective function and constraints with respect to controls to select a small subset of these that may be used. The authors in [18]

explore two alternative approaches, both of which impose a hard limit on the number of allowed control adjustments. The first approach involves formulating the problem as a Mathematical Program with Equilibrium Constraints (MPEC), while the second and more promising one approximates the integral constraint that limits the number of control adjustments with a smooth non-linear inequality. In our view, using a hard limit for the number of allowed control adjustments may not be desirable. The reason is that an adequate limit is difficult to determine, and a poor choice for it may cause feasibility issues. Finally, the authors in [56] and [88] try to quantify the cost of adjusting control devices and use algorithms based on meta-heuristics that also handle the discrete variables associated with certain controls such as transformer tap ratios and switched shunt admittances.

Although several techniques have been explored in the literature for addressing the OPF deficiencies mentioned above, widely-used commercial OPF software still rely on common rounding techniques for handling discrete variables, and provide limited functionality for obtaining a practical number of control actions. For example, the commercial OPF packages reviewed for this work either do not provide ways to obtain a small number of control actions, or allow minimizing the number of control adjustments by using V-shaped penalty functions and alternating between a Linear Programming (LP) solver and a power flow solver. Approaches of this type typically lack robustness since alternating between solving the linearized OPF problem and the power flow problem is typically only locally convergent. This is especially true when a locally-convergent power flow method such as the widely-used NR method of Section 3.3 is applied.

In this chapter, new techniques for jointly addressing these OPF aspects are described and results assessing their effectiveness and computational requirements on mid-size power networks are presented. The idea explored for handling discrete variables consists of alternating between making progress towards solving a continuous relaxation, and evaluating different discrete variable choices in parallel. The technique used for obtaining a practical number of control actions is based on using a smooth approximation of a convex sparsity-inducing penalty that has been widely used for solving problems in Machine Learning and Statistics.

## 4.2 The Optimal Power Flow Problem

The OPF problem considered here is given by

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & P_c(y) + \|Cy\|_0 \\ \text{subject to} \quad & f(x,y) = 0 \\ & h(x,y) \leq 0 \\ & Dy \in \mathcal{D}. \end{aligned} \tag{4.1}$$

The vector $x$ consists of bus voltage angles, voltage magnitudes of unregulated buses, and generator reactive powers. The vector $y$ consists of adjustments of generator active powers, voltage magnitude set points of regulated buses, transformer tap ratios, phase shifts of phase-shifting transformers, and switched shunt admittances. The constraint $f(x,y) = 0$ denotes the power flow equations (2.37). The constraint $h(x,y) \leq 0$ represents generator active power limits (2.28), generator reactive power limits (2.29), bus voltage magnitude limits (2.1), branch thermal limits (2.17), and control device limits (2.19), (2.20) and (2.27). The constraint $Dy \in \mathcal{D}$ enforces discreteness of tap ratios and switched shunt admittances. The matrix $D$ extracts the sub-vector of $y$ that contains these discrete variables. Finally, $P_c(y)$ is the cost of active power generation, where $P_c$ is a separable convex quadratic function [77], and $\|Cy\|_0$ is the cost of making adjustments of generator dispatches, voltage set points, transformer tap ratios, phase shifts and switched shunt admittances. The matrix $C$ is a diagonal matrix of adjustment costs and the function $\|\cdot\|_0$ is the "zero norm" [85], which is defined by

$$\|z\|_0 := |\{\ z_i \mid z_i \neq 0\}|, \tag{4.2}$$

where $|\cdot|$ gives the cardinality of a set (number of elements).

## 4.3 Modeling Approach

Solving problem (4.1) is difficult due to the non-convex power flow equations, the discrete nature of the zero norm, the discreteness constraint of certain controls, and the fact that electric power networks are typically large. To make it more tractable, a few smoothing and approximation techniques are applied.

### 4.3.1 Sparse Control Adjustments

Using a function of the form of $\|Cy\|_0$ for quantifying the cost of making control adjustments provides two complications: First, the "zero norm" function $\|\cdot\|_0$ is not continuous and hence also not differentiable. Second, determining the cost of each control adjustment, *i.e.*, the entries of the diagonal matrix $C$, is difficult. Hence, the alternative considered here consists of using a smooth penalty function that is known to induce sparsity, and a trade-off parameter that defines its relative importance with respect to generation cost. This function is given by

$$P_s(y) := \frac{1}{n_s} \sum_i \sqrt{\left(\frac{y_i}{\Delta y_i}\right)^2 + \epsilon}, \tag{4.3}$$

where $\epsilon$ is a small positive scalar, *e.g.*, $10^{-6}$, $n_s$ is the size of the vector $y$, $\Delta y_i$ is a normalization factor given by

$$\Delta y_i := \max\{y_i^{\max} - y_i^{\min}, \eta\}, \tag{4.4}$$

$y_i^{\max}$ and $y_i^{\max}$ are control adjustment limits, and $\eta$ is a small positive scalar, *e.g.*, $10^{-4}$ [44]. Figure 4.1 shows a one-variable version of this function. Aside from inducing sparsity, this penalty also adds convexity and "regularization" to the problem, which make it easier to solve. This also means that the size of the control adjustments obtained are likely to become smaller as the emphasis on reducing the number of control adjustments is increased, at least on problems for which achieving feasibility is not difficult.

The penalty function $P_s$ considered here assigns equal weights to all control adjustments. However, in practice, it may be desirable to favor certain adjustments over others, or to consider certain adjustments as a last resort. This may be achieved easily by including non-negative weights $w_i$ inside the summation terms of the expression that defines $P_s$. To favor certain control adjustments, one would set their weights to be small. To consider certain controls as a last resort, one would set their weights to be very large.

### 4.3.2 Voltage and Branch Thermal Limits

Voltage magnitude limits (2.1) and branch thermal limits (2.17) are somewhat soft constraints that are desirable to satisfy but not mandatory, and hence may be treated as such [77] [91]. Branch thermal limit violations may be discouraged indirectly by discouraging

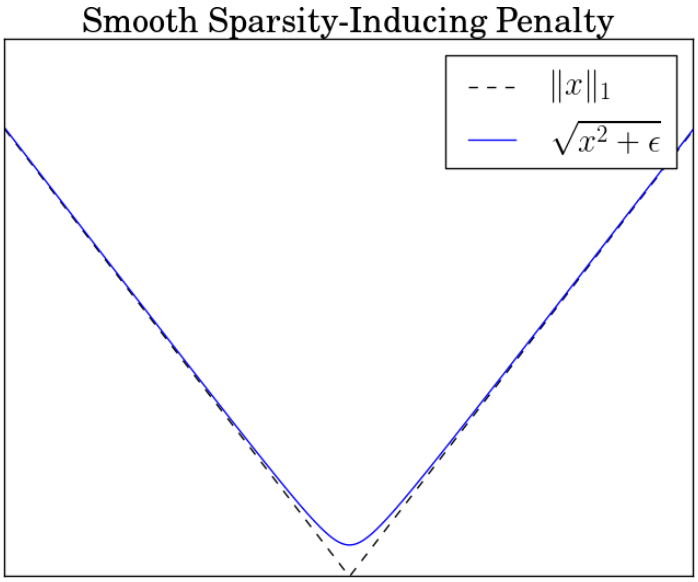


Figure 4.1: Smooth sparsity-inducing penalty.

large voltage angle differences across branches [16] [77] [54]. Hence, the penalty function

$$P_l(x,y) := \frac{1}{n}\sum_k \left(\frac{v_k - \bar{v}_k}{\Delta v_k}\right)^2 + \frac{1}{n_b}\sum_{(k,m)\in\mathcal{B}} \left(\frac{\theta_{km}}{\pi}\right)^2 \tag{4.5}$$

is considered here for discouraging voltage and branch thermal limit violations. In the above expression, $n$ is the number of buses, $\mathcal{B}$ and $n_b$ are the set and number of branches, respectively, $\bar{v}_k$ and $\Delta v_k$ are defined by

$$\bar{v}_k := 0.5(v_k^{\max} + v_k^{\min}) \tag{4.6}$$

$$\Delta v_k := \max\{v_k^{\max} - v_k^{\min}, \eta\}, \tag{4.7}$$

where $\eta$ is a small positive scalar, *e.g.*, $10^{-4}$, and $\theta_{km}$ is as defined in (2.15).

### 4.3.3 Variable Bounds

For handling simple scalar variable bounds of the form $u^{\min} \leq u \leq u^{\max}$, where $u^{\min}$ and $u^{\max} \in \mathbb{R}$, such as generator active and reactive power limits and control device limits, an alternative to the well known barrier and active-set strategies is explored. This alternative readily allows warm or infeasible starts and avoids dealing explicitly with the combinatorial problem of identifying the inequalities that are active at the solution. The approach consists

of enforcing bounds using the smooth nonlinear equality constraints

$$\psi_+(u) := \Delta u_+ + \epsilon^2/\Delta\bar{u} - \sqrt{\Delta u_+^2 + \epsilon^4/\Delta\bar{u}^2 + \epsilon^2} = 0 \tag{4.8}$$

$$\psi_-(u) := \Delta u_- + \epsilon^2/\Delta\bar{u} - \sqrt{\Delta u_-^2 + \epsilon^4/\Delta\bar{u}^2 + \epsilon^2} = 0, \tag{4.9}$$

where

$$\Delta u_+ := u^{\max} - u \tag{4.10}$$

$$\Delta u_- := u - u^{\min} \tag{4.11}$$

$$\Delta\bar{u} := \max\{u^{\max} - u^{\min}, \epsilon\}, \tag{4.12}$$

and $\epsilon$ is small a positive scalar, *e.g.*, $10^{-4}$. The key properties of these constraints are presented in the next theorems, whose proofs use results about the constraint functions $\psi_+$ and $\psi_+$ derived in Appendix C.

**Theorem 4.3.1.** *The functions $\psi_+$ and $\psi_-$ satisfy the property*

$$\max\left\{\ |\psi_+(u)|,\ |\psi_-(u)|\ \right\} \leq \epsilon \quad \iff \quad u^{\min} - \frac{\epsilon}{1 + \frac{\Delta\bar{u}}{\epsilon}} \leq u \leq u^{\max} + \frac{\epsilon}{1 + \frac{\Delta\bar{u}}{\epsilon}}. \tag{4.13}$$

*Proof.* Let $b = \epsilon^2/\Delta\bar{u}$. From the definition of $\Delta\bar{u}$, it holds that $\Delta\bar{u} \geq \epsilon$ and hence that $1 \geq \epsilon/\Delta\bar{u}$. Multiplying both sides by $\epsilon$ gives $\epsilon \geq b$. Hence, Lemma C.0.16 applies for both $\psi_+$ and $\psi_-$ and gives

$$|\psi_+(u)| \leq \epsilon \quad \iff \quad -\frac{b}{1 + \frac{b}{\epsilon}} \leq \Delta u_+ \tag{4.14}$$

$$|\psi_-(u)| \leq \epsilon \quad \iff \quad -\frac{b}{1 + \frac{b}{\epsilon}} \leq \Delta u_-. \tag{4.15}$$

From the definitions of $\Delta u_+$ and $\Delta u_-$, it holds that

$$-\frac{b}{1 + \frac{b}{\epsilon}} \leq \Delta u_+ \quad \iff \quad u \leq u^{\max} + \frac{b}{1 + \frac{b}{\epsilon}} \tag{4.16}$$

$$-\frac{b}{1 + \frac{b}{\epsilon}} \leq \Delta u_- \quad \iff \quad u \geq u^{\min} - \frac{b}{1 + \frac{b}{\epsilon}}. \tag{4.17}$$

Relation (4.13) follows from these and the fact that

$$\frac{b}{1+\frac{b}{\epsilon}} = \frac{\epsilon}{1+\frac{\Delta\bar{u}}{\epsilon}}. \tag{4.18}$$

□

Theorem 4.3.1 shows that with a feasibility tolerance of $\epsilon$, constraints (4.8) and (4.9) force $u$ to be inside an interval that is only slightly larger than $[u^{\min}, u^{\max}]$. For example, for a transformer tap ratio limit, with $\epsilon = 10^{-4}$ and a control range of $u^{\max} - u^{\min} = 0.2$ per unit, the maximum limit violation is

$$\frac{\epsilon}{1+\frac{\Delta\bar{u}}{\epsilon}} = 5 \times 10^{-8}. \tag{4.19}$$

**Theorem 4.3.2.** *If $u^{\max} - u^{\min} \geq \epsilon$, then*

$$\psi_+(u) = 0 \quad \textit{and} \quad \psi_-(u) = 0 \quad \Longleftrightarrow \quad u = \frac{u^{\max} + u^{\min}}{2}. \tag{4.20}$$

*Proof.* Let $b = \epsilon^2/\Delta\bar{u}$. From Lemma C.0.17, it holds that

$$\psi_+(u) = 0 \quad \Longleftrightarrow \quad \Delta u_+ = \frac{\epsilon^2}{2b} \tag{4.21}$$

$$\psi_-(u) = 0 \quad \Longleftrightarrow \quad \Delta u_- = \frac{\epsilon^2}{2b}. \tag{4.22}$$

If $u^{\max} - u^{\min} \geq \epsilon$, then the definition of $\Delta\bar{u}$ gives $\Delta\bar{u} = u^{\max} - u^{\min}$. Hence

$$\Delta u_+ = \Delta u_- = \frac{\Delta\bar{u}}{2} = \frac{u^{\max} - u^{\min}}{2}. \tag{4.23}$$

Then, the definitions of $\Delta u_+$ and $\Delta u_-$ give that

$$u = u^{\max} - \Delta u_+ = \frac{u^{\max} + u^{\min}}{2} \tag{4.24}$$

$$u = u^{\min} + \Delta u_- = \frac{u^{\max} + u^{\min}}{2}, \tag{4.25}$$

which proves (4.20). □

Theorem 4.3.2 shows that if the allowed variable range is non-trivial, satisfying constraints (4.8) and (4.9) exactly forces the variable to the midpoint of its range. Hence, the midpoint minimizes the infeasibilities. This implies that even with a nonzero feasibility tolerance, these constraints have a slight bias towards favoring points close to the midpoint of the allowed range.

Figure 4.2 provides an illustration of $\psi_+$ and $\psi_-$ as well as the $u$-interval they enforce with a feasibility tolerance of $\epsilon$.

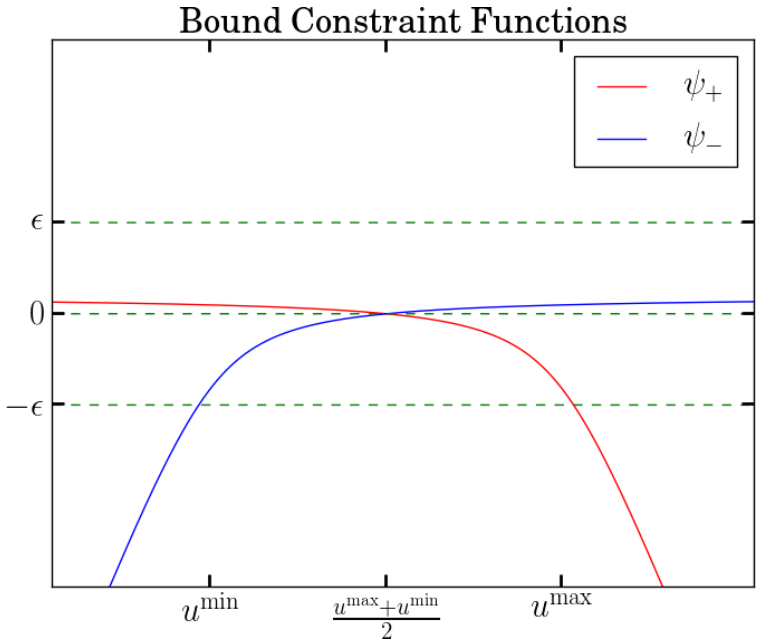


Figure 4.2: Bound constraint functions $\psi_+$ and $\psi_-$.

This approach for treating bound constraints is used for enforcing active power limits of generators (2.28), reactive power limits of generators (2.29), tap ratio limits of tap-changing transformers (2.19), phase shift limits of phase-shifting transformers (2.20), and susceptance limits of switched shunt devices (2.27).

A potential downside of using this approach for handling bounds as opposed to a barrier or an active set method, is that variables may violate excessively their bounds, especially during the initial iterations of an algorithm when using a poor initial point. This may result in functions of these variables being evaluated at points that have little or no physical meaning, and ill-conditioning may occur. One strategy that is used here to help overcome this issue is to include terms in the objective function that encourage variables to take on

physically meaningful values.

### 4.3.4 New Problem Formulation

Combining the ideas presented in Sections 4.3.1, 4.3.2 and 4.3.3, a more tractable OPF problem formulation is obtained:

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & \varphi(x,y) := \gamma_c P_c(y) + \gamma_s P_s(y) + \gamma_l P_l(x,y) \\ \text{subject to} \quad & c(x,y) = 0 \\ & Dy \in \mathcal{D}. \end{aligned} \tag{4.26}$$

Here, the smooth nonlinear constraint $c(x, y) = 0$ consists of the power flow equations (2.37) as well as the variable bounds expressed in terms of the functions $\psi_+$ and $\psi_-$ defined in Section 4.3.3. The positive scalars $\gamma_c$, $\gamma_s$ and $\gamma_l$ control the trade-off between generation cost, control sparsity, and soft-constraint violations.

As in the power flow problem formulation of Section 3.4, quadratic regularization may be added in order to make the objective function strongly convex in both $x$ and $y$ and encourage physically meaningful iterates. As described in Section 3.5, this may be exploited by the solution algorithm for constructing sufficient descent directions and ensuring global convergence.

## 4.4 Solution Algorithm

The main idea of the algorithm considered for solving the OPF problem as formulated in (4.26) consists of alternating between making progress towards solving a continuous relaxation of the problem, where all variables are treated as continuous, and evaluating different discrete variables choices. The evaluation of discrete variable choices is performed in a distributed manner by executing parallel algorithms that make progress towards solving (4.26) without modifying the discrete variables. The different discrete variable choices, or trial discrete points, are constructed using information obtained from the continuous relaxation. In particular, five strategies for constructing these points are explored.

For evaluating the performance of this approach, a benchmark method that resembles typical methods used in practice is considered. This method consists of solving the continuous relaxation of the problem, rounding the discrete variables to their nearest valid discrete

values, and then solving the problem again while keeping the discrete variables fixed.

In the sequel, derivatives are assumed to be with respect to the optimization variables $x$ and $y$. In particular, the notation $\nabla$ is used for denoting the gradient with respect to both $x$ and $y$.

### 4.4.1 Continuous Relaxation

The algorithm used for solving the continuous relaxation of problem (4.26), *i.e.*,

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & \varphi(x,y) \\ \text{subject to} \quad & c(x,y) = 0, \end{aligned} \tag{4.27}$$

is analogous to the power flow algorithm described in Section 3.5. In particular, it consists of approximately solving the sequence of subproblems

$$\underset{x,y}{\text{minimize}} \quad L(x,y,\lambda_k,\mu_k) := \mu_k\varphi(x,y) - \mu_k\lambda_k^T c(x,y) + \frac{1}{2}\|c(x,y)\|_2^2, \tag{4.28}$$

indexed by $k \in \mathbb{Z}_+$, where $\{\lambda_k\}$ are Lagrange multiplier estimates, and $\{\mu_k\}$ are positive penalty parameters. Search directions are obtained by using either the exact Hessian of $L$ or a positive definite approximation of it, as described in Section 3.5.3. Lagrange multiplier estimates are updated by using a regularized first-order update, as described in Section 3.5.4. Since the objective function in the OPF problem is not an artificial element added to the problem to help obtain feasibility, as in the proposed power flow problem formulation described in Section 3.4, the termination condition consists of satisfying both feasibility and first-order optimality according to the required feasibility and optimality tolerance. That is, satisfying

$$\|c_k\|_\infty < \epsilon_f \quad \text{and} \quad \|\nabla\varphi_k - J_k^T\lambda_k\|_\infty < \epsilon_o(\|\nabla\varphi_k\|_\infty + \gamma_k\|\lambda_k\|_\infty), \tag{4.29}$$

where $c_k$, $\nabla\varphi_k$ and $J_k$ are $c$, $\nabla\varphi$ and the Jacobian of $c$ at $(x_k, y_k)$, respectively, $\epsilon_f$ and $\epsilon_o$ are feasibility and optimality tolerances, respectively, and

$$\gamma_k := \max_{i,j} |J_{ij}(x_k, y_k)|. \tag{4.30}$$

This algorithm is referred to as *crOPF*, which stands for *continuous relaxation OPF*.

Key parameters of the crOPF algorithm are shown in Table 4.1. More details about the algorithm and about the role of its parameters can be found in Sections 3.4 and 3.5.

Table 4.1: Parameters of the crOPF algorithm.

| name | default value | description |
|---|---|---|
| $\gamma_c \in \mathbb{R}_{++}$ | $1 \times 10^0$ | weight for generation cost |
| $\gamma_s \in \mathbb{R}_{++}$ | $1 \times 10^1$ | weight for sparsity penalty |
| $\gamma_l \in \mathbb{R}_{++}$ | $1 \times 10^2$ | weight for soft-constraint penalty |
| $\kappa \in \mathbb{R}_{++}$ | $1 \times 10^1$ | coefficient for computing initial $\mu$ |
| $\beta_s \in (0,1)$ | $2 \times 10^{-1}$ | factor for decreasing $\mu$ significantly |
| $\beta_l \in (0,1)$ | $9 \times 10^{-1}$ | factor for decreasing $\mu$ slightly |
| $\tau_s \in (0,1)$ | $2 \times 10^{-1}$ | factor that defines required reductions in $\|\nabla L\|_\infty$ |
| $\tau_f \in (0,1)$ | $2 \times 10^{-1}$ | factor that defines desirable reductions in $\|c\|_\infty$ |
| $\epsilon_f \in (0,1)$ | $1 \times 10^{-4}$ | feasibility tolerance for $c(x) = 0$ in per unit base power |
| $\epsilon_o \in (0,1)$ | $1 \times 10^{-4}$ | tolerance for first-order optimality conditions |
| $\zeta_s \in \mathbb{R}_{++}$ | $1 \times 10^{-2}$ | threshold for using second-order power flow information |
| $\rho_1 \in \mathbb{R}_{++}$ | $1 \times 10^0$ | coefficient for regularization in first-order $\lambda$ update |

### 4.4.2 Exploration of Discrete Space

As mentioned above, the approach considered here for obtaining optimal or near-optimal discrete variable values consists of alternating between two algorithms. The first algorithm, or *guide*, makes progress towards solving the continuous relaxation of the problem. The second algorithm, or *evaluator*, uses information obtained from the relaxation to construct a valid discrete point, and evaluates this point by making progress towards solving (4.26) without modifying the discrete variables. By using several evaluator algorithms, many trial discrete points may be evaluated in parallel in order to find better solution estimates. The OPF algorithm that utilizes this strategy for exploring the space of discrete variables is referred to as *alOPF*, which stands for *alternating OPF*.

More specifically, the method performs the following four steps every outer iteration:

1. **Relaxation:** make progress towards solving the continuous relaxation.
2. **Construction:** construct trial discrete points using a specific strategy.
3. **Evaluation:** evaluate the constructed trial discrete points in parallel.

4. **Update:** update the current solution estimate with information obtained from the "most promising" discrete point evaluation.

The relaxation step consists of performing one outer iteration of the crOPF algorithm, including dual variable update, towards solving the continuous relaxation problem

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & \varphi(x,y) \\ \text{subject to} \quad & c(x,y) = 0. \end{aligned} \tag{4.31}$$

That is, solving

$$\underset{x,y}{\text{minimize}} \quad L(x,y,\lambda_k,\mu_k) \tag{4.32}$$

until $\|\nabla L\|_\infty$ is reduced by a factor $\tau_s$ (Table 4.1), where $\lambda_k$ is the current vector of Lagrange multiplier estimates, and $\mu_k$ is the current penalty parameter. The current solution estimate $(x_k, y_k)$ of problem (4.26) is used as initial point. The new primal point obtained is denoted by $(\tilde{x}_k, \tilde{y}_k)$. A new Lagrange multiplier estimate is then obtained by solving

$$\underset{\lambda}{\text{minimize}} \quad \|Z^T(\nabla\tilde{\varphi}_k - \tilde{J}_k^T\lambda)\|_2^2 + \rho_1\mu_k\|\lambda - \lambda_k\|_2^2 \tag{4.33}$$

using the procedure described in Section 3.5.4, where the columns of $Z$ span the null space of the matrix $[\,0\ D\,]$ that extracts the discrete variables (transformer tap ratios, switched shunt susceptances), $\tilde{\varphi}_k$ and $\tilde{J}_k$ are $\varphi$ and the Jacobian of $c$ at $(\tilde{x}_k, \tilde{y}_k)$, respectively, and $\rho_1$ is a positive scalar (Table 4.1). The solution of (4.33) is denoted by $\tilde{\lambda}_k$. The use of the null space matrix $Z$ here is designed to encourage $\tilde{\lambda}_k$ to satisfy the optimality conditions of the problem that considers the discrete variables fixed.

The construction step consists of using a specific strategy, as described below, to construct $m \in \mathbb{N}$ trial discrete points $(\tilde{x}_k^i, \tilde{y}_k^i, \tilde{\lambda}_k^i)$, $i \in [m]$, to evaluate.

The evaluation step consists of evaluating each of these $m$ trial discrete points to determine how the discrete variables should be updated. More specifically, for each $i$, the primal-dual trial point $(\tilde{x}_k^i, \tilde{y}_k^i, \tilde{\lambda}_k^i)$ is evaluated by using it as a starting point and solving

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & \varphi(x,y) \\ \text{subject to} \quad & c(x,y) = 0 \\ & Dy = D\tilde{y}_k^i \end{aligned} \tag{4.34}$$

with the crOPF algorithm, using $\mu_k$ as initial penalty parameter and moving only in the null space of $[\,0 \;\; D\,]$ until $\|Z^T \nabla L\|_\infty$ is reduced by a factor $\tau_s$ with respect to that at $(x_k, y_k, \lambda_k, \mu_k)$.

The update step consists of setting the new solution estimate $(x_{k+1}, y_{k+1}, \lambda_{k+1})$ and new penalty parameter $\mu_{k+1}$ to the ones obtained by the "best" trial-point evaluation that achieved the required progress. More specifically, after an evaluator algorithm, say the $i$-th one, achieves the required progress, it informs all the other evaluator algorithms about its achieved objective value, say $\varphi_i$. After receiving this information, the $j$-th evaluator, where $j \neq i$, stops immediately if $\varphi_j > (1+\xi)\varphi_i$, where $\varphi_j$ is the objective value associated with its current iterate, and $\xi \in (0,1)$. If $\varphi_j \leq (1+\xi)\varphi_i$, it continues working towards achieving the required progress. After all evaluator algorithms stop, the solution estimate and penalty parameter obtained from the evaluator algorithm that achieved the required progress and obtained the lowest objective value are used for starting the next iteration of the algorithm.

The termination condition consists of satisfying the optimality conditions of the problem solved by an evaluator algorithm to the required accuracy, *i.e.*, satisfying

$$\|c_k\|_\infty < \epsilon_f \quad \text{and} \quad \|Z^T(\nabla\varphi_k - J_k^T\lambda_k)\|_\infty < \epsilon_o(\|\nabla\varphi_k\|_\infty + \gamma_k\|\lambda_k\|_\infty) \tag{4.35}$$

with a valid discrete point, where $c_k$, $\nabla\varphi_k$, $J_k$ and $\gamma_k$ are as defined in Section 4.4.1.

