Nonlinear Optimal Power Flow by Interior and Non-Interior Point Methods

by

Geraldo Leite Torres

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical Engineering

Waterloo, Ontario, Canada, 1998

The University of Waterloo requires the signatures of all persons using or photocopying this thesis. Please sign below, and give address and date.

Abstract

Optimization of power systems is one of the areas where *interior-point* (IP) methods are being applied extensively; due to the size and special features of these problems, IP methods have computationally proven to be a viable alternative for their solution. In this thesis, we propose and investigate a number of IP methods for solving large *nonlinear programming* (NLP) problems. The IP method that we study belongs to the class of *infeasible primal-dual path-following* methods. Four *higher-order* variants of this IP method are considered as well: (i) the predictor-corrector method, (ii) the perturbed composite Newton method, (iii) the multiple predictor-corrector method, and (iv) the multiple centrality corrections method. The proposed IP algorithms are then applied to specialized *optimal power flow* (OPF) problems that use voltages either in polar or in rectangular coordinates. When formulated in rectangular coordinates, some OPF variants have quadratic objective and quadratic constraints. Such quadratic features allow for ease of matrix setup and inexpensive incorporation of second-order information in *higher-order* variants of the IP method.

A *non-interior-point* (NIP) method for solving nonlinear OPF problems is also proposed in this thesis. Unlike IP methods, the NIP method handles the complementarity conditions for optimality in such a way that the strict positivity conditions are not required to be satisfied at every iterate. This approach derives from reformulations of *complementarity problems* as nonlinear systems of equations, and allows for a Newton-type method to be used. To reformulate the OPF problem as a nonlinear system of equations, we handle the complementarity conditions by means of an *NCP-function*, which is a function $\psi_\mu : I\!R^2 \mapsto I\!R$ that satisfies the property: $\psi_\mu(a,b) = 0 \Leftrightarrow a > 0$, $b > 0$ and $ab = \mu$, for any $\mu > 0$. Since the non-negativity of any limit point is automatically assured by NCP-functions, without imposing additional conditions, the initial point and the iterates do not necessarily have to stay in the positive orthant.

In this thesis, we have derived the proposed IP and NIP algorithms based on an NLP problem form that is suitable to express most OPF problems. Many important issues for the efficient implementation of these algorithms, as related to nonlinear OPF solution, are discussed in detail. Numerical results illustrate the viability of the proposed algorithms as applied to several power networks that range in size from 14 to 2098 buses.

Acknowledgements

I always had in mind that the Brazilian people would be the first on this page. Through CAPES agency and the Universidade Federal de Pernambuco (UFPE) they anonymously provided all the financial support that I needed.

I would like to express my gratitude to my supervisor, Dr. Victor Quintana, for the opportunity to do this research, guidance, friendship, and the various activities on optimization in which we have been partners. I thank Dr. Anthony Vannelli and Dr. Alan George, who were always willing to help me. The professionalism and constructive suggestions of Dr. Paul Calamai are greatly appreciated. I am also grateful to Dr. Claudio Cañizares and my external examiner, Dr. Francisco Galiana.

A special thanks goes to Mrs. Lorna Van Mossel, whom I am a sincere friend of, and who I also like calling my "Canadian mother". Through her I developed priceless friendships. In particular, I mention Jim and Lynn Harris (late friend), and Dave and Katherine Hare.

I am also indebted to the Departamento de Engenharia Elétrica e Sistemas de Potência at UFPE for the leave of absence to pursue graduate studies. In particular, I thank my colleagues Dr. Manoel Afonso and Dr. Maria Carvalho, friends without whom I might never have pursued graduate studies. I also thank the sincere encouragement received from my colleagues Renato Ribeiro and Ednaldo Miranda (late friend).

My stay in Canada was an opportunity for great friendships. First of all, I mention Sammy Aiau, Cecilio and Giselle Pimentel, André Plaisant, and Geraldo and Bete Milioli. I would also like to mention Carlson and Daniela Cabral, Everton de Oliveira, Francisco and Neusa Somera, Germano and Rosário Torres, Marcelo Bessa, Pedro de Abreu, and Zambroni and Marli de Souza. The friendship and caring of Mrs. Quintana is also appreciated. Playing with the little Brazilians Matheus de Souza, Lucas Pimentel and Izaura Milioli were my most enjoyable moments in Waterloo.

I also thank my office colleagues Vladimir Lucic and Kingsley Fregene for the friendly convivial, and my friends and fellow graduate students José Medina, Marcelino Madrigal, Rodrigo Fuentes and Daniel German. I thank my friends in Brazil that constantly sent me messages with a word of concern, in particular Siomara Durand and Abílio Muniz.

Finally, and most of all, I thank my parents, Alexandre and Socorro, who put the education of their six children above everything. In the same way I thank my brothers and sisters, in particular Adígina and Adalci, who took care of many practical things while I was away from home.

To my parents, Alexandre and Socorro

Contents

List of Tables

List of Figures

Glossary

We summarize here, for convenience of reference, some of the notation and terminology used in this thesis.

$\mathbb{R}^n$	: n-dimensional Euclidean space.
$\mathbb{R}^n_+$	: nonnegative orthant of $\mathbb{R}^n$. If $x \in \mathbb{R}^n_+$, then $x_i \geq 0$ for $i = 1, \ldots, n$.
$\mathbb{R}^n_{++}$	: strictly positive orthant of $\mathbb{R}^n$. If $x \in \mathbb{R}^n_{++}$, then $x_i > 0$ for $i = 1, \ldots, n$.
$\|\cdot\|_2$, $\|\cdot\|$	: Euclidean norm. If $x \in \mathbb{R}^n$, then $\|x\|_2 = (\sum_{i=1}^n x_i^2)^{1/2}$.
$\|\cdot\|_\infty$	: l_∞-norm. If $x \in \mathbb{R}^n$, then $\|x\|_\infty = \max_i \|x_i\|$.
ln	: natural logarithm: $\log_e$.
u	: vectors of ones of appropriate dimension: $u = (1, 1, \ldots, 1)^T$.
0	: zero vector or zero matrix.
I_n	: identity matrix in $\mathbb{R}^{n \times n}$.

Scalars:

k	: iteration index: $k = 1, 2, 3, \ldots$.
m	: number of equality constraints.
n	: number of primal variables x_i.
p	: number of nonlinear functional bound constraints.
q	: number of simple bound constraints.
r	: number of primal and dual variables: $r = 4p + 4q + n + m$.

Index sets:

$\mathcal{E}$	: set of indices (with $\|\mathcal{E}\|$ elements) of load buses eligible for shunt var control.
$\mathcal{F}$	: set of indices (with $\|\mathcal{F}\|$ elements) of load buses with fixed var sources.
$\mathcal{G}$	: set of indices (with $\|\mathcal{G}\|$ elements) of generator buses.
$\mathcal{N}$	: set of indices (with $\|\mathcal{N}\|$ elements) of all buses in the system.
$\tilde{\mathcal{N}}$	: set of indices (with $\|\tilde{\mathcal{N}}\|$ elements) of all buses in the system but the slack bus.
$\mathcal{N}_i$	: set of indices (with $\|\mathcal{N}_i\|$ elements) of all buses directly connected to bus i.
$\mathcal{B}$	: set of ordered index pairs (i,j) (with $\|\mathcal{B}\|$ elements) of sending-end and receiving-end buses of all branches in the system (transmission lines and transformers): $\mathcal{B} := \{(i,j) \mid i \in \mathcal{N}, j \in \mathcal{N}_i \text{ and } j > i\}$
$\mathcal{T}$	: set of ordered index pairs (i,j) (with $\|\mathcal{T}\|$ elements) of sending-end and receiving-end buses of all transformers with LTC: $\mathcal{T} \subset \mathcal{B}$.

Functions and vectors:

$\boldsymbol{x} \in I\!R^n$	: vector of primal variables.
$\widehat{\boldsymbol{x}} \in I\!R^q$	: vector of primal variables that have finite bounds: $\underline{\boldsymbol{x}} \leq \widehat{\boldsymbol{x}} \leq \overline{\boldsymbol{x}}$.
$f : I\!R^n \mapsto I\!R$	: scalar objective function.
$\nabla_{\boldsymbol{x}} f : I\!R^n \mapsto I\!R^n$	: gradient vector of $f(\boldsymbol{x})$.
$\boldsymbol{g} : I\!R^n \mapsto I\!R^m$	: nonlinear function-vector of equality constraints.
$\boldsymbol{h} : I\!R^n \mapsto I\!R^p$	: nonlinear function-vector of functional variables, that have lower bound $\underline{\boldsymbol{h}}$ and upper bound $\overline{\boldsymbol{h}}$.
$\boldsymbol{s}_1,\ \boldsymbol{s}_2 \in I\!R^p_+$	: primal slack vectors: $\underline{\boldsymbol{h}} + \boldsymbol{s}_1 = \boldsymbol{h}(\boldsymbol{x})$ and $\boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{s}_2 = \overline{\boldsymbol{h}}$.
$\boldsymbol{s}_3,\ \boldsymbol{s}_4 \in I\!R^q_+$	: primal slack vectors: $\underline{\boldsymbol{x}} + \boldsymbol{s}_3 = \widehat{\boldsymbol{x}}$ and $\widehat{\boldsymbol{x}} + \boldsymbol{s}_4 = \overline{\boldsymbol{x}}$.
$\boldsymbol{z}_1 \in I\!R^p_+,\ \boldsymbol{z}_2 \in I\!R^p$	: Lagrange multiplier vectors related to $\underline{\boldsymbol{h}} \leq \boldsymbol{h}(\boldsymbol{x}) \leq \overline{\boldsymbol{h}}$.
$\boldsymbol{z}_3 \in I\!R^q_+,\ \boldsymbol{z}_4 \in I\!R^q$	: Lagrange multiplier vectors related to $\underline{\boldsymbol{x}} \leq \widehat{\boldsymbol{x}} \leq \overline{\boldsymbol{x}}$.
$\boldsymbol{y} \in I\!R^m$	: Lagrange multiplier vector related to $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0}$.

$w \in \mathbb{R}^r$	: vector of all primal and dual variables: $w := (s_1, s_2, s_3, s_4, z_1, z_2, z_3, z_4, x, y)^T$.
$L_\mu : \mathbb{R}^r \mapsto \mathbb{R}$	: Lagrangian function parameterized by μ.
$\nabla_x L_\mu : \mathbb{R}^r \mapsto \mathbb{R}^n$	: gradient vector of $L_\mu(w;\mu)$ with respect to x.
$\nabla_w L_\mu : \mathbb{R}^r \mapsto \mathbb{R}^r$	: gradient vector of $L_\mu(w;\mu)$ with respect to w.

Matrices:

$B \in \mathbb{R}^{\lvert\mathcal{N}\rvert \times \lvert\mathcal{N}\rvert}$	: bus susceptance matrix.
$G \in \mathbb{R}^{\lvert\mathcal{N}\rvert \times \lvert\mathcal{N}\rvert}$	: bus conductance matrix.
$\nabla_x g : \mathbb{R}^n \mapsto \mathbb{R}^{n \times m}$	: transposed Jacobian matrix of $g(x)$: $\nabla_x g(x) := [\nabla_x g_1(x), \nabla_x g_2(x), \ldots, \nabla_x g_m(x)]$.
$\nabla_x h : \mathbb{R}^n \mapsto \mathbb{R}^{n \times p}$	: transposed Jacobian matrix of $h(x)$: $\nabla_x h(x) := \left[\nabla_x h_1(x), \nabla_x h_2(x), \ldots, \nabla_x h_p(x)\right]$.
$\nabla^2_{xx} f : \mathbb{R}^n \mapsto \mathbb{R}^{n \times n}$	: Hessian matrix of $f(x)$.
$\nabla^2_{xx} g_i : \mathbb{R}^n \mapsto \mathbb{R}^{n \times n}$	: Hessian matrix of the component $g_i(x)$ of $g(x)$.
$\nabla^2_{xx} h_i : \mathbb{R}^n \mapsto \mathbb{R}^{n \times n}$	: Hessian matrix of the component $h_i(x)$ of $h(x)$.
$\nabla^2_{xx} L_\mu : \mathbb{R}^r \mapsto \mathbb{R}^{n \times n}$	: Hessian matrix of $L_\mu(w;\mu^k)$ with respect to x.
$\nabla^2_{ww} L_\mu : \mathbb{R}^r \mapsto \mathbb{R}^{r \times r}$	: Hessian matrix of $L_\mu(w;\mu^k)$ with respect to w.
S_1, S_2, S_3, S_4	: diagonal matrices constructed from the vectors s_1, s_2, s_3 and s_4, respectively. For example, $S_1 := \text{diag}(s_{1_1}, s_{1_2}, \ldots, s_{1_p})$.
$Z_1, \widehat{Z}_2, Z_3, \widehat{Z}_4$	: diagonal matrices constructed from the vectors z_1, $z_1 + z_2$, z_3 and $z_3 + z_4$, respectively.

Acronyms:

IP	: Interior-Point.
KKT	: Karush-Kuhn-Tucker.
LCP	: Linear Complementarity Problem.
LP	: Linear Programming.
LTC	: under Load Tap Changer.
MCC	: Multiple Centrality Corrections.

ML	: Maximum Loadability.
MLS	: Minimum Load Shedding.
MPC	: Multiple Predictor-Corrector.
NCP	: Nonlinear Complementarity Problem.
NIP	: Non-Interior-Point.
NLP	: NonLinear Programming.
OPF	: Optimal Power Flow.
QP	: Quadratic Programming.
PCN	: Perturbed Composite Newton.
RPD	: Reactive Power Dispatch.
SLP	: Sequential Linear Programming.
SQP	: Sequential Quadratic Programming.