Algorithm 4 shows pseudocode for the alOPF algorithm.

**Algorithm 4** alOPF algorithm

Given $(x_0, y_0, \lambda_0)$ and parameters from Table 4.1
$k \leftarrow 0$
Set initial penalty $\mu_0$ as described in Section 3.5.1
**while** True **do**
    **if** $(x_k, y_k, \lambda_k)$ satisfies (4.35) with $Dy_k \in \mathcal{D}$ **then**
        **return** $(x_k, y_k, \lambda_k)$
    **end if**
    Obtain $(\tilde{x}_k, \tilde{y}_k, \tilde{\lambda}_k)$ by solving (4.32) until $\|\nabla L\|_\infty$ is reduced by $\tau_s$, and solving (4.33)
    Construct discrete points $(\tilde{x}_k^i, \tilde{y}_k^i, \tilde{\lambda}_k^i)$, $i \in [m]$, using a specific strategy
    Evaluate points by solving (4.34), fixing $Dy$, until $\|Z^T\nabla L\|_\infty$ is reduced by $\tau_s$
    Set $x_{k+1}$, $y_{k+1}$ $\lambda_{k+1}$ and $\mu_{k+1}$ with the ones from the "best" evaluation
    $k \leftarrow k+1$
**end while**

**Construction Strategies for Trial Discrete Points**

Five strategies for constructing trial discrete points are considered. Two of these allow creating one trial discrete point, while the others allow creating many of these for performing evaluations in parallel. Each of the construction strategies is described next.

1. **Rounding:** This simple construction strategy takes the point obtained by the relaxation step, *i.e.*, $(\tilde{x}_k, \tilde{y}_k, \tilde{\lambda}_k)$, and simply rounds the discrete variables to their nearest valid discrete values. Hence, only one trial discrete point is constructed.

2. **Coordinate Descent:** This strategy takes the point obtained by the rounding strategy, say $(\hat{x}, \hat{y}, \hat{\lambda})$, and replaces $\hat{y}$ with $\hat{y} + \Delta\hat{y}$, where $\Delta\hat{y}$ partially solves the problem

$$\begin{aligned} \underset{\Delta\hat{y}}{\text{minimize}} \quad & L(\hat{x}_k, \hat{y} + \Delta\hat{y}, \hat{\lambda}, \mu) \\ \text{subject to} \quad & \Delta\hat{y} \in \mathcal{Y}. \end{aligned} \tag{4.36}$$

   Here, $\mu$ is the current penalty parameter and $\mathcal{Y}$ is a discrete set such that $\Delta\hat{y} \in \mathcal{Y}$ only moves the discrete variables either to the nearest right or left valid discrete values of the discrete variables in $\tilde{y}_k$. Coordinate descent [51] is used to partially solve (4.36), and the coordinate to modify during each iteration is chosen as the one with the largest sensitivity with respect to $L$. Only a relatively small number iterations are performed, *e.g.*, 50, and only one trial discrete point is obtained.

3. **Thresholding:** This strategy partitions the interval $[0, 1]$ into $m-1$ intervals of equal size. The partition points are denoted by $\alpha_i$, $i \in [m]$. For each $i \in [m]$, each discrete variable in $\tilde{y}_k$, the point obtained by the relaxation step, is rounded to the nearest right valid discrete value if it is more than $\alpha_i$ of the way there from the nearest left discrete value. Otherwise, it is rounded to the nearest left discrete value. For this strategy, $m$ trial discrete points are constructed, one for each threshold $\alpha_i$.

4. **Optimal Thresholding:** This strategy partitions the interval $[0, 1]$ into $\tilde{m} - 1$ intervals of equal size, where $\tilde{m} \gg m$. The partition points are denoted by $\alpha_i$, $i \in [\tilde{m}]$. For each $i \in [\tilde{m}]$, each discrete variable in $\tilde{y}_k$, the point obtained by the relaxation step, is rounded to the nearest right or left valid discrete value using threshold $\alpha_i$, as done by the thresholding strategy. The set $\mathcal{I} \subset [\tilde{m}]$ corresponding to the $m$ thresholds that result in the lowest value of augmented Lagrangian $L$ is identified. The $m$ trial

points constructed by this strategy are the ones obtained by rounding according to the optimal thresholds $\alpha_i$, $i \in \mathcal{I}$.

5. **Discrete Line Search:** This strategy uses the current point $(x_k, y_k, \lambda_k)$ and the point $(\tilde{x}_k, \tilde{y}_k, \tilde{\lambda}_k)$ obtained from the relaxation step to construct a primal-dual search direction given by

$$\Delta x = \tilde{x}_k - x_k \tag{4.37}$$

$$\Delta y = \tilde{y}_k - y_k \tag{4.38}$$

$$\Delta \lambda = \tilde{\lambda}_k - \lambda_k. \tag{4.39}$$

Then, $m$ equally-spaced points along the primal-dual line segment

$$(x_k, y_k, \lambda_k) + \alpha(\Delta x, \Delta y, \Delta \lambda),\ \alpha \in [0, 1], \tag{4.40}$$

are obtained by using equally spaced $\alpha_i \in [0, 1]$, $i \in [m]$, including the endpoints. The $m$ trial points constructed by this strategy are obtained by rounding the discrete variables of each of the points on the line segment to their nearest valid discrete value.

### 4.4.3 Benchmark Method

To evaluate the effectiveness in obtaining higher-quality solutions and the computational requirements of the alOPF algorithm with the various construction strategies, a benchmark algorithm is considered. This benchmark algorithm represents algorithms commonly used in practice for handling discrete variables in OPF. It consists of first solving the problem using the crOPF algorithm and treating all variables as continuous, then rounding the discrete variables to their nearest valid discrete values, and then resolving the problem again with the crOPF algorithm but keeping the discrete variables fixed. This algorithm is referred to as *rrOPF*, which stands for *round and resolve OPF*.

## 4.5 Implementation

The OPF algorithms described here were implemented in the Python Programming Language using Python's C API for speeding up function evaluations and matrix constructions. The linear algebra operations were performed using the Python packages `Numpy` and `Scipy`,

and the sequential version of the multifrontal sparse direct solver `MUMPS` [3] through the Python wrapper `pymumps`. Parallelization of the evaluation of trial discrete points for exploring the discrete variable space was implemented using Python's multiprocessing module.

## 4.6 Experiments

Results of various experiments that assess the performance and computational requirements of the described OPF techniques are presented. In particular, results regarding the effectiveness of the smooth sparsity-inducing penalty for obtaining a manageable number of control actions are shown as well as ones regarding the trade-off between solution quality and number of control actions. Lastly, results comparing the solution properties and computational requirements of the crOPF and rrOPF algorithms with those of the alOPF algorithm with each of the construction strategies are also presented.

The computer used for performing the experiments was an 8-core PC with Intel® Core™ i7 CPU, 2.80 GHz, 8 GB of RAM, and running the operating system Ubuntu 12.04.

The test cases used consisted of four OPF problems from two different mid-size North American System Operators. The data obtained consisted of load flow data only and did not include generation cost data. Therefore, the same quadratic cost function was used for all generators, each having a linear-term coefficient of 20 and a quadratic-term coefficient of 0.01, as is done in the examples of [68]. Table 4.2 shows the properties of the OPF test cases, including number of buses, number of branches (transmission lines and transformers), number and properties of discrete variables corresponding to transformer tap ratios, number and properties of discrete variables corresponding to switched shunt devices, and total system load. Cases C and D were constructed from Cases A and B by increasing the loads by factors of 2 and 1.5, respectively, while maintaining power factors. Transformer tap and switched shunt position data is shown as $\mu \pm \sigma$, where $\mu$ is the average and $\sigma$ is the standard deviation.

### 4.6.1 Sparse Control Adjustment Profiles

The first experiment consisted of displaying the control adjustments utilized by the crOPF algorithm on each of the test cases for small and large sparsity weights in order to determine the effectiveness of the smooth sparsity-inducing penalty $P_s$. Figures 4.4-4.7 show the

Table 4.2: Properties of OPF test cases.

| name | buses | branches | taps | positions | shunts | positions | load (MW) |
|---|---|---|---|---|---|---|---|
| Case A | 2454 | 2781 | 1038 | $21 \pm 9$ | 170 | $2 \pm 1$ | 7320 |
| Case B | 2468 | 3215 | 135 | $30 \pm 6$ | 160 | $2 \pm 1$ | 39200 |
| Case C | 2454 | 2781 | 1038 | $21 \pm 9$ | 170 | $2 \pm 1$ | 14640 |
| Case D | 2468 | 3215 | 135 | $30 \pm 6$ | 160 | $2 \pm 1$ | 58800 |

results obtained. The left plots correspond to small sparsity weights while the right plots correspond to large sparsity weights. The adjustment of a control device is given as a fraction of its control range. For example, for a tap adjustment of $\Delta t$, the quantity shown in the figure is $\Delta t/(t^{\max} - t^{\min})$, where $t^{\max}$ and $t^{\min}$ are the tap ratio's upper and lower limits, respectively. Adjustments of each type of control device are shown with a different color. Figure 4.3 shows the control quantity associated with each color. The plots also show the number of control actions and cost obtained on each of the cases. A control action is defined as a control adjustment that is greater than 2% of its control range. The cost shown on the plots is given by $\gamma_c P_c + \gamma_l P_l$, *i.e.*, the part of the objective function $\varphi$ that measures solution quality in terms of generation cost, voltage magnitude profiles, and phase angle differences across branches.

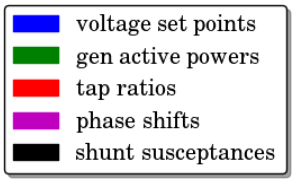


Figure 4.3: Colors associated with OPF control quantities.

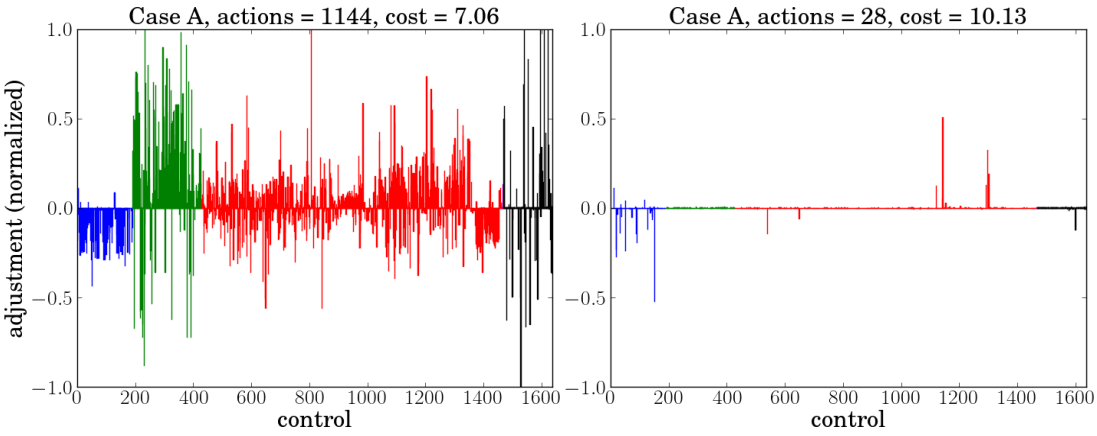


Figure 4.4: Control adjustment profiles for OPF Case A.

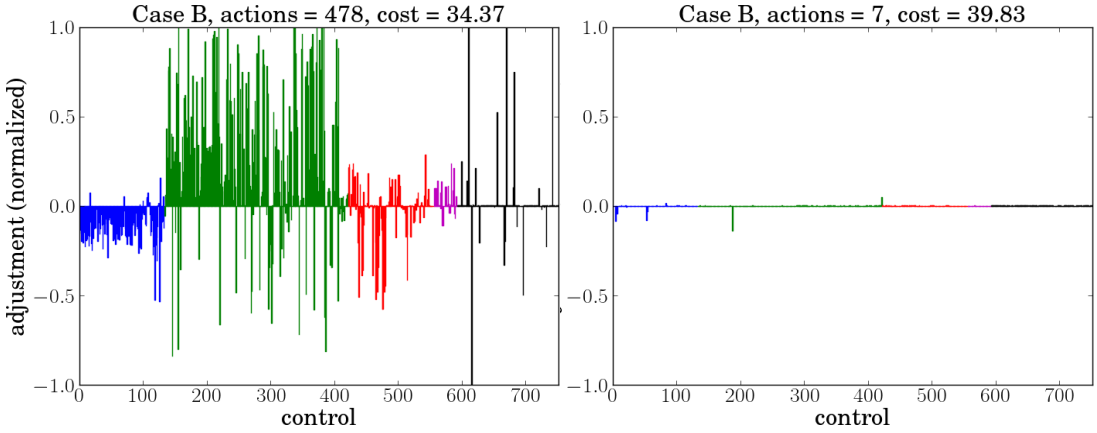


Figure 4.5: Control adjustment profiles for OPF Case B.

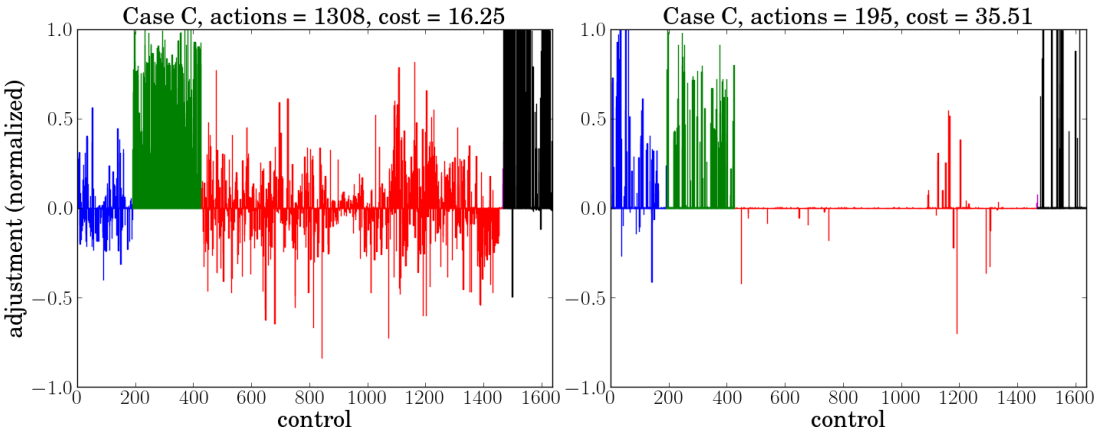


Figure 4.6: Control adjustment profiles for OPF Case C.

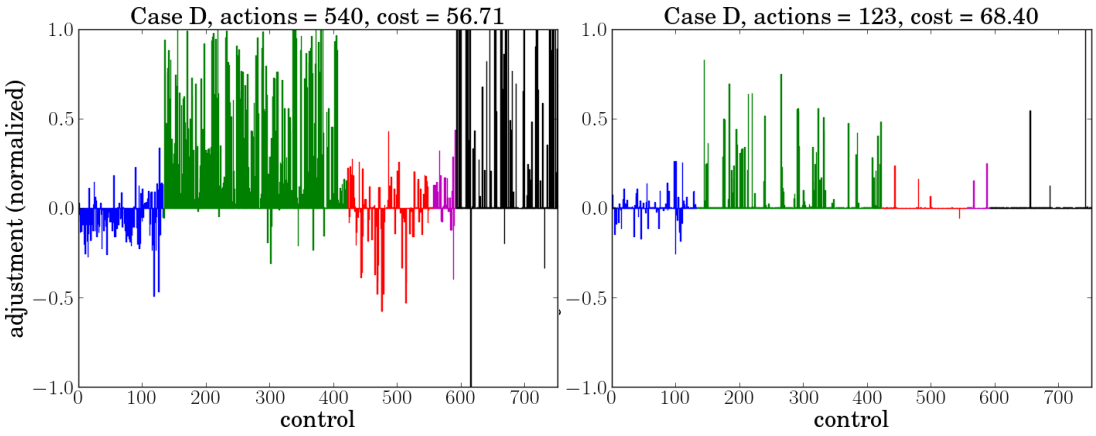


Figure 4.7: Control adjustment profiles for OPF Case D.

As the figures show, the smooth penalty $P_s$ allows reducing the number of control actions. As expected, the solution quality (as measured by the cost) obtained by allowing many control actions is better than the one obtained with only a few. This trade-off between cost or solution quality and number of control actions was investigated more in depth by the experiment described next.

### 4.6.2 Cost-Sparsity Trade-Off

The second experiment consisted of characterizing the trade-off between cost and number of control actions, where the cost refers to the part of the objective function that measures solution quality in terms of generation cost, voltage magnitude profiles and phase angle differences across branches. To do this, the crOPF algorithm was applied to each case for increasing values of the sparsity weight $\gamma_s$, while keeping the weights $\gamma_c$ and $\gamma_l$ associated with generation cost and soft constraints fixed at 1 and 10, respectively. Figures 4.8-4.11 show the results obtained. In the figures, the left plots show how the cost $\gamma_c P_c + \gamma_l P_l$ varies as the sparsity weight $\gamma_s$ is increased, while the right plots show how the number of control actions varies as the sparsity weight $\gamma_s$ is increased. The cost curves are shown in units of cost obtained with the smallest sparsity weight.

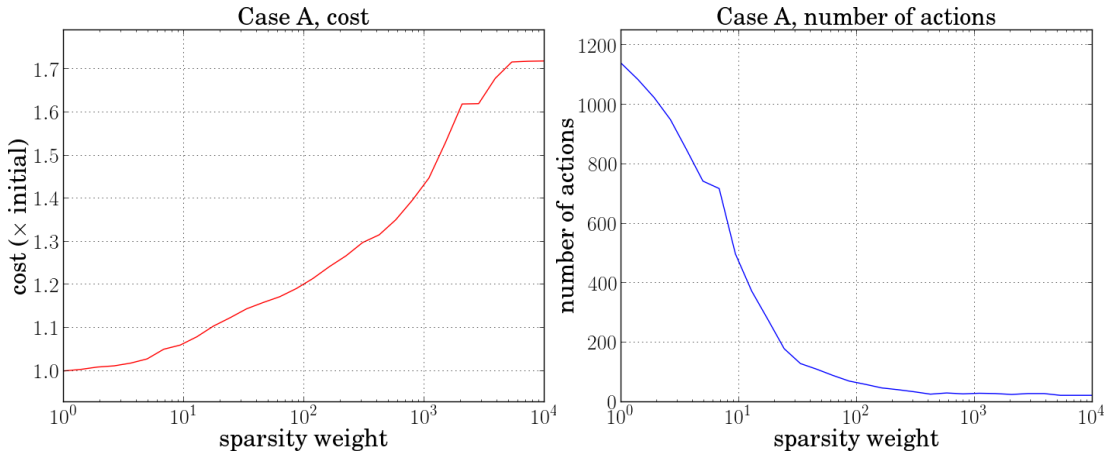


Figure 4.8: Trade-off between cost and number of control actions for OPF Case A.

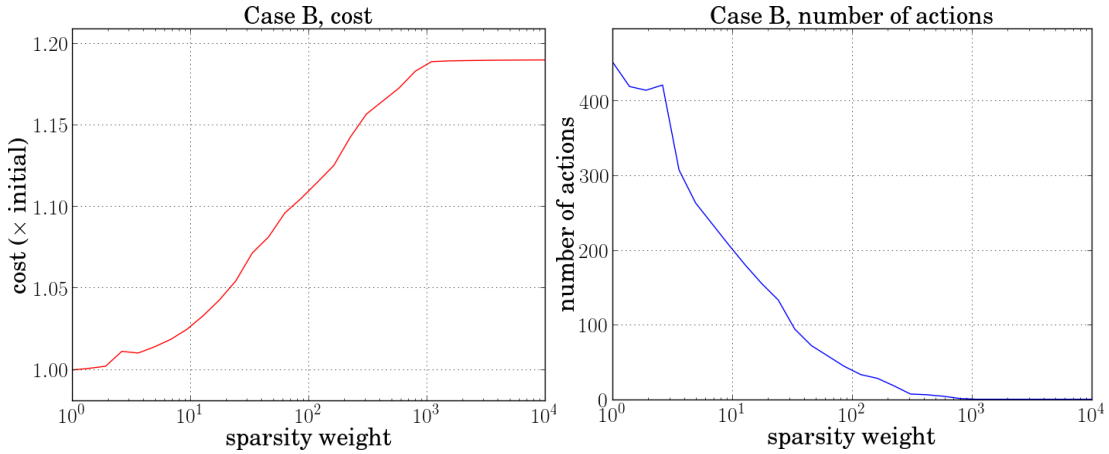


Figure 4.9: Trade-off between cost and number of control actions for OPF Case B.

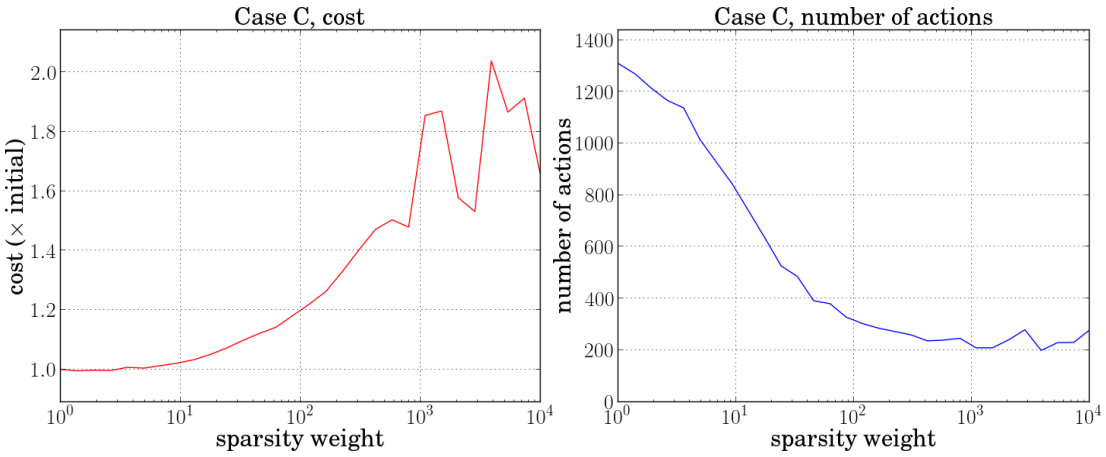


Figure 4.10: Trade-off between cost and number of control actions for OPF Case C.

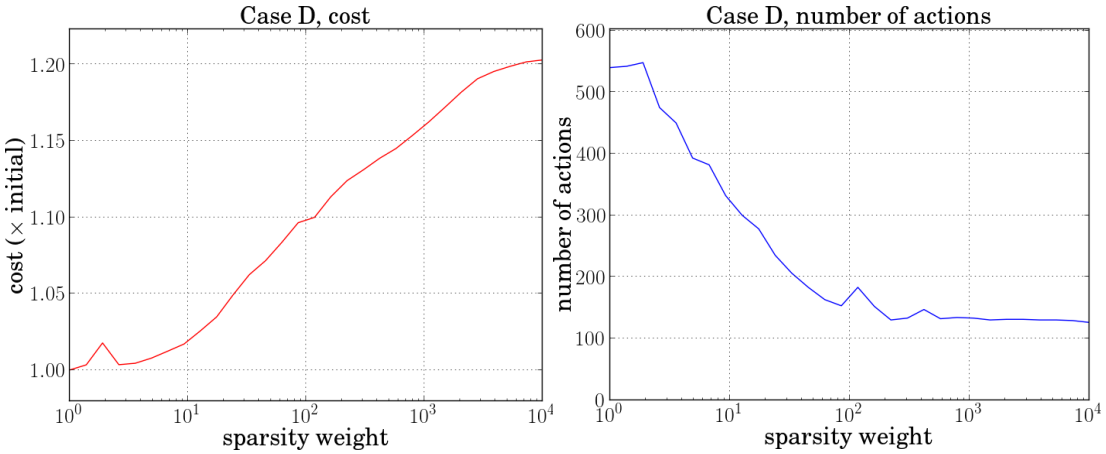


Figure 4.11: Trade-off between cost and number of control actions for OPF Case D.

As the figures show, increasing the sparsity weight $\gamma_s$ while keeping the generation cost and soft-constraint penalty weights $\gamma_c$ and $\gamma_l$, respectively, fixed, reduces the number of control actions at the expense of degradation in solution quality. For the simple cases, *i.e.*, Cases A and B, the number of control actions is reduced to 0 or almost 0. This makes sense since for these simple cases obtaining feasibility is not an issue. On the other hand, for the hard cases, *i.e.*, Cases C and D, the number of control actions does not go to zero as $\gamma_s$ is increased. This is because these cases require significant control adjustments to allow the power network to accommodate the required transmission of power.

### 4.6.3 Method Comparison

The third experiment consisted of comparing the solution properties and computational requirements of all the methods on each of the OPF cases with various sparsity weights. In particular, values of sparsity weight $\gamma_s$ of 1, 10 and 100 were used while keeping $\gamma_c$ and $\gamma_l$ fixed at 1 and 100, respectively. It was observed that the quality of the solution obtained by all the methods had a significant dependence on the parameter $\kappa$, which determines the initial penalty parameter (Section 3.5.1), and hence the initial importance of minimizing the objective function relative to that of reducing the infeasibilities. Hence, for each of the cases, each method was executed with $\kappa \in \{1, 10\}$ and the best solution obtained was recorded. For a few cases, some methods failed with such values of $\kappa$ and hence different values had to be used (shown in bold). For alOPF algorithms, the relaxation step was stopped if it required more than 100 iterations. Also, the parameter $\xi$ used for determining whether an evaluator algorithm should stop, as described in Section 4.4.2, was set to 0.5. For the alOPF algorithm with optimized thresholding strategy, a number of $\tilde{m} = 50$ thresholds were considered. For the alOPF algorithm with coordinate descent strategy, a number of 50 coordinates were considered. Tables 4.5-4.16 show the data collected. In these tables, the strategies used with the alOPF method to construct trial discrete points are given as codes, and these codes are described in Table 4.3. In the tables of results, the column labeled $\Delta\varphi$ (%) shows the difference in objective value obtained by the method with respect to that obtained by the crOPF method. The column labeled "rank" gives a value between 1 and 7 that ranks the method according to objective value achieved for a particular case and sparsity weight, where 1 is given to the one that obtained the lowest value. The columns labeled $t$ ($s$) and $t/t_0$ show the time in seconds and the time ratio with respect to that of the crOPF method, respectively. The columns labeled $v^{\max}$ and $v^{\min}$ show the solution's maximum and minimum bus voltage magnitudes, respectively, in per unit bus nominal voltage. The column labeled "actions" shows the number of control actions utilized by the method for solving a specific case. Lastly, the column labeled $\kappa$ shows the value of the parameter $\kappa$ used, which is either 1 or 10 unless the method failed with both of these values.

The results obtained on each of the cases were aggregated in order to find an overall ranking of methods in terms of finding solutions with the lowest objective value, and to determine how much longer on average they are with respect to the crOPF method. The solution quality ranking was obtained by adding all the individual ranks given to each method, and then ranking them again based on the obtained rank sum. Again, the value

1 was assigned to the one with the lowest value, *i.e.*, the "best" one. Table 4.4 shows the aggregate results obtained.

Table 4.3: Codes for rounding strategies of alOPF algorithms.

| code | strategy | parallel processes |
|---|---|---|
| RO 1 | Rounding | 1 |
| CD 1 | Coordinate Descent | 1 |
| TH $m$ | Thresholding | $m$ |
| OT $m$ | Optimal Thresholding | $m$ |
| DL $m$ | Discrete Line Search | $m$ |

Table 4.4: Method comparison - aggregate results.

| method | rank sum | final rank | average $t/t_0$ |
|---|---|---|---|
| crOPF | 27 | 1 | 1.00 |
| rrOPF | 58 | 5 | 1.85 |
| alOPF - RO 1 | 58 | 5 | 2.47 |
| alOPF - CD 1 | 72 | 7 | 3.15 |
| alOPF - TH 8 | 39 | 3 | 5.29 |
| alOPF - OT 8 | 41 | 4 | 7.10 |
| alOPF - DL 8 | 37 | 2 | 4.51 |

Table 4.5: Method comparison on OPF Case A with $\gamma_s = 1$.

| method | $\Delta\varphi$ (%) | rank | $t$ ($s$) | $t/t_0$ | $v^{\max}$ | $v^{\min}$ | actions | $\kappa$ |
|---|---|---|---|---|---|---|---|---|
| crOPF | 0.0000 | 1 | 9.82 | 1.00 | 1.06 | 0.89 | 1138 | 10 |
| rrOPF | 0.4698 | 6 | 24.79 | 2.52 | 1.06 | 0.89 | 1113 | 10 |
| alOPF - RO 1 | 0.4587 | 3 | 22.80 | 2.32 | 1.09 | 0.89 | 1124 | 10 |
| alOPF - CD 1 | 0.4950 | 7 | 20.17 | 2.05 | 1.06 | 0.89 | 1116 | 10 |
| alOPF - TH 8 | 0.4644 | 4 | 52.64 | 5.36 | 1.06 | 0.89 | 1105 | 10 |
| alOPF - OT 8 | 0.4122 | 2 | 47.86 | 4.87 | 1.06 | 0.89 | 1112 | 10 |
| alOPF - DL 8 | 0.4692 | 5 | 39.48 | 4.02 | 1.09 | 0.89 | 1117 | 10 |

Table 4.6: Method comparison on OPF Case A with $\gamma_s = 10$.

| method | $\Delta\varphi$ (%) | rank | $t$ $(s)$ | $t/t_0$ | $v^{\max}$ | $v^{\min}$ | actions | $\kappa$ |
|---|---|---|---|---|---|---|---|---|
| crOPF | 0.0000 | 1 | 8.53 | 1.00 | 1.06 | 0.90 | 466 | 10 |
| rrOPF | 0.4510 | 2 | 21.39 | 2.51 | 1.06 | 0.90 | 480 | 10 |
| alOPF - RO 1 | 2.4033 | 7 | 18.59 | 2.18 | 1.07 | 0.90 | 534 | 1 |
| alOPF - CD 1 | 1.9751 | 6 | 21.71 | 2.55 | 1.06 | 0.90 | 526 | 1 |
| alOPF - TH 8 | 1.2181 | 5 | 49.87 | 5.85 | 1.07 | 0.90 | 488 | 10 |
| alOPF - OT 8 | 1.0525 | 3 | 52.88 | 6.20 | 1.06 | 0.90 | 495 | 1 |
| alOPF - DL 8 | 1.0781 | 4 | 43.05 | 5.05 | 1.07 | 0.90 | 480 | 10 |

Table 4.7: Method comparison on OPF Case A with $\gamma_s = 100$.

| method | $\Delta\varphi$ (%) | rank | $t$ $(s)$ | $t/t_0$ | $v^{\max}$ | $v^{\min}$ | actions | $\kappa$ |
|---|---|---|---|---|---|---|---|---|
| crOPF | 0.0000 | 1 | 4.30 | 1.00 | 1.09 | 0.90 | 64 | 1 |
| rrOPF | 3.6033 | 4 | 8.36 | 1.94 | 1.09 | 0.89 | 118 | 1 |
| alOPF - RO 1 | 3.5530 | 2 | 20.94 | 4.87 | 1.09 | 0.89 | 118 | 1 |
| alOPF - CD 1 | 3.7115 | 7 | 29.35 | 6.83 | 1.09 | 0.89 | 121 | 10 |
| alOPF - TH 8 | 3.7006 | 6 | 34.75 | 8.09 | 1.09 | 0.87 | 117 | 1 |
| alOPF - OT 8 | 3.6111 | 5 | 67.86 | 15.79 | 1.09 | 0.89 | 119 | 10 |
| alOPF - DL 8 | 3.6025 | 3 | 42.46 | 9.88 | 1.09 | 0.89 | 116 | 10 |

Table 4.8: Method comparison on OPF Case B with $\gamma_s = 1$.

| method | $\Delta\varphi$ (%) | rank | $t$ $(s)$ | $t/t_0$ | $v^{\max}$ | $v^{\min}$ | actions | $\kappa$ |
|---|---|---|---|---|---|---|---|---|
| crOPF | 0.0000 | 3 | 8.32 | 1.00 | 1.37 | 0.90 | 452 | 10 |
| rrOPF | 0.0552 | 5 | 8.59 | 1.03 | 1.37 | 0.89 | 450 | 1 |
| alOPF - RO 1 | 0.0139 | 4 | 12.63 | 1.52 | 1.37 | 0.89 | 447 | 1 |
| alOPF - CD 1 | 0.5218 | 7 | 25.46 | 3.06 | 1.36 | 0.89 | 485 | 10 |
| alOPF - TH 8 | −0.0029 | 2 | 36.92 | 4.44 | 1.37 | 0.89 | 450 | 1 |
| alOPF - OT 8 | 0.2197 | 6 | 81.86 | 9.84 | 1.37 | 0.89 | 485 | 10 |
| alOPF - DL 8 | −0.0376 | 1 | 14.20 | 1.71 | 1.37 | 0.89 | 447 | 1 |

Table 4.9: Method comparison on OPF Case B with $\gamma_s = 10$.

| method | $\Delta\varphi$ (%) | rank | $t$ $(s)$ | $t/t_0$ | $v^{\max}$ | $v^{\min}$ | actions | $\kappa$ |
|---|---|---|---|---|---|---|---|---|
| crOPF | 0.0000 | 1 | 3.85 | 1.00 | 1.37 | 0.88 | 199 | 1 |
| rrOPF | 0.0968 | 7 | 6.94 | 1.80 | 1.37 | 0.88 | 203 | 1 |
| alOPF - RO 1 | 0.0724 | 4 | 9.40 | 2.44 | 1.37 | 0.88 | 203 | 1 |
| alOPF - CD 1 | 0.0726 | 6 | 11.46 | 2.98 | 1.37 | 0.88 | 203 | 1 |
| alOPF - TH 8 | 0.0644 | 2 | 25.59 | 6.65 | 1.37 | 0.88 | 204 | 10 |
| alOPF - OT 8 | 0.0647 | 3 | 25.98 | 6.75 | 1.37 | 0.88 | 203 | 1 |
| alOPF - DL 8 | 0.0724 | 4 | 15.03 | 3.91 | 1.37 | 0.88 | 203 | 1 |

Table 4.10: Method comparison on OPF Case B with $\gamma_s = 100$.

| method | $\Delta\varphi$ (%) | rank | $t$ $(s)$ | $t/t_0$ | $v^{\max}$ | $v^{\min}$ | actions | $\kappa$ |
|---|---|---|---|---|---|---|---|---|
| crOPF | 0.0000 | 1 | 3.42 | 1.00 | 1.41 | 0.89 | 38 | 10 |
| rrOPF | 0.7130 | 4 | 4.25 | 1.24 | 1.41 | 0.89 | 45 | 1 |
| alOPF - RO 1 | 0.7133 | 6 | 9.49 | 2.77 | 1.41 | 0.89 | 45 | 1 |
| alOPF - CD 1 | 0.7133 | 6 | 12.41 | 3.63 | 1.41 | 0.89 | 45 | **0.8** |
| alOPF - TH 8 | 0.6935 | 2 | 17.98 | 5.26 | 1.41 | 0.89 | 43 | 1 |
| alOPF - OT 8 | 0.7130 | 4 | 16.29 | 4.76 | 1.41 | 0.89 | 45 | 1 |
| alOPF - DL 8 | 0.7128 | 3 | 24.84 | 7.26 | 1.41 | 0.89 | 45 | 10 |

Table 4.11: Method comparison on OPF Case C with $\gamma_s = 1$.

| method | $\Delta\varphi$ (%) | rank | $t$ $(s)$ | $t/t_0$ | $v^{\max}$ | $v^{\min}$ | actions | $\kappa$ |
|---|---|---|---|---|---|---|---|---|
| crOPF | 0.0000 | 5 | 15.83 | 1.00 | 1.20 | 0.84 | 1308 | 10 |
| rrOPF | −0.2149 | 4 | 27.15 | 1.72 | 1.20 | 0.85 | 1262 | 10 |
| alOPF - RO 1 | 0.0863 | 6 | 32.67 | 2.06 | 1.21 | 0.86 | 1288 | 10 |
| alOPF - CD 1 | 1.7402 | 7 | 37.71 | 2.38 | 1.21 | 0.87 | 1286 | 10 |
| alOPF - TH 8 | −0.5571 | 2 | 61.02 | 3.86 | 1.21 | 0.86 | 1265 | 1 |
| alOPF - OT 8 | −0.7563 | 1 | 73.93 | 4.67 | 1.21 | 0.86 | 1261 | 1 |
| alOPF - DL 8 | −0.5155 | 3 | 58.36 | 3.69 | 1.21 | 0.86 | 1266 | 1 |

Table 4.12: Method comparison on OPF Case C with $\gamma_s = 10$.

| method | $\Delta\varphi$ (%) | rank | $t$ $(s)$ | $t/t_0$ | $v^{\max}$ | $v^{\min}$ | actions | $\kappa$ |
|---|---|---|---|---|---|---|---|---|
| crOPF | 0.0000 | 3 | 14.22 | 1.00 | 1.21 | 0.85 | 823 | 10 |
| rrOPF | 0.4113 | 4 | 25.60 | 1.80 | 1.21 | 0.86 | 827 | 10 |
| alOPF - RO 1 | 26.4800 | 7 | 26.91 | 1.89 | 1.30 | 0.87 | 912 | 1 |
| alOPF - CD 1 | 11.0077 | 6 | 37.94 | 2.67 | 1.24 | 0.87 | 952 | 1 |
| alOPF - TH 8 | −0.1192 | 1 | 49.941 | 3.51 | 1.20 | 0.86 | 838 | 1 |
| alOPF - OT 8 | 0.5295 | 5 | 81.54 | 5.73 | 1.21 | 0.86 | 830 | 1 |
| alOPF - DL 8 | −0.0958 | 2 | 51.52 | 3.62 | 1.21 | 0.86 | 815 | 1 |

Table 4.13: Method comparison on OPF Case C with $\gamma_s = 100$.

| method | $\Delta\varphi$ (%) | rank | $t$ $(s)$ | $t/t_0$ | $v^{\max}$ | $v^{\min}$ | actions | $\kappa$ |
|---|---|---|---|---|---|---|---|---|
| crOPF | 0.0000 | 1 | 12.31 | 1.00 | 1.36 | 0.89 | 307 | 10 |
| rrOPF | 22.2815 | 7 | 33.91 | 2.75 | 1.37 | 0.88 | 476 | 10 |
| alOPF - RO 1 | 1.4732 | 6 | 20.19 | 1.64 | 1.35 | 0.88 | 359 | 1 |
| alOPF - CD 1 | 1.1743 | 4 | 34.76 | 2.82 | 1.35 | 0.89 | 358 | 1 |
| alOPF - TH 8 | 1.0062 | 2 | 74.84 | 6.08 | 1.35 | 0.88 | 351 | 1 |
| alOPF - OT 8 | 1.0178 | 3 | 101.43 | 8.24 | 1.35 | 0.88 | 352 | 1 |
| alOPF - DL 8 | 1.1808 | 5 | 72.79 | 5.91 | 1.35 | 0.89 | 353 | 1 |

Table 4.14: Method comparison on OPF Case D with $\gamma_s = 1$.

| method | $\Delta\varphi$ (%) | rank | $t$ $(s)$ | $t/t_0$ | $v^{\max}$ | $v^{\min}$ | actions | $\kappa$ |
|---|---|---|---|---|---|---|---|---|
| crOPF | 0.0000 | 4 | 10.43 | 1.00 | 1.39 | 0.84 | 540 | 10 |
| rrOPF | −0.0220 | 3 | 16.00 | 1.53 | 1.39 | 0.85 | 540 | 10 |
| alOPF - RO 1 | 4.6796 | 7 | 28.17 | 2.70 | 1.43 | 0.87 | 530 | 10 |
| alOPF - CD 1 | 0.8473 | 6 | 30.57 | 2.93 | 1.40 | 0.84 | 543 | 1 |
| alOPF - TH 8 | −0.1609 | 2 | 41.03 | 3.93 | 1.39 | 0.84 | 533 | 10 |
| alOPF - OT 8 | 0.1584 | 5 | 81.86 | 7.85 | 1.39 | 0.84 | 570 | 1 |
| alOPF - DL 8 | −0.1850 | 1 | 35.06 | 3.36 | 1.39 | 0.84 | 535 | 10 |

Table 4.15: Method comparison on OPF Case D with $\gamma_s = 10$.

| method | $\Delta\varphi$ (%) | rank | $t$ ($s$) | $t/t_0$ | $v^{\max}$ | $v^{\min}$ | actions | $\kappa$ |
|---|---|---|---|---|---|---|---|---|
| crOPF | 0.0000 | 5 | 10.82 | 1.00 | 1.39 | 0.84 | 329 | 10 |
| rrOPF | 0.2529 | 6 | 17.62 | 1.63 | 1.40 | 0.84 | 332 | 10 |
| alOPF - RO 1 | −0.4479 | 3 | 24.17 | 2.23 | 1.39 | 0.85 | 330 | **9** |
| alOPF - CD 1 | 0.5692 | 7 | 27.00 | 2.49 | 1.39 | 0.85 | 399 | 1 |
| alOPF - TH 8 | −0.4213 | 4 | 43.61 | 4.03 | 1.39 | 0.85 | 327 | 10 |
| alOPF - OT 8 | −0.4535 | 2 | 47.07 | 4.35 | 1.39 | 0.85 | 334 | 1 |
| alOPF - DL 8 | −0.4664 | 1 | 30.49 | 2.82 | 1.39 | 0.85 | 330 | 10 |

Table 4.16: Method comparison on OPF Case D with $\gamma_s = 100$.

| method | $\Delta\varphi$ (%) | rank | $t$ ($s$) | $t/t_0$ | $v^{\max}$ | $v^{\min}$ | actions | $\kappa$ |
|---|---|---|---|---|---|---|---|---|
| crOPF | 0.0000 | 1 | 7.95 | 1.00 | 1.41 | 0.82 | 149 | 1 |
| rrOPF | 0.3516 | 6 | 13.87 | 1.75 | 1.41 | 0.82 | 146 | 10 |
| alOPF - RO 1 | 0.3154 | 3 | 23.48 | 2.96 | 1.41 | 0.82 | 145 | **80** |
| alOPF - CD 1 | 0.3154 | 3 | 26.75 | 3.37 | 1.41 | 0.82 | 145 | **80** |
| alOPF - TH 8 | 0.3570 | 7 | 51.22 | 6.45 | 1.41 | 0.82 | 146 | 1 |
| alOPF - OT 8 | 0.2854 | 2 | 48.99 | 6.16 | 1.41 | 0.82 | 145 | 1 |
| alOPF - DL 8 | 0.3337 | 5 | 23.36 | 2.94 | 1.41 | 0.82 | 145 | **0.01** |

As the aggregate results show, the crOPF algorithm is the best one in terms of finding a solution with the lowest objective value. This is expected since this algorithm solves a continuous relaxation of the problem, whose (global) optimum is a lower bound for the original problem. Among the algorithms that obtain valid discrete points, the alOPF algorithm with a discrete line search strategy is the best, followed by the alOPF algorithms with thresholding-based strategies. An interesting result is that the algorithm with the simple thresholding strategy outperformed the optimized one, *i.e.*, the one that selects the best thresholds according to the value of the augmented Lagrangian. This may be attributed to the trial discrete points constructed from the thresholds selected by the optimized strategy having less diversity, which reduces the ability of the algorithm to explore different regions of the discrete variable space. The benchmark rrOPF algorithm and the alOPF algorithm with the simple rounding strategy are tied and occupy the penultimate ranking position in terms of solution quality. This result suggests that perhaps no benefits are obtained by rounding the discrete variables during the early stages of the algorithm. Lastly, the worst-performing algorithm was the alOPF algorithm with the greedy coordinate descent

strategy. This result reinforces the idea that it is important to maintain exploration when selecting trial discrete points, and not focus only on points that appear locally promising.

In terms of time, the aggregate results show that the alOPF algorithms that utilize several parallel processes are the slowest. This is attributed partly to communication overhead and the fact that these algorithms typically take more iterations due to followings paths with more gradual approach to feasibility. In particular, the algorithm with discrete line search strategy is about 4.5 times slower than the crOPF method, while the algorithm with simple thresholding strategy is about 5.3 times slower. The alOPF algorithm with optimized thresholding strategy is slower than these two since it requires extra computations for finding the subset of most promising thresholds.

From the aggregate results, it may be concluded that the alOPF algorithm with discrete line search strategy is the best algorithm that enforces the discreteness constraints. This is because its solution quality ranking is behind only that of the crOPF algorithm and it is faster than the alOPF algorithm with the simple thresholding strategy. However, the individual case results show that this method failed to solve one case with the regular values of $\kappa$. This suggests that it may be slightly less robust than the similarly ranked but slower alOPF algorithm with simple thresholding strategy. Algorithms that also had convergence issues are the alOPF algorithms with simple rounding and coordinate descent strategies. These algorithms failed to solve two cases with the regular values of $\kappa$ considered. It was observed that these convergence issues were caused by the generation of poor search directions during the evaluation of the trial discrete points. This is likely attributed to exploring an ill-conditioned region of the problem after using a poor initial (trial) point.

A final observation is that for several cases, the crOPF method, which solves a relaxed problem, did not achieve the lowest objective value. This happened mostly for cases having small sparsity weights. A possible explanation for this may be that such problems have excessive flexibility. This makes them ill-conditioned and results in algorithms making large initial discrete variable adjustments that later complicate their progress towards the solution. The algorithms that enforce discreteness may suffer less from this because they reduce this flexibility by temporarily fixing the discrete variables.

## 4.7 Conclusions

In this chapter, two important challenges of OPF problems have been addressed, namely, that of handling discrete variables and of obtaining a manageable number of control adjustments. For addressing these challenges, several modeling and optimization techniques were explored. In particular, the use of a smooth sparsity-inducing penalty function was considered for obtaining a small number of control adjustments without complicating the problem. In fact, the penalty function considered adds convexity and "regularization" to the problem, which can be exploited by the solution algorithm to improve its stability and convergence properties, as done in the power flow method of Chapter 3. The algorithm considered for handling discrete variables is based on alternating between a guide algorithm that attempts to solve a continuous relaxation of the problem, and one or more evaluator algorithms that explore different choices of discrete variables in parallel. Both guide and evaluator algorithms are based on the augmented Lagrangian method described in Chapter 3 for solving power flow problems. Results of experiments were presented that showed the effectiveness of the smooth sparsity-inducing penalty in reducing the number of control adjustments as well as the trade-off that exists between this number and solution quality. Also, results were presented that compared the performance of the alternating algorithm with that of one that solves a continuous relaxation of the problem and a benchmark method that represents current practices. Several strategies for constructing trial discrete points were considered, and it was found that ones based on discrete line search and simple thresholding provided the best results in terms of objective value, outperforming the benchmark method while being only slightly slower. The results suggested that an important factor for obtaining better solutions with valid discrete values is exploration, and that greedy approaches do not perform well.

# Chapter 5

# Online Security Assessment

## 5.1 Background and Overview

Power systems have numerous types of security constraints that system operators need to monitor constantly, such as voltage stability constraints, branch thermal overload limits, bus voltage limits, and dynamic stability constraints [55] [57]. Traditionally, security assessment practices have typically consisted of first performing offline studies hours or even days in advance due to their high computational cost. Critical quantities, *e.g.*, load levels and power transfer levels, are determined and one or two-dimensional nomograms describing the security boundaries are constructed. During real-time operations, the pre-selected critical quantities are monitored and nomograms are consulted in order to assess whether the system is secure [57]. In the past, these practices were acceptable since systems had predictable power flow directions and load patterns [57], and were operated in a very conservative manner. Now, with industry restructuring, systems are less predictable and tend to operate closer to their limits [90], and this is expected to become more common with the planned increase of variable and distributed generation. Consequently, security assessments need to be performed much more frequently, more quantities need to be monitored, and information that is more concise needs to be presented to operators [57].

Several authors have explored new ways for performing security assessments more efficiently and effectively. In particular, efforts have been made for determining ways for representing security boundaries and for determining their proximity to the current operating point. In [55], the authors consider various security constraints and describe an approach for approximating boundaries using hyperplanes in the space of several pre-selected critical

quantities. In [41], the authors follow the same approach and focus exclusively on thermal security boundaries. In particular, they represent the boundaries using hyperplanes in the space of "cut-set angles", which are a particular choice of critical quantities that are related to voltage angle differences across branches of transmission corridors. The construction of the hyperplanes is done offline by stressing the critical quantities in different directions and performing system simulations until boundary points are obtained. These points are then used for setting up linear systems of equations that define the hyperplanes. An alternative approach for representing security boundaries and performing security assessment is described in [57] and [90]. It consists of using an Artificial Neural Network (ANN) for mapping values of pre-selected critical quantities to system security measures, and hence for representing the boundary of a general set of security constraints. The ANN, once trained, can be used online for constructing nomograms and assessing system security easily. However, the training procedure requires identifying a set of critical quantities of manageable size and performing offline system simulations. Another approach for determining bounds on thermal security margins is described in [19]. The approach uses a linear system model based on sensitivities and expresses security margins in terms of changes in generator and load powers. The authors test the approach on a power network of 80 buses, but do not address the considerable computational challenges associated with realistic networks that are much larger. These computational challenges are considered in our previous work [31], where an approach based on a linear system model for determining and visualizing critical thermal boundaries in the space of generator and load powers is also explored. The explored techniques are shown to be efficient for large cases of up to 45000 buses, but are based on approximate thermal limits that only consider branch active power flows.

In this chapter, efficient techniques are described for determining and visualizing security boundaries of power systems for online security assessment. The main goals of these techniques are to avoid repetitive and time-consuming computer simulations, and to discover and consider critical quantities that operators may not be aware of when assessing security. A simple linear model of a power network is used and only a pre-contingency case is considered with thermal overload and bus voltage magnitude limits as security constraints. This simple model allows establishing an analytic framework and developing techniques for filtering the most critical constraints and approximating the geometry of their boundaries quickly. The results obtained with these approximate, but fast techniques, may then be leveraged by more accurate and computationally expensive methods, such as ones based on

a nonlinear system model.

## 5.2 The Critical Operating Boundaries Problem

The Critical Operating Boundaries (COB) problem considered here consists of three parts:

1. Determine the "nearest" reachable voltage and thermal overload boundaries from the current operating point, where distance is measured in terms of changes in generator and load powers.

2. Determine effective and practical adjustments in generation and load powers that move the system away from the nearest reachable boundaries.

3. Determine how to visualize this information on a two-dimensional plane.

Mathematically, this problem may be formulated as follows: Let the power flow equations (2.37) be expressed as

$$Yy - f(x) = 0, \tag{5.1}$$

where $y$ is the vector of generator and load powers, $Y$ is a "power injection" matrix that associates injections (or consumptions) with buses, $x$ is the vector of bus voltage magnitudes and angles, and $f$ gives the power leaving each bus through shunt devices and branches for a given $x$. Also, let the function $h$ be such that the constraints $h_i(x) \geq 0$, $i \in [n_c]$, where $n_c \in \mathbb{N}$ and $[n_c] := \{1, \ldots, n_c\}$, represent all bus voltage magnitude limits (2.1) and branch thermal overload limits (2.17). Lastly, let $y_0$ be the current vector of generator and load powers, and $x_0$ the system state associated with $y_0$. Also assume that $h_i(x_0) > 0$ for each $i \in [n_c]$. Then, the first part of the problem consists of finding a small constraint index set $\bar{\mathcal{N}} \subset [n_c]$ of size $\bar{n}_c \ll n_c$ such that $\delta_i \leq \delta_j$ for all $i \in \bar{\mathcal{N}}$ and $j \in [n_c] \setminus \bar{\mathcal{N}}$, where $\delta_i$ is the distance from the current point to the boundary of $h_i(x) \geq 0$ in the space of generator and load powers. That is, $\delta_i$ is the optimal value of

$$\begin{aligned} \underset{\Delta y, x}{\text{minimize}} \quad & \|\Delta y\|_2 \\ \text{subject to} \quad & Y(y_0 + \Delta y) - f(x) = 0 \\ & h_i(x) = 0 \\ & y^{\min} \leq y_0 + \Delta y \leq y^{\max}, \end{aligned} \tag{5.2}$$

where $y^{\min}$ and $y^{\max}$ are bounds on generator and load powers. By convention, when problem (5.2) is infeasible, $\delta_i = +\infty$. The second part of the problem consists of finding directions $\Delta y$ for moving $y_0$ in such a way that a measure that quantifies how close the system is to the boundaries of the constraints in $\bar{\mathcal{N}}$ is improved. These directions should be practical in the sense that they should only involve a few actions, *i.e.*, be sparse. Lastly, the third part of the problem consists of visualizing the boundary $\{ x \mid h_i(x) = 0 \}$ of each constraint $i \in \bar{\mathcal{N}}$.

## 5.3 Modeling Approach

A linear approximation of both the power flow equations and the security constraints is used to simplify the problem and be able to perform fast analyses. The linearization of the power flow equations is obtained from a first-order Taylor expansion of the function $f$ about the current operating point $x_0$. This gives the linear system

$$Ax = b + Yy, \tag{5.3}$$

where

$$A := J(x_0) \in \mathbb{R}^{2n \times 2n} \quad \text{and} \quad b = J(x_0)x_0 - f(x_0) \in \mathbb{R}^{2n}, \tag{5.4}$$

and $J$ is the Jacobian of $f$, which is assumed to be non-singular. The vector $y \in \mathbb{R}^{n_y}$, where $n_y \in \mathbb{N}$, is composed of active powers of generators and loads. Reactive power consumptions of loads and reactive power injections of non-regulating generators are obtained from their active powers by assuming constant power factors. Generator reactive power limits are ignored, and hence regulated voltage magnitudes are kept fixed at their set points. To achieve this, for each $k \in \mathcal{R}$, where $\mathcal{R}$ is the set of regulated buses, the row of $A$ associated with the reactive power mismatch at bus $k$ is set to zero everywhere except for the entry corresponding to the bus voltage magnitude $v_k$, which is set to one. The corresponding entry of $b$ is set to the bus voltage magnitude set point $v_k^t$, and the corresponding row of $Y$ is set to zero everywhere. A similar technique is used for keeping the slack bus voltage magnitude and angle fixed at their given values.