Chapter 1

Introduction

The main purpose of a power system is to provide its consumers of electricity with power above a certain level of quality, and as economically as possible. Loosely speaking, quality of power supply is measured in terms of constancy of frequency and of voltage, and level of reliability. Frequency control is closely related to active power control, whereas voltage control is closely related to reactive power control [50]. Incidentally, the constantly changing load demand for active and reactive powers, and unforeseen changes in network configuration, can result in voltage levels that are well outside tolerable limits and, most likely, violate utility and consumers equipment operation restrictions.

To correct undesirable operation conditions, power system operators are required to constantly control the production, absorption, and flow of power at all levels in the system. This is done by adjusting system control variables such as generator outputs, transformer tap settings, shunt var sources, and so forth. Deciding on an optimal control action, aiming at the secure and economic operation of a power system, is an extremely difficult task, which is best performed by the *optimal power flow* (OPF) tool at power system control centers [73]. The OPF tool is a sophisticated computational procedure that uses mathematical programming techniques to find an optimal setting of the system control variables, subject to a reasonably large set of specified physical and operational constraints.

The OPF problem is inevitably a very large non-convex *nonlinear programming* (NLP) problem, that is complicated in realistic applications by the presence of a large number of discrete variables such as transformer tap ratios, shunt capacitor susceptances, and so forth. Given its significance in power system planning and operation activities, OPF has been the subject of intensive research for nearly four decades [38, 65]. Gradient techniques

were the first approaches used to solve an OPF problem [18], which was mathematically formulated for the first time by Carpentier in 1962. Since then, improvements in OPF procedures have been achieved in two main ways: (i) more efficient problem formulation, and (ii) improved mathematical techniques. The main techniques for solving the OPF problem include reduced gradient methods, methods based on augmented Lagrangian and exact penalty functions, and, mainly, local approximation-based approaches such as *sequential linear programming* (SLP) and *sequential quadratic programming* (SQP).

The SLP and SQP approaches have been widely used in power system optimization. Nowadays, they can take advantage of efficient *interior-point* (IP) methods for solving the LP and QP subproblems [72, 79]. However, the convergence success of SLP and SQP is, among other things, highly dependent on the existence of a good initial operating point, which not always does occur. Incidentally, as we explain below, there has been a growing need to solve the OPF problem in a nonlinear manner. An acclaimed algorithm to solve the OPF in a nonlinear manner was proposed by Sun et al. [66, 1984]. It combines together the Newton's method (for unconstrained optimization) with a Lagrange multiplier method (for optimization with equalities) and penalty functions (for handling inequalities). Well designed data structures, which allow for block-factorization, and efficient use of sparsity techniques made such an algorithm very attractive and successful at the time. Its computational efficiency, however, greatly relies on the efficient identification of the binding constraints.

In the new scenario of a deregulated electricity market, the trend is for power systems to be operated closer to ultimate limits. Competition in a deregulated market, with utilities eager for better profits, likely involve large power transactions that may lead the systems to heavily loading conditions. Also, load growth, coupled with financial and regulatory constraints that often severely limit the expansion of generation and transmission facilities, have led power systems to operate closer to their transfer limit. Power systems should then be carefully monitored and controlled in order to avoid voltage, electromechanical, and control stability problems. Towards this purpose, an OPF procedure at power system control centers plays a major role. From the above, an OPF is now required to deal with strong nonlinearities in power system behavior; local approximation-based optimization techniques will be less attractive to cope with stressed operation conditions [36].

In the emerging deregulated electricity market, with numerous sellers of electric energy in the same area, new challenges are being imposed to the OPF procedure. For example, the following applications are listed in [22]:

- To calculate the maximum load at a subset of buses, for voltage collapse analysis, or calculate the minimum load shedding to avoid voltage collapse.
- To calculate the maximum load that can safely be delivered at a given bus, or set of buses, in order to define contracts with large power consumers.
- To calculate the maximum active power that can safely be transferred from one area of the network to another, to define inter-utility transactions.
- To calculate the maximum active power between any pair of buses in order to define maximum wheeling transaction capability.

These OPF variants are highly nonlinear problems, and solving them can be very difficult. Various conditions under which an OPF algorithm may fail to converge are discussed in [2].

1.1 Research Motivation

In this research, we strive for the efficient numerical solution of large-scale NLP problems (with thousands of variables and constraints) that can be expressed in the standard form:

$$\begin{array}{lrcl} \text{minimize} & f(\boldsymbol{x}) & & \\ \text{subject to} & g(\boldsymbol{x}) & = & \boldsymbol{0}, \\ & \underline{\boldsymbol{h}} \leq \boldsymbol{h}(\boldsymbol{x}) & \leq & \overline{\boldsymbol{h}}, \\ & \underline{\boldsymbol{x}} \leq \widehat{\boldsymbol{I}}\boldsymbol{x} & \leq & \overline{\boldsymbol{x}}, \end{array} \tag{1.1}$$

motivated by the fact that the structure of most OPF problems is essentially contained in such a form, where

- $\boldsymbol{x} \in I\!R^n$: is a vector of explicit decision variables, including the control and nonfunctional dependent (state) variables. (A control variable is one which can be manipulated, whereas a dependent variable depends on other variables and cannot be directly manipulated.)
- $f : I\!R^n \mapsto I\!R$: is a scalar function that represents the power system's planning or operation optimization goal such as generation cost, power losses in the transmission system, load shedding, and so forth.
- $g : I\!R^n \mapsto I\!R^m$: is a nonlinear vector that contains conventional power balance equations, occasionally augmented by a few special equality constraints

such as power balances across boundaries as in a pool operation, or flows that are set to a given value, and so forth.

- $h : I\!R^n \mapsto I\!R^p$: is a nonlinear vector of functional variables, with lower bound $\underline{h}$ and upper bound $\overline{h}$, corresponding to physical and specified operational limits on the system.
- $\widehat{x} \in I\!R^q$: (implicit as $\widehat{I}x$) is a vector with the components of x that have finite bounds, with lower bound $\underline{x}$ and upper bound $\overline{x}$, corresponding to physical and specified operational limits on the system.
- $\widehat{I} \in I\!R^{q \times n}$: is an incidence matrix to obtain $\widehat{x}$ from x, formed by the rows of $I_n \in I\!R^{n \times n}$ that have the indices of the bounded variables.

To solve the NLP problem (1.1), we will employ *interior-point* (IP) and *non-interior-point* (NIP) methods for large-scale NLP.

Over the past fifteen years, research on IP methods has experienced an awesome expansion, both in theory and computational practice. The first known IP method is usually attributed to Frisch [26, 1955], which is a *logarithmic barrier method* that was later (1960s) extensively studied by Fiacco and McCormick [21, 1968] to solve nonlinearly inequality constrained problems. Interestingly enough, it was in the LP research area, in 1984, that the superb computational efficiency of IP methods was first demonstrated in practice [47], with initial reluctant acceptance and later with enthusiastic recognition by the research community. In 1979, Khachiyan [48] presented an *ellipsoid method* that would solve an LP problem in *polynomial time*, meaning that the number of operations to obtain the optimum is bounded by a polynomial in the problem dimension. Unfortunately, his method proved, in practice, to be computationally inferior to the simplex method.

As mentioned above, the greatest breakthrough in the IP research field took place in 1984 when Karmarkar [47] came up with a new IP method for LP, reporting solution times up to 50 times faster than the simplex method. Since then, several IP methods have been proposed and implemented. IP methods are usually classified into three main categories: (i) *projective methods* [47], (ii) *affine-scaling methods* [53], and (iii) *primal-dual methods* [32, 76]. Projective methods include Karmarkar's original algorithm, and are responsible for the great interest established to the IP research area. Soon after 1984, affine-scaling methods were obtained as simplifications of projective methods. Their reduced computational complexity and simplicity made them very popular at the time. They were also among the most effective in practice. Primal-dual methods can be subdivided into: (i) *path-following methods* [30] and (ii) *potential reduction methods* [58].

The first theoretical results for primal-dual path-following methods are due to Megiddo [56, 1986], who proposed to apply a logarithmic barrier method to the primal and dual problems simultaneously. His primal-dual path-following algorithm performs better than earlier IP algorithms. The primal-dual algorithms that incorporate predictor and corrector steps, such as Mehrotra's predictor-corrector technique [57, 1992], are currently accepted as the computationally most effective variants. Further improvements to Mehrotra's technique were later achieved through the use of multiple corrector steps [10, 31]. Nowadays, IP method variants are being extended to solve all kind of programs: from linear to nonlinear, and from convex to non-convex. Recently, the development of IP methods to directly solve NLP problems [19, 64, 71, 78] has been motivated by the superb performance of IP methods for LP and QP.

Optimization of power system operations is one of the areas where IP methods are being applied extensively [13, 33, 40, 54, 55, 69, 77]; due to the size and special features of these problems, IP methods have computationally proven to be a viable alternative for their solution. Among the various applications of IP methods in power systems are the solution of the minimum load shedding [34] and maximum loadability [40] problems; these are highly nonlinear variants of the OPF problem that very unlikely can be solved by an LP-based approach. An attractive feature of the primal-dual IP approach is that feasibility is usually attained during the iterative process, as optimality is approached; this means that the power balance equations need not be satisfied at the initial point. This feature is particularly important to solving minimum load shedding problems, where power flow unsolvability is an issue.

Previously proposed OPF algorithms have mostly used voltages in polar coordinates, possibly due to the excellent performance and widespread use of decoupled power flow programs which use voltage polar coordinates. Polar coordinates are more intuitive because voltage magnitudes and phase angles are usually taken as state variables, and these have a physical meaning. Rectangular coordinates, on the contrary, have been totally neglected in OPF studies. In this thesis, we study the advantages and disadvantages of using the polar and rectangular coordinates in nonlinear OPF solution by primal-dual IP methods.

We have observed that some OPF variants when formulated in rectangular form have quadratic objective and quadratic constraints. Desirable properties of a quadratic function are: (i) its Hessian is constant, (ii) its Taylor-series expansion terminates at the second-order term without truncation error, and (iii) the second-order term of its Taylor-series is easily evaluated. Such quadratic features allow for ease of matrix setup and inexpensive

incorporation of second-order information in higher-order IP method variants. With the polar coordinates, on the other hand, we find that voltage-magnitude bounds are handled in a more straightforward manner. These issues have not been studied in great detail in previous works; we address them in the current thesis research.

The OPF optimality conditions can be regarded as a particular case of the *nonlinear complementarity problem* (NCP). In the last few years, growing attention has been paid to approaches that reformulate *complementarity problems* as nonlinear systems of equations [6, 11,16,20,23,44–46], so that a Newton-type method can be used in their solution. There are several possibilities to redefine an NCP as a system of equations. Recently, reformulations that handle the complementarity conditions by means of *NCP-functions* have attracted a lot of attention from researchers.

A function $\psi : I\!R^2 \mapsto I\!R$ is said to be an NCP-function if it satisfies the property:

$$\psi(a,b) = 0 \quad \Longleftrightarrow \quad a \geq 0, \quad b \geq 0 \quad \text{and} \quad ab = 0.$$

Since the non-negativity of any limit point is automatically assured by an NCP-function (as long as the iterative process converges to a stationary point of the Lagrangian function), without imposing additional conditions, the initial point and the iterates do not necessarily have to stay in the positive orthant, unlike IP methods. Recently developed techniques for solving complementarity problems have not been applied to OPF solution. As we employ these techniques, we further expand the state-of-the-art of OPF solution procedures.

1.2 Research Objectives

We define as the main objectives underlying the current thesis research, the following:

- To formulate OPF problems by using voltages in rectangular and in polar coordinates, and to identify, afterwards, the major computational advantages and disadvantages of both voltage representations.
- To develop a primal-dual IP algorithm for solving OPF problems in a nonlinear manner. As in previous works, this IP algorithm development is a direct extension of its similar for LP. However, we aim to study different step length schemes and updating formulae of the barrier parameter, study the influence of various parameters in the algorithm convergence, elaborate on different analytical procedures to compute the Newton directions, and propose initialization heuristics suitable for the OPF solution.

- To develop extensions to NLP of several successful higher-order IP methods for LP. The higher-order IP variants studied are: (i) the predictor-corrector method [57], (ii) the perturbed composite Newton method [67], (iii) the multiple predictor-corrector method [10], and (iv) the multiple centrality corrections method [31]. The approach (i) was previously extended to nonlinear OPF solution by Wu, Debs and Marsten [77, 1994], whereas the approaches (ii), (iii) and (iv) have not been extended at all to power systems optimization.

- To develop the first OPF approach that handles the complementarity conditions for optimality by means of an NCP-function. This is an original OPF algorithm, where we take advantage of recent mathematical development to solve complementarity problems. We use an NCP-function to transform the whole *Karush-Kuhn-Tucker* (KKT) conditions into a system of nonlinear equations; such a nonlinear system is solved afterwards by a Newton-type method, which we describe in detail.

  The research carried out under this topic is the major contribution of this thesis.

- To discuss many issues that are related to the efficient implementation of IP and NIP algorithms for NLP, in connection with the solution of OPF problems. Particularly, to discuss heuristics for initialization of the algorithms; to discuss the assembling of matrices in rectangular and in polar coordinates; to discuss data structures; and to discuss the solution of the linear systems.

- To implement the proposed algorithms in Fortran 77 and perform computational experiments to gain insight into the various methods; to compare competing methods; to tune parameters in the algorithms; and to compare the polar and the rectangular "versions" of the OPF algorithm.

1.3 Outline of the Thesis

The remainder of this thesis is organized as follows. In Chapter 2, we describe formulations of three variants of the OPF problem, namely, (i) the minimum transmission power losses, (ii) the maximum loadability, and (iii) the minimum load shedding problems. Problem (i) is formulated both in rectangular and in polar coordinates.

In Chapters 3 and 4, we deal with primal-dual IP algorithms suitable for solving the NLP problem (1.1). Specifically, in Chapter 3, a primal-dual IP method is developed in detail. We describe two analytical approaches to compute the Newton directions, three

schemes to compute the step lengths, two updating formulae of the barrier parameter, and describe the convergence test. In Chapter 4, we describe various higher-order variants of the IP method presented in Chapter 3, namely, (i) the predictor-corrector method, (ii) the composite Newton method, (iii) the multiple predictor-corrector method, and (iv) the multiple centrality corrections method.

In Chapter 5, a new NIP continuation method for solving the nonlinear OPF problem is proposed. This algorithm development is based on a recently proposed *NCP-function* to solve complementarity problems. We propose a reformulation of the KKT conditions as a nonlinear system of equations and then describe a Newton-type algorithm that is used to solve such a nonlinear system.

In Chapter 6, we discuss many points and issues that are directly related to efficient implementations of the IP and NIP algorithms that are described in Chapters 3, 4 and 5. The emphasis, however, is on implementations for solving the OPF problems that are described in Chapter 2. Particularly, we propose four heuristics for initialization of the algorithms, describe the assembling of matrices for the rectangular and the polar "versions" of the OPF problem, discuss data structures and some code fragments, and discuss the solution of the linear systems.

In Chapter 7, we discuss extensive numerical results obtained with implementations of the IP and NIP algorithms to solve the *reactive power dispatch* OPF variant. The main purposes of the computational tests are: (i) to gain insight into the various methods, (ii) to compare competing methods, (iii) to tune parameters in the algorithms, and (iv) to compare the rectangular and the polar "versions" of the OPF problem. In these numerical tests, we employ eleven power systems that range in size from 14 to 2098 buses.

Finally, a summary and conclusions are presented in Chapter 8. We highlight the main contributions of this thesis and provide a list of potential research directions that, we believe, could further expand the state-of-the-art of OPF solution procedures.

Chapter 2

Optimal Power Flow Problem

For efficient, reliable and economic operation of a power system, several levels of controls that involve a complex array of devices have to be selected and properly coordinated. In the operation of a power system, the load demand for active and reactive powers is continuously changing and often results in voltage levels that are well outside tolerable limits and, most likely, violate utility and consumers equipment operation restrictions. To repair unacceptable operating conditions, power system operators are required to continuously control the production, absorption, and flow of power at all levels in the system, by adjusting system control variables such as generation outputs, transformer tap settings, phase shifter angles, shunt capacitor susceptances, shunt reactor suceptances, and so forth.

Due to the fact that a power system is fed by many generating units, and supplies power to a vast number of loads that are dispersed over large geographical areas, the task of maintaining voltages within the required limits is a difficult one. Voltage control is recognized as closely related to reactive power control; the proper selection and coordination of equipment for controlling reactive power and voltage are among the major challenges of power engineering [50]. This task can be efficiently performed by the *optimal power flow* (OPF) procedure at power system control centers. The OPF is an elaborated computational tool that uses optimization techniques to find the state of a power system that optimizes a given performance index while satisfying a set of physical and operational constraints.

An OPF problem can be posed in various different forms [65]. This chapter deals with the formulation of three variants of the OPF class, namely, (i) the *reactive power dispatch* (RPD) problem, (ii) the *maximum loadability* (ML) problem, and (iii) the *minimum load shedding* (MLS) problem. The objective of the RPD problem, as formulated in this thesis,

is to minimize the active power losses in the transmission system subject to the power flow balance equations, the operating limits on voltages and reactive power output of generators, and the physical limits on shunt susceptances and transformer tap settings.

The ML problem aims at determining the maximum load increase a power system can withstand while satisfying utility and consumers device operation restrictions [40]. In many situations in the operation of a power system, load shedding schemes are utilized to reduce the actual load to a level that can be safely supplied by available generation. The MLS problem is to determine the minimum load shedding necessary for restoring feasibility of operation, or even for restoring solvability of the power flow equations, otherwise unsolvable [34]. Such a situation occurs, for instance, as the system undergoes a severe contingency. From the above, the ML and MLS procedures are, therefore, valuable tools in voltage collapse studies. As such, they deal with highly nonlinear problems.

Previously proposed OPF formulations have mostly used voltages expressed in polar coordinates, possibly due to the excellent performance and widespread use of decoupled power flow programs that employ polar coordinates. Polar coordinates are more intuitive because voltage magnitudes and phase angles are usually taken as state variables, and these have a direct physical meaning. Although voltages expressed in rectangular coordinates has provided some sort of numerical advantage in several applications, no substantial attention has been paid to this voltage representation form in OPF studies.

The rectangular coordinates space has been utilized, among others, by Iwamoto and Tamura [42, 1981] in power flow studies, by Iwamoto, Kusano and Quintana [41, 1989] in state estimation studies, by Galiana and Zeng [27, 1992] in the study of load flow behavior in the proximity of a Jacobian singularity, and by Overbye and Klump [60, 1996] in the calculation of power system low-voltage solutions. In this thesis, we study the advantages and disadvantages of using the rectangular and the polar variable spaces in nonlinear OPF solution by primal-dual IP methods. Rectangular coordinates was first utilized in OPF solution by IP methods by Torres, Quintana and Lambert-Torres [70, 1996].

This chapter is organized as follows. In the next section, we present some notation, definitions and power equations used in the OPF formulations. In Section 2.2, the RPD problem is formulated in rectangular and in polar coordinates. We show that the constraint functions in the rectangular coordinates "version" are quadratic functions of the voltage components. In Sections 2.3 and 2.4, we describe fairly simple formulations of the ML and MLS problems. We include these problems to emphasize the current need for solving OPF problems in a nonlinear manner. Final remarks in Section 2.5 close the chapter.

2.1 Some Definitions and Power Equations

The following sets of indices are utilized throughout this thesis. We denote by $\mathcal{N}$ the set of all buses (nodes) in the system, by $\widetilde{\mathcal{N}}$ the set of all buses but the slack bus, by $\mathcal{G}$ the set of *generator* buses, by $\mathcal{F}$ the set of load buses with *fixed* shunt var sources, and by $\mathcal{E}$ the set of load buses *eligible* for shunt var control. Furthermore, the following inter-sets relations hold: $\mathcal{N} = \mathcal{G} \cup \mathcal{F} \cup \mathcal{E}$ and $\mathcal{G} \cap \mathcal{F} = \mathcal{G} \cap \mathcal{E} = \mathcal{F} \cap \mathcal{E} = \emptyset$. We denote by $\mathcal{N}_i$ the set of all buses directly connected to bus i. We define the set of ordered index pairs

$$\mathcal{B} := \{(i,j) \mid i \in \mathcal{N},\ j \in \mathcal{N}_i \text{ and } j > i\}$$

as the set of sending-end (i) and receiving-end (j) buses of all branches (transmission lines and transformers) in the system. We let $\mathcal{T} \subset \mathcal{B}$ be the set of sending-end (i) and receiving-end (j) buses of the transformers with under *load tap changer* (LTC) device. By $|\mathcal{N}|$ we denote the size (cardinality) of the set $\mathcal{N}$, by $|\widetilde{\mathcal{N}}|$ the size of the set $\widetilde{\mathcal{N}}$, and so forth.

We express the (complex) bus-voltage at bus i ($\hat{V}_i$) in rectangular coordinates as

$$\hat{V}_i := e_i + jf_i, \qquad \text{for all } i \in \mathcal{N}$$

where e_i and f_i are the real and imaginary components of $\hat{V}_i$, respectively, and j here is the imaginary unity ($\sqrt{-1}$). Without loss of generality, we assign the index 1 to the slack bus, and assume that this bus provides the system angular reference with $e_1 := V_1$ and $f_1 := 0$. The net active power (P_i) and reactive power (Q_i) injections into bus i are defined as

$$P_i := P_{G_i} - P_{L_i}, \qquad \text{for all } i \in \mathcal{N}$$
$$Q_i := Q_{G_i} - Q_{L_i}, \qquad \text{for all } i \in \mathcal{N}$$

where P_{G_i}, P_{L_i}, Q_{G_i} and Q_{L_i} are the active power generation, active power load, reactive power generation and reactive power load at bus i, respectively. For a given voltage profile and network topology, the net power injections into bus $i \in \mathcal{N}$ are computed from

$$P_i(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) = G_{ii}(e_i^2 + f_i^2) + e_i \sum_{j \in \mathcal{N}_i} (G_{ij}e_j - B_{ij}f_j) + f_i \sum_{j \in \mathcal{N}_i} (G_{ij}f_j + B_{ij}e_j), \qquad (2.1)$$

$$Q_i(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) = B_{ii}(e_i^2 + f_i^2) + f_i \sum_{j \in \mathcal{N}_i} (G_{ij}e_j - B_{ij}f_j) - e_i \sum_{j \in \mathcal{N}_i} (G_{ij}f_j + B_{ij}e_j), \qquad (2.2)$$

where $\boldsymbol{e} \in \mathbb{R}^{|\mathcal{N}|}$ and $\boldsymbol{f} \in \mathbb{R}^{|\mathcal{N}|}$ are vectors with the voltage components, G_{ij} is the ijth element of the bus conductance matrix $\boldsymbol{G} \in \mathbb{R}^{|\mathcal{N}| \times |\mathcal{N}|}$, B_{ij} is the ijth element of the bus susceptance matrix $\boldsymbol{B} \in \mathbb{R}^{|\mathcal{N}| \times |\mathcal{N}|}$, and $\boldsymbol{t} \in \mathbb{R}^{|\mathcal{T}|}$ is the vector of transformer tap settings, which are implicit in some elements of $\boldsymbol{G}$ and $\boldsymbol{B}$.

The active (P_{ij}) and reactive (Q_{ij}) power flows in the branches are computed from

$$P_{ij}(e,f,t) = t_{ij}^2 g_{ij}(e_i^2 + f_i^2) - t_{ij}g_{ij}(e_ie_j + f_if_j) - t_{ij}b_{ij}(e_if_j - e_jf_i), \tag{2.3}$$

$$Q_{ij}(e,f,t) = -t_{ij}^2(b_{ij} + b_{ij}^{\mathrm{sh}})(e_i^2 + f_i^2) + t_{ij}g_{ij}(e_if_j - e_jf_i) + t_{ij}b_{ij}(e_ie_j + f_if_j), \tag{2.4}$$

where g_{ij}, b_{ij} and b_{ij}^{sh} are the series conductance, the series susceptance and the shunt susceptance of the branch $(i,j) \in \mathcal{B}$, respectively. In (2.3) and (2.4), we have $t_{ij} = 1$ if the branch (i,j) is a transmission line, and $b_{ij}^{\mathrm{sh}} = 0$ if the branch (i,j) is a transformer. The tap setting representation used is shown in Figure 2.1. As shown in [70], the active power losses in the transmission system can be expressed in the form

$$P_{\mathrm{loss}}(e,f,t) = e^T G e + f^T G f. \tag{2.5}$$

REMARK 2.1 *The Equations* (2.1) *through* (2.5) *are quadratic functions of the bus-voltage rectangular coordinates* e *and* f.

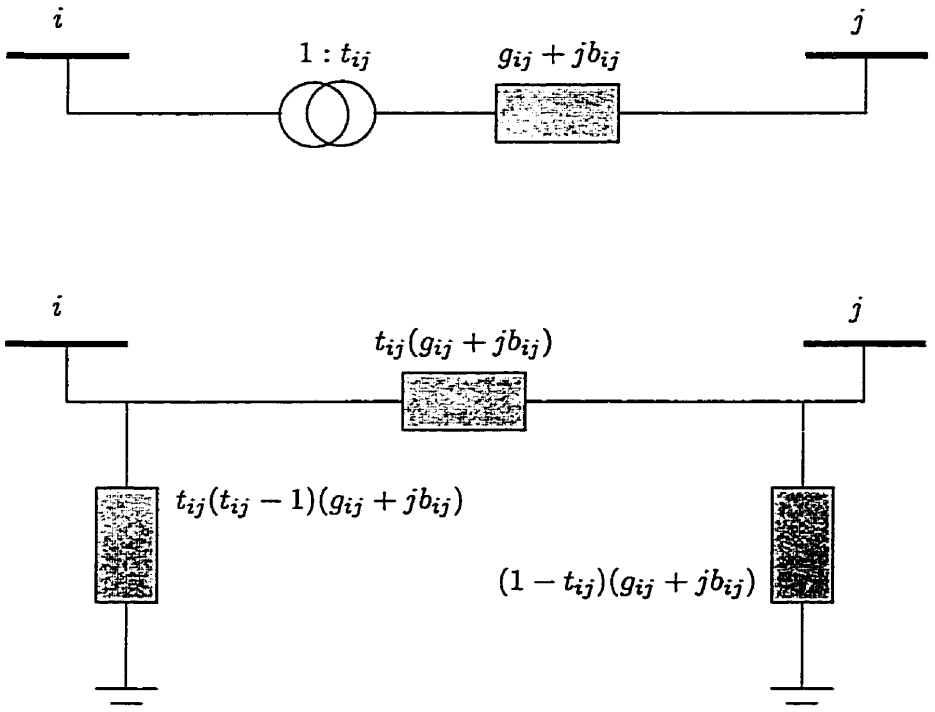

Figure 2.1: Tap setting representation and the transformer Π-model.