The linearization of the security constraints, which includes bus voltage limits (2.1) and branch thermal overload limits (2.17), is obtained from first-order expansions of smooth approximations of the branch current magnitudes $|i_{km}|$ and $|i_{mk}|$, for each $(k, m) \in \mathcal{B}$,

where $\mathcal{B}$ is the set of branches. In particular, the smooth approximation of $|i_{km}|$ used is

$$\left(\Re\{i_{km}\}^2 + \Im\{i_{km}\}^2 + \rho\right)^{1/2}, \tag{5.5}$$

where $\rho$ is some small positive scalar, *e.g.*, $10^{-8}$, and $\Re\{\cdot\}$ and $\Im\{\cdot\}$ give the real and imaginary parts of a complex number, respectively. An analogous approximation is used for $|i_{mk}|$. The resulting linear inequality security constraints are denoted by

$$Cx \geq d, \tag{5.6}$$

where $C \in \mathbb{R}^{n_c \times 2n}$, $d \in \mathbb{R}^{n_c}$, and $n_c$ is the number of security constraints, *i.e.*, $n_c = 2n+2|\mathcal{B}|$. Hence, the linear system model considered here is given by

$$Ax = b + Yy, \quad Cx \geq d. \tag{5.7}$$

Since $A$ is non-singular, the state vector $x$ can be eliminated from the above system to obtain a set of linear inequalities in terms of $y$ only. These inequalities are given by

$$Qy \geq t, \tag{5.8}$$

where the matrix $Q \in \mathbb{R}^{n_c \times n_y}$ and vector $t \in \mathbb{R}^{n_c}$ are given by

$$Q := CA^{-1}Y \quad \text{and} \quad t := d - CA^{-1}b. \tag{5.9}$$

One difficulty of dealing with these constraints given in terms of $y$ is that, unlike $A$, $Y$ and $C$, the matrix $Q$ is dense, and hence constructing this matrix is impractical for mid to large-scale networks. The techniques described below for analyzing these constraints are designed to utilize the matrix $Q$ only via matrix-vector products, and hence the elements of this matrix need not be known explicitly.

The particular security metric used to determine how far a point $y$ is from the security boundaries is given by the concave function

$$M(y) = \sum_{i \in \mathcal{N}} \log(q_i^T y - t_i), \tag{5.10}$$

where $\bar{\mathcal{N}}$ is the set of indices corresponding to the nearest reachable constraints, and $q_i$ is the $i$-th row of $Q$. The larger the value of $M(y)$ is, the more "interior" and hence more secure the point $y$ is. This function will be used later to identify recommended operator actions for moving the system away from critical operating boundaries.

## 5.4 Solution Algorithm

The algorithms described next for solving the Critical Operating Boundaries or COB problem use the linear system model described in the previous section that involves $y$ only, *i.e.*, $Qy \geq t$. Furthermore, they assume that the initial vector $y_0$ of generator and load powers is strictly feasible, *i.e.*, $Qy_0 > t$, and that $Q$ does not have zero rows. These assumptions are made only for keeping the exposition simple. In the implementation, rows of $Q$ that have negligible norm are easily detected to avoid dividing by a near-zero number, and infeasible points are handled by adding a few safeguards to the techniques described in the sequel.

### 5.4.1 Nearest Reachable Boundaries

The algorithm for finding the nearest boundaries consists of first finding the nearest boundaries while ignoring bounds on the power vector $y$, and then filtering the ones that can actually be reached by considering such bounds.

Using the linear system model in terms of $y$ only, *i.e.*, $Qy \geq t$, and ignoring the bounds on $y$, the distance $\delta_i$ to the boundary of the $i$-th security constraint $q_i^T y \geq t_i$ from the current point $y_0$ is given by

$$\begin{array}{ll} \underset{\Delta y}{\text{minimize}} & \|\Delta y\|_2 \\ \text{subject to} & q_i^T(y_0 + \Delta y) = t_i, \end{array} \tag{5.11}$$

where $q_i$ is the $i$-th row of $Q$. The next lemma characterizes this distance.

**Lemma 5.4.1.** *The solution of problem* (5.11) *is given by* $-\alpha_i \hat{q}_i$*, where*

$$\alpha_i := \frac{q_i^T y_0 - t_i}{\|q_i\|_2} \quad \textit{and} \quad \hat{q}_i := \frac{q_i}{\|q_i\|_2}, \tag{5.12}$$

*and the associated objective value,* i.e., $\delta_i$*, is equal to* $\alpha_i$*.*

*Proof.* Let $\Delta y^* := -\alpha_i \hat{q}_i$. Then

$$q_i^T(y_0 + \Delta y^*) = q_i^T y_0 - \alpha_i q_i^T \hat{q}_i \tag{5.13}$$

$$= q_i^T y_0 - \frac{q_i^T y_0 - t_i}{\|q_i\|_2}\|q_i\|_2 \tag{5.14}$$

$$= t_i, \tag{5.15}$$

so $\Delta y^*$ is a feasible point of (5.11). Since every feasible point may be expressed as

$$\Delta y = \Delta y^* + Z\eta, \tag{5.16}$$

for some $\eta$, where $Z$ is a matrix whose columns span the subspace of all vectors orthogonal to $q_i$, it follows that

$$\|\Delta y\|_2^2 = \|\Delta y^* + Z\eta\|_2^2 \tag{5.17}$$

$$= \|\Delta y^*\|_2^2 + \|Z\eta\|_2^2 + 2(\Delta y^*)^T Z\eta \tag{5.18}$$

$$= \|\Delta y^*\|_2^2 + \|Z\eta\|_2^2 - 2\frac{\alpha_i}{\|q_i\|_2} q_i^T Z\eta. \tag{5.19}$$

Since $q_i^T Z = 0$, it follows that

$$\|\Delta y\|_2^2 = \|\Delta y^*\|_2^2 + \|Z\eta\|_2^2 \geq \|\Delta y^*\|_2^2, \tag{5.20}$$

so $\Delta y^*$ is the feasible point with smallest norm, and such norm is clearly $\alpha_i$. □

Lemma 5.4.1 just states that the distance in the $y$ space from the current point $y_0$ to the boundary of the security constraint $q_i^T y \geq t_i$ is $\alpha_i$, and that the associated change in $y$ to reach the nearest boundary point is given by $-\alpha_i \hat{q}_i$.

The difficulty in determining the distance to the boundary of each security constraint, and hence for determining the nearest ones, is that $\|q_i\|_2$ and hence $q_i$ is needed for each $i \in [n_c]$. From (5.9), $q_i^T = c_i^T A^{-1} Y$, where $c_i$ is the $i$-th row of $C$, and hence constructing each $q_i$ requires solving the linear system

$$A^T \xi_i = c_i. \tag{5.21}$$

The difficulty is that $\xi_i$ and the resulting $q_i = Y^T \xi_i$ are not sparse in general. Therefore,

doing this for all $n_c$ security constraints is not practical, since $n_c$ is large ($n_c = 2n + 2|\mathcal{B}|$). For example, for a 2.5k-bus system, $n_c$ is typically greater than 10k. For this reason, the approach considered here for determining boundary distances is based on estimating each $\|q_i\|_2$ using matrix-vector products only, and hence without having to construct each $q_i$. The idea of estimating the norms of the rows of a matrix via matrix-vector products has been explored in the context of matrix equilibration [14]. It is based on the relation given by the following lemma.

**Lemma 5.4.2.** *Given any constant vector $q$ and random vector $z$ composed of independent and identically distributed random variables of zero mean and unit variance, it holds that*

$$\mathrm{E}\big((q^T z)^2\big) = \|q\|_2^2, \tag{5.22}$$

*where $\mathrm{E}(\cdot)$ gives the expected value of a random variable.*

*Proof.* This relation follows easily from the linearity of expectation:

$$\begin{aligned} \mathrm{E}((q^T z)^2) &= \mathrm{E}(q^T z q^T z) && (5.23)\\ &= \mathrm{E}(q^T z z^T q) && (5.24)\\ &= q^T \mathrm{E}(z z^T) q && (5.25)\\ &= q^T q. && (5.26) \end{aligned}$$

□

This relation just states that $\|q\|_2^2$ is the variance of the random variable $q^T z$. Hence, $\|q_i\|_2^2$ may be estimated by drawing random samples $z_1, z_2, \ldots, z_k$ of $z$, and computing the sample variance of $q_i^T z$, which is given by

$$\sigma_{ki}^2 := \frac{1}{k} \sum_{j=1}^{k} (q_i^T z_j)^2. \tag{5.27}$$

In fact, this may be done simultaneously for all $q_i$ using

$$\sigma_k^2 := \frac{1}{k} \sum_{j=1}^{k} w_j \circ w_j, \tag{5.28}$$

where $\circ$ denotes element-wise product, $w_j := C\zeta_j$, and $\zeta_j$ solves $A\zeta_j = Yz_j$. From the results of Lemma 5.4.1, boundary distance estimates are then obtained by

$$\tilde{\alpha}_{ki} := \frac{q_i^T y_0 - t_i}{\sigma_{ki}},\ i \in [n_c]. \tag{5.29}$$

The number of samples $k$ of the random vector $z$ must be large enough such that the distance estimates $\tilde{\alpha}_k := \{\tilde{\alpha}_{ki}\}_{i\in[n_c]}$ are accurate enough to allow identifying the nearest boundaries, and small enough such that the procedure is much faster than the naive approach of constructing each constraint normal $q_i$, $i \in [n_c]$. The strategy considered here for determining this number exploits the fact that, when $z$ is a vector of independent and identically distributed Gaussian random variables, *i.e.*, $z \sim \mathcal{N}(0, I)$, where $I$ is the identity matrix, $q_i^T z$ is Gaussian with zero mean and variance $\|q_i\|_2^2$, *i.e.*, $q_i^T z \sim \mathcal{N}(0, \|q_i\|_2^2)$, for each $i \in [n_c]$. From this, it follows that $k\frac{\sigma_{ki}^2}{\|q_i\|_2^2}$ has chi-squared distribution with $k$ degrees of freedom, *i.e.*,

$$k\frac{\sigma_{ki}^2}{\|q_i\|_2^2} \sim \chi_k^2. \tag{5.30}$$

Hence, for each $i \in [n_c]$, positive scalars $L_{ki}^1$ and $L_{ki}^2$ are known such that

$$L_{ki}^1 \leq k\frac{\sigma_{ki}^2}{\|q_i\|_2^2} \leq L_{ki}^2 \tag{5.31}$$

with high probability, *e.g.*, 0.9. Equivalently,

$$\sigma_{ki}\sqrt{\frac{k}{L_{ki}^2}} \leq \|q_i\|_2 \leq \sigma_{ki}\sqrt{\frac{k}{L_{ki}^1}} \tag{5.32}$$

with high probability. From these confidence bounds on $\|q_i\|_2$, confidence bounds on the actual boundary distance $\alpha_i$ are obtained, and these are given by

$$\alpha_{ki}^{\max} := \frac{q_i^T y_0 - t_i}{\sigma_{ki}}\sqrt{\frac{L_{ki}^2}{k}} \tag{5.33}$$

$$\alpha_{ki}^{\min} := \frac{q_i^T y_0 - t_i}{\sigma_{ki}}\sqrt{\frac{L_{ki}^1}{k}}, \tag{5.34}$$

for each $i \in [n_c]$. These confidence bounds are used within the sampling procedure as follows: After drawing a few samples, say $k = 5$, and updating $\sigma_k^2$ using (5.28), a set

$\tilde{\mathcal{N}} \subset [n_c]$ of size $\tilde{n}_c \in \mathbb{N}$, where $\tilde{n}_c \ll n_c$, and such that $\tilde{\alpha}_{ki} \leq \tilde{\alpha}_{kj}$ for all $i \in \tilde{\mathcal{N}}$ and $j \in [n_c] \setminus \tilde{\mathcal{N}}$, is obtained. This is done by finding the indices of the $\tilde{n}_c$ smallest $\tilde{\alpha}_{ki}$, $i \in [n_c]$, using the *introselect* algorithm [65]. As an example, $\tilde{n}_c$ may be 100 for a network with $n_c$ around 10k. Then, the threshold distance

$$\alpha_k^{th} := \max\{\ \alpha_{ki}^{\max} \mid i \in \tilde{\mathcal{N}}\ \} \tag{5.35}$$

is computed, and constraints $i \in [n_c]$ such that $\alpha_{ki}^{\min} > \alpha_k^{th}$ are identified and subsequently ignored in the sampling process. The process is repeated until the distance estimates to the boundaries of the "relevant constraints", *i.e.*, the ones for which $\alpha_{ki}^{\min} \leq \alpha_k^{th}$, converge. This is tested by the condition

$$\left(\sum_{i\in\mathcal{H}}(\tilde{\alpha}_{ki} - \tilde{\alpha}_{(k-1)i})^2\right)^{1/2} \leq \epsilon\left(\sum_{i\in\mathcal{H}}(\tilde{\alpha}_{ki})^2\right)^{1/2}, \tag{5.36}$$

where $\epsilon$ is some small positive scalar, *e.g.*, $10^{-2}$, and $\mathcal{H} \subset [n_c]$ is the set of indices of the relevant constraints.

Bounds on the power vector $y$ are considered for the constraints associated with the index set $\tilde{\mathcal{N}}$, and an index set $\bar{\mathcal{N}} \subset \tilde{\mathcal{N}}$ of constraints that can be violated is extracted. More specifically, for each security constraint $i \in \tilde{\mathcal{N}}$, the quantity $\nu_i$ is computed, where

$$\nu_i := \inf\{\ q_i^T y - t_i \ \mid\ y^{\min} \leq y \leq y^{\max}\ \}. \tag{5.37}$$

The set $\bar{\mathcal{N}}$ is then given by

$$\bar{\mathcal{N}} = \{\ i \in \tilde{\mathcal{N}} \ \mid\ \nu_i \leq 0\ \}, \tag{5.38}$$

and its size is denoted by $\bar{n}_c$. To obtain $\nu_i$, $q_i$ is needed. Hence, $q_i$ needs to be constructed for each $i \in \tilde{\mathcal{N}}$, but this is practical since $\tilde{n}_c$ is small. Moreover, only one $q_i$ needs to be stored at a given time and discarded after $\nu_i$ is determined. The value of $\nu_i$ can be easily obtained analytically, as the next lemma shows.

**Lemma 5.4.3.** *The solution $\nu_i$ of problem* (5.37) *is given by*

$$\nu_i = q_i^T y_i^* - t_i, \tag{5.39}$$

*where* $y_i^* \in \mathbb{R}^{n_y}$ *is the vector defined by*

$$y_{ij}^* = \begin{cases} y_j^{\max}, & \text{if } q_{ij} \leq 0 \\ y_j^{\min}, & \text{otherwise,} \end{cases} \tag{5.40}$$

*for each* $j \in [n_y]$.

*Proof.* This result follows immediately from the fact that $y_{ij}^*$ satisfies

$$y_{ij}^* = \operatorname{argmin}\{\ q_{ij} z \mid z \in [y_j^{\min}, y_j^{\max}]\ \}, \tag{5.41}$$

for each $j \in [n_y]$. □

Algorithm 5 shows the pseudocode for finding the small index set $\bar{\mathcal{N}}$ associated with the nearest reachable security boundaries.

**Algorithm 5** Nearest Reachable Boundaries

Given $\tilde{n}_c \ll n_c$, $p \in \mathbb{N}$, $\epsilon > 0$
$k \leftarrow 1$
**while** True **do**
    Sample random vector $z_k \sim \mathcal{N}(0, I)$
    Update sample variance vector $\sigma_k^2$ using (5.28)
    **if** $k$ mod $p = 0$ **then**
        Compute $\tilde{\alpha}_k$ and bounds $\alpha_k^{\max}$ and $\alpha_k^{\min}$ using (5.29), (5.33) and (5.34)
        Construct set $\tilde{\mathcal{N}}$ of $\tilde{n}_c$ nearest constraints according to $\tilde{\alpha}_k$
        Compute threshold distance $\alpha_k^{th}$ using (5.35)
        Construct set $\mathcal{H}$ of relevant constraints using $\alpha_k^{th}$
        **if** Distance estimates of relevant constraints satisfy (5.36) **then**
            *break*
        **end if**
    **end if**
    $k \leftarrow k + 1$
**end while**
Compute $\nu_i$ for each $i \in \tilde{\mathcal{N}}$ using (5.39)
Construct set $\bar{\mathcal{N}}$ by removing indices from $\tilde{\mathcal{N}}$ such that $\nu_i > 0$.

### 5.4.2 Distributed Approach for Improving Linear Model

The constraints corresponding to the index set $\bar{\mathcal{N}}$, which are denoted by $\bar{Q}y \geq \bar{t}$, are a small subset of the nearest reachable constraints according to the linear model. However,

the distances and orientations of the boundaries of these constraints are only approximations of the real ones. It may be that errors inherent in the linear model result in many constraints begin incorrectly left out of the set $\bar{\mathcal{N}}$ of nearest reachable ones. To improve the accuracy of the distance and orientation of the boundaries, and to try to avoid incorrectly leaving out critical operating boundaries, a distributed approach is considered.

The main idea of the approach consists of determining $m \in \mathbb{N}$ key locations where to re-linearize the system, repeat the distance estimation and filtering algorithms of Section 5.4.1 for each new system, and then update the original linear model with the information obtained.

More specifically, given a number $m$, then $m$ locations are determined in the space of generator and load powers in such a way that they allow exploring the subspace containing the nearest boundaries of the constraints obtained so far, *i.e.*, $\bar{Q}y \geq \bar{t}$. This is done by considering the matrix

$$M_s := \bar{Q}^T D^{-1}, \tag{5.42}$$

where $D$ is a diagonal matrix with diagonals $-\tilde{\alpha}_i \sigma_i$, $i \in \bar{\mathcal{N}}$, $\tilde{\alpha}$ is the vector of boundary distance estimates, as defined in (5.29), and $\sigma$ is the vector of norm estimates of constraint normals, as defined in (5.28). This matrix is referred to as the *security matrix* and will also play an important role in the visualization techniques considered below. The $m$ left singular vectors $\{u_i\}_{i\in[m]}$ associated with the $m$ largest singular values of $M_s$ are obtained. This is done by first finding the (orthonormal) eigenvectors $\{v_i\}_{i\in[m]}$ associated with the $m$ largest eigenvalues $\{\lambda_i\}_{i\in[m]}$ of $M_s^T M_s$ using an implicitly restarted Lanczos method [45], and then setting

$$u_i = \frac{M_s v_i}{\sqrt{\lambda_i}}, \tag{5.43}$$

for each $i \in [m]$. This procedure only uses matrix-vector products with $M_s$ (and $M_s^T$), and hence $\bar{Q}$ need not be stored. For each direction $u_i$, the scalar factor $\beta_i$ for reaching the nearest boundary of the constraints $\bar{Q}y \geq \bar{t}$ along $u_i$ is obtained by first computing the vector

$$\gamma := (\bar{t} - \bar{Q}y_0) \circ \bar{\eta}_i, \tag{5.44}$$

where $\eta_i$ is the element-wise reciprocal of $\bar{Q}u_i$, and then setting $\beta_i = \gamma_{j^*}$, where $j^*$ is given by

$$j^* = \text{argmin}\{\ |\gamma_s| \ \mid \ s \in [\bar{n}_c]\ \}. \tag{5.45}$$

The key locations used for re-linearizing the system are then given by

$$y^i = y_0 + \tau\beta_i u_i,\ i \in [m], \tag{5.46}$$

where $\tau \in (0,1)$, *e.g.*, 0.7, is user-defined.

For each $i \in [m]$, the exact system state $x^i$ associated with the power vector $y^i$ is obtained by solving the nonlinear power flow equations (5.1) using the Newton-Raphson algorithm described in Section 3.3. It may be that this power flow algorithm fails due to the initial point not being close enough to the solution, or due to the power flow equations having no solution for the power vector $y^i$. In this case, the power flow algorithm is applied repeatedly along points in the line segment joining $y_0$ and $y^i$, as is done by continuation methods [2] [17], and the point nearest to $y^i$ for which the algorithm succeeds is used. At this point, the linear model described in Section 5.3 is constructed, giving

$$Q^i y \geq t^i, \tag{5.47}$$

where

$$Q^i := C^i(A^i)^{-1}Y \quad \text{and} \quad t^i := d^i - C^i(A^i)^{-1}b^i. \tag{5.48}$$

The superscript $i$ is used here to differentiate this model from the ones obtained at other points, *i.e.*, at $y_0$ and $y^j$, $j \in [m] \setminus \{i\}$.

For each $i \in [m]$, the distance estimation and filtering algorithms of Section 5.4.1 are used to obtain a small subset of nearest reachable constraints given by $\bar{Q}^i y \geq \bar{t}^i$. Subsequently, from the constraints $\bar{Q}^i y \geq \bar{t}^i$, only the ones whose constraint normal $q^i$ makes an angle $\theta$ less than $\vartheta$, say $\pi/3$, with $-\beta_i u_i$ are selected and used for updating the original linear model. The reason for doing this is that such constraints are likely to be more accurate for the linear system obtained at $y^i$ than for the original one obtained at $y_0$. Figure 5.1 shows the boundary $q^T y = t$ of a constraint based on the original model obtained at $y_0$, the boundary $(q^i)^T y = t^i$ of the same constraint from the model obtained at $y^i$, and the angle $\theta$ the new constraint normal $q^i$ makes with $-\beta_i u_i$.

The information obtained from the linear models constructed at the different points $y^i$, $i \in [m]$, is used to update the original linear model as follows: Suppose that for the original

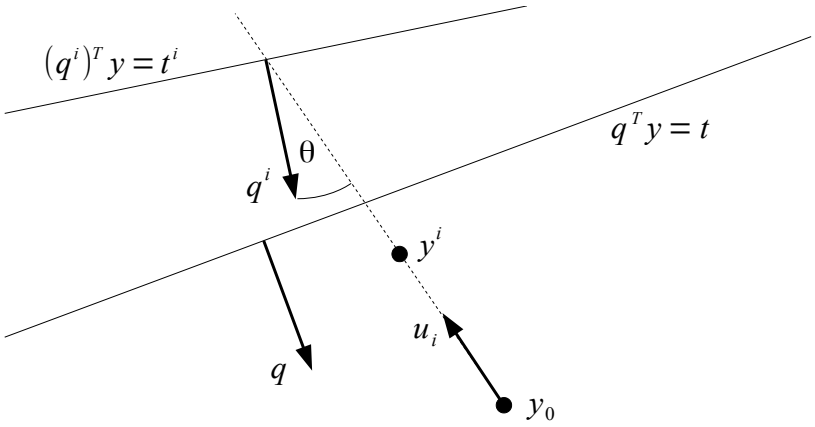


Figure 5.1: Angle between new constraint normal $q^i$ and $-\beta_i u_i$.

constraint $q^T y \geq t$, the variations

$$(q^i)^T y \geq t^i,\ i \in \mathcal{I}, \tag{5.49}$$

where $\mathcal{I} \subset [m]$, are obtained from the linear models constructed at points $y^i$, $i \in \mathcal{I}$. Then, the new constraint is taken as $(q')^T y \geq t'$, where

$$q' := \zeta_0 q + \frac{\zeta_1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} q^i \tag{5.50}$$

$$t' := \zeta_0 t + \frac{\zeta_1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} t^i, \tag{5.51}$$

and $\zeta_0$ and $\zeta_1 \in (0, 1)$ such that $\zeta_0 + \zeta_1 = 1$, *e.g.*, $\zeta_0 = 0.25$ and $\zeta_1 = 0.75$. The following lemma states an important property of $q'$ and $t'$.

**Lemma 5.4.4.** *The new constraint normal $q'$ and right hand side $t'$ given by* (5.50) *and* (5.51)*, respectively, satisfy*

$$q' = Y^T A^{-T} \bar{c} \tag{5.52}$$

$$t' = \bar{\delta} - \bar{c}^T A^{-1} b, \tag{5.53}$$

*where*

$$\bar{c} := \zeta_0 c + \frac{\zeta_1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} A^T (A^i)^{-T} c^i \tag{5.54}$$

$$\bar{\delta} := \zeta_0 t + \frac{\zeta_1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} t^i + \bar{c}^T A^{-1} b, \tag{5.55}$$

*and $c$ and $c^i$ are the normals of the state security constraints obtained at $y_0$ and $y^i$, respectively, corresponding to the constraint $q^T y \geq t$.*

*Proof.* For each $i \in [m]$, using $I = A^{-T} A^T$ and the definition of $Q^i$, it follows that

$$q^i = Y^T (A^i)^{-T} c^i \tag{5.56}$$

$$= Y^T A^{-T} A^T (A^i)^{-T} c^i. \tag{5.57}$$

Hence, from the definition of $q'$,

$$q' = \zeta_0 q + \frac{\zeta_1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} q^i \tag{5.58}$$

$$= \zeta_0 Y^T A^{-T} c + \frac{\zeta_1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} Y^T A^{-T} A^T (A^i)^{-T} c^i \tag{5.59}$$

$$= Y^T A^{-T} \left( \zeta_0 c + \frac{\zeta_1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} A^T (A^i)^{-T} c^i \right) \tag{5.60}$$

$$= Y^T A^{-T} \bar{c}. \tag{5.61}$$

Moreover,

$$t' = \zeta_0 t + \frac{\zeta_1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} t^i \tag{5.62}$$

$$= \zeta_0 t + \frac{\zeta_1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} t^i + \bar{c}^T A^{-1} b - \bar{c}^T A^{-1} b \tag{5.63}$$

$$= \bar{\delta} - \bar{c}^T A^{-1} b. \tag{5.64}$$

□

The results from Lemma 5.4.4 show that the original constraint $q^T y \geq t$ may be updated with information from the new linear models by updating only the state security constraints

$Cx \geq d$, and not the power flow equations $Ax = b + Yy$. By doing this, the state security constraints $Cx \geq d$ are adjusted to compensate for errors not only in $C$ and $d$, but also in $A$ and $b$, and adjustments to one constraint do not affect the accuracy of any other one. Figure 5.2 shows the updated constraint for the simple example shown in Figure 5.1.

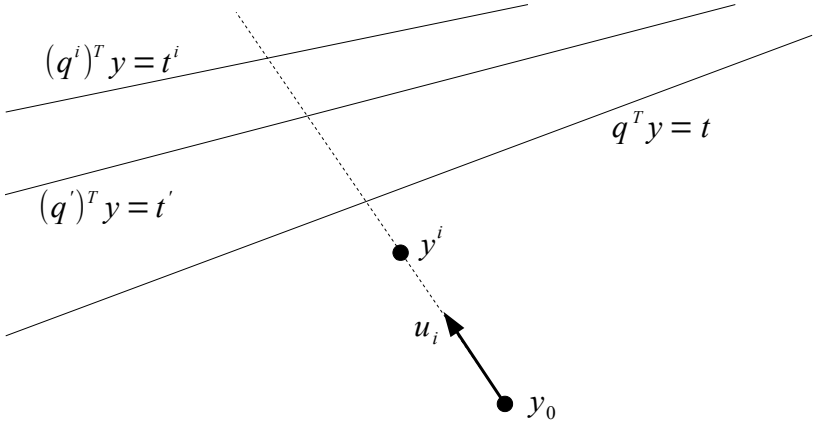


Figure 5.2: Resulting constraint $(q')^T y \geq t'$ from averaging $q^T y \geq t$ and $(q^i)^T y \geq t^i$.