In power system studies, the most common representation of (complex) bus-voltages is the polar coordinates,

$$\hat{V}_i := V_i \exp(j\theta_i), \qquad \text{for all } i \in \mathcal{N}$$

where $V_i = (e_i^2 + f_i^2)^{1/2}$ and $\theta_i = \arctan(f_i/e_i)$ are the voltage magnitude and phase angle of $\hat{V}_i$, respectively. We assume that the system angular reference is defined as $\theta_1 := 0°$. For a given voltage profile and network configuration, the net power injections into bus $i \in \mathcal{N}$ are computed from

$$P_i(v,\theta,t) = G_{ii}V_i^2 + V_i \sum_{j\in\mathcal{N}_i} V_j\big[G_{ij}\cos(\theta_i-\theta_j) + B_{ij}\sin(\theta_i-\theta_j)\big], \tag{2.6}$$

$$Q_i(v,\theta,t) = -B_{ii}V_i^2 + V_i \sum_{j\in\mathcal{N}_i} V_j\big[G_{ij}\sin(\theta_i-\theta_j) - B_{ij}\cos(\theta_i-\theta_j)\big], \tag{2.7}$$

where $v \in \mathbb{R}^{|\mathcal{N}|}$ and $\theta \in \mathbb{R}^{|\mathcal{N}|}$ are the vectors of bus-voltage magnitudes and voltage phase angles, respectively. The active and reactive power flows in the branches are computed from

$$P_{ij}(v,\theta,t) = t_{ij}^2 g_{ij}V_i^2 - t_{ij}V_iV_j\big[g_{ij}\cos(\theta_i-\theta_j) + b_{ij}\sin(\theta_i-\theta_j)\big], \tag{2.8}$$

$$Q_{ij}(v,\theta,t) = -t_{ij}^2(b_{ij}+b_{ij}^{\mathrm{sh}})V_i^2 - t_{ij}V_iV_j\big[g_{ij}\sin(\theta_i-\theta_j) - b_{ij}\cos(\theta_i-\theta_j)\big]. \tag{2.9}$$

The active power losses in the transmission system can be obtained from

$$P_{\mathrm{loss}}(v,\theta,t) = \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}_i} g_{ij}\left[V_i^2 + V_j^2 - 2V_iV_j\cos(\theta_i-\theta_j)\right]. \tag{2.10}$$

Whether in rectangular or polar coordinates, the branch apparent power flow is given by

$$S_{ij}(\cdot,\cdot,t) = \sqrt{P_{ij}^2(\cdot,\cdot,t) + Q_{ij}^2(\cdot,\cdot,t)} \tag{2.11}$$

where $(\cdot,\cdot)$ stands for either (e,f) or (v,θ). The complete definition of power flow requires knowledge of four variables at each bus i in the system: P_i, Q_i, V_i and θ_i. In the standard (non-optimal) power flow problem, two of these four variables are known a priori and the aim of the power flow is to solve for the remaining two variables.

2.2 Minimum Transmission Power Losses

In this section, a particular case of the nonlinear OPF problem—the *reactive power dispatch* (RPD) problem—is formulated both in rectangular and in polar coordinates. As usual, it is assumed that the active power injections in all buses but the slack bus are known and remain fixed at the *economic dispatch* (ED) scheduled values. The RPD problem is to determine the reactive power controls, such as generator outputs, shunt susceptances and transformer taps, that minimize the transmission active power losses while satisfying the power balance equations, the operating limits on voltages and reactive power generations, and the physical limits on shunt susceptances and transformer tap settings.

2.2.1 Rectangular Coordinates

The RPD problem in rectangular coordinates is derived from the NLP problem (1.1) as

- $\boldsymbol{x} \in \mathbb{R}^n$: includes the voltage rectangular coordinates $\boldsymbol{e}$ and $\boldsymbol{f}$ but f_1 ($f_1 := 0$), and the transformer tap settings $\boldsymbol{t}$.
- $f : \mathbb{R}^n \mapsto \mathbb{R}$: can be the transmission active power losses, as given by $P_{\text{loss}}(\boldsymbol{e}, \boldsymbol{f}, \boldsymbol{t})$ in (2.5), or the active power injection into the slack bus, as given by $P_1(\boldsymbol{e}, \boldsymbol{f}, \boldsymbol{t})$ in (2.1).
- $\boldsymbol{g} : \mathbb{R}^n \mapsto \mathbb{R}^m$: includes the bus active and reactive power balance constraints

$$P_i(\boldsymbol{e}, \boldsymbol{f}, \boldsymbol{t}) + P^0_{L_i} - P^{ED}_{G_i}, \qquad \text{for all } i \in \widetilde{\mathcal{N}}, \tag{2.12a}$$

$$Q_i(\boldsymbol{e}, \boldsymbol{f}, \boldsymbol{t}) + Q^0_{L_i} - Q^{ED}_{G_i}, \qquad \text{for all } i \in \mathcal{F}, \tag{2.12b}$$

  occasionally augmented by a few special equality constraints. The superscript $(^{ED})$ stands for the *economic dispatch* scheduling, and the superscript $(^0)$ here stands for the base-case load level.

- $\boldsymbol{h} : \mathbb{R}^n \mapsto \mathbb{R}^p$: includes the bus voltage and reactive power functional variables

$$Q_i(\boldsymbol{e}, \boldsymbol{f}, \boldsymbol{t}), \qquad \text{for all } i \in \mathcal{G}, \tag{2.13a}$$

$$Q_i(\boldsymbol{e}, \boldsymbol{f}, \boldsymbol{t}), \qquad \text{for all } i \in \mathcal{E}, \tag{2.13b}$$

$$e_i^2 + f_i^2, \qquad \text{for all } i \in \mathcal{N}, \tag{2.13c}$$

  with the lower operating limits

$$\underline{\boldsymbol{h}} = \begin{pmatrix} \underline{Q}_i, & \text{for all } i \in \mathcal{G} \\ \underline{Q}_i, & \text{for all } i \in \mathcal{E} \\ \underline{V}_i^2, & \text{for all } i \in \mathcal{N} \end{pmatrix} \tag{2.14a}$$

  and the upper operating limits

$$\overline{\boldsymbol{h}} = \begin{pmatrix} \overline{Q}_i, & \text{for all } i \in \mathcal{G} \\ \overline{Q}_i, & \text{for all } i \in \mathcal{E} \\ \overline{V}_i^2, & \text{for all } i \in \mathcal{N} \end{pmatrix}. \tag{2.14b}$$

  The reactive power limits in (2.14) are defined as follows:

$$\underline{Q}_i := \begin{cases} \underline{Q}_{G_i} - Q^0_{L_i}, & \text{if } i \in \mathcal{G}, \\ \underline{b}_i - Q^0_{L_i} + Q^{ED}_{G_i}, & \text{if } i \in \mathcal{E}. \end{cases} \tag{2.15a}$$

$$\overline{Q}_i := \begin{cases} \overline{Q}_{G_i} - Q^0_{L_i}, & \text{if } i \in \mathcal{G}, \\ \overline{b}_i - Q^0_{L_i} + Q^{ED}_{G_i}, & \text{if } i \in \mathcal{E}. \end{cases} \tag{2.15b}$$

where $\underline{b}_i$ and $\overline{b}_i$ denote the allowable minimum and maximum shunt susceptances at bus i, respectively.

- $\widehat{I}\boldsymbol{x} \in \mathbb{R}^q$: includes the variables in $\boldsymbol{x}$ that have finite bounds, that is, $\widehat{\boldsymbol{x}} = \boldsymbol{t}$, with $\underline{\boldsymbol{x}} = \underline{\boldsymbol{t}}$ and $\overline{\boldsymbol{x}} = \overline{\boldsymbol{t}}$.

Branch flow constraints can be appropriately incorporated into $\boldsymbol{h}(\boldsymbol{x})$ in one of three forms: (i) in terms of the branch active power flows (P_{ij}), (ii) in terms of the branch reactive power flows (Q_{ij}), or (iii) in terms of the square of the moduli of the branch current flows,

$$\begin{aligned} I_{ij}^2(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) = {} & b_{ij}^{\mathrm{sh}}(b_{ij}^{\mathrm{sh}} + 2b_{ij})(e_i^2 + f_i^2) - 2b_{ij}^{\mathrm{sh}} b_{ij}(f_i f_j + e_i e_j) \\ & + 2b_{ij}^{\mathrm{sh}} g_{ij}(e_j f_i - e_i f_j) + (g_{ij}^2 + b_{ij}^2)\big[(e_i - e_j)^2 + (f_i - f_j)^2\big]. \end{aligned} \tag{2.16}$$

A properly chosen (small) subset of branches $\{(i,j)\} \subset \mathcal{B}$ most likely will account for all branch flow violations. Notice that the right-hand sides of (2.3), (2.4) and (2.16) are quadratic functions of the voltage components $\boldsymbol{e}$ and $\boldsymbol{f}$.

Also, notice that the reactive power injections from shunt capacitors and shunt reactors (b_i) are handled as implicit variables in (2.13b). Shunt susceptances can be modeled as explicit variables too (the usual way indeed). To do so, we express the inequality constraint (2.13b) in $\boldsymbol{h}(\boldsymbol{x})$ as the equality constraint

$$Q_i(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) - b_i + Q_{L_i}^0 - Q_{G_i}^{ED} = 0, \qquad \text{for all } \; i \in \mathcal{E} \tag{2.17}$$

which is, afterwards, incorporated into $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0}$. The following redefinitions are required: $\boldsymbol{x} = (\boldsymbol{e},\boldsymbol{f},\boldsymbol{t},\boldsymbol{b})^T$, $\underline{\boldsymbol{x}} = (\underline{\boldsymbol{t}},\underline{\boldsymbol{b}})^T$ and $\overline{\boldsymbol{x}} = (\overline{\boldsymbol{t}},\overline{\boldsymbol{b}})^T$.

In compact form, the RPD problem in rectangular coordinates can be mathematically stated as follows:

$$\begin{array}{lrcll} \text{minimize} & P_{\mathrm{loss}}(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) & & & \\ \text{subject to:} & & & & \\ & P_i(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) + P_{L_i}^0 - P_{G_i}^{ED} & = & 0, & \text{for all } \; i \in \widetilde{\mathcal{N}}, \\ & Q_i(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) + Q_{L_i}^0 - Q_{G_i}^{ED} & = & 0, & \text{for all } \; i \in \mathcal{F}, \\ \underline{Q}_i \le & Q_i(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) & \le & \overline{Q}_i, & \text{for all } \; i \in \mathcal{G}, \\ \underline{Q}_i \le & Q_i(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) & \le & \overline{Q}_i, & \text{for all } \; i \in \mathcal{E}, \\ \underline{V}_i^2 \le & e_i^2 + f_i^2 & \le & \overline{V}_i^2, & \text{for all } \; i \in \mathcal{N}, \\ -\overline{Q}_{ij} \le & Q_{ij}(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) & \le & \overline{Q}_{ij}, & \text{for a } \{(i,j)\} \subseteq \mathcal{B}, \\ \underline{t}_{ij} \le & t_{ij} & \le & \overline{t}_{ij}, & \text{for all } \; (i,j) \in \mathcal{T}. \end{array} \tag{2.18}$$

REMARK 2.2 *Except for the few terms involving the tap settings* $\boldsymbol{t}$*, which are implicit in some elements of* $\boldsymbol{G}$ *and* $\boldsymbol{B}$*, all nonlinear functions in the RPD problem* (2.18) *are quadratic. Other OPF variants with quadratic objective and constraints are the minimization of reactive power losses in the transmission system,* $\boldsymbol{f}^T\boldsymbol{B}\boldsymbol{f} + \boldsymbol{e}^T\boldsymbol{B}\boldsymbol{e}$*, and the minimization of cost of power generations with quadratic cost curves,* $\sum_{i\in\mathcal{G}}(a_i + b_i P_{G_i} + c_i P_{G_i}^2)$.

REMARK 2.3 *Compared with more highly nonlinear functions, numerical advantages that stem from the quadratic OPF formulation are: (i) the Taylor-series expansion of a quadratic function* $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$ *terminates at the second-order term without truncation error,*

$$f(\boldsymbol{x}^k + \Delta\boldsymbol{x}) = f(\boldsymbol{x}^k) + (\boldsymbol{x}^k)^T\boldsymbol{A}\Delta\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{x}^T\boldsymbol{A}\Delta\boldsymbol{x}, \tag{2.19}$$

(ii) the Hessian matrix of $f(\boldsymbol{x})$ *is constant* ($\nabla^2_{\boldsymbol{x}\boldsymbol{x}} f(\boldsymbol{x}) = \boldsymbol{A}$)*, and (iii) the higher-order term in* (2.19) *is easily computed as* $f(\Delta\boldsymbol{x})$*. Such features allow for ease of matrix setup and inexpensive incorporation of second-order information in higher-order IP methods.*

2.2.2 Polar Coordinates

The RPD problem in polar coordinates can be similarly derived from the NLP problem (1.1), as follows:

- $\boldsymbol{x} \in I\!R^n$: includes the voltage magnitudes $\boldsymbol{v}$ and the voltage phase-angles $\boldsymbol{\theta}$ but θ_1 ($\theta_1 := 0°$), and the transformer tap settings $\boldsymbol{t}$.
- $f : I\!R^n \mapsto I\!R$: can be the transmission active power losses, as given by $P_{\text{loss}}(\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t})$ in (2.10), or the active power injection into the slack bus, as given by $P_1(\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t})$ in (2.6).
- $g : I\!R^n \mapsto I\!R^m$: includes the bus active and reactive power balance constraints

$$P_i(\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t}) + P^0_{L_i} - P^{ED}_{G_i}, \qquad \text{for all } i \in \widetilde{\mathcal{N}}, \tag{2.20a}$$
$$Q_i(\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t}) + Q^0_{L_i} - Q^{ED}_{G_i}, \qquad \text{for all } i \in \mathcal{F}, \tag{2.20b}$$

  occasionally augmented by a few special equality constraints.