One important property to consider is whether the vectors

$$\nu^i := A^T (A^i)^{-T} c^i, \tag{5.65}$$

for $i \in \mathcal{I}$, are sparse. In practice, one may consider only entries of $\nu_i$ that are large enough, *e.g.*,

$$\nu^i_j \geq \epsilon_1 + \epsilon_2 \|\nu^i\|_\infty, \tag{5.66}$$

where $\epsilon_1$ and $\epsilon_2$ are small positive scalars. Results regarding this matter are reported in Section 5.6.

### 5.4.3 Recommended Power Adjustments

To move the system away from the boundaries of the nearest constraints, *i.e.*, the ones corresponding to the set $\bar{\mathcal{N}}$, and increase security as measured by $M(y)$, several directions in the $y$ space are obtained. The first and most effective direction $d^*$ (among the ones obtained) is given by

$$d^* = \nabla M(y_0) = \bar{Q}^T \lambda, \tag{5.67}$$

where $\bar{Q}$ is the matrix whose rows are given by $q_i$, $i \in \bar{\mathcal{N}}$, and $\lambda$ is the vector whose entries are given by $(q_i^T y_0 - ti)^{-1}$, $i \in \bar{\mathcal{N}}$. Again, $\bar{Q}$ need not be constructed explicitly since only matrix-vector products are needed for obtaining $\lambda$ and $\bar{Q}^T\lambda$. If $\nabla M(y_0) \neq 0$, then $d^*$ is an ascent direction for $M(y_0)$ and hence moving along $d^*$ in the vicinity of $y_0$ increases the security margin. If $\nabla M(y_0) = 0$, then $y_0$ is the "most secure" point with respect to the constraints in $\bar{\mathcal{N}}$, since it is a global maximum of $M(y)$. However, the direction $d^*$ may involve a number of generator and load power adjustments that are impractical for operators to execute. Hence, directions that have similar effects to $d^*$ but that involve only a few power adjustments are desirable.

Two practical directions are considered here, each of which involves only one power adjustment. Other directions involving more power adjustments may be obtained by using similar ideas to the ones described next. The directions considered here are given by

$$d_1 = e_i e_i^T d^* \quad \text{and} \quad d_2 = e_j e_j^T d^*, \tag{5.68}$$

where $e_i$ and $e_j$ are elementary vectors, *i.e.*, having one entry equal to one and the rest zero, and $i$ and $j$ satisfy

$$i = \operatorname{argmax} \left\{ \ |d_s^*| \ \mid \ s \in [n_y] \ \right\} \tag{5.69}$$

$$j = \operatorname{argmax} \left\{ \ |d_s^*| \ \mid \ s \in [n_y] \setminus \{i\} \ \right\}. \tag{5.70}$$

As the next lemma shows, these directions have desirable properties.

**Lemma 5.4.5.** *The directions $d_1$ and $d_2$, as defined in* (5.68)*, are orthogonal and satisfy*

$$\nabla M(y_0)^T d_1 \geq 0, \quad \nabla M(y_0)^T d_2 \geq 0, \tag{5.71}$$

*and*

$$\|d_1 - d^*\|_2 \leq \|d_2 - d^*\|_2 \leq \|e_s e_s^T d^* - d^*\|_2, \ \forall s \in [n_y] \setminus \{i, j\}. \tag{5.72}$$

*Proof.* Orthogonality is trivial since $i \neq j$. The inequalities (5.71) follow from the fact that for any $s \in [n_y]$,

$$\nabla M(y_0)^T e_s e_s^T d^* = (\nabla M(y_0)_s)^2. \tag{5.73}$$

Lastly, the inequalities (5.72) follow from (5.69), (5.70) and the identity

$$\sum_{r \neq k} a_r^2 = \sum_{r \neq m} a_r^2 + (a_m^2 - a_k^2), \tag{5.74}$$

for all vectors $a \in \mathbb{R}^{n_y}$ and indices $k$ and $m \in [n_y]$. □

Lemma 5.4.5 shows that the "practical" directions $d_1$ and $d_2$ are both ascent directions of $M(y)$ unless $d_i^*$ or $d_j^*$ is zero, which from the definition of $i$ and $j$ would imply that $\nabla M(y_0)$ is zero or has at most one nonzero entry. Moreover, it also shows that these two directions are the closest directions to $d^*$, in the Euclidean sense, that have only one nonzero power adjustment.

Here, it has been assumed that all generators and loads of the system are controllable by the system operator. In practice, operators may be able to adjust only a subset of these or prefer adjusting some, *e.g.*, generators, over others, *e.g.*, loads, due to their economic effects. Restricting power adjustments to controllable devices may be incorporated into the procedure described above by only considering directions for which all entries associated with uncontrollable devices are zero. Favoring some adjustments over others due to economic factors cannot be directly incorporated into the procedure described here and it is a subject of future research.

### 5.4.4 Visualization

After obtaining the nearest reachable boundaries $\bar{Q}y \geq \bar{t}$ associated with the index set $\bar{\mathcal{N}}$, and recommended power adjustments $d^*$, $d_1$ and $d_2$ for improving security margins, this information needs to be presented to operators in a concise manner. Two approaches are considered for constructing a simple two-dimensional visualization of this information. They are based on finding a suitable two-dimensional plane that captures information about the nearest boundaries, and constructing a representation of the boundaries and recommended power adjustments on that plane.

A pair of orthonormal vectors $u_1$ and $u_2$ in the space of generator and load powers define a plane $P$ on that space given by

$$P = \text{span}\{u_1, u_2\}. \tag{5.75}$$

Any point on that plane may be expressed as $y = y_0 + Uw$, where $U$ is the matrix whose

columns are $u_1$ and $u_2$, and $w \in \mathbb{R}^2$. On that plane, the nearest reachable boundaries $\bar{Q}y \geq \bar{t}$ are given by

$$\bar{Q}y \geq \bar{t} \iff \bar{Q}(y_0 + Uw) \geq \bar{t} \tag{5.76}$$

$$\iff \bar{Q}Uw \geq \bar{t} - \bar{Q}y_0 \tag{5.77}$$

$$\iff \hat{Q}w \geq \hat{t}, \tag{5.78}$$

where $\hat{Q} = \bar{Q}U$ and $\hat{t} = \bar{t} - \bar{Q}y_0$. As the next lemma shows, boundary distances on the plane are greater than or equal to boundaries in the original space.

**Lemma 5.4.6.** *For any constraint $q^T y \geq t$ and two-dimensional plane $P$ spanned by orthogonal vectors $u_1$ and $u_2$, the boundary distance on $P$ is greater than or equal to the boundary distance in the original space.*

*Proof.* Using (5.77), the constraint on the plane is given by $q^T Uw \geq t - q^T y_0$. Applying Lemma 5.4.1 to both constraints $q^T y \geq t$ and $q^T Uw \geq t - q^T y_0$, and using the fact that $y_0$ corresponds to $w = 0$ and is strictly feasible, the boundary distance in the original space is given by

$$\delta := \frac{q^T y_0 - t}{\|q\|_2}, \tag{5.79}$$

while the one on the plane is given by

$$\bar{\delta} := \frac{q^T y_0 - t}{\|U^T q\|_2}. \tag{5.80}$$

Since $U$ has orthonormal columns, it holds that

$$\|U^T q\|_2 \leq \|U^T\|_2 \|q\|_2 \leq \|q\|_2. \tag{5.81}$$

Hence, it follows that $\bar{\delta} \geq \delta$. □

On a plane $P$ spanned by the two orthonormal columns of a matrix $U$, any recommended direction of generator and load power adjustments may be projected to obtain a corresponding direction on the plane. For one such a direction $d \in \mathbb{R}^{n_y}$, the projected direction is given by the vector $U^T d$ in terms of the basis given by the orthonormal columns of $U$.

One criterion for choosing a visualization plane is to try to maintain boundary distances of the nearest boundaries as much as possible. This is desirable since showing a boundary that is very close to the operating point as being far away may give operators a false sense of security. A particular plane that attempts to maintain boundary distances giving more emphasis to the ones that are nearest is the plane $P_s$ given by

$$P_s = \text{span}\{u_1, u_2\}, \tag{5.82}$$

where $u_1$ and $u_2$ are the left singular vectors associated with the two largest singular values of the security matrix $M_s$ defined in Section 5.4.2. This plane is referred to as the *security plane*, and its key property is formalized by the next theorem.

**Theorem 5.4.7.** *The matrix $U^*$ whose orthonormal columns span the security plane $P_s$ solves*

$$\begin{aligned} \underset{U}{\textit{maximize}} \quad & \sum_{i \in \bar{\mathcal{N}}} \beta_i^2 \|U^T q_i\|_2^2 \\ \textit{subject to} \quad & U^T U = I \\ & U \in \mathbb{R}^{n_y \times 2}, \end{aligned} \tag{5.83}$$

*with weights*

$$\beta_i = -(\tilde{\alpha}_i \sigma_i)^{-1}, \ i \in \bar{\mathcal{N}}, \tag{5.84}$$

*where $\tilde{\alpha}$ is the vector of boundary distance estimates, as defined in* (5.29)*, and $\sigma$ is the vector of norm estimates of constraint normals, as defined in* (5.28)*.*

*Proof.* Letting $a_i = \beta_i q_i$, for each $i \in \bar{\mathcal{N}}$, and $A$ the matrix whose columns are $a_i$, $i \in \bar{\mathcal{N}}$, the objective function of (5.83) may be expressed as

$$\sum_{i \in \bar{\mathcal{N}}} \beta_i^2 \|U^T q_i\|_2^2 = \sum_{i \in \bar{\mathcal{N}}} (\beta_i q_i)^T U U^T (\beta_i q_i) \tag{5.85}$$

$$= \sum_{i \in \bar{\mathcal{N}}} a_i^T U U^T a_i \tag{5.86}$$

$$= \text{Tr}(A^T U U^T A) \tag{5.87}$$

$$= \text{Tr}(U^T A A^T U), \tag{5.88}$$

where $\mathrm{Tr}(\cdot)$ gives the trace of a matrix. Hence, problem (5.83) is equivalent to

$$\begin{aligned} \underset{U}{\text{maximize}} \quad & \mathrm{Tr}(U^T A A^T U) \\ \text{subject to} \quad & U^T U = I \\ & U \in \mathbb{R}^{n_y \times 2}. \end{aligned} \tag{5.89}$$

It is well known that the solution of (5.89) is given by the matrix $U$ whose columns $u_1$ and $u_2$ are the eigenvectors corresponding to the two largest eigenvalues of $AA^T$ [25] [42] [66]. Hence, $u_1$ and $u_2$ are the left singular vectors corresponding to the two largest singular values of $A$. Since the matrix $A$ is equal to the security matrix $M_s$ defined in (5.42) for weights given by (5.84), it follows that the matrix $U^*$ used to define the security plane is the solution of (5.83) with such weights. □

Since boundary distances on a plane are never smaller than in the original space, as shown by Lemma 5.4.6, maintaining boundary distances is equivalent to minimizing boundary distances on the plane. Since the boundary distance of a plane constraint $q^T U w \geq t - q^T y_0$ is given by

$$\frac{q^T y_0 - t}{\|U^T q\|_2}, \tag{5.90}$$

minimizing its boundary distance over $U$ is equivalent to maximizing the quantity $\|U^T q\|_2^2$. From this, it is clear that the security plane $P_s$, which uses a matrix $U$ that solves (5.83), attempts to maintain boundary distances giving more emphasis to the ones that are closer to the current point $y_0$.

Inevitably, the boundaries of some of the constraints in $\bar{\mathcal{N}}$ that are not nearest to the point $y_0$, *i.e.*, the ones that are given low weights by the security plane, may appear on $P_s$ further than they are in the original space. Hence, adjustments may be made to the right hand sides of the resulting plane constraints to correct for this. More specifically, $\hat{t}$ in (5.78) may be adjusted such that

$$\hat{t}_i = -\tilde{\alpha}_i \|U^T q_i\|_2, \tag{5.91}$$

for each $i \in \bar{\mathcal{N}}$, where $\tilde{\alpha}$ is as defined in (5.29).

Another criterion for choosing a visualization plane is to capture as much information as possible about the geometry of the boundaries as seen by specific control actions. In this case, two control actions $\Delta y_1$ and $\Delta y_2$ characterizing generator and load power adjustments

may be used to define a *control plane*. This plane, which is denoted by $P_c$, is given by

$$P_c = \text{span}\{u_1, u_2\}, \tag{5.92}$$

where $u_1$ and $u_2$ are unit vectors parallel to $\Delta y_1$ and $\Delta y_2$, respectively. In particular, the sparse recommended adjustments $d_1$ and $d_2$ defined Section 5.4.3 may be used for this purpose.

#### Scale and Boundaries

To draw the boundaries of the plane constraints $\hat{Q}w \geq \hat{t}$, a scale must be determined along with boundaries on $w$. This may be done by setting

$$w^{\max} = \quad s_f \delta_{\min} e \tag{5.93}$$

$$w^{\min} = -s_f \delta_{\min} e, \tag{5.94}$$

where $s_f$ is a scaling factor, *e.g.*, 4 or 5, $\delta_{\min}$ is the distance to the closest boundary, and $e \in \mathbb{R}^2$ is the vector of ones.

### 5.4.5 Tracing Nonlinear Boundaries

Let $P$ be the plane defined by

$$P = \text{span}\{u_1, u_2\}, \tag{5.95}$$

where $u_1$ and $u_2$ are unit vectors in the space of generator and load powers. Then, the boundary of a security constraint, say $h_i(x) \leq 0$, on $P$, is given by

$$\mathcal{T} := \{\ w \in \mathbb{R}^2 \ \mid\ \Phi(x, w) = 0 \text{ for some } x \in \mathbb{R}^{2n}\ \}, \tag{5.96}$$

where the function $\Phi$ is defined as

$$\Phi(x, w) := \begin{bmatrix} Y(y_0 + Uw) - f(x) \\ h_i(x) \end{bmatrix}, \tag{5.97}$$

$U$ is the matrix with columns $u_1$ and $u_2$, and $Y$, $y_0$ and $f$ are as defined in Section 5.2.

Given bounds $w^{\min}$ and $w^{\max} \in \mathbb{R}^2$, the set of $w$ such that

$$w \in \mathcal{T} \cap \{w \in \mathbb{R}^2 \ \mid \ w^{\min} \leq w \leq w^{\max} \} \tag{5.98}$$

may be found by sweeping the value of one of the two entries of $w$, say $w_1$, and obtaining the corresponding values of $w_2$ using the Newton-Raphson method with a predictor-corrector strategy, as is commonly done in homotopy-based methods [23]. This procedure may be stopped when the NR method fails or when $w$ reaches its bounds. Figure 5.3 illustrates this approach. There, the predictor step predicts the value $\tilde{w}_2$ for the new value $w_1'$ of $w_1$. This prediction typically results in the point $(w_1', \tilde{w}_2)$ not being in the desired set. The corrector step is hence applied at this point to restore feasibility, giving the new point $(w_1', w_2')$.

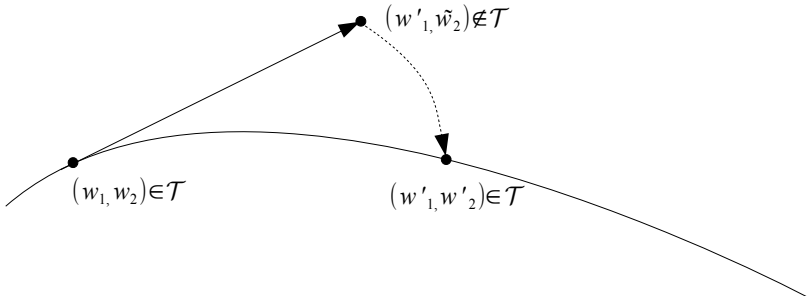


Figure 5.3: Prediction-correction strategy for tracing nonlinear boundaries.

Information from the linear constraint on the plane, say $\hat{q}^T w \geq \hat{t}$, corresponding to the constraint $h_i(x) \geq 0$ may be exploited for simplifying the tracing of the nonlinear boundary. This may be done by obtaining the point on the linear version of the boundary that is closest to the current operating point $y_0$, *i.e.*,

$$\hat{y} := y_0 + U\hat{w}, \tag{5.99}$$

where $\hat{w} = \hat{q}\hat{t}/\|\hat{q}\|_2^2$, and using $\hat{w}$ and $\hat{x} := A^{-1}(b + Y\hat{y})$ to start the predictor-corrector procedure. Moreover, the orientation of $\hat{q}$ may be used for determining whether to use $w_1$ or $w_2$ as the independent or "sweeping" variable. For example, if $|\hat{q}_2| > |\hat{q}_1|$, then $w_1$ may be preferable as the independent variable.

## 5.5 Implementation

All the algorithms described were implemented in the Python programming language. The linear algebra operations were performed using the Python packages `Numpy` and `Scipy`, and the sequential version of the multifrontal sparse direct solver `MUMPS` [3] through the Python wrapper `pymumps`. Parallelization of the re-linearizations of the system for improving the model accuracy was implemented using Python's multiprocessing module.

## 5.6 Experiments

Experimental results showing the performance of the algorithms described for determining and visualizing branch thermal and voltage magnitude limits are presented. These include results regarding the accuracy of the linear model, the effectiveness of the matrix-free stochastic approach for estimating constraint boundary distances, sample linear and nonlinear boundary visualizations on different planes, and execution time profiles of all the main algorithm steps.

The COB test cases used were obtained from three power networks of two different North American System Operators. The data obtained for these cases consisted of load flow data, which included branch thermal limits and generator active power limits. For voltage magnitude limits, values of $v \pm 0.1$ per unit bus nominal voltage were used, where $v$ is the bus voltage magnitude given in the base case data. For load active power limits, values of 0 and $2P$ were used, where $P$ is the load active power given in the base case data. Table 5.1 shows properties of the COB test cases. Cases A and B had one and four branch thermal violations, respectively, at the given base case power. To obtain an initial strictly feasible state, the thermal bounds of the corresponding violated constraints were set to a value 10% larger than the branch current magnitude. Case C was a problematic power flow case for which active and reactive power injections had to be modified significantly in order to make it solvable by the available power flow solvers. The resulting case had several thermal violations, and these were eliminated by increasing the corresponding thermal bounds as done for Cases A and B. Since the resulting Case C had a significant number of thermal bounds that were somewhat arbitrary, this case was only used for evaluating the computational requirements of the algorithms.

Table 5.1: Properties of COB test cases.

| name | buses | branches | voltage constraints | thermal constraints |
|---|---|---|---|---|
| Case A | 2454 | 2781 | 4908 | 5562 |
| Case B | 2468 | 3215 | 4936 | 6430 |
| Case C | 45286 | 58994 | 90572 | 117988 |

### 5.6.1 Model Accuracy

The accuracy of the linear model described in Section 5.3 near the critical operating boundaries was studied. This was done by first extracting the 200 nearest linear boundaries and filtering the reachable ones using the algorithms described in Section 5.4.1. For each of the obtained linear boundaries, the boundary point $y$ nearest to the current point $y_0$ was obtained, which is of the form $y = y_0 - \alpha q$, where $q$ is the constraint normal and $\alpha$ is some positive scalar. Then, the constraint quantity, *i.e.*, bus voltage magnitude or branch current magnitude, corresponding to the system state at point $y$ was obtained using both the linear and nonlinear models, and the error between these two was computed. These errors were then plotted as a function of the distance of the boundary point from the current point $y_0$. This procedure was repeated twice after updating the linear model using the distributed algorithm described in Section 5.4.2.

For some of the constraints, it was not possible to obtain the system state based on the nonlinear model corresponding to the nearest point on the linear boundary. This was due to the point on the linear boundary being near or beyond the boundary of the feasible region of the nonlinear power flow equations. The procedure used for trying to obtain the nonlinear system state consisted of solving a sequence of closely-related power flow problems with the NR method while gradually changing the loading conditions in the direction of $-q$. When this procedure failed due to approaching the feasibility boundary and encountering a near-singular Jacobian of the power flow constraints, the loading conditions and system state associated with the last problem solved in the sequence were used for evaluating the accuracy of the linear model. This was observed to occur mostly for voltage boundaries for Case B, for which it affected around 50% of the constraints considered. For thermal constraints or for Case A, this was observed to occur only for about 5% of the constraints considered.

The distributed algorithm for updating the linear model used $m = 8$ parallel processes, considered $\tilde{n}_c = 100$ nearest constraints, used a maximum angle $\vartheta = \pi/3$, and used weights

$\zeta_0 = 0.25$ and $\zeta_1 = 0.75$ for the averaging step. Figures 5.4 and 5.5 show the results obtained. In the figures, voltage errors are shown in units or per unit nominal voltage, while current magnitude errors are shown as percentages of branch thermal ratings.

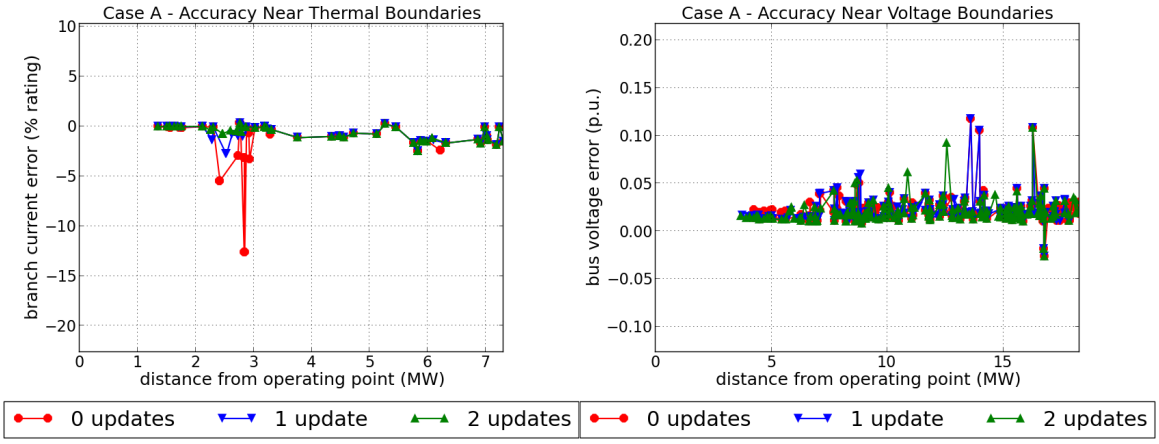


Figure 5.4: Model accuracy near boundaries of COB Case A.

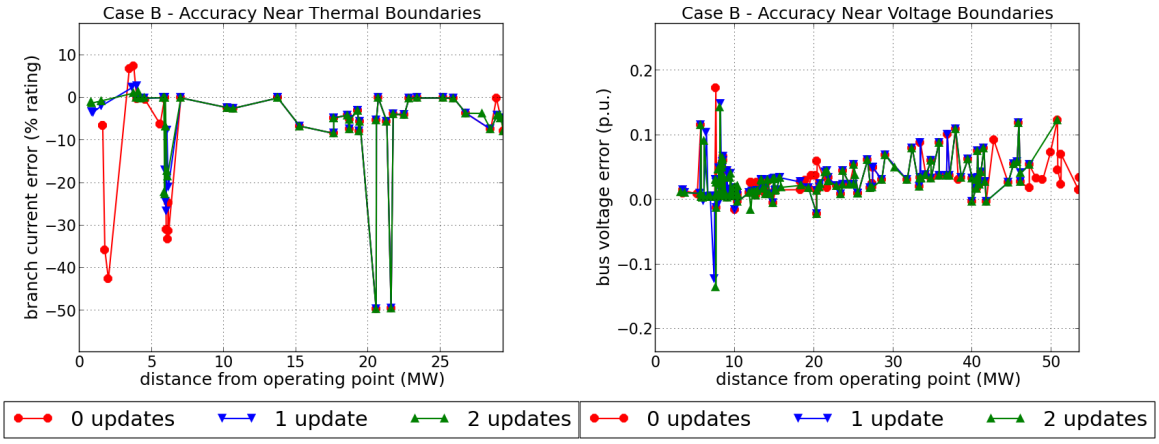


Figure 5.5: Model accuracy near boundaries of COB Case B.

From the plots, several observations can be made: For Case A, the model errors associated with the few nearest voltage boundaries are fairly small ($\leq 0.02$ p.u.), and, except for a few constraints, so are the errors associated with the thermal boundaries ($\leq 3\%$). For Case B, the linear model has large errors associated with both voltage and thermal boundaries that are near the current operating point. The distributed algorithm described in Section 5.4.2 reduces significantly the large errors associated with near thermal boundaries of Case A and Case B, and the most significant error reductions occur after the first model update.

This algorithm also reduces slightly the model errors associated with the few nearest voltage boundaries of Case A. However, it does not reduce the large errors associated with a few near voltage boundaries of Case B. One possible reason for this is the construction and use of ill-conditioned linear systems at the new linearization points due to these points being near the power flow feasibility boundary. To investigate this, the condition number of the Jacobian of the power flow constraints of Case B was obtained at several points. These points included the near-boundary points used to compare the linear and nonlinear system models that resulted in larger errors, and the corresponding new linearization points used by the distributed approach. It was found that at these points the condition numbers were around $3 \times 10^8$ and only slightly larger than those obtained at the current point of Case B and at points associated with small model errors, which were around $1 \times 10^8$. Also, compared to the condition number obtained at the current point of Case A, they were about one order of magnitude larger. These numbers suggest that using poorly-conditioned linear systems is not the cause of the inability of the distributed approach to reduce the large model errors associated with the near voltage boundaries of Case B. Hence, it is conjectured that this inability is attributed to the fact that Case B is near the power flow feasibility boundary, as already noted, where voltages have a more nonlinear relation with powers (see PV curves in [26]). Lastly, another observation is that most of the thermal errors are under-estimations, while the voltage errors are over-estimations. These properties require further investigation and are subject of future research.

With regards to the operation of the distributed algorithm for updating the linear model, two important quantities were noted. One of these quantities was the percentage of nonzero entries of the new state-space constraint normals (5.65) returned by each parallel process. It was observed that this number on average was about 5% for Case A and about 25% for Case B by applying (5.66) with $\epsilon_1 = 10^{-7}$ and $\epsilon_2 = 10^{-7}$. The other important quantity was the number of constraints returned by each process to be used for updating the linear model. It was observed that with the maximum angle $\vartheta = \pi/3$, this number was on average about 5 for both cases.

### 5.6.2 Nearest Boundaries

The effectiveness of the matrix-free stochastic algorithm described in Section 5.4.1 for estimating boundary distances and filtering the nearest ones was tested. This was done by using the algorithm to obtain an approximate set of the 500 nearest constraints for each test

case, and determining the percentage of the actual nearest 500 and nearest 50 contained in that set. A brute force approach that constructs each constraint normal was used for determining the actual nearest constraints (according to the linear model). Tables 5.2-5.4 show the results obtained. The stochastic algorithm was executed 10 times for each case, and hence results are shown as $\mu \pm \sigma$, where $\mu$ denotes the mean and $\sigma$ the standard deviation of a particular quantity. An interesting note is that although the algorithm for estimating boundary distances was derived assuming Gaussian random vectors $z$, it was observed to require fewer samples when using $z$ consisting of independent and identically distributed uniform random variables with zero mean and unit variance. The results shown here were obtained using this enhancement.

Table 5.2: Near-constraint filtering on COB Case A.

| algorithm | time (s) | samples | % top 500 | % top 50 |
|---|---|---|---|---|
| stochastic | $0.08 \pm 0.004$ | $53 \pm 3$ | $95.6 \pm 0.5$ | $100 \pm 0$ |
| brute-force | 13.92 | - | 100 | 100 |

Table 5.3: Near-constraint filtering on COB Case B.

| algorithm | time (s) | samples | % top 500 | % top 50 |
|---|---|---|---|---|
| stochastic | $0.08 \pm 0.006$ | $52 \pm 4$ | $94.7 \pm 1.0$ | $100 \pm 0$ |
| brute-force | 15.66 | - | 100 | 100 |

Table 5.4: Near-constraint filtering on COB Case C.

| algorithm | time (s) | samples | % top 500 | % top 50 |
|---|---|---|---|---|
| stochastic | $1.28 \pm 0.09$ | $53 \pm 4$ | $95.1 \pm 1.6$ | $100 \pm 0$ |
| brute-force | 4836.34 | - | 100 | 100 |

The results show that the matrix-free stochastic approach is effective for estimating boundary distances and filtering the nearest constraints, and is suitable for online applications due to its speed. On the other hand, the brute-force approach is significantly slower and impractical for large cases such as Case C, for which it takes more than 1 hour to determine boundary distances.

### 5.6.3 Visualization

The 200 nearest constraints of both Case A and Case B were obtained and from these the ones that are reachable within the generator and load power bounds were extracted using the algorithms of Section 5.4.1. The resulting constraints were reconstructed on both the security and control planes, as described in Section 5.4.4. Figures 5.7 and 5.8 show the plots obtained and Figure 5.6 provides the legend for the plots. It is noted that for constraint reconstructions on the security plane, boundary distances on the plane were adjusted to reflect distances on the original space in order to visualize more boundaries. For reconstructions on the control plane, boundary distances reflect actual distances on the plane. Also, for security-plane plots, the single-control recommended directions $d_1$ and $d_2$ are shown as $d_1 + d_2$, while for the control-plane plots, they are shown as separate directions since they are the basis for the plane.

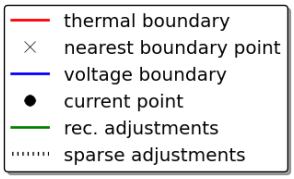


Figure 5.6: Legend for boundary plots.

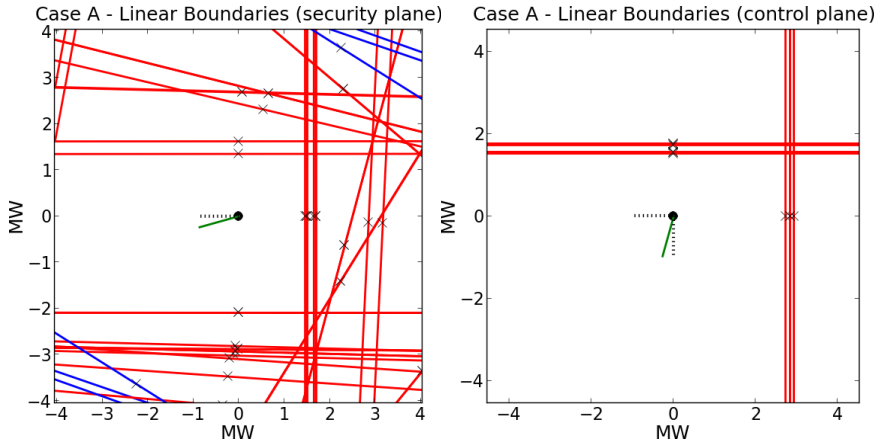


Figure 5.7: Linear boundaries of COB Case A.

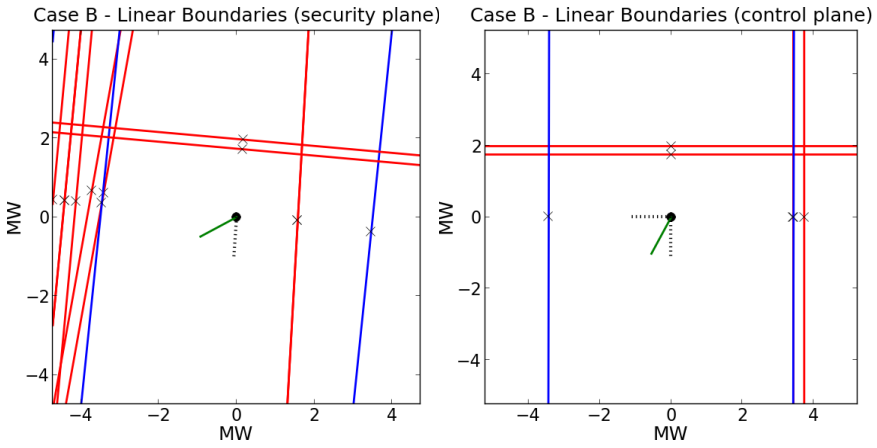


Figure 5.8: Linear boundaries of COB Case B.

As the figures show, both cases appear to have boundaries that are only a few MW away from the current operating point. These small MW distances can be attributed to the following reasons: Both cases had branches with relatively small thermal limits (with respect to other branches in the network) that were close to being reached at the current operating point. For example, Case A has a branch with a thermal limit of 13 MVA and an apparent power flow of 12 MVA at the current operating point. Case B, as stated before, is a more ill-conditioned case for which the current operating point is closer to the boundary of the power flow constraints, around which voltages tend to change more rapidly (see PV curves in [26]). Also, distances shown in the figures only include MW changes of generators and loads, and not the implicit MVAr changes due to our assumption that power factors remain constant (Section 5.3). Finally, distances shown are in terms of Euclidean distances in the space of generator and load powers, *i.e.*, $\|\Delta y\|_2$, which are smaller than total absolute power changes, *i.e.*, $\|\Delta y\|_1$. They are related by the inequalities

$$\|\Delta y\|_2 \leq \|\Delta y\|_1 \leq \sqrt{n_y}\|\Delta y\|_2, \tag{5.100}$$

where $n_y$ is the dimension of the vector $\Delta y$. Considering other ways to measure distance, such as using the total absolute changes in generator and load powers, is subject of future research.

To compare linear boundaries with their nonlinear counterparts, the algorithm described in Section 5.4.5 was used to trace the nonlinear boundaries. This was done for only a few of the nearest voltage and thermal security constraints. Figure 5.9 shows the linear and

nonlinear boundaries of the nearest voltage constraints of Case A, while Figure 5.10 shows the linear and nonlinear boundaries of the nearest thermal constraints of Case B. For both of these figures, the plots reflect actual distances on the plane.

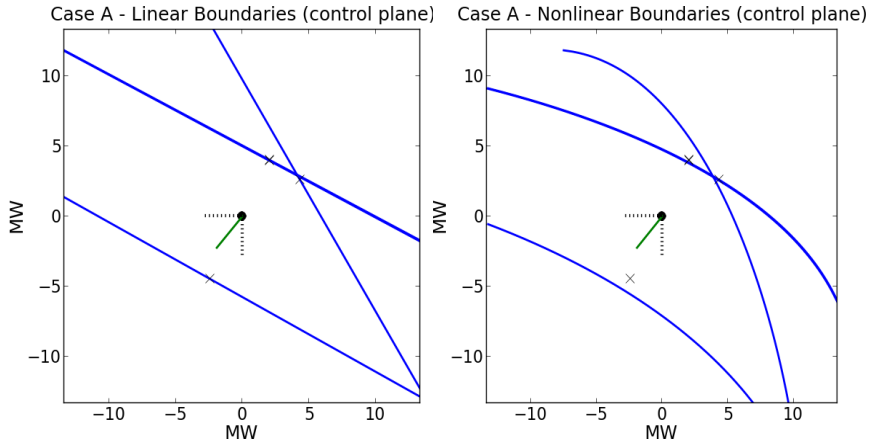


Figure 5.9: Linear and nonlinear voltage boundaries of COB Case A.

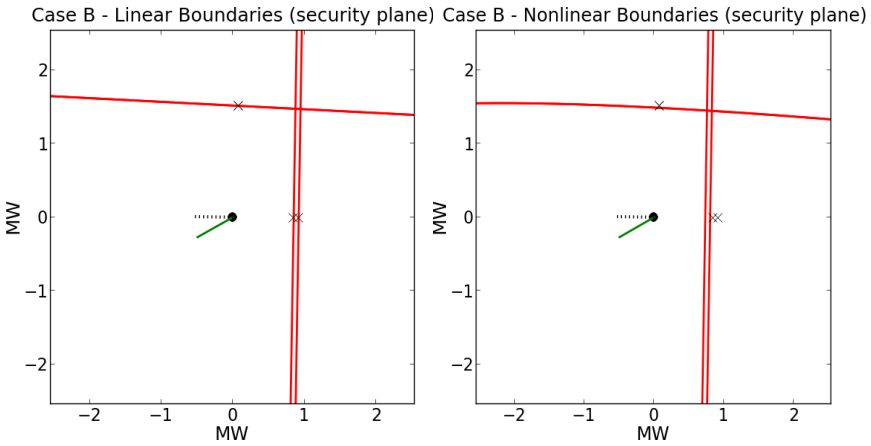


Figure 5.10: Linear and nonlinear thermal boundaries of COB Case B.

### 5.6.4 Computational Requirements

The computational requirements of the tasks performed for determining and visualizing critical security boundaries were evaluated. This was done by measuring the time spent by each of the following tasks: finding the nonlinear system state, constructing the linear model, filtering the 100 nearest constraints, filtering the ones that are reachable within the power bounds, finding recommended adjustments, obtaining a basis for the security plane,

reconstructing the constraints on the plane, and tracing the linear boundaries for creating the display. Tables 5.5-5.7 show the results obtained for each of the test cases.

Table 5.5: Execution times of algorithms on COB Case A

| task | time (s) | time (%) |
|---|---|---|
| finding nonlinear state | 0.065 | 4.09 |
| constructing linear model | 1.041 | 65.42 |
| filtering 100 nearest constr. | 0.057 | 3.58 |
| filtering reachable constr. | 0.143 | 8.99 |
| finding rec. adjustments | 0.003 | 0.19 |
| finding security plane basis | 0.076 | 4.78 |
| reconstructing constr. on plane | 0.004 | 0.25 |
| tracing linear boundaries | 0.202 | 12.70 |
| Total | 1.591 | 100.0 |

Table 5.6: Execution times of algorithms on COB Case B

| task | time (s) | time (%) |
|---|---|---|
| finding nonlinear state | 0.057 | 3.43 |
| constructing linear model | 1.174 | 70.59 |
| filtering 100 nearest constr. | 0.079 | 4.75 |
| filtering reachable constr. | 0.145 | 8.72 |
| finding rec. adjustments | 0.003 | 0.18 |
| finding security plane basis | 0.061 | 3.67 |
| reconstructing constr. on plane | 0.004 | 0.24 |
| tracing linear boundaries | 0.140 | 8.42 |
| Total | 1.663 | 100.00 |

Table 5.7: Execution times of algorithms on COB Case C

| task | time (s) | time (%) |
|---|---|---|
| finding nonlinear state | 2.620 | 8.49 |
| constructing linear model | 23.662 | 76.64 |
| filtering 100 nearest constr. | 0.875 | 2.83 |
| filtering reachable constr. | 2.379 | 7.70 |
| finding rec. adjustments | 0.050 | 0.16 |
| finding security plane basis | 1.079 | 3.49 |
| reconstructing constr. on plane | 0.073 | 0.24 |
| tracing linear boundaries | 0.139 | 0.45 |
| Total | 30.877 | 100.00 |

The results show that by far the most time-consuming task is constructing the linear model. This task involves constructing the matrices and vectors $A$, $b$, $Y$, $y_0$, $C$ and $d$, as described in Section 5.3. The reason for this task being slow is that, in the current implementation, the routines for constructing these matrices and vectors are written in Python, which is particularly slow for this type of operations. This is the reason why the routines for performing function evaluations in the power flow and OPF algorithms of Chapters 3 and 4, respectively, were implemented in C and wrapped using Python's C API. From experience, it is expected that if these routines are implemented in C, they would be around 100 times faster and the entire procedure would take around 0.5 seconds for Cases A and B, and around 7 seconds for the Case C. In this case, the execution time of the algorithm would be dominated by the time spent filtering reachable constraints and drawing linear boundaries, for Cases A and B, and by the time spent finding the nonlinear state (solving the initial power flow given the current loading conditions) and filtering the reachable constraints for Case C.

The computational requirements of the tasks for updating the linear model were also evaluated. For doing this, the distributed algorithm for updating the linear model was executed using 8 parallel processes for Cases A and B, and 3 parallel processes for Case C. The reason for using only 3 processes for the large case is that the computer used for the experiment did not have enough memory to run more. Tables 5.8-5.10 show the times required by each task. The second through fifth tasks involve obtaining the system state based on the nonlinear model (solving around three power flow problems), constructing the

linear model by linearizing the system at the new state, and filtering the nearest and reachable constraints. They are performed by parallel processes and hence time values shown for these tasks are averages. The last two tasks involve applying the filtering algorithms again in order to obtain a new set of nearest reachable constraints based on the updated linear model.

Table 5.8: Execution times of model-update tasks on COB Case A

| task | time (s) | time (%) |
|---|---|---|
| finding new linearization locations | 0.124 | 4.13 |
| finding nonlinear state (ave.) | 0.355 | 11.83 |
| constructing linear model (ave.) | 1.795 | 59.83 |
| filtering 100 nearest constr. (ave.) | 0.122 | 4.07 |
| filtering reachable constr. (ave.) | 0.233 | 7.77 |
| updating state security constraints | 0.009 | 0.30 |
| transferring data to/from workers | 0.162 | 5.40 |
| re-filtering 100 nearest constr. | 0.057 | 1.90 |
| re-filtering reachable constr. | 0.143 | 4.77 |
| Total | 3.000 | 100.00 |

Table 5.9: Execution times of model-update tasks on COB Case B

| task | time (s) | time (%) |
|---|---|---|
| finding new linearization locations | 0.091 | 2.67 |
| finding nonlinear state (ave.) | 0.373 | 10.95 |
| constructing linear model (ave.) | 2.057 | 60.37 |
| filtering 100 nearest constr. (ave.) | 0.142 | 4.17 |
| filtering reachable constr. (ave.) | 0.249 | 7.31 |
| updating state security constraints | 0.027 | 0.79 |
| transferring data to/from workers | 0.244 | 7.16 |
| re-filtering 100 nearest constr. | 0.079 | 2.32 |
| re-filtering reachable constr. | 0.145 | 4.26 |
| Total | 3.407 | 100.00 |

Table 5.10: Execution times of model update tasks on COB Case C

| task | time (s) | time (%) |
|---|---|---|
| finding new linearization locations | 1.295 | 3.18 |
| finding nonlinear state (ave.) | 5.324 | 13.07 |
| constructing linear model (ave.) | 23.581 | 57.85 |
| filtering 100 nearest constr. (ave.) | 1.123 | 2.76 |
| filtering reachable constr. (ave.) | 2.643 | 6.49 |
| updating state security constraints | 0.085 | 0.21 |
| transferring data to/from workers | 3.442 | 8.45 |
| re-filtering 100 nearest constr. | 0.875 | 2.15 |
| re-filtering reachable constr. | 2.379 | 5.84 |
| Total | 40.747 | 100.00 |

From the execution times of the tasks for updating the linear model, it is observed that the most time consuming ones are constructing the linear model and finding the nonlinear state. Again, with a C implementation of the routines for constructing of the linear model, as in the PF and OPF algorithms of Chapters 3 and 4, the construction task would be computationally negligible and updating the linear model would take around 1.5 seconds for Cases A and B, and around 15 seconds for Case C.

Lastly, the computational requirements of the algorithm for tracing nonlinear boundaries were evaluated. This was done by measuring the time required for tracing various voltage and thermal boundaries for each case. For a total number of boundary points equal to 200, it was found that the tracing algorithm required roughly about 6, 6, and 80 seconds for Cases A, B and C, respectively, per boundary. Hence, tracing one nonlinear boundary alone is much more computationally expensive than determining and visualizing several critical linearized boundaries. From this, it is concluded that for any application that traces nonlinear boundaries using the approach described in Section 5.4.5, different boundaries should be traced in parallel.

## 5.7 Conclusions

In this chapter, the problem of determining and visualizing critical security boundaries was considered. In particular, a modeling approach was described that is based on a linear system model and a problem transformation that characterizes security boundaries in terms of generation and load powers. Efficient filtering algorithms were proposed for estimating

boundary distances, finding the nearest ones, and determining the ones that can actually be reached or violated within the given power bounds. With respect to the set of nearest and reachable constraints, a technique for determining power adjustments that move the system away from the boundaries was described. A visualization procedure was then proposed for showing the nearest reachable boundaries as well as the recommended power adjustments on a two dimensional diagram. This procedure consisted of first finding a suitable plane and then re-constructing the constraints on that plane. Two planes were considered: one that attempts to preserve boundary distances while giving more emphasis to the ones that are closer to the operating point, and another that captures boundaries that are relevant to specific control actions. Experimental results showing the efficiency and effectiveness of the filtering and visualization techniques were presented as well as sample visualizations.

The accuracy of the linear system model considered was evaluated, and it was found that it provided reasonable accuracy for many but not all of the nearest security boundaries. To try to improve its accuracy, a distributed algorithm was described that re-linearizes the system at key locations and incorporates the information obtained into the original system model. Experimental results presented showed that this algorithm was most effective in reducing model errors associated with near thermal boundaries, and least effective in reducing model errors associated with near voltage boundaries. It was conjectured that the reason for this is that voltage boundaries are more nonlinear than thermal boundaries, especially for ill-conditioned cases that are closer to their loadability limit.

Lastly, a technique for tracing nonlinear boundaries that utilizes information from the linear boundary approximations was described. This technique is based on applying the Newton-Raphson process repeatedly using a prediction-correction strategy. It was found that tracing only one nonlinear boundary using this approach was much more computationally expensive that determining and visualizing linearized boundaries with the proposed approach. Based on this, it was concluded that tracing nonlinear boundaries should be done in parallel in order to be potentially suitable for online applications.

# Chapter 6

# Conclusions

In this work, modeling and numerical optimization techniques were explored for overcoming the limitations associated with current algorithms used for power network operations and planning. Current algorithms are typically based on heuristics and have worked well in the past, but their limitations are compromising system efficiency and reliability as grids become more complex, variable and unpredictable, and operate closer to their limits.

With regards to scenario analysis, current power flow algorithms lack robustness, and this complicates system analyses and potentially limits the ability of operators and planners to assess system performance and security. To address this, a new problem formulation was explored together with robust optimization techniques. The resulting algorithm was shown to be much more robust and informative than a benchmark algorithm representing current practices. The algorithm was also found to be slower than the benchmark method for simple cases. Hence, it was suggested that the new algorithm be used on challenging scenarios for which current algorithms fail or sensitivity information is required. Future directions for enhancing this work may be to develop a hybrid algorithm that combines the fast NR algorithm with the robust techniques explored here to produce a single tool. Also, techniques for reducing the number of iterations of the proposed algorithm may be explored in the future by using a statistical approach. The large number of iterations required by the proposed method is partly attributed to a competition between the objective function and the constraints. The objective function used in this work was the same for all cases. Statistics or Machine Learning techniques may be used for determining the best objective function, or equivalently, the most suitable prior distribution of a power flow solution, for a specific network based on historical data. Lastly, ways of extending the techniques explored

here to handle other problem elements that are typically handled with heuristics in current algorithms, such as DC lines, may be explored.

With regards to system planning and control, current optimal power flow algorithms typically use inadequate rounding techniques for handling discrete variables, and may recommend an impractical number of control actions. These deficiencies were addressed here by exploring the use of a smooth sparsity-inducing penalty function and parallel processing. The smooth sparsity-inducing function was found to be effective for obtaining a manageable number of control actions. The parallel processing approach, which was built on top of the robust techniques proposed for solving power flow problems, was found to allow obtaining higher-quality solutions than a benchmark method representing current practices. Future directions for enhancing this work may be to test the proposed techniques for handling discrete variables on large-scale cases using computer clusters. Other ways of parallelizing the exploration of the discrete-variable space may be considered. For example, instead of using the solution estimate of the "best" discrete point evaluation for starting the next iteration, one may use the best two or more of these and start several new iterations in parallel. Also, the techniques proposed here for handling discrete variables may be compared or enhanced with other techniques that have been proposed in the OPF literature, such as sequential MILP and progressive rounding.

With regards to online security assessment, current methods rely on operator experience and computer simulations that are computationally expensive for determining and visualizing critical operating boundaries. These deficiencies were addressed by exploring efficient techniques based on a linear system model and matrix-free methods. These techniques were found to allow quick identification of critical thermal and voltage security boundaries, and determination of suitable visualization planes and scales. The accuracy of the results obtained was found to be acceptable for most but not all of the system boundaries. However, a proposed distributed approach for improving the linear system model was found to be effective for reducing the model errors associated with the nearest thermal boundaries. Future directions for enhancing this work may be to consider other ways of measuring boundary distances. One example would be to use the total absolute changes in generator and load powers instead of the Euclidean distance. Ways of incorporating reactive power limits of generators and their effect on voltage regulation need to be explored. The proposed techniques may be also extended to consider post-contingency constraints and more complex

operating boundaries. Post-contingency constraints may be considered without naively repeating the entire procedure for each contingency. This may be done by exploiting the fact that the linear model of a contingency case is very similar to the one of the base case, and that contingency effects may be mostly local. Considering more complex operating boundaries, such as voltage stability boundaries, is expected to be challenging since the applicability of a linear model for this is questionable. Lastly, ways of incorporating and taking advantage of the techniques described here in current security assessment practices need to be determined. The algorithms presented here, given information about the current loading conditions, can quickly provide a visualization of the critical thermal and (simple) voltage boundaries and power adjustments to improve security. The visualization plane and scale obtained are dynamic and change according to the loading conditions and state of the system. The information obtained with the proposed techniques may be used to define "power transfer studies" that rely less on historical data and obtain fast approximate boundary visualizations. These may then be enhanced by applying more accurate but computationally expensive methods based on system simulations to improve security awareness.

# Appendix A

# Global Convergence

Under the assumption that bounded power mismatches implies bounded voltage magnitudes, it is shown that the sequence of iterates $\{x_k\}$ of the power flow algorithm described in Section 3.5 has a feasible point (according to the required tolerance) or a limit point that is a stationary point of the function $c^T c$ (restricted to the set of $x$ such that $Ax = b$). This result is obtained without assuming that the Jacobian of the power flow equations is full rank. In the following proofs, the subscript $i$ is used for buses or terms inside a sum or product series, the subscript $j$ is used to associate an object with the $j$-th inner iteration of subproblem (3.36), and the subscript $k$ is used to associate an object with the $k$-th outer iteration. The continuity of the functions involved is also used throughout. To simplify the notation, derivatives are assumed to be with respect to $x$ and hence a subscript $x$ is not used for derivatives of functions that include other variables.

The reader is reminded that $x$ is a vector of voltage magnitudes $\{v_i\}_{i\in[n]\setminus\{s\}}$, voltage angles $\{\theta_i\}_{i\in[n]\setminus\{s\}}$, generator reactive powers $\{Q_i^g\}_{i\in\mathcal{R}}$, and positive and negative regulated voltage magnitude deviations $\{y_i\}_{i\in\mathcal{R}}$ and $\{z_i\}_{i\in\mathcal{R}}$, respectively. The function $\Phi$ is the smooth vector-valued function whose scalar function entries are given by (3.23) and (3.24). The objective function $\varphi$ is a strongly convex non-negative quadratic function, and the function $f$ is the vector-valued function of active and reactive power mismatches. The function $c$ consists of both $f$ and $\Phi$, as defined in (3.34). Throughout this section, $J$ is used to denote the Jacobian of $c$, and $Z$ is used to denote the matrix whose columns span the null space of $A$, the Jacobian of the linear equality constraints.

When dealing with outer iterations, $\varphi_k$, $f_k$, $\Phi_k$, $c_k$ and $J_k$ are used to denote $\varphi(x_k)$, $f(x_k)$, $\Phi(x_k)$, $c(x_k)$ and $J(x_k)$, respectively. Similar shorthand notation is used when dealing

with inner iterations.

**Assumption 1.** *For all $M > 0$, the set $\{x \mid f(x)^T f(x) \leq M\}$ has bounded voltage magnitudes $\{v_i\}_{i\in[n]\setminus\{s\}}$.*

**Lemma A.0.1.** *For any given penalty parameter $\mu > 0$ and vector $\lambda$, the search directions $\{p_j\}_{j\in\mathbb{Z}_+}$ used during the solution of subproblem* (3.36) *are descent directions. That is, they satisfy*

$$\nabla L_\mu(x_j, \lambda)^T p_j < 0 \tag{A.1}$$

*for each $j \in \mathbb{Z}_+$ such that $Z^T \nabla L_\mu(x_j, \lambda) \neq 0$.*

*Proof.* Suppose not. Then, for some $j$ such that $Z^T g_j \neq 0$, where $g_j := \nabla L_\mu(x_j, \lambda)$, $g_j^T p_j \geq 0$. Hence, $p_j$ does not satisfy (3.48) and must have been obtained from

$$Z^T \tilde{H}_\mu(x_j, \lambda) Z q_j = -Z^T g_j \tag{A.2}$$

and $p_j = Z q_j$. In that case, since $\tilde{H}_\mu(x_j, \lambda) \succ 0$, $Z$ is full rank and $Z^T g_j \neq 0$, it follows that $q_j \neq 0$ and

$$g_j^T p_j = g_j^T Z q_j = -q_j^T Z^T \tilde{H}_\mu(x_j, \lambda) Z q_j < 0, \tag{A.3}$$

which is a contradiction. □

Since $\varphi(x)$ is non-negative for all $x$, it is clear that for any given $\mu$ and $\lambda$, the function $L_\mu(\cdot, \lambda)$ is bounded below. From this, Lemma A.0.1, and the continuity of the functions involved, it can be shown that there exists a step length that satisfies the strong Wolfe conditions (3.5) and (3.6), provided that $c_1$ and $c_2$ satisfy $0 < c_1 < c_2 < 1$ [67]. From this, the following lemma is obtained.

**Lemma A.0.2.** *For any given $\mu > 0$ and vector $\lambda$, the sequence $\{L_\mu(x_j, \lambda)\}_{j\in\mathbb{Z}_+}$ is monotonically non-increasing and convergent, where $\{x_j\}_{j\in\mathbb{Z}_+}$ are the inner iterates generated during the solution of subproblem* (3.36).

*Proof.* Let $h_j$ and $g_j$ denote $L_\mu(x_j, \lambda)$ and $\nabla L_\mu(x_j, \lambda)$, respectively, for each $j$. From Lemma A.0.1, for each $j$ such that $Z^T g_j \neq 0$, it holds that $g_j^T p_j < 0$. Hence, for each such $j$, the line search procedure gives $\alpha_j > 0$ such that

$$h_{j+1} \leq h_j + c_1 \alpha_j g_j^T p_j. \tag{A.4}$$

This implies that $h_{j+1} \leq h_j$ for all $j$ and hence that the sequence $\{h_j\}$ is monotonically non-increasing. Since $L_\mu(\cdot, \lambda)$ is bounded below, the sequence $\{h_j\}$ is bounded and hence also convergent. If $Z^T g_j = 0$ for some $j$, the process clearly stops and there is nothing more to check. □

Using Lemma A.0.2 and exploiting the properties of the objective function $\varphi$, a compactness result can be obtained for the inner iterates.