- $h : I\!R^n \mapsto I\!R^p$: includes the bus reactive power functional variables

$$Q_i(\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t}), \qquad \text{for all } i \in \mathcal{G}, \tag{2.21a}$$
$$Q_i(\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t}), \qquad \text{for all } i \in \mathcal{E}, \tag{2.21b}$$

with the lower operating limits

$$\underline{h} = \begin{pmatrix} \underline{Q}_i, & \text{for all } i \in \mathcal{G} \\ \underline{Q}_i, & \text{for all } i \in \mathcal{E} \end{pmatrix}, \tag{2.22a}$$

and the upper operating limits

$$\overline{h} = \begin{pmatrix} \overline{Q}_i, & \text{for all } i \in \mathcal{G} \\ \overline{Q}_i, & \text{for all } i \in \mathcal{E} \end{pmatrix}. \tag{2.22b}$$

The lower ($\underline{Q}_i$) and the upper ($\overline{Q}_i$) reactive power limits in (2.22) are as defined in (2.15).

- $\widehat{I}\boldsymbol{x} \in I\!R^q$: includes the variables in $\boldsymbol{x}$ that have finite bounds, that is, $\widehat{\boldsymbol{x}} = (\boldsymbol{v}, \boldsymbol{t})^T$, with $\underline{\boldsymbol{x}} = (\underline{\boldsymbol{v}}, \underline{\boldsymbol{t}})^T$ and $\overline{\boldsymbol{x}} = (\overline{\boldsymbol{v}}, \overline{\boldsymbol{t}})^T$.

Branch flow limits can be incorporated into $\boldsymbol{h}(\boldsymbol{x})$ in one of three ways: (i) in terms of the branch active power flows (P_{ij}), (ii) in terms of the branch reactive power flows (Q_{ij}), or (iii) in terms of the branch apparent power flows (S_{ij}).

In compact form, the RPD problem in polar coordinates can be mathematically stated as follows:

$$\begin{array}{lrcl} \text{minimize} & P_{\text{loss}}(\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t}) & & \\ \text{subject to:} & & & \\ & P_i(\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t}) + P^0_{L_i} - P^{ED}_{G_i} = 0, & & \text{for all } i \in \widetilde{\mathcal{N}}, \\ & Q_i(\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t}) + P^0_{L_i} - Q^{ED}_{G_i} = 0, & & \text{for all } i \in \mathcal{F}, \\ \underline{Q}_i \le & Q_i(\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t}) & \le \overline{Q}_i, & \text{for all } i \in \mathcal{G}, \\ \underline{Q}_i \le & Q_i(\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t}) & \le \overline{Q}_i, & \text{for all } i \in \mathcal{E}, \\ -\overline{Q}_{ij} \le & Q_{ij}(\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t}) & \le \overline{Q}_{ij}, & \text{for a } \{(i,j)\} \subseteq \mathcal{B}, \\ \underline{V}_i \le & V_i & \le \overline{V}_i, & \text{for all } i \in \mathcal{N}, \\ \underline{t}_{ij} \le & t_{ij} & \le \overline{t}_{ij}, & \text{for all } (i,j) \in \mathcal{T}. \end{array} \tag{2.23}$$

REMARK 2.4 *The nonlinear OPF, whether in rectangular or polar coordinates, is non-convex since nonlinear equality constraints and nonlinear functional bounds of the form* $\underline{\boldsymbol{h}} \le \boldsymbol{h}(\boldsymbol{x}) \le \overline{\boldsymbol{h}}$ *(for the shape of the functions* $h_i(\boldsymbol{x})$ *of the OPF problem) cannot define a convex region.*

2.3 Maximum Loadability

In this section, the *maximum loadability* (ML) problem is formulated as an optimization problem. Our interest in this problem is twofold: (i) to provide another example of an OPF

problem with quadratic functions, and (ii) to emphasize the current need for solving OPF problems in a nonlinear manner.

For a given feasible operating state of a power system, the ML problem is to determine the maximum load increase the power system can withstand (either total system load, or the load in a given area of the system, or the load at a particular bus, or the load at a set of buses) while satisfying utility and consumers equipment operation restrictions. As such, the ML procedure is a valuable tool in voltage collapse studies [40]. Depending on how an ML problem is formulated, its solution can be used to define contracts with large power consumers, to define inter-utility transactions, and so forth. A particular formulation of the ML problem, that uses voltage in rectangular coordinates, is mathematically stated as

$$
\begin{array}{lrcll}
\text{minimize} & -\lambda & & & \\
\text{subject to:} & & & & \\
 & P_i(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) + P^0_{L_i}(1+\lambda) - P^{ED}_{G_i} & = & 0, & \text{for all } i \in \tilde{\mathcal{N}}, \\
 & Q_i(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) + Q^0_{L_i}(1+\lambda) - Q^{ED}_{G_i} & = & 0, & \text{for all } i \in \mathcal{F}, \\
\underline{Q}_i \le & Q_i(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) + Q^0_{L_i}\lambda & \le & \overline{Q}_i, & \text{for all } i \in \mathcal{G}, \\
\underline{Q}_i \le & Q_i(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) + Q^0_{L_i}\lambda & \le & \overline{Q}_i, & \text{for all } i \in \mathcal{E}, \\
\underline{V}_i^2 \le & e_i^2 + f_i^2 & \le & \overline{V}_i^2, & \text{for all } i \in \mathcal{N}, \\
\underline{I}_{ij}^2 \le & I_{ij}^2(\boldsymbol{e},\boldsymbol{f},\boldsymbol{t}) & \le & \overline{I}_{ij}^2, & \text{for a } \{(i,j)\} \subseteq \mathcal{B}, \\
\underline{t}_{ij} \le & t_{ij} & \le & \overline{t}_{ij}, & \text{for all } (i,j) \in \mathcal{T},
\end{array}
\tag{2.24}
$$

where λ is the load parameter. The decision vector is defined in (2.24) as $\boldsymbol{x} := (\boldsymbol{e}, \boldsymbol{f}, \boldsymbol{t}, \lambda)^T$. The formulation (2.24) is rather simple; however, it complies with our objectives. Notice that,

- all nonlinear functions in (2.24) are quadratic functions of the voltage components $\boldsymbol{e}$ and $\boldsymbol{f}$; and
- to solve (2.24), we start from a feasible operating state—by this we mean that the constraints in (2.24) can be satisfied for $\lambda = 0$—and then increase the load parameter λ until the feasible set of (2.24) in the space defined by $(\boldsymbol{e}, \boldsymbol{f}, \boldsymbol{t}) \in I\!R^{|\mathcal{N}|} \times I\!R^{|\tilde{\mathcal{N}}|} \times I\!R^{|\mathcal{T}|}$ ceases to exist, at least locally, for the new load level: $(1+\lambda)P^0_{L_i}$ and $(1+\lambda)Q^0_{L_i}$. Therefore, the ML problem deals with highly nonlinear operating conditions.

Yet in connection with the ML formulation (2.24), also notice that,

- the load parameter λ appears in the power equations only, and load increase is allowed

for all buses. However, many applications are concerned with load increase in a subset of buses only or even with load increase at a particular bus;

- a single parameter λ is used for the active ($P^0_{L_i}$) and the reactive ($Q^0_{L_i}$) power loads. This means that the *power factor* of the loads will remain constant. Yet the loads have been modeled as constant power; load modeling, however, plays an important role in voltage collapse analysis [9].

2.4 Minimum Load Shedding

In many situations in the operation of a power system, load shedding schemes are utilized to reduce the actual load to a level that can be safely supplied by available generation and network topology. This kind of situation may happen, for instance, when a heavily loading system undergoes the outage of a major facility such as an important transmission line, transformer, or generation unit. Some situations might be so severe that the power flow equations have no real solution [34]. Load curtailment becomes necessary to restore solvability of the power flow equations and feasibility of operation as well. The *minimum load shedding* (MLS) problem appropriately deals with these situations.

A particular formulation of the MLS problem, that uses voltage in rectangular coordinates, is mathematically stated as

$$
\begin{array}{lrcll}
\text{minimize} & \lambda & & & \\
\text{subject to:} & & & & \\
 & P_i(e,f,t) + P^0_{L_i}(1-\lambda) - P^{ED}_{G_i} & = & 0, & \text{for all } i \in \tilde{\mathcal{N}}, \\
 & Q_i(e,f,t) + Q^0_{L_i}(1-\lambda) - Q^{ED}_{G_i} & = & 0, & \text{for all } i \in \mathcal{F}, \\
\underline{Q}_i \leq & Q_i(e,f,t) - Q^0_{L_i}\lambda & \leq & \overline{Q}_i, & \text{for all } i \in \mathcal{G}, \\
\underline{Q}_i \leq & Q_i(e,f,t) - Q^0_{L_i}\lambda & \leq & \overline{Q}_i, & \text{for all } i \in \mathcal{E}, \\
\underline{V}_i^2 \leq & e_i^2 + f_i^2 & \leq & \overline{V}_i^2, & \text{for all } i \in \mathcal{N}, \\
\underline{I}_{ij}^2 \leq & I_{ij}^2(e,f,t) & \leq & \overline{I}_{ij}^2, & \text{for a } \{(i,j)\} \subseteq \mathcal{B}, \\
\underline{t}_{ij} \leq & t_{ij} & \leq & \overline{t}_{ij}, & \text{for all } (i,j) \in \mathcal{T},
\end{array}
\tag{2.25}
$$

where λ is the load shedding parameter. The decision vector is defined as $\boldsymbol{x} := (\boldsymbol{e}, \boldsymbol{f}, \boldsymbol{t}, \lambda)^T$. Notice that,

- all nonlinear functions in (2.25) are quadratic functions of the voltage components $\boldsymbol{e}$ and $\boldsymbol{f}$;

- a single load shedding parameter λ is used for the active and the reactive power loads of all buses. Individual load shedding parameter for each bus, say, λ_i, should be used instead, in order to the utility service be able to distinguish the importance of one consumer from the other;

- if the power balance equations are the only constraints in (2.25), then the MLS is only concerned with restoring solvability of the power flow equations. In such a case, a solution always exist. However, if operational limits are included, and, in addition, a single parameter λ is considered, then a solution can no longer be guaranteed; the feasible region could be empty for all λ.

2.5 Final Remarks

In this chapter, three variants of the broad class of OPF problems have been described, namely, (i) the reactive power dispatch problem, (ii) the maximum loadability problem, and (iii) the minimum load shedding problem. The RPD problem has been formulated both in rectangular and in polar coordinates. The major differences between the formulations in rectangular and in polar coordinates have been identified, so far, as

- voltage-magnitude bounds are handled as functional bounds in rectangular coordinates,

$$\underline{V}_i^2 \leq e_i^2 + f_i^2 \leq \overline{V}_i^2, \qquad \text{for all } \; i \in \mathcal{N},$$

  and as simple bounds in polar coordinates,

$$\underline{V}_i \leq V_i \leq \overline{V}_i, \qquad \text{for all } \; i \in \mathcal{N}.$$

- the nonlinear functions (objective function and constraints) of the formulations in rectangular coordinates are quadratic functions of the voltage components $\boldsymbol{e}$ and $\boldsymbol{f}$. Such a quadratic feature allows for ease of matrix setup and inexpensive incorporation of second-order information in higher-order IP methods.

Analytical and numerical comparisons of the formulations (2.18) and (2.23) are presented in Chapters 6 and 7.

Chapter 3

An Interior-Point Method for Nonlinear Programming

In this chapter, we describe in great detail the mathematical development of an infeasible primal-dual *interior-point* (IP) method for solving the *nonlinear programming* (NLP) problem (1.1). Thus, the described IP method is suitable for solving the *optimal power flow* (OPF) problems presented in Chapter 2. In the next section, we describe the basic ideas behind the Fiacco and McCormick's (classical) logarithmic barrier method, so that we can point out some differences between the classical approach and the "modern" ones. A short survey of recently developed IP methods for NLP is also presented. In Sections 3.2 through 3.6, we proceed with a detailed derivation of our infeasible primal-dual IP method for NLP.

Specifically, in Section 3.2, we describe a transformed problem on which the IP algorithm operates, establish the first-order optimality conditions for the original and the transformed problems, and present some definitions and considerations. In Section 3.3, we discuss two different analytical approaches for computing the Newton directions: (i) the augmented (natural) equation system, and (ii) a reduced equation system. In Section 3.4, we discuss a number of procedures to compute the step sizes in the Newton directions. This is an important issue in non-convex optimization that is further discussed in Chapter 5. In Section 3.5, we show how the barrier parameter is reduced. Two updating formulae of μ are discussed. In Section 3.6, we describe the convergence test. An outline of the primal-dual IP algorithm is presented in Section 3.7. Finally, in Section 3.8, we make some remarks on the described IP algorithm and highlight some of the contributions.