**Lemma A.0.3.** *For any $\mu > 0$ and vector $\lambda$, the sequence of inner iterates $\{x_j\}_{j \in \mathbb{Z}_+}$ generated during the solution of subproblem* (3.36) *lies in a compact set.*

*Proof.* From Lemma A.0.2, there exist $N_1$, $N_2 > 0$ such that

$$N_1 \leq \mu\varphi_j - \mu\lambda^T c_j + \frac{1}{2}\|c_j\|_2^2 \leq N_2 \tag{A.5}$$

for all $j$. This implies that the sequence $\{\varphi_j\}$ is bounded. Since $\varphi$ is non-negative, quadratic and strongly convex, it can be expressed as

$$\varphi(x) = (x - a)^T A(x - a) + b, \tag{A.6}$$

where $a$ is some vector, $A$ is a positive definite matrix, and $b$ is a non-negative scalar. Since

$$\begin{aligned} \|x_j\|_2 &\leq \|x_j - a\|_2 + \|a\|_2 && \text{(A.7)} \\ &= \|A^{-1/2}A^{1/2}(x_j - a)\|_2 + \|a\|_2 && \text{(A.8)} \\ &\leq \|A^{-1/2}\|_2\|A^{1/2}(x_j - a)\|_2 + \|a\|_2 && \text{(A.9)} \\ &\leq \|A^{-1/2}\|_2\varphi_j^{1/2} + \|a\|_2 && \text{(A.10)} \end{aligned}$$

for all $j$, it follows that $\{x_j\}$ is a bounded sequence and hence that it lies in a compact set. □

With Lemmas A.0.2 and A.0.3, the following theorem is obtained.

**Theorem A.0.4.** *For any given $\mu > 0$ and vector $\lambda$, if $Z^T \nabla L_\mu(x_j, \lambda) \neq 0$ for all $j \in \mathbb{Z}_+$, the iterates $\{x_j\}_{j \in \mathbb{Z}_+}$ generated during the solution of subproblem* (3.36) *satisfy*

$$\lim_{j \to \infty} \frac{\nabla L_\mu(x_j, \lambda)^T p_j}{\|q_j\|_2} = 0, \tag{A.11}$$

*where* $p_j = Zq_j$.

*Proof.* Let $h_j$ and $g_j$ denote $L_\mu(x_j, \lambda)$ and $\nabla L_\mu(x_j, \lambda)$, respectively, for all $j$, and suppose that (A.11) does not hold. Then, there exists an $\epsilon > 0$ and a countably infinite set $\mathcal{I} \subset \mathbb{Z}_+$ such that for all $j \in \mathcal{I}$,

$$-\frac{g_j^T p_j}{\|q_j\|_2} \geq \epsilon. \tag{A.12}$$

From condition (3.5) of the line search, for all such $j \in \mathcal{I}$,

$$h_j - h_{j+1} \geq -c_1 \alpha_j g_j^T p_j \tag{A.13}$$

$$= -c_1 \alpha_j \|q_j\|_2 \left( \frac{g_j^T p_j}{\|q_j\|_2} \right) \tag{A.14}$$

$$\geq c_1 \alpha_j \|q_j\|_2 \epsilon. \tag{A.15}$$

From Lemma A.0.2, $h_j - h_{j+1} \to 0$ as $j \to \infty$ so (A.15) implies that the sequence $\{\alpha_j \|q_j\|_2\}_{j \in \mathcal{I}}$ converges to 0. From condition (3.6) of the line search, for all $j \in \mathcal{I}$,

$$\left(g_{j+1} - g_j\right)^T Z q_j \geq -(1 - c_2) g_j^T p_j. \tag{A.16}$$

From this, the Cauchy-Schwarz inequality and (A.12), it follows that for all $j \in \mathcal{I}$,

$$\frac{\|Z^T g_{j+1} - Z^T g_j\|_2}{1 - c_2} \geq -\frac{g_j^T p_j}{\|q_j\|_2} \geq \epsilon. \tag{A.17}$$

From Lemma A.0.3, the sequence $\{x_j\}$ lies in some compact set $\mathcal{K}$. Hence, the continuous function $Z^T \nabla L_\mu(\cdot, \lambda)$ is actually uniformly continuous in $\mathcal{K}$. This implies that there exists a $\delta$ such that for all $y$ and $z \in \mathcal{K}$ such that $\|y - z\|_2 < \delta$,

$$\|Z^T \nabla L_\mu(y, \lambda) - Z^T \nabla L_\mu(z, \lambda)\|_2 < \frac{\epsilon(1 - c_2)}{2}. \tag{A.18}$$

Now, since $\{\alpha_j \|q_j\|_2\}_{j \in \mathcal{I}}$ converges to zero, so does $\{\alpha_j \|p_j\|_2\}_{j \in \mathcal{I}}$ and hence there exists an $l \in \mathcal{I}$ such that

$$\alpha_l \|p_l\|_2 < \delta. \tag{A.19}$$

For such $l$,

$$\|x_{l+1} - x_l\|_2 = \alpha_l \|p_l\|_2 < \delta \tag{A.20}$$

so

$$\frac{\|Z^T g_{l+1} - Z^T g_l\|_2}{1 - c_2} < \frac{\epsilon}{2}, \tag{A.21}$$

which contradicts (A.17). Therefore, (A.11) must hold. □

An important property of the Hessian approximation used for computing the search directions is now shown. This property allows showing that the search directions used by the algorithm are not only descent directions, as shown in Lemma A.0.1, but sufficient descent directions.

**Lemma A.0.5.** *For any given $\mu > 0$ and vector $\lambda$, there exists a $\eta > 0$ such that $\tilde{H}_\mu(x_j, \lambda)$, as defined in* (3.42)*, satisfies*

$$\kappa(Z^T \tilde{H}_\mu(x_j) Z) < \eta \tag{A.22}$$

*for all inner iterations $j \in \mathbb{Z}_+$, where $\kappa(\cdot)$ gives the condition number of its argument.*

*Proof.* From Lemma A.0.3, $\{x_j\}$ lies in some compact set $\mathcal{K}$. Since, $\|\cdot\|_2$ and $Z^T J^T J Z$ are continuous functions, the image of $\mathcal{K}$ under their composition is compact. Hence, there exists a $\sigma > 0$ such that for each $j$,

$$\lambda_{\max}\big(Z^T J_j^T J_j Z\big) = \|Z^T J_j^T J_j Z\|_2 < \sigma, \tag{A.23}$$

where $\lambda_{\max}(\cdot)$ gives the largest eigenvalue of its argument. Letting $A = Z^T \nabla^2 \varphi_j Z$, which is positive definite and independent of $x_j$, and $B_j = Z^T J_j^T J_j Z$, which is positive semi-definite, the following inequalities hold:

$$\lambda_{\min}(\mu A + B_j) = \inf_{\|v\|_2=1} v^T(\mu A + B_j)v \tag{A.24}$$

$$\geq \mu \inf_{\|v\|_2=1} v^T A v + \inf_{\|v\|_2=1} v^T B_j v \tag{A.25}$$

$$= \mu\lambda_{\min}(A) + \lambda_{\min}(B_j) \tag{A.26}$$

$$\geq \mu\lambda_{\min}(A) > 0, \tag{A.27}$$

where $\lambda_{\min}(\cdot)$ gives the smallest eigenvalue of its argument. Similarly,

$$\lambda_{\max}(\mu A + B_j) = \sup_{\|v\|_2=1} v^T(\mu A + B_j)v \tag{A.28}$$

$$\leq \mu \sup_{\|v\|_2=1} v^T A v + \sup_{\|v\|_2=1} v^T B_j v \tag{A.29}$$

$$= \mu\lambda_{\max}(A) + \lambda_{\max}(B_j) \tag{A.30}$$

$$< \mu\lambda_{\max}(A) + \sigma. \tag{A.31}$$

Letting $\tilde{H}_j$ denote $\tilde{H}_\mu(x_j, \lambda)$, these inequalities imply that for all $j$,

$$\kappa(Z^T\tilde{H}_j Z) = \frac{\lambda_{\max}(Z^T\tilde{H}_j Z)}{\lambda_{\min}(Z^T\tilde{H}_j Z)} = \frac{\lambda_{\max}(\mu A + B_j)}{\lambda_{\min}(\mu A + B_j)} < \frac{\mu\lambda_{\max}(A) + \sigma}{\mu\lambda_{\min}(A)}. \tag{A.32}$$

□

**Lemma A.0.6.** *For any given $\mu > 0$ and vector $\lambda$, there exists a $\rho > 0$ such that the search directions $\{p_j\}_{j\in\mathbb{Z}_+}$ computed during the solution of subproblem* (3.36) *satisfy*

$$-\frac{\nabla L_\mu(x_j, \lambda)^T p_j}{\|Z^T\nabla L_\mu(x_j, \lambda)\|_2\|q_j\|_2} > \varrho, \tag{A.33}$$

*for all $j \in \mathbb{Z}_+$ such that $Z^T\nabla L_\mu(x_j, \lambda) \neq 0$, where $p_j = Zq_j$.*

*Proof.* From Lemma A.0.5, there exists an $\eta > 0$ such that for all $j$,

$$\kappa(Z^T\tilde{H}_j Z) < \eta, \tag{A.34}$$

where $\tilde{H}_j$ denotes $\tilde{H}_\mu(x_j, \lambda)$. Letting $g_j$ and $C_j$ denote $\nabla L_\mu(x_j, \lambda)$ and $Z^T\tilde{H}_j Z$, respectively,

it follows that for all $j$ such that $Z^T g_j \neq 0$ and $q_j$ was computed using $C_j$,

$$-\frac{g_j^T p_j}{\|Z^T g_j\|_2 \|q_j\|_2} = \frac{g_j^T Z C_j^{-1} Z^T g_j}{\|Z^T g_j\|_2 \|C_j^{-1} Z^T g_j\|_2} \tag{A.35}$$

$$\geq \frac{\lambda_{\min}(C_j^{-1})}{\lambda_{\max}(C_j^{-1})} \tag{A.36}$$

$$= \frac{\lambda_{\min}(C_j)}{\lambda_{\max}(C_j)} \tag{A.37}$$

$$= \frac{1}{\kappa(C_j)} > \frac{1}{\eta}. \tag{A.38}$$

For $j$ such that $Z^T g_j \neq 0$ and $q_j$ was not computed using $C_j$, the exact Hessian was used and $p_j$ must have passed condition (3.48). Hence, letting $\varrho = \min\{\xi, 1/\eta\}$ completes the proof, where $\xi$ is the predefined small positive scalar described in Section 3.5.3. □

With Lemmas A.0.6 and Theorem A.0.4, it can be proved that the vPF algorithm always solves each subproblem (3.36) to the required accuracy.

**Theorem A.0.7.** *For any given $\mu > 0$ and vector $\lambda$, the iterates $\{x_j\}_{j \in \mathbb{Z}_+}$ generated during the solution of subproblem* (3.36) *satisfy either*

$$Z^T \nabla L_\mu(x_l, \lambda) = 0 \tag{A.39}$$

*for some $l \in \mathbb{Z}_+$, or*

$$\lim_{j \to \infty} Z^T \nabla L_\mu(x_j, \lambda) = 0. \tag{A.40}$$

*Proof.* If $l$ exists such that $Z^T \nabla L_\mu(x_l, \lambda) = 0$, the theorem is proved. Otherwise, Lemma A.0.6 gives that

$$-\frac{g_j^T p_j}{\|q_j\|_2} > \varrho \|Z^T g_j\|_2 \tag{A.41}$$

for all $j$, where $g_j$ denotes $\nabla L_\mu(x_j, \lambda)$. From Theorem A.0.4, the left-hand side of this inequality goes to zero as $j$ goes to infinity. Hence,

$$\lim_{j \to \infty} Z^T g_j = 0. \tag{A.42}$$

□

**Corollary A.0.7.1.** *Given a penalty parameter $\mu > 0$, vector $\lambda$ and any subproblem optimality tolerance $\delta > 0$, the inner level of the vPF algorithm always finds $\bar{x}$ such that*

$$\|Z^T \nabla L_\mu(\bar{x}, \lambda)\|_\infty < \delta \tag{A.43}$$

*in a finite number of iterations.*

*Proof.* This results follows immediately from Theorem A.0.7. □

Properties of the outer iterates generated by the vPF algorithm are now proved. First, it is shown that the sequence of values of the Augmented Lagrangian function associated with the outer iterates is bounded.

**Lemma A.0.8.** *Given any initial point $x_0$, initial penalty parameter $\mu_0 > 0$, and initial vector of Lagrange multiplier estimates $\lambda_0$, if $\|c(x_k)\|_\infty \geq \epsilon_f$ for all $k \in \mathbb{Z}_+$, the sequence $\{L_{\mu_k}(x_k, \lambda_k)\}_{k \in \mathbb{Z}_+}$ is bounded, where $\{x_k\}_{k \in \mathbb{Z}_+}$ are the outer iterates generated by the vPF algorithm.*

*Proof.* Suppose $\|c_k\|_\infty \geq \epsilon_f$ for all $k$. Hence, the vPF algorithm never terminates and generates an infinite sequence of outer iterates $x_k$. By the equivalency of norms in finite dimensional spaces, there exists some $\epsilon$ such that $\|c_k\|_2 \geq \epsilon$ for all $k$. During outer iteration $k$, the inner level of the algorithm takes $x_k$, $\mu_k$ and $\lambda_k$ and generates the next outer iterate $x_{k+1}$. By Lemma A.0.2, it must be that

$$L_{\mu_k}(x_{k+1}, \lambda_k) \leq L_{\mu_k}(x_k, \lambda_k), \tag{A.44}$$

for all $k$. Equivalently,

$$\mu_k \varphi_{k+1} - \mu_k \lambda_k^T c_{k+1} + \frac{1}{2}\|c_{k+1}\|_2^2 \leq \mu_k \varphi_k - \mu_k \lambda_k^T c_k + \frac{1}{2}\|c_k\|_2^2. \tag{A.45}$$

By construction (last paragraph of Section 3.5.4), there exists some $M > 0$ such that $\|\lambda_k\|_2 \leq M$ for all $k$. Hence, using the Cauchy-Schwarz inequality, the fact that $\mu_k$ is

non-increasing, and the non-negativity of $\varphi$, it follows that

$$\mu_k\varphi_{k+1} - \mu_k M\|c_{k+1}\|_2 + \frac{1}{2}\|c_{k+1}\|_2^2 \leq \mu_k\varphi_k + \mu_k M\|c_k\|_2 + \frac{1}{2}\|c_k\|_2^2 \tag{A.46}$$

$$\mu_k\varphi_{k+1} + \frac{1}{2}\|c_{k+1}\|_2^2\Big(1 - \frac{2\mu_k M}{\|c_{k+1}\|_2}\Big) \leq \mu_k\varphi_k + \frac{1}{2}\|c_k\|_2^2\Big(1 + \frac{2\mu_k M}{\|c_k\|_2}\Big) \tag{A.47}$$

$$\mu_{k+1}\varphi_{k+1} + \frac{1}{2}\|c_{k+1}\|_2^2\Big(1 - \frac{2\mu_k M}{\epsilon}\Big) \leq \mu_k\varphi_k + \frac{1}{2}\|c_k\|_2^2\Big(1 + \frac{2\mu_k M}{\epsilon}\Big) \tag{A.48}$$

for all $k$. Since $\mu_k \downarrow 0$ as $k \to \infty$, there is some $K \in \mathbb{N}$ such that for all $k \geq K$,

$$1/2 < 1 - 2\mu_k M/\epsilon < 1. \tag{A.49}$$

For such $k$,

$$\Big(1 - \frac{2\mu_k M}{\epsilon}\Big)\Big(\mu_{k+1}\varphi_{k+1} + \frac{1}{2}\|c_{k+1}\|_2^2\Big) \leq \Big(1 + \frac{2\mu_k M}{\epsilon}\Big)\Big(\mu_k\varphi_k + \frac{1}{2}\|c_k\|_2^2\Big), \tag{A.50}$$

or

$$\mu_{k+1}\varphi_{k+1} + \frac{1}{2}\|c_{k+1}\|_2^2 \leq \eta_k\Big(\mu_k\varphi_k + \frac{1}{2}\|c_k\|_2^2\Big), \tag{A.51}$$

where

$$\eta_k := \frac{1 + \frac{2\mu_k M}{\epsilon}}{1 - \frac{2\mu_k M}{\epsilon}}. \tag{A.52}$$

It follows that for $k > K$,

$$\mu_k\varphi_k + \frac{1}{2}\|c_k\|_2^2 \leq \Big(\mu_K\varphi_K + \frac{1}{2}\|c_K\|_2^2\Big)\prod_{i=K}^{k-1}\eta_i. \tag{A.53}$$

Now, for $k \geq K$, it holds that

$$\log\eta_k = \log\Big(1 + \frac{2\mu_k M}{\epsilon}\Big) - \log\Big(1 - \frac{2\mu_k M}{\epsilon}\Big). \tag{A.54}$$

Hence, the bounds

$$1 - \frac{1}{y} \leq \log y \leq y - 1, \tag{A.55}$$

for all $y \in \mathbb{R}_{++}$, imply that

$$\log\eta_k \leq \mu_k\frac{2M}{\epsilon} + \mu_K\frac{4M}{\epsilon} = \mu_k\frac{6M}{\epsilon}. \tag{A.56}$$

Since, $\mu_k \leq \beta_l^k \mu_0$, where $\beta_l \in (0,1)$, there exists some $N > 0$ such that for all $k > K$,

$$\log \prod_{i=N}^{k-1} \eta_i = \sum_{i=N}^{k-1} \log \eta_i \leq \frac{6M\mu_0}{\epsilon} \sum_{i=N}^{\infty} \beta_l^i < N. \tag{A.57}$$

This implies that for all $k > K$,

$$\mu_k \varphi_k + \frac{1}{2}\|c_k\|_2^2 \leq \Big(\mu_K \varphi_K + \frac{1}{2}\|c_K\|_2^2\Big) e^N \tag{A.58}$$

and hence that $\{L_{\mu_k}(x_k, \lambda_k)\}$ is a bounded sequence. □

With the boundedness result of Lemma A.0.8, the boundedness of other important quantities can be shown.

**Lemma A.0.9.** *If the vPF algorithm never terminates and Assumption 1 holds, the voltage magnitudes $\{v_i\}_{i\in[n]\setminus\{s\}}$, generator reactive powers $\{Q_i^g\}_{i\in\mathcal{R}}$, and regulated voltage magnitude deviations $\{y_i\}_{i\in\mathcal{R}}$ and $\{z_i\}_{i\in\mathcal{R}}$ associated with each of the outer iterates $\{x_k\}_{k\in\mathbb{Z}_+}$ are uniformly bounded. Moreover, the sequence $\{J_k\}_{k\in\mathbb{Z}_+}$ of Jacobian matrices is bounded.*

*Proof.* If the vPF algorithm never terminates, $\|c_k\|_\infty \geq \epsilon_f$ for all $k$. Then, by Lemma A.0.8, the sequence $\{L_{\mu_k}(x_k, \lambda_k)\}$ is bounded. From the definition of $L_\mu$, the sequence $\{\|c_k\|_2\}$ is bounded. Then, from the definition of $c$ (Section 3.5), $\{\|f_k\|_2\}$ must be bounded. Hence, Assumption 1 gives that the voltage magnitudes $\{v_i\}$ associated with the outer iterates are uniformly bounded. The uniform boundedness of the generator reactive powers $\{Q_i^g\}$ then follows from the boundedness of reactive power mismatches and voltage magnitudes, and from (2.40). Similarly, from the definition of $c$, $\{\|\Phi_k\|_2\}$ is also bounded. The uniform boundedness of $\{y_i\}$ and $\{z_i\}$ then follows from the uniform boundedness of $\{Q_i^g\}$, the fact that $\{\Phi_k\}$ is a bounded sequence, and from (3.23) and (3.24). Lastly, the boundedness of $\{J_k\}$ is implied from the uniform boundedness of $\{v_i\}$, $\{Q_i^g\}$, $\{y_i\}$ and $\{z_i\}$, and the fact that angles only appear inside sine and cosine functions. □

**Lemma A.0.10.** *If the vPF algorithm never terminates, the sequence $\{\mu_k \nabla \varphi_k\}_{k\in\mathbb{Z}_+}$ satisfies $\mu_k \nabla \varphi_k \to 0$ as $k \to \infty$, where $\{x_k\}_{k\in\mathbb{Z}_+}$ are the outer iterates.*

*Proof.* If the vPF algorithm never terminates, $\|c_k\|_\infty \geq \epsilon_f$ for all $k$. Then, by Lemma A.0.8, the sequence $\{L_{\mu_k}(x_k, \lambda_k)\}$ is bounded. From the definition of $L_\mu$, the sequence $\{\mu_k \varphi_k\}$ is

bounded. Since $\varphi$ is non-negative, quadratic and strongly convex, it can be expressed as

$$\varphi(x) = (x-a)^T A(x-a) + b, \tag{A.59}$$

where $a$ is some vector, $A$ is a positive definite matrix, and $b$ is a non-negative scalar. Hence, $\{\mu_k \varphi_k\}$ bounded implies that there exists some $M > 0$ such that for all $k$,

$$\|A^{1/2}(x-a)\|_2 \leq \sqrt{\frac{M}{\mu_k}}. \tag{A.60}$$

It follows that

$$\mu_k \|\nabla \varphi_k\|_2 = \mu_k \|2A(x-a)\|_2 \tag{A.61}$$

$$\leq \mu_k 2 \|A^{1/2}\|_2 \|A^{1/2}(x-a)\|_2 \tag{A.62}$$

$$\leq 2\sqrt{\mu_k M}\|A^{1/2}\|_2. \tag{A.63}$$

The result then follows from the fact that $\mu_k \to 0$ as $k \to \infty$. □

The main result about the convergence of the vPF algorithm can now be proved.

**Theorem A.0.11.** *Under Assumption 1, the vPF algorithm either finds during some iteration $l \in \mathbb{Z}_+$ an outer iterate $x_l$ that is feasible according to the required tolerance,* i.e.*, that satisfies $\|c_l\|_\infty < \epsilon_f$, or it generates a sequence of outer iterates $\{x_k\}_{k\in\mathbb{Z}_+}$ that satisfies*

$$\lim_{k\to\infty} Z^T J_k^T c_k = 0. \tag{A.64}$$

*From $\{x_k\}_{k\in\mathbb{Z}_+}$, a bounded sequence $\{\tilde{x}_k\}_{k\in\mathbb{Z}_+}$ can be constructed such that*

$$J(\tilde{x}_k)^T c(\tilde{x}_k) = J_k^T c_k, \tag{A.65}$$

*for all $k$, by translating the voltage angles of $\{x_k\}_{k\in\mathbb{Z}_+}$ to $[-\pi, \pi]$. This new sequence is guaranteed to have a limit point $x^*$ that satisfies*

$$Z^T J(x^*)^T c(x^*) = 0. \tag{A.66}$$

*Proof.* Suppose that Assumption 1 holds. If for some $l \in \mathbb{Z}_+$, the outer iterate $x_l$ satisfies $\|c(x_l)\|_\infty < \epsilon_f$, the result holds. Otherwise the vPF algorithm does not terminate and it

generates a sequence of outer iterates $\{x_k\}$ such that

$$\|Z^T \nabla L_{\mu_k}(x_{k+1}, \lambda_k)\|_\infty < \delta_k, \tag{A.67}$$

where $\delta_k \downarrow 0$. Hence,

$$\nu_k := \|Z^T \nabla L_{\mu_k}(x_{k+1}, \lambda_k)\|_2 \to 0 \tag{A.68}$$

as $k \to \infty$. Now, for all $k$,

$$\|Z^T J_{k+1}^T c_{k+1}\|_2 = \|Z^T \nabla L_{\mu_k}(x_{k+1}, \lambda_k) - \mu_k Z^T \nabla \varphi_{k+1} + \mu_k Z^T J_{k+1}^T \lambda_k\|_2 \tag{A.69}$$

$$\leq \nu_k + \mu_k \|Z\|_2 \|\nabla \varphi_{k+1}\|_2 + \mu_k \|Z\|_2 \|J_{k+1}\|_2 \|\lambda_k\|_2 \tag{A.70}$$

$$\leq \nu_k + \frac{\mu_{k+1}}{\beta_s} \|Z\|_2 \|\nabla \varphi_{k+1}\|_2 + \mu_k \|Z\|_2 \|J_{k+1}\|_2 \|\lambda_k\|_2, \tag{A.71}$$

where the last inequality follow from $\mu_{k+1} \geq \beta_s \mu_k$. Since $\{\lambda_k\}$ is bounded by construction and $\{J_k\}$ is bounded from Lemma A.0.9, the last term on the right hand side of (A.71) goes to zero as $k \to 0$. From Lemma A.0.10, the middle term goes to zero as $k \to 0$. Hence, the entire right hand side of (A.71) goes to zero and (A.64) holds.

Since the voltage angles appear in $c$ and $J$ only inside sine and cosine functions, they can be translated to lie inside $[-\pi, \pi]$ by adding or subtracting multiplies of $2\pi$ to create a sequence $\{\tilde{x}_k\}$ with uniformly bounded angles that satisfies

$$J(\tilde{x}_k)^T c(\tilde{x}_k) = J_k^T c_k, \tag{A.72}$$

for all $k$. The sequence $\{\tilde{x}_k\}$ hence lies in a compact set and has a limit point $x^*$ that satisfies (A.66). □

This completes the proof of the convergence of the vPF algorithm. It has been shown that this algorithm either terminates with a feasible point according to the required tolerance, or that it generates a sequence of iterates that has a limit point that is a stationary point of the function $c^T c$ (restricted to the set of $x$ such that $Ax = b$).

# Appendix B

# Lossy Networks

In Section 3.5.5, *lossy networks* were introduced as power networks for which active power losses are positive for any set of complex bus voltages that are not all zero. In this section, a simple characterization of these networks based on the nodal admittance matrix is proved, and also that for these networks, bounded power mismatches implies bounded voltage magnitudes.

**Lemma B.0.12.** *A power network is a lossy network if an only if the Hermitian matrix* $\tilde{G}$*, as defined in* (2.35)*, is positive definite.*

*Proof.* Let $\{P_k^g\}_{k\in[n]}$ be the active powers injected by generators and $\{P_k^l\}_{k\in[n]}$ the active powers consumed by loads at each bus of the network. It is known that the total active power injected into the system must equal the total active power consumed by loads plus the total active power lost in the system. Hence

$$\sum_{k\in[n]} P_k^g - \sum_{k\in[n]} P_k^l = L, \tag{B.1}$$

where $L$ denotes the total active power losses. For each $k \in [n]$, let $w_k$ be the complex voltage at bus $k$, *i.e.*,

$$w_k := v_k e^{j\theta_k}. \tag{B.2}$$

Then, from (2.38),

$$\sum_{k\in[n]} P_k^g - \sum_{k\in[n]} P_k^l = \Re\left\{ \sum_{k\in[n]} \sum_{m\in[n]} v_k v_m Y_{km}^* e^{j(\theta_k-\theta_m-\phi_{km})} \right\} \tag{B.3}$$

$$= \Re\left\{ \sum_{k\in[n]} \sum_{m\in[n]} w_k w_m^* \tilde{Y}_{km}^* \right\} \tag{B.4}$$

$$= \Re\{w^* \tilde{Y}^* w\}, \tag{B.5}$$

where $\tilde{Y}$ is as defined in (2.34), $\Re\{\cdot\}$ gives the real part of its argument, and $*$ denotes conjugate transpose. From (2.35) and (2.36), it follow that

$$w^* \tilde{Y}^* w = w^* (\tilde{G} + j\tilde{B})^* w \tag{B.6}$$

$$= w^* \tilde{G}^* w - j w^* \tilde{B}^* w. \tag{B.7}$$

Since both $\tilde{G}$ and $\tilde{B}$ are Hermitian, $\tilde{G} = \tilde{G}^*$, $\tilde{B} = \tilde{B}^*$, and hence that $w^*\tilde{G}^*w$ and $w^*\tilde{B}^*w$ are real. Therefore,

$$\sum_{k\in[n]} P_k^g - \sum_{k\in[n]} P_k^l = w^* \tilde{G} w, \tag{B.8}$$

so $L = w^*\tilde{G}w$ and that for any $w \neq 0$, $L > 0$ if and only if $\tilde{G} \succ 0$. □

**Lemma B.0.13.** *For all lossy networks and for all $M$, the set $\{x \mid f(x)^T f(x) \leq M\}$ has bounded voltage magnitudes, where $x$ and $f$ are the vector of power flow variables and the vector-valued function of power mismatches, respectively, as defined in Section 3.4.*

*Proof.* Suppose the network is lossy and let $M > 0$. Let $x$ be such that $f(x)^T f(x) \leq M$ and $w$ the corresponding vector of complex bus voltages. Also, let $\mathcal{I}$ denote the set $[n] \setminus \{s\}$, *i.e.*, the set of all buses except the slack. Then, using results derived in Lemma B.0.12,

$$\Re\left\{ \sum_{k\in\mathcal{I}} \sum_{m\in[n]} w_k w_m^* \tilde{Y}_{km}^* \right\} = \Re\left\{ \sum_{k\in[n]} \sum_{m\in[n]} w_k w_m^* \tilde{Y}_{km}^* \right\} - \Re\left\{ \sum_{m\in[n]} w_s w_m^* \tilde{Y}_{sm}^* \right\} \tag{B.9}$$

$$= w^* \tilde{G} w - \Re\{w_s w^* \eta\}, \tag{B.10}$$

where $\eta^*$ is the $s$-th row of $\tilde{Y}$, and $*$ denotes conjugate transpose. Since $w_s$ is a complex constant in the power flow problem, let $\nu$ to denote the constant complex vector $w_s\eta$. From

(B.10) and the definition of $f$,

$$w^*\tilde{G}w - \Re\{w^*\nu\} = \sum_{k\in\mathcal{I}} \left(P_k^g - P_k^l\right) - \sum_{i\in\mathcal{A}} f_i(x), \tag{B.11}$$

where $\mathcal{A}$ is the set of indices of the vector $f(x)$ that correspond to active power mismatches at buses $k \in [n] \setminus \{s\}$. Hence,

$$\lambda_{\min}(\tilde{G})\|w\|^2 - \|w\|\|\nu\| \leq \sum_{k\in\mathcal{I}} \left|P_k^g - P_k^l\right| + \|f(x)\|_1, \tag{B.12}$$

where $\|\cdot\|$ is the norm induced by the inner product $\langle a, b\rangle := b^*a$, for any complex vectors $a$ and $b$. Since $P_k^g$ and $P_k^l$ are constants (independent of $x$) for $k \in \mathcal{I}$, the quantity

$$N := \sum_{k\in\mathcal{I}} \left|P_k^g - P_k^l\right| \tag{B.13}$$

is also a constant. Then, using the inequalities

$$\|f\|_1 \leq m\|f\|_\infty \leq m\sqrt{f^T f} \leq m\sqrt{M}, \tag{B.14}$$

where $m$ is the dimension of $f(x)$, it follows that

$$\lambda_{\min}(\tilde{G})\|w\|^2 - \|w\|\|\nu\| \leq N + m\sqrt{M}. \tag{B.15}$$

From Lemma B.0.12, $\lambda_{\min}(\tilde{G}) > 0$. From this and the fact that $N$ and $M$ are independent of $x$, it can be concluded that the set

$$\{w \mid f(x)^T f(x) \leq M\} \tag{B.16}$$

is bounded. □

# Appendix C

# Bound Constraint Function

Let the function $\psi$ be defined by

$$\psi(a) := a + b - \sqrt{a^2 + b^2 + \epsilon^2}, \tag{C.1}$$

where $b$ and $\epsilon \in \mathbb{R}_{++}$.

**Lemma C.0.14.** *The derivative of $\psi$ satisfies*

$$0 < \psi'(a) < 2, \tag{C.2}$$

*for all $a \in \mathbb{R}$, and hence $\psi$ is monotonically strictly increasing.*

*Proof.* The derivative of $\psi$ is given by

$$\psi'(a) = 1 - \frac{a}{\sqrt{a^2 + b^2 + \epsilon^2}}. \tag{C.3}$$

Since $b$ and $\epsilon$ are positive, it follows that

$$\sqrt{a^2 + b^2 + \epsilon^2} > \sqrt{a^2} = |a|. \tag{C.4}$$

Hence,

$$-1 < -\frac{a}{\sqrt{a^2 + b^2 + \epsilon^2}} < 1. \tag{C.5}$$

Adding 1 to these inequalities gives (C.2). □

**Lemma C.0.15.** *The function $\psi$ satisfies*

$$\lim_{a\to-\infty} \psi(a) = -\infty \tag{C.6}$$

$$\lim_{a\to\infty} \psi(a) = b, \tag{C.7}$$

*and*

$$-\infty < \psi(a) < b, \tag{C.8}$$

*for all $a \in \mathbb{R}$.*

*Proof.* First, the limit (C.6) follows easily from the fact that $-\sqrt{a^2+b^2+\epsilon^2} \to -\infty$ as $a \to -\infty$. To show the other limit, consider $a > 0$ and let $f$ and $g$ be such that

$$f(a) = 1 - \sqrt{1 + \frac{b^2+\epsilon^2}{a^2}} \quad \text{and} \quad g(a) = \frac{1}{a}. \tag{C.9}$$

Then

$$\psi(a) = a + b - \sqrt{a^2+b^2+\epsilon^2} \tag{C.10}$$

$$= b + \frac{1 - \sqrt{1 + \frac{b^2+\epsilon^2}{a^2}}}{\frac{1}{a}} \tag{C.11}$$

$$= b + \frac{f(a)}{g(a)}. \tag{C.12}$$

Clearly, $f(a) \to 0$ and $g(a) \to 0$ as $a \to \infty$. Also,

$$\frac{f'(a)}{g'(a)} = -\frac{b^2+\epsilon^2}{\sqrt{a^2+b^2+\epsilon^2}} \to 0 \tag{C.13}$$

as $a \to \infty$. Hence, by L'Hôpital's Rule, it follows that

$$\lim_{a\to\infty} \psi(a) = b + \lim_{a\to\infty} \frac{f(a)}{g(a)} = b + \lim_{a\to\infty} \frac{f'(a)}{g'(a)} = b. \tag{C.14}$$

From Lemma C.0.14, $\psi$ is monotonically strictly increasing and hence (C.8) follows. □

**Lemma C.0.16.** *If $\epsilon \geq b$, the function $\psi$ satisfies*

$$|\psi(a)| \leq \epsilon \quad \iff \quad -\frac{b}{1+\frac{b}{\epsilon}} \leq a < \infty. \tag{C.15}$$

*Proof.* If $\epsilon \geq b$, if follows from Lemma C.0.15 that

$$|\psi(a)| \leq \epsilon \quad \iff \quad -\epsilon \leq \psi(a). \tag{C.16}$$

From Lemma C.0.14, $\psi$ is monotonically strictly increasing so

$$-\epsilon \leq \psi(a) \quad \iff \quad \psi^{-1}(-\epsilon) \leq a. \tag{C.17}$$

Since

$$\begin{aligned}
\psi\left(-\frac{b}{1+\frac{b}{\epsilon}}\right) &= \psi\left(-\frac{b\epsilon}{b+\epsilon}\right) && \text{(C.18)}\\
&= -\frac{b\epsilon}{b+\epsilon} + b - \frac{1}{b+\epsilon}\sqrt{b^2\epsilon^2 + b^2(b+\epsilon)^2 + \epsilon^2(b+\epsilon)^2} && \text{(C.19)}\\
&= \frac{b^2}{b+\epsilon} - \frac{1}{b+\epsilon}\sqrt{b^4 + 2b^2\epsilon(b+\epsilon) + \epsilon^2(b+\epsilon)^2} && \text{(C.20)}\\
&= \frac{b^2}{b+\epsilon} - \frac{1}{b+\epsilon}\sqrt{(b^2 + \epsilon(b+\epsilon))^2} && \text{(C.21)}\\
&= \frac{b^2}{b+\epsilon} - \frac{b^2 + \epsilon(b+\epsilon)}{b+\epsilon} && \text{(C.22)}\\
&= -\epsilon, && \text{(C.23)}
\end{aligned}$$

it holds that

$$\psi^{-1}(-\epsilon) = -\frac{b}{1+\frac{b}{\epsilon}}. \tag{C.24}$$

The relation (C.15) follows from this, (C.16) and (C.17). □

**Lemma C.0.17.** *The function $\psi$ satisfies*

$$\psi(a) = 0 \quad \iff \quad a = \frac{\epsilon^2}{2b}. \tag{C.25}$$

*Proof.* The result follows from

$$\psi(u) = 0 \iff a + b = \sqrt{a^2 + b^2 + \epsilon^2} \tag{C.26}$$

$$\iff a^2 + 2ab + b^2 = a^2 + b^2 + \epsilon^2 \tag{C.27}$$

$$\iff 2ab = \epsilon^2 \tag{C.28}$$

$$\iff a = \frac{\epsilon^2}{2b}. \tag{C.29}$$

□

# Bibliography


[1] E. Acha, C.R. Fuerte-Esquivel, H. Ambriz-Pérez, and C. Angeles-Camacho. *FACTS: Modelling and Simulation in Power Networks*. John Wiley & Sons, Inc., 2005.

[2] V. Ajjarapu and C. Christy. The continuation power flow: A tool for steady state voltage stability analysis. *IEEE Transactions on Power Systems*, 7(1):416–423, 1992.

[3] P.R. Amestoy, I.S. Duff, J.Y. L'Excellent, and J. Koster. MUMPS: A general purpose distributed memory sparse solver. In T. Sorevik, F. Manne, A.H. Gebremedhin, and R. Moe, editors, *Applied Parallel Computing. New Paradigms for HPC in Industry and Academia*, volume 1947 of *Lecture Notes in Computer Science*, pages 121–130. Springer Berlin Heidelberg, 2001.

[4] G. Andersson. Modelling and analysis of electric power systems. *ETH Zurich*, September 2008.

[5] J.S. Arora, M.W. Huang, and C.C. Hsieh. Methods for optimization of nonlinear problems with discrete variables: A review. *Structural Optimization*, 8(2-3):69–85, 1994.

[6] U.A. Bakshi. *DC Machines And Synchronous Machines*. Technical Publications, 2007.

[7] M.E. Baran and F.F. Wu. Optimal capacitor placement on radial distribution systems. *IEEE Transactions on Power Delivery*, 4(1):725–734, January 1989.

[8] D.B. Bedoya and C.A Castro. Computation of power systems minimum voltage stability security margins. In *International Conference on Power System Technology*, pages 1–7, October 2006.

[9] H.Y. Benson, D.F. Shanno, and R.J. Vanderbei. Interior-point methods for nonconvex nonlinear programming: Complementarity constraints. *Operations Research and Financial Engineering*, pages 1–20, 2002.

[10] R. Bent, G.L. Toole, and A Berscheid. Transmission network expansion planning with complex power flow models. *IEEE Transactions on Power Systems*, 27(2):904–912, May 2012.

[11] D.P. Bertsekas. Constrained optimization and lagrange multiplier methods. *Computer Science and Applied Mathematics, Boston: Academic Press*, 1, 1982.

[12] P.R. Bijwe and S.M. Kelapure. Nondivergent fast power flow methods. *IEEE Transactions on Power Systems*, 18(2):633–638, May 2003.

[13] C.M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006.

[14] A.M. Bradley. *Algorithms for the Equilibration of Matrices and Their Application to Limited-Memory Quasi-Newton Methods*. PhD thesis, Stanford University, May 2010.

[15] L.M.C. Braz, C.A. Castro, and C.A.F. Murati. A critical evaluation of step size optimization based load flow methods. *IEEE Transactions on Power Systems*, 15(1):202–207, February 2000.

[16] M.B. Cain, R.P. O'Neill, and A. Castillo. History of optimal power flow and formulations. *Federal Energy Regulatory Commission*, 2012.

[17] C.A. Canizares and F.L. Alvarado. Point of collapse and continuation methods for large AC/DC systems. *IEEE Transactions on Power Systems*, 8(1):1–8, 1993.

[18] F. Capitanescu, W. Rosehart, and L. Wehenkel. Optimal power flow computations with constraints limiting the number of control actions. In *IEEE PowerTech Conference, Bucharest*, pages 1–8, June 2009.

[19] F. Capitanescu and T. Van Cutsem. Evaluating bounds on voltage and thermal security margins under power transfer uncertainty. In *Power Systems Computation Conference (PSCC)*, June 2002.

[20] F. Capitanescu and L. Wehenkel. Optimal power flow computations with a limited number of controls allowed to move. *IEEE Transactions on Power Systems*, 25(1):586–587, February 2010.

[21] F. Capitanescu and L. Wehenkel. Sensitivity-based approaches for handling discrete variables in optimal power flow computations. *IEEE Transactions on Power Systems*, 25(4):1780–1789, November 2010.

[22] P. Chen. Hessian matrix vs. GaussNewton Hessian matrix. *SIAM Journal on Numerical Analysis*, 49(4):1417–1435, 2011.

[23] H.D. Chiang, A.J. Flueck, K.S. Shah, and N. Balu. CPFLOW: A practical tool for tracing power system steady-state stationary behavior due to load and generation variations. *IEEE Transactions on Power Systems*, 10(2):623–634, 1995.

[24] H.D. Chiang and H. Li. On-line ATC evaluation for large-scale power systems: Framework and tool. In J.H. Chow, F.F. Wu, and J. Momoh, editors, *Applied Mathematics for Restructured Electric Power Systems*, Power Electronics and Power Systems, pages 63–103. Springer US, 2005.

[25] I.D. Coope and P.F. Renaud. Trace inequalities with applications to orthogonal regression and matrix nearness problems. In *Journal of Inequalities in Pure and Applied Mathematics*, 2009.

[26] M.L. Crow. *Computational Methods for Electric Power Systems*. Electric Power Engineering Series. Taylor & Francis, 2002.

[27] S. Cvijic, P. Feldmann, and M. Hie. Applications of homotopy for solving AC power flow and AC optimal power flow. In *IEEE Power and Energy Society General Meeting*, pages 1–8, 2012.

[28] M.E. El-Hawary. *Electrical Power Systems: Design and Analysis*. IEEE Press Series on Power Engineering. Wiley, 1995.

[29] J. Ellson, E. Gansner, L. Koutsofios, S. North, G. Woodhull, and Lucent Technologies. Graphviz - Open source graph drawing tools. In *Lecture Notes in Computer Science*, pages 483–484. Springer-Verlag, 2001.

[30] U. Eminoglu, M.H. Hocaoglu, and T. Yalcinoz. Transmission line shunt and series compensation with voltage sensitive loads. *International Journal of Electrical Engineering Education*, 46(4):354–369, 2009.

[31] R. Entriken, W. Murray, and T. Tinoco De Rubira. Linear analysis for determining and visualizing critical thermal boundaries of power systems. In *IEEE Conference on Innovative Smart Grid Technologies (to appear)*, November 2015.

[32] J.H. Eto and R.J. Thomas. Computational needs for the next generation electric grid proceedings. *Department of Energy*, 2011.

[33] P.E. Gill, W. Murray, and M.H. Wright. *Practical optimization.* Academic Press, 1981.

[34] L.L. Grigsby. *Power System Stability and Control, Third Edition.* The Electric Power Engineering Handbook. Taylor & Francis, 2012.

[35] J.F. Gutierrez, M.F. Bedrinana, and C.A. Castro. Critical comparison of robust load flow methods for ill-conditioned systems. In *IEEE PowerTech Conference, Trondheim*, pages 1–6, 2011.

[36] Electric Power Research Institute. Application of advanced data processing, mathematical techniques and computing technologies in control centers: Enhancing speed and robustness of power flow computation, December 2012. Technical Update.

[37] Electric Power Research Institute. New technologies and methods to improve computational speed and robustness of power flow analysis, December 2013. Technical Update.

[38] Electric Power Research Institute. Critical operating boundaries: Automated identification and visualization of thermal and voltage limits, December 2014. Technical Update.

[39] Electric Power Research Institute. GPU-based contingency analysis and AC-OPF techniques for operations, December 2014. Technical Update.

[40] S. Iwamoto and Y. Tamura. A load flow calculation method for ill-conditioned power systems. *IEEE Transactions on Power Apparatus and Systems*, PAS-100(4):1736–1743, April 1981.

[41] T. Jiang, H. Jia, Y. Jiang, and J. Zhao. Cutset-angle based wide area thermal security region and its application in china southern power grid. *International Transactions on Electrical Energy Systems*, 2014.

[42] E. Kokiopoulou, J. Chen, and Y. Saad. Trace optimization and eigenproblems in dimension reduction methods. *Numerical Linear Algebra with Applications*, 18(3):565–602, 2011.

[43] N. Krejić, J.M. Martnez, M. Mello, and E.A. Pilotta. Validation of an augmented Lagrangian algorithm with a Gauss-Newton Hessian approximation using a set of hard-spheres problems. *Computational Optimization and Applications*, 16(3):247–263, 2000.

[44] S. Lee, H. Lee, P. Abbeel, and A.Y. Ng. Efficient L1 regularized logistic regression. In *Proceedings of the 21st National Conference on Artificial Intelligence*, 2006.

[45] R.B. Lehoucq, D.C. Sorensen, and C. Yang. *ARPACK Users' Guide: Solution of Large-scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. Software, Environments, Tools. SIAM, 1998.

[46] G. Leoniopoulos. Efficient starting point of load-flow equations. *International Journal of Electrical Power and Energy Systems*, 16(6):419–422, 1994.

[47] S. Leyffer. Complementarity constraints as nonlinear equations: Theory and numerical experience. In *Preprint ANL/MCS-P1054-0603, Mathematics and Computer Science Division, Argonne National Laboratory*, pages 169–208. Springer, 2003.

[48] L. Liu, X. Wang, X. Ding, and H. Chen. A robust approach to optimal power flow with discrete variables. *IEEE Transactions on Power Systems*, 24(3):1182–1190, August 2009.

[49] M. Liu, S.K. Tso, and Y. Cheng. An extended nonlinear primal-dual interior-point algorithm for reactive-power optimization of large-scale power systems with discrete control variables. *IEEE Transactions on Power Systems*, 17(4):982–991, November 2002.

[50] W.H. Liu, A.D. Papalexopoulos, and W.F. Tinney. Discrete shunt controls in a newton optimal power flow. *IEEE Transactions on Power Systems*, 7(4):1509–1518, November 1992.

[51] D.G. Luenberger and Y. Ye. *Linear and Nonlinear Programming*. International Series in Operations Research & Management Science. Springer, 2008.

[52] P.J. Macfie, G.A. Taylor, M.R. Irving, P. Hurlock, and H. Wan. Proposed shunt rounding technique for large-scale security constrained loss minimization. *IEEE Transactions on Power Systems*, 25(3):1478–1485, August 2010.

[53] J. Machowski, J. Bialek, and J. Bumby. *Power System Dynamics: Stability and Control*. Wiley, 2008.

[54] R. Madani, S. Sojoudi, and J. Lavaei. Convex relaxation for optimal power flow problem: Mesh networks. In *Submitted to the 2013 Asilomar Conference*, 2013.

[55] Y.V. Makarov, S. Lu, X. Guo, J.F. Gronquist, P. Du, T.B. Nguyen, and J.W. Burns. *Wide Area Security Region: Final Report*. Pacific Northwest National Laboratory, PNNL-19331, 2010.

[56] T. Malakar and S.K. Goswami. Active and reactive dispatch with minimum control movements. *International Journal of Electrical Power & Energy Systems*, 44(1):78–87, 2013.

[57] J.D. McCalley, S. Wang, Q.L. Zhao, Guozhong Z., R.T. Treinen, and A.D. Papalexopoulos. Security boundary visualization for systems operation. *IEEE Transactions on Power Systems*, 12(2):940–947, May 1997.

[58] J.M. Mendel. *Lessons in Estimation Theory for Signal Processing, Communications, and Control*. Pearson Education, 1995.

[59] F. Milano. Continuous Newton's method for power flow analysis. *IEEE Transactions on Power Systems*, 24(1):50–57, February 2009.

[60] W. Murray. *Newton-Type Methods*. John Wiley & Sons, Inc., 2010.

[61] W. Murray and K.M. Ng. An algorithm for nonlinear optimization problems with binary variables. *Computational Optimization and Applications*, 47(2):257–288, 2010.

[62] W. Murray, T. Tinoco De Rubira, and A. Wigington. Improving the robustness of Newton-based power flow methods to cope with poor initial points. In *Proceedings of the 45th Annual North American Power Symposium*, September 2013.

[63] W. Murray, T. Tinoco De Rubira, and A. Wigington. A robust and informative method for solving large-scale power flow problems. *Submitted to Computational Optimization and Applications*, November 2014.

[64] W. Murray, T. Tinoco De Rubira, and A. Wigington. Optimal power flow with limited and discrete controls. In *IEEE Conference on Innovative Smart Grid Technologies (to appear)*, November 2015.

[65] David R. Musser. Introspective sorting and selection algorithms. *Software: Practice and Experience*, 27(8):983–993, August 1997.

[66] T.T. Ngo, M. Bellalij, and Y. Saad. The trace ratio optimization problem. *SIAM Review*, 54(3):545–569, 2012.

[67] J. Nocedal and S. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, 2006.

[68] T. Overbye. ECE 476, Power System Analysis, Lecture 16, Economic Dispatch. University Lecture, Department of Electrical and Computer Engineering, University of Illinois, 2011.

[69] R.P. ONeill, T. Dautel, and E. Krall. Recent ISO software enhancements and future software and modeling plans. *Federal Energy Regulatory Commission, Washington, DC, Technical Report*, 2011.

[70] A.D. Papalexopoulos, Carl F. Imparato, and F.F. Wu. Large-scale optimal power flow: Effects of initialization, decoupling and discretization. *IEEE Transactions on Power Systems*, 4(2):748–759, May 1989.

[71] H. Pieper. *Algorithms for Mathematical Programs with Equilibrium Constraints with Applications to Deregulated Electricity Markets*. PhD thesis, Stanford University, June 2001.

[72] P. Pourbeik, P.S. Kundur, and C.W. Taylor. The anatomy of a power grid blackout - Root causes and dynamics of recent major blackouts. *IEEE Power and Energy Magazine*, 4(5):22–29, September 2006.

[73] C. Roman and W. Rosehart. Complementarity model for generator buses in opf-based maximum loading problems. *IEEE Transactions on Power Systems*, 20(1):514–516, 2005.

[74] W. Rosehart, C. Roman, and A. Schellenberg. Optimal power flow with complementarity constraints. *IEEE Transactions on Power Systems*, 20(2):813–822, 2005.

[75] F. Saccomanno. *Electric power systems: Analysis and control*. IEEE Press Series on Power Engineering. IEEE Press, 2003.

[76] J. Scudder and F.L. Alvarado. *Step Size Optimization in a Polar Newton Power Flow*. University of Wisconsin-Madison. Department of Electrical and Computer Engineering. 1981.

[77] S.A. Soman, S.A. Khaparde, and S. Pandit. *Computational Methods for Large Sparse Power Systems Analysis: An Object Oriented Approach*. Kluwer international series in engineering and computer science. Kluwer Academic Publishers, 2002.

[78] S.A. Soman, K. Parthasarathy, and D. Thukaram. Curtailed number and reduced controller movement optimization algorithms for real time voltage/reactive power control. *IEEE Transactions on Power Systems*, 9(4):2035–2041, November 1994.

[79] B. Stott. Effective starting process for Newton-Raphson load flows. *Proceedings of the Institution of Electrical Engineers*, 118(8):983–987, 1971.

[80] R.A. Tapia. Diagonalized multiplier methods and quasi-Newton methods for constrained optimization. *Journal of Optimization Theory and Applications*, 22(2):135–194, 1977.

[81] W.F. Tinney, J.M. Bright, K.D. Demaree, and B. A. Hughes. Some deficiencies in optimal power flow. *IEEE Transactions on Power Systems*, 3(2):676–683, May 1988.

[82] J. Valenzuela and M. Mazumdar. Probabilistic unit commitment under a deregulated market. In B.F. Hobbs, M.H. Rothkopf, R.P. ONeill, and H. Chao, editors, *The Next Generation of Electric Power Unit Commitment Models*, volume 36 of *International Series in Operations Research & Management Science*, pages 139–152. Springer US, 2001.

[83] C.D. Vournas, M. Karystianos, and N.G. Maratos. Bifurcation points and loadability limits as solutions of constrained optimization problems. In *IEEE Power Engineering Society Summer Meeting*, volume 3, pages 1883–1888, 2000.

[84] X.F. Wang, Y. Song, and M. Irving. *Modern Power Systems Analysis*. Power Electronics and Power Systems. Springer, 2008.

[85] J. Weston, A. Elisseeff, B. Scholkopf, and P. Kaelbling. Use of the zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

[86] D. White, A. Roschelle, P. Peterson, D. Schlissel, B. Biewald, and W. Steinhurst. The 2003 blackout: Solutions that wont cost a fortune. *The Electricity Journal*, 16(9):43–53, 2003.

[87] X.P. Zhang. *Restructured Electric Power Systems: Analysis of Electricity Markets with Equilibrium Models*. IEEE Press Series on Power Engineering. Wiley, 2010.

[88] Y. Zhang and Z. Ren. Optimal reactive power dispatch considering costs of adjusting the control devices. *IEEE Transactions on Power Systems*, 20(3):1349–1356, August 2005.

[89] J. Zhao, H.D. Chiang, P. Ju, and H. Li. On PV-PQ bus type switching logic in power flow computation. In *Power Systems Computation Conference (PSCC)*, Glasgow, Scotland, 2008.

[90] G. Zhou and J.D. McCalley. Composite security boundary visualization. *IEEE Transactions on Power Systems*, 14(2):725–731, May 1999.

[91] J. Zhu. *Optimization of Power System Operation*. IEEE Press Series on Power Engineering. Wiley, 2009.

[92] R.D. Zimmerman, C.E. Murillo-Sánchez, and R.J. Thomas. MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Transactions on Power Systems*, 26(1):12–19, February 2011.