3.1 Classical and "Modern" Logarithmic Barrier Methods

The *logarithmic barrier function* approach is usually attributed to Frisch [26, 1955], and was developed by Fiacco and McCormick in the 1960s [21, 1968] for solving general inequality constrained problems of the form

$$\text{minimize} \quad f(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{h}(\boldsymbol{x}) \geq \boldsymbol{0}, \tag{3.1}$$

where the scalar function $f : I\!R^n \mapsto I\!R$ and the nonlinear vector-function $\boldsymbol{h} : I\!R^n \mapsto I\!R^p$ are twice continuously differentiable in $\Omega := \{\boldsymbol{x} \in I\!R^n \mid \boldsymbol{h}(\boldsymbol{x}) \geq \boldsymbol{0}\}$. It is assumed that at least one point $\boldsymbol{x}^0$ exists for which $\boldsymbol{h}(\boldsymbol{x}^0) > \boldsymbol{0}$. That is, the region Ω has a nonempty interior.

Fiacco and McCormick's (classical) logarithmic barrier function approach to solve (3.1) incorporates the inequality constraints into the objective function by means of a logarithmic barrier function, to transform the constrained problem into a sequence of unconstrained problems of the form

$$\text{minimize} \quad \left\{ f_\mu(\boldsymbol{x}; \mu^k) := f(\boldsymbol{x}) - \mu^k \sum_{i=1}^{p} \ln h_i(\boldsymbol{x}) \right\}, \tag{3.2}$$

where μ^k is a positive *barrier parameter* that is monotonically decreased to zero as iterations progress, that is, $\mu^0 > \mu^1 > \cdots > \mu^k > \cdots > \mu^\infty = 0$. An outline of the *classical logarithmic barrier algorithm* for solving (3.1) is presented below.

Algorithm 3.1 Fiacco and McCormick's Logarithmic Barrier Algorithm

STEP 0: Choose $\mu^0 > 0$ and an initial point $\boldsymbol{x}^0$ such that $\boldsymbol{h}(\boldsymbol{x}^0) > \boldsymbol{0}$; set $k \leftarrow 0$.

STEP 1: Check whether $\boldsymbol{x}^k$ qualifies as an approximate local minimizer for (3.1). If so, stop with $\boldsymbol{x}^k$ as the solution.

STEP 2: Compute the unconstrained minimum $\boldsymbol{x}(\mu^k) \leftarrow \min f_\mu(\boldsymbol{x}; \mu^k)$. Let $\boldsymbol{x}^{k+1} = \boldsymbol{x}(\mu^k)$.

STEP 3: Choose $\mu^{k+1} < \mu^k$, set $k \leftarrow k + 1$ and return to STEP 1.

Fiacco and McCormick show in [21] that under certain conditions and sufficiently small μ^k, by letting μ^k decrease to zero ($\mu^k \downarrow 0$), the sequence $\{\boldsymbol{x}(\mu^k)\}$ of minimizers of $f_\mu(\boldsymbol{x}; \mu^k)$ forms a continuously differentiable path converging to $\boldsymbol{x}^*$, where $\boldsymbol{x}^*$ is a local minimizer of (3.1). The path defined by $\{\boldsymbol{x}(\mu^k)\}$ is known as the *barrier trajectory*.

Several major difficulties were quickly noted with using the classical logarithmic barrier approach to solve (3.1), as discussed in [75] and elsewhere. Initially, a major problem was

the need to determine an initial feasible point, which can be as difficult as solving the actual problem. A second major problem relates to severe numerical difficulties (with the numerical techniques available at the time), even if the constrained problem (3.1) is well-conditioned. The estimates of the Lagrange multipliers for the active constraints ($h_i(\boldsymbol{x}) = 0$) are calculated by taking ratios of two quantities tending to zero, which is numerically unstable. Along the trajectory that approaches the solution, the Hessian matrix of $f_\mu(\boldsymbol{x}; \mu^k)$ becomes increasingly ill-conditioned, and, in the limit ($\mu^k \downarrow 0$), is singular. Other major difficulties are the need for a very careful line search algorithm, the choice of the initial μ^0, and the subsequent scheme for reducing μ^k at each step. Several modified barrier functions were proposed to remedy these difficulties; see [3] for a survey.

After Karmarkar's announcement, in 1984, of a fast polynomial-time IP method for LP [47], this optimization area received much attention from researchers around the world and, accordingly, experienced an awesome progress both in theory and in practice. We shall not attempt to list these research efforts, which has been done in [32,76] and elsewhere, and restrict our attention to IP methods for general NLP. Recently, the development of primal-dual IP methods for general NLP (see [1, 19, 64, 71, 78]) has been motivated by the superb computational performance of primal-dual IP methods for LP and QP. Wright in [74, 1991] discusses the structure of several IP methods for LP and their extensions to NLP.

Breitfeld and Shanno in [4, 1994], and Shanno, Breitfeld and Samantiraki in [64, 1995] and elsewhere describe several IP algorithms for NLP and present computational results as well. Yamashita and Yabe in [78, 1996] propose a class of primal-dual IP algorithms for nonlinearly constrained problems. They provide proofs of local, superlinear and quadratic convergence of these algorithms, and include some computational results. El-Bakry et al. in [19, 1996] discuss a primal-dual IP method for LP and its extension to general NLP. They claim that their algorithm can be implemented so that it is locally and q-quadratically convergent under only the standard Newton method assumptions. A global convergence theory and some computational experiments are presented too.

Akrotirianakis and Rustem in [1, 1997] also describe an IP method for general NLP. They claim global convergence of the algorithm based on a monotonic decrease of a merit function; computational results are not reported. Byrd, Gilbert and Nocedal in [7, 1996] and Byrd, Hribar and Nocedal in [8, 1997] have developed IP methods for NLP within a *trust region* framework. Vanderbei and Shanno in [71, 1997] describe an IP method for NLP that has shown, in practice, to be robust and efficient. Major modifications, as compared with its like for LP and QP, are the consideration of a merit function and an altered search

direction to assure that a descent direction for the merit function is obtained. Gay, Overton and Wright in [28, 1997] address issues such as merit functions, line search procedures, adjusting the barrier parameter, etc, in implementing a specific primal-dual IP method.

The infeasible primal-dual IP algorithm for NLP that we describe in this chapter shares many similarities with the IP algorithms developed by Wu, Debs and Marsten [77, 1994], Granville [33, 1994], and Martínez, Gómez and Quintana [54, 1996]; all of them are direct extensions from IP methods for LP, and have proven to be computationally efficient when solving large nonlinear OPF problems, despite OPF non-convexity.

3.2 Transformed Problem and Optimality Conditions

The first step in the derivation of our infeasible IP algorithm to solve the NLP problem (1.1) is to add the nonnegative slack variables $(s_1, s_2, s_3, s_4) \in \mathbb{R}^p_+ \times \mathbb{R}^p_+ \times \mathbb{R}^q_+ \times \mathbb{R}^q_+$ to (1.1), in order to transform all inequalities into equalities, as follows:

$$\begin{aligned}
\text{minimize} \quad & f(x) \\
\text{subject to:} \quad & g(x) = 0, \\
& -s_1 - s_2 + \overline{h} - \underline{h} = 0, \\
& -h(x) - s_2 + \overline{h} = 0, \\
& -s_3 - s_4 + \overline{x} - \underline{x} = 0, \\
& -\widehat{I}x - s_4 + \overline{x} = 0, \quad s_1 \geq 0,\ s_2 \geq 0,\ s_3 \geq 0,\ s_4 \geq 0.
\end{aligned} \tag{3.3}$$

The logarithmic barrier approach handles the remaining inequalities in (3.3) implicitly, by incorporating them into logarithmic barrier terms that are appended to the objective function, yielding the following *transformed problem*:

$$\begin{aligned}
\text{minimize} \quad & f(x) - \mu^k \sum_{i=1}^{p} \left(\ln s_{1_i} + \ln s_{2_i} \right) - \mu^k \sum_{i=1}^{q} \left(\ln s_{3_i} + \ln s_{4_i} \right) \\
\text{subject to:} \quad & g(x) = 0, \\
& -s_1 - s_2 + \overline{h} - \underline{h} = 0, \\
& -h(x) - s_2 + \overline{h} = 0, \\
& -s_3 - s_4 + \overline{x} - \underline{x} = 0, \\
& -\widehat{I}x - s_4 + \overline{x} = 0, \quad s_1 > 0,\ s_2 > 0,\ s_3 > 0,\ s_4 > 0,
\end{aligned} \tag{3.4}$$

where $\mu^k > 0$ is the *barrier parameter* that is forced to monotonically decrease to zero.

The strict positivity conditions $(s_1, s_2, s_3, s_4) \in \mathbb{R}^p_{++} \times \mathbb{R}^p_{++} \times \mathbb{R}^q_{++} \times \mathbb{R}^q_{++}$, which are imposed to define the logarithmic terms, are handled implicitly. The sequence of parameters $\mu^0 > \mu^1 > \cdots > \mu^\infty = 0$ generates a sequence of subproblems that are defined by (3.4), with k as the subproblem index. Under certain conditions (see [21]), that include the regularity assumptions, as μ^k approaches zero, the sequence $\{x(\mu^k)\}$ that can be generated by solving the transformed problem (3.4) approaches x^*, where x^* is a local minimizer of (3.3) and, therefore, a local minimizer of the original problem (1.1). The sequence of points $\{x(\mu^k)\}$ is said to define the *barrier trajectory* for the subproblem (3.4).

Necessary optimality conditions for the equality-constrained problem (3.4), with μ^k fixed, can be derived from the *Lagrangian function* $L_\mu(w;\mu^k)$, which is defined as

$$\begin{aligned} L_\mu(w;\mu^k) :=& f(x) - \mu^k \sum_{i=1}^{p} \big(\ln s_{1_i} + \ln s_{2_i} \big) - \mu^k \sum_{i=1}^{q} \big(\ln s_{3_i} + \ln s_{4_i} \big) - y^T g(x) \\ & - z_1^T \big(- s_1 - s_2 + \overline{h} - \underline{h} \big) - z_2^T \big(- h(x) - s_2 + \overline{h} \big) \\ & - z_3^T \big(- s_3 - s_4 + \overline{x} - \underline{x} \big) - z_4^T \big(- \widehat{I} x - s_4 + \overline{x} \big), \end{aligned} \tag{3.5}$$

where $(y, z_1, z_2, z_3, z_4) \in \mathbb{R}^m \times \mathbb{R}^p \times \mathbb{R}^p \times \mathbb{R}^q \times \mathbb{R}^q$ are vectors of *Lagrange multipliers*, commonly called *dual variables*, and $w := (s_1, s_2, s_3, s_4, z_1, z_2, z_3, z_4, x, y) \in \mathbb{R}^r$.

A local minimizer of (3.4) is characterized by a stationary point of $L_\mu(w;\mu^k)$, which must satisfy the *Karush-Kuhn-Tucker* (KKT) first-order necessary conditions [24, 76],

$$\nabla_{s_1} L_\mu = -\mu^k S_1^{-1} u + z_1 = 0, \tag{3.6a}$$

$$\nabla_{s_2} L_\mu = -\mu^k S_2^{-1} u + z_1 + z_2 = 0, \tag{3.6b}$$

$$\nabla_{s_3} L_\mu = -\mu^k S_3^{-1} u + z_3 = 0, \tag{3.6c}$$

$$\nabla_{s_4} L_\mu = -\mu^k S_4^{-1} u + z_3 + z_4 = 0, \tag{3.6d}$$

$$\nabla_{z_1} L_\mu = s_1 + s_2 - \overline{h} + \underline{h} = 0, \tag{3.6e}$$

$$\nabla_{z_2} L_\mu = h(x) + s_2 - \overline{h} = 0, \tag{3.6f}$$

$$\nabla_{z_3} L_\mu = s_3 + s_4 - \overline{x} + \underline{x} = 0, \tag{3.6g}$$

$$\nabla_{z_4} L_\mu = \widehat{I} x + s_4 - \overline{x} = 0, \tag{3.6h}$$

$$\nabla_x L_\mu = \nabla_x f(x) - \nabla_x g(x) y + \nabla_x h(x) z_2 + \widehat{I}^T z_4 = 0, \tag{3.6i}$$

$$\nabla_y L_\mu = -g(x) = 0, \tag{3.6j}$$

where $\nabla_x f : \mathbb{R}^n \mapsto \mathbb{R}^n$ is the gradient of $f(x)$, $\nabla_x g : \mathbb{R}^n \mapsto \mathbb{R}^{n \times m}$ is the transposed Jacobian of $g(x)$, $\nabla_x h : \mathbb{R}^n \mapsto \mathbb{R}^{n \times p}$ is the transposed Jacobian of $h(x)$, $S_1 := \operatorname{diag}(s_{1_i})$, $S_2 := \operatorname{diag}(s_{2_i})$, $S_3 := \operatorname{diag}(s_{3_i})$, $S_4 := \operatorname{diag}(s_{4_i})$, and $u := (1, 1, \ldots, 1)^T$ stands for vectors of appropriate dimensions whose elements are all equal to one.

An interpretation to the KKT equations (3.6) is as follows. The Equations (3.6j) and (3.6e)–(3.6h), together with the implicit conditions $(s_1, s_2, s_3, s_4) \in I\!R^p_+ \times I\!R^p_+ \times I\!R^q_+ \times I\!R^q_+$, ensure *primal feasibility*. The Equation (3.6i), together with the implicit conditions $(z_1, z_1 + z_2, z_3, z_3 + z_4) \in I\!R^p_+ \times I\!R^p_+ \times I\!R^q_+ \times I\!R^q_+$, are referred to as *dual feasibility*. The Equations (3.6a)–(3.6d) are usually called the *μ-complementarity conditions*, perturbations of the standard *complementarity conditions* for (3.3). The *μ-complementarity conditions* can be represented in various mathematically equivalent forms. Of these, the form

$$S_1 z_1 - \mu^k u = 0, \tag{3.7a}$$

$$S_2(z_1 + z_2) - \mu^k u = 0, \tag{3.7b}$$

$$S_3 z_3 - \mu^k u = 0, \tag{3.7c}$$

$$S_4(z_3 + z_4) - \mu^k u = 0, \tag{3.7d}$$

is the "least nonlinear" in the sense that the Hessian matrix $\nabla^2_{ww} L_\mu$ (defined below) is independent of μ^k and asymptotically reflects the condition of the original problem—(1.1)—as $\mu^k \downarrow 0$. As we let μ^k approach zero in the barrier subproblems, the perturbed KKT conditions (3.6) closely approximate the KKT conditions for the NLP problem (3.3).

The complementarity conditions state that none of the complementarity pairs, $(s^*_{1_i}, z^*_{1_i})$, $(s^*_{2_i}, z^*_{1_i} + z^*_{2_i})$, $(s^*_{3_j}, z^*_{3_j})$ and $(s^*_{4_j}, z^*_{3_j} + z^*_{4_j})$, can have both of their arguments nonzero, or, equivalently, that inactive constraints ($s^* \neq 0$) must have a zero multiplier ($z^* = 0$). If there is no i such that $s^*_{1_i} = z^*_{1_i} = 0$ or $s^*_{2_i} = z^*_{1_i} + z^*_{2_i} = 0$, and no j such that $s^*_{3_j} = z^*_{3_j} = 0$ or $s^*_{4_j} = z^*_{3_j} + z^*_{4_j} = 0$, then *strict complementarity* is said to hold. A case where both values are zero is an intermediate state between a constraint being *strongly active* and being *inactive*. These three conditions are illustrated in Figure 3.1. The total residual of the complementarity conditions, defined by

$$\rho := s_1^T z_1 + s_2^T(z_1 + z_2) + s_3^T z_3 + s_4^T(z_3 + z_4), \tag{3.8}$$

is called the *complementarity gap*. If w is both primal and dual feasible, then ρ is usually called the *duality gap*.

A strictly feasible starting point is not mandatory for the primal-dual IP method, but the strict positivity conditions $(s_1, s_2, s_3, s_4) > 0$ and $(z_1, z_1 + z_2, z_3, z_3 + z_4) > 0$ must be satisfied at every point. The IP iterates start from a point w^0 that satisfy the strict positivity conditions; in order to preserve this condition, subsequent IP iterations follow a trajectory in the positive orthant of the space of *complementarity products* $(s_i z_i)$. Feasibility is attained during the iterative process, as optimality is approached or reached.

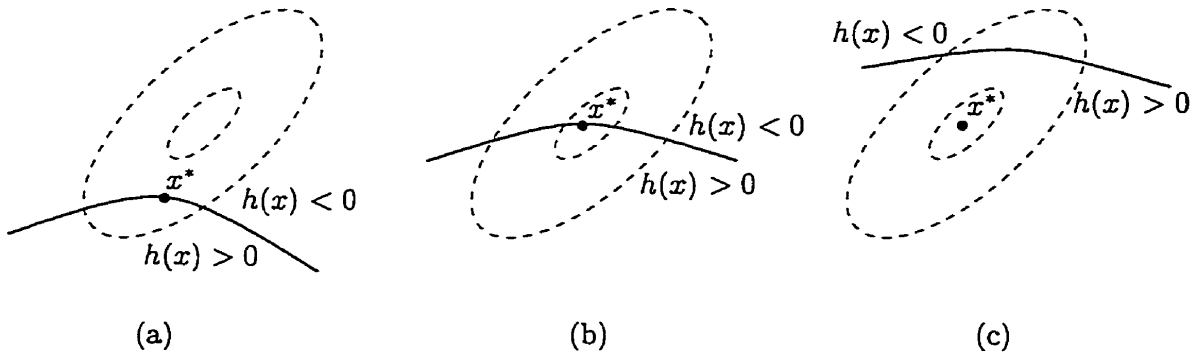

Figure 3.1: Complementarity for $\min_x\{f(x) \mid h(x) \geq 0\}$. Inequality constraint is (a) *strongly active* when $z^* > 0$ and $h(x^*) = 0$, (b) *weakly active* when $z^* = h(x^*) = 0$, and (c) *inactive* when $z^* = 0$ and $h(x^*) > 0$ (Figure 9.1.2 from [24], slightly modified.)

Primal-dual IP iterates invariably apply *one step* of Newton's method for nonlinear equations to the KKT system (3.6), compute a step size in the Newton direction, update the variables, and reduce μ^k. The algorithm terminates when primal infeasibility, dual infeasibility and the complementarity gap fall below predetermined tolerances.

We explain below, in very simple terms, a role played by the barrier parameter μ^k in most primal-dual IP algorithms. Let the KKT conditions for the NLP problem (3.3) be given by

$$\nabla_w L(w) := \begin{pmatrix} S_1 z_1 \\ S_2(z_1 + z_2) \\ S_3 z_3 \\ S_4(z_3 + z_4) \\ s_1 + s_2 - \overline{h} + \underline{h} \\ h(x) + s_2 - \overline{h} \\ s_3 + s_4 - \overline{x} + \underline{x} \\ \widehat{I}x + s_4 - \overline{x} \\ \nabla_x f(x) - \nabla_x g(x) y + \nabla_x h(x) z_2 + \widehat{I}^T z_4 \\ -g(x) \end{pmatrix} = 0, \tag{3.9}$$

$(s_1, s_2, s_3, s_4) \geq 0$ and $(z_1, z_1 + z_2, z_3, z_3 + z_4) \geq 0$. A new estimate for w^k can be computed using *one step* of the Newton's method to find zeros of nonlinear functions, as applied to (3.9). The iterates have the general form

$$w^{k+1} = w^k - \alpha^k [\nabla^2_{ww} L(w^k)]^{-1} \nabla_w L(w^k), \tag{3.10}$$

where $\nabla^2_{ww}L(w^k)$ is the Jacobian of $\nabla_w L(w)$, and $\alpha^k \in (0,1]$ is a damping factor to enhance convergence and keep the nonnegative variables strictly positive instead of just nonnegative. Zhang's approach [80] to explain this necessity is followed closely here.

Consider any of the complementarity equations in the KKT system (3.9), say, $s_{l_i} z_{l_i} = 0$. The Newton equation for $s_{l_i} z_{l_i} = 0$, at a given point $(s^k_{l_i}, z^k_{l_i})$, is

$$s^k_{l_i} \Delta z_{l_i} + z^k_{l_i} \Delta s_{l_i} = -s^k_{l_i} z^k_{l_i}.$$

If one of the variables, say, $z^k_{l_i}$ is zero, then the Newton equation becomes $s^k_{l_i} \Delta z_{l_i} = 0$, leading to a zero update, $\Delta z_{l_i} = 0$. Consequently, $z^k_{l_i}$ will remain zero all the time once it becomes zero, which is fatal because the algorithm will never be able to recover once it sets a variable to zero by "mistake."

Next, consider that the perturbed KKT system (3.6) is expressed in the form

$$\nabla_w L_\mu(w;\mu^k) = \begin{pmatrix} S_1 z_1 \\ S_2(z_1+z_2) \\ S_3 z_3 \\ S_4(z_3+z_4) \\ s_1 + s_2 - \overline{h} + \underline{h} \\ h(x) + s_2 - \overline{h} \\ s_3 + s_4 - \overline{x} + \underline{x} \\ \widehat{I}x + s_4 - \overline{x} \\ \nabla_x f(x) - \nabla_x g(x) y + \nabla_x h(x) z_2 + \widehat{I}^T z_4 \\ -g(x) \end{pmatrix} - \begin{pmatrix} \mu^k u \\ \mu^k u \\ \mu^k u \\ \mu^k u \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = 0. \quad (3.11)$$

Even if we keep the nonnegative variables strictly positive, we would still expect difficulty in recovering from a situation where a variable is adversely set to too small a value. Notice that the perturbed complementarity conditions in (3.11) reduce the chances of such mistakes at early stages by driving all the complementarity pairs to zero at the same pace, say, $s^k_i z^k_i = \mu^k \to 0$ as $k \to \infty$ for every index i. If we express the iterates generated by (3.11) in terms of $\nabla_w L(w^k)$ and $\nabla^2_{ww}L(w^k)$, similar to (3.10), we obtain

$$w^{k+1} = w^k - \alpha^k \left[\nabla^2_{ww}L(w^k)\right]^{-1} [\nabla_w L(w^k) - \mu^k \widehat{u}], \quad (3.12)$$

where $\widehat{u} = (u, 0)$, with $u \in I\!R^{2p+2q}$ and $0 \in I\!R^{r-2p-2q}$. Notice that the search direction in (3.12) has two components: (i) the "pure" Newton direction, $-\left[\nabla^2_{ww}L(w^k)\right]^{-1}\nabla_w L(w^k)$, known as the *affine-scaling direction*, and (ii) the *centering direction*, $\left[\nabla^2_{ww}L(w^k)\right]^{-1}(\mu^k \widehat{u})$, a component that is not seen in (3.10), which drives the nonnegative variables away from the boundary.

3.3 Computing the Newton Directions

Although the KKT system (3.6) is a nonlinear equation system, its solution is usually approximated by a *single* iteration of Newton's method—the Newton direction is only a means to follow the path of minimizers parameterized by μ^k. Such an approximate solution can be obtained either by solving all equations together or by solving an equivalent reduced system. This reduced system is obtained by eliminating variables by substitution. Both solution approaches are described in this section.

3.3.1 Solving the Augmented System

As we take the first-order terms in a Taylor-series approximation of the KKT system (3.6), around the point w^k, we obtain the following symmetric indefinite system:

$$\nabla^2_{ww} L_\mu(w^k) \begin{pmatrix} \Delta s_1 \\ \Delta s_2 \\ \Delta s_3 \\ \Delta s_4 \\ \Delta z_1 \\ \Delta z_2 \\ \Delta z_3 \\ \Delta z_4 \\ \Delta x \\ \Delta y \end{pmatrix} = - \begin{pmatrix} -\mu^k (S_1^k)^{-1} u + z_1^k \\ -\mu^k (S_2^k)^{-1} u + z_1^k + z_2^k \\ -\mu^k (S_3^k)^{-1} u + z_3^k \\ -\mu^k (S_4^k)^{-1} u + z_3^k + z_4^k \\ s_1^k + s_2^k - \overline{h} + \underline{h} \\ h(x^k) + s_2^k - \overline{h} \\ s_3^k + s_4^k - \overline{x} + \underline{x} \\ \widehat{I} x^k + s_4^k - \overline{x} \\ \nabla_x f(x^k) - \nabla_x g(x^k) y^k + \nabla_x h(x^k) z_2^k + \widehat{I}^T z_4^k \\ -g(x^k) \end{pmatrix}, \quad (3.13)$$

where, as we drop most superscripts k,

$$\nabla^2_{ww} L_\mu(w) = \begin{bmatrix} \mu^k S_1^{-2} & 0 & 0 & 0 & I_p & 0 & 0 & 0 & 0 & 0 \\ 0 & \mu^k S_2^{-2} & 0 & 0 & I_p & I_p & 0 & 0 & 0 & 0 \\ 0 & 0 & \mu^k S_3^{-2} & 0 & 0 & 0 & I_q & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu^k S_4^{-2} & 0 & 0 & I_q & I_q & 0 & 0 \\ I_p & I_p & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I_p & 0 & 0 & 0 & 0 & 0 & 0 & \nabla_x h^T & 0 \\ 0 & 0 & I_q & I_q & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I_q & 0 & 0 & 0 & 0 & \widehat{I} & 0 \\ 0 & 0 & 0 & 0 & 0 & \nabla_x h & 0 & \widehat{I}^T & \nabla^2_{xx} L_\mu & -\nabla_x g \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\nabla_x g^T & 0 \end{bmatrix} \quad (3.14)$$

and

$$\nabla^2_{xx} L_\mu(w^k) = \nabla^2_{xx} f(x^k) - \sum_{j=1}^{m} y_j^k \nabla^2_{xx} g_j(x^k) + \sum_{j=1}^{p} z_{2_j}^k \nabla^2_{xx} h_j(x^k). \tag{3.15}$$

The symmetric matrices $\nabla^2_{xx} f(x)$, $\nabla^2_{xx} g_j(x)$ and $\nabla^2_{xx} h_j(x)$ are Hessians of the objective function $f(x)$, the constraint function $g_j(x)$, and the constraint function $h_j(x)$, respectively. To form the Newton system (3.13), computing $\nabla^2_{xx} L_\mu(w^k)$ demands the greatest effort. In an OPF solution, evaluating (3.15) is more efficiently done if we use voltage in rectangular coordinates; in which case, the Hessians $\nabla^2_{xx} f(x)$, $\nabla^2_{xx} g_j(x)$ and $\nabla^2_{xx} h_j(x)$ are constant.

3.3.2 Solving a Reduced System

Let us now consider the solution of the KKT system (3.6) as the solution of an equivalent *reduced system* of equations in terms of x and y. Such a reduced system can be written as

$$d(x, y; \mu^k) = 0, \tag{3.16a}$$

$$-g(x) = 0, \tag{3.16b}$$

where $d(x, y; \mu^k)$ is defined below. To transform the KKT system (3.6) into (3.16), we need to eliminate, by substitution, the variable vectors z_2 and z_4 from the KKT equation (3.6i). These variables are eliminated in the process as we define the Newton system. By taking the first-order terms in a Taylor-series approximation of (3.6i) around w^k, we obtain

$$\begin{aligned} \nabla^2_{xx} L_\mu(w^k)\Delta x - \nabla_x g(x^k)\Delta y + \nabla_x h(x^k)\Delta z_2 + \widehat{I}^T \Delta z_4 \\ \approx -\nabla_x f(x^k) + \nabla_x g(x^k) y^k - \nabla_x h(x^k) z_2^k - \widehat{I}^T z_4^k. \end{aligned} \tag{3.17}$$

To eliminate Δz_2 from (3.17), first we consider the linear approximations that are obtained from the KKT equations (3.6e), (3.6f), (3.6a) and (3.6b), which are

$$\Delta s_2 = -\nabla_x h(x^k)^T \Delta x, \tag{3.18a}$$

$$\Delta s_1 = -\Delta s_2, \tag{3.18b}$$

$$\Delta z_1 = -\mu^k (S_1^k)^{-2} \Delta s_1, \tag{3.18c}$$

$$\Delta z_2 = -\mu^k (S_2^k)^{-2} \Delta s_2 - \Delta z_1. \tag{3.18d}$$

Then, by manipulating the incremental equations (3.18), we express Δz_2 in terms of Δx, as follows:

$$\Delta z_2 = \mu^k \left[(S_1^k)^{-2} + (S_2^k)^{-2} \right] \nabla_x h(x^k)^T \Delta x. \tag{3.19}$$

To eliminate Δz_4 from (3.17), first we consider the linear approximations that are obtained from the KKT equations (3.6g), (3.6h), (3.6c) and (3.6d), which are

$$\Delta s_4 = -\widehat{I}\Delta x, \tag{3.20a}$$

$$\Delta s_3 = -\Delta s_4, \tag{3.20b}$$

$$\Delta z_3 = -\mu^k (S_3^k)^{-2} \Delta s_3, \tag{3.20c}$$

$$\Delta z_4 = -\mu^k (S_4^k)^{-2} \Delta s_4 - \Delta z_3. \tag{3.20d}$$

Then, by manipulating the incremental equations (3.20), we express Δz_4 in terms of Δx, as follows:

$$\Delta z_4 = \mu^k \left[(S_3^k)^{-2} + (S_4^k)^{-2} \right] \widehat{I} \Delta x. \tag{3.21}$$

By substituting the linear approximations (3.19) and (3.21) into (3.17), we obtain

$$\begin{aligned} \left[\nabla^2_{xx} L_\mu(w^k) + \mu^k \nabla_x h(x^k) \left((S_1^k)^{-2} + (S_2^k)^{-2} \right) \nabla_x h(x^k)^T + \mu^k \widehat{I}^T \left((S_3^k)^{-2} + (S_4^k)^{-2} \right) \widehat{I} \right] \Delta x \\ -\nabla_x g(x^k) \Delta y \approx -\nabla_x f(x^k) + \nabla_x g(x^k) y^k - \nabla_x h(x^k) z_2^k - \widehat{I}^T z_4^k \end{aligned} \tag{3.22}$$

Finally, the reduced linear system can be expressed as

$$\begin{bmatrix} \nabla_x d(w^k)^T & -\nabla_x g(x^k) \\ -\nabla_x g(x^k)^T & 0 \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} -d(w^k) \\ g(x^k) \end{pmatrix}, \tag{3.23}$$

where $\nabla_x d : I\!R^r \mapsto I\!R^{n \times n}$ is the transposed Jacobian matrix of the reduced function vector $d(x, y; \mu^k)$, that is evaluated at the current point by

$$\begin{aligned} \nabla_x d(w^k)^T = & \nabla^2_{xx} L_\mu(w^k) + \mu^k \nabla_x h(x^k) \left[(S_1^k)^{-2} + (S_2^k)^{-2} \right] \nabla_x h(x^k)^T \\ & + \mu^k \widehat{I}^T \left[(S_3^k)^{-2} + (S_4^k)^{-2} \right] \widehat{I}, \end{aligned} \tag{3.24}$$

and

$$d(w^k) := \nabla_x L_\mu(w^k) = \nabla_x f(x^k) - \nabla_x g(x^k) y^k + \nabla_x h(x^k) z_2^k + \widehat{I}^T z_4^k. \tag{3.25}$$

The coefficient matrix of the reduced system (3.23) is also symmetric indefinite since the three terms in the right-hand side of (3.24) are symmetric matrices. We use the reduced system (3.23) to examine, in Chapter 7, the handling of voltage bounds in the rectangular and the polar "versions" of the OPF problem..

To compute the Newton direction through the reduced system approach, we first solve for Δx and Δy in (3.23); then, we solve for Δs_2, Δs_1, Δz_1 and Δz_2 using (3.18); and solve for Δs_4, Δs_3, Δz_3 and Δz_4 using (3.20). The numerical solution of the linear indefinite systems (3.13) and (3.23) is discussed in Chapter 6.

3.4 Computing Step Lengths and Updating Variables

New values of the primal and the dual variables are computed from

$$x^{k+1} = x^k + \alpha_P^k \Delta x, \tag{3.26a}$$
$$s_1^{k+1} = s_1^k + \alpha_P^k \Delta s_1, \tag{3.26b}$$
$$s_2^{k+1} = s_2^k + \alpha_P^k \Delta s_2, \tag{3.26c}$$
$$s_3^{k+1} = s_3^k + \alpha_P^k \Delta s_3, \tag{3.26d}$$
$$s_4^{k+1} = s_4^k + \alpha_P^k \Delta s_4, \tag{3.26e}$$

and

$$y^{k+1} = y^k + \alpha_D^k \Delta y, \tag{3.27a}$$
$$z_1^{k+1} = z_1^k + \alpha_D^k \Delta z_1, \tag{3.27b}$$
$$z_2^{k+1} = z_2^k + \alpha_D^k \Delta z_2, \tag{3.27c}$$
$$z_3^{k+1} = z_3^k + \alpha_D^k \Delta z_3, \tag{3.27d}$$
$$z_4^{k+1} = z_4^k + \alpha_D^k \Delta z_4, \tag{3.27e}$$

where the scalars $\alpha_P^k \in (0, 1]$ and $\alpha_D^k \in (0, 1]$ are *step lengths* in the Newton direction in the primal and the dual spaces, respectively.

Desirable features of a step towards a local minimizer are that it simultaneously holds the strict positivity conditions and provides a sufficient decrease in both complementarity gap and infeasibility. However, to properly balance these three goals (generally competing among them) we need to set an appropriate *merit function* and perform a *line search* along the Newton direction, aiming at minimizing this merit function. We discuss merit functions and some line search procedures in Chapter 5. Three computationally inexpensive, however, rather simple, procedures to obtain the step lengths are considered below.

3.4.1 Scheme–A: Separate Primal and Dual Steps

This is the simplest step length procedure and, by far, the most commonly used procedure in implementations of primal-dual IP methods for LP. It considers separate step lengths in the primal and dual spaces, and its primary and only goal is holding the strict positivity conditions. The *primal step length* α_P^k and the *dual step length* α_D^k are both calculated by finding the smallest of the maximum step lengths of all variables with strict positivity

conditions, as follows:

$$\alpha_s^{\max} = \min\left\{ \min_{i=1:p}\left\{ \frac{-s_{1_i}^k}{\Delta s_{1_i}} \middle| \Delta s_{1_i} < 0 \right\}, \min_{i=1:p}\left\{ \frac{-s_{2_i}^k}{\Delta s_{2_i}} \middle| \Delta s_{2_i} < 0 \right\}, \right.$$
$$\left. \min_{i=1:q}\left\{ \frac{-s_{3_i}^k}{\Delta s_{3_i}} \middle| \Delta s_{3_i} < 0 \right\}, \min_{i=1:q}\left\{ \frac{-s_{4_i}^k}{\Delta s_{4_i}} \middle| \Delta s_{4_i} < 0 \right\} \right\}, \tag{3.28}$$

$$\alpha_z^{\max} = \min\left\{ \min_{i=1:p}\left\{ \frac{-z_{1_i}^k}{\Delta z_{1_i}} \middle| \Delta z_{1_i} < 0 \right\}, \min_{i=1:p}\left\{ \frac{-\widehat{z}_{2_i}^k}{\Delta \widehat{z}_{2_i}} \middle| \Delta \widehat{z}_{2_i} < 0 \right\}, \right.$$
$$\left. \min_{i=1:q}\left\{ \frac{-z_{3_i}^k}{\Delta z_{3_i}} \middle| \Delta z_{3_i} < 0 \right\}, \min_{i=1:q}\left\{ \frac{-\widehat{z}_{4_i}^k}{\Delta \widehat{z}_{4_i}} \middle| \Delta \widehat{z}_{4_i} < 0 \right\} \right\}, \tag{3.29}$$

$$\alpha_P^k = \min\left\{1,\ \alpha_0 \alpha_s^{\max}\right\}, \tag{3.30}$$

$$\alpha_D^k = \min\left\{1,\ \alpha_0 \alpha_z^{\max}\right\}, \tag{3.31}$$

where $\widehat{z}_2 := z_1 + z_2$, $\widehat{z}_4 := z_3 + z_4$, and $\alpha_0 \in (0,1)$ is a *step reduction factor* to ensure that the next point will satisfy the strict positivity conditions, that is, $(s_1, s_2, s_3, s_4) > \mathbf{0}$ and $(z_1, \widehat{z}_2, z_3, \widehat{z}_4) > \mathbf{0}$. A commonly used value is $\alpha_0 = 0.99995$; see [52].

3.4.2 Scheme–B: Single Common Step

Separate step lengths in the primal and the dual spaces is an advantage of primal-dual IP methods for LP, and has proven highly efficient in practice, reducing the number of iterations to convergence by 10%–20% on typical problems (see [52]). For general NLP, however, the interdependence of primal and dual variables—as clearly shown in the dual feasibility condition (3.6i)—does not rigorously allows for separate step lengths in the primal and dual spaces. In such a case, a single common step length to update the primal and dual variables shall be computed from

$$\alpha_P^k = \alpha_D^k \leftarrow \min\left\{\alpha_P^k,\ \alpha_D^k\right\}. \tag{3.32}$$

In spite of the above mentioned coupling between primal and dual variables in the dual feasibility conditions, a single common step length (see [77]) and separate step lengths (see [33]) have both performed well in nonlinear OPF solution.

3.4.3 Scheme–C: Dual Box Constraint

This step length procedure is derived from the IP method proposed by Yamashita and Yabe [78, 1996]. The primal step length α_P^k in (3.26) is obtained by the Scheme A above

whereas the dual step length α_D^k in (3.27) is obtained by a different procedure. Following Yamashita and Yabe's method, the dual step length is the largest step $\alpha_D^k \leq 1$ that satisfies

$$\min\left\{\frac{\mu^k}{\varphi^k(s_{1_i}^k + \alpha_P^k \Delta s_{1_i})}, z_{1_i}^k\right\} \leq z_{1_i}^k + \alpha_D^k \Delta z_{1_i} \leq \max\left\{\frac{\vartheta^k \mu^k}{s_{1_i}^k + \alpha_P^k \Delta s_{1_i}}, z_{1_i}^k\right\}, \quad i = 1:p \tag{3.33a}$$

$$\min\left\{\frac{\mu^k}{\varphi^k(s_{2_i}^k + \alpha_P^k \Delta s_{2_i})}, \widehat{z}_{2_i}^k\right\} \leq \widehat{z}_{2_i}^k + \alpha_D^k \Delta \widehat{z}_{2_i} \leq \max\left\{\frac{\vartheta^k \mu^k}{s_{2_i}^k + \alpha_P^k \Delta s_{2_i}}, \widehat{z}_{2_i}^k\right\}, \quad i = 1:p \tag{3.33b}$$

$$\min\left\{\frac{\mu^k}{\varphi^k(s_{3_i}^k + \alpha_P^k \Delta s_{3_i})}, z_{3_i}^k\right\} \leq z_{3_i}^k + \alpha_D^k \Delta z_{3_i} \leq \max\left\{\frac{\vartheta^k \mu^k}{s_{3_i}^k + \alpha_P^k \Delta s_{3_i}}, z_{3_i}^k\right\}, \quad i = 1:q \tag{3.33c}$$

$$\min\left\{\frac{\mu^k}{\varphi^k(s_{4_i}^k + \alpha_P^k \Delta s_{4_i})}, \widehat{z}_{4_i}^k\right\} \leq \widehat{z}_{4_i}^k + \alpha_D^k \Delta \widehat{z}_{4_i} \leq \max\left\{\frac{\vartheta^k \mu^k}{s_{4_i}^k + \alpha_P^k \Delta s_{4_i}}, \widehat{z}_{4_i}^k\right\}, \quad i = 1:q \tag{3.33d}$$

where φ^k and ϑ^k are positive numbers that satisfy

$$\varphi^k > \max\left\{1, \frac{2\mu^k}{(1-\alpha_0)\min\left\{\min\{S_1^k z_1^k\}, \min\{S_2^k \widehat{z}_2^k\}, \min\{S_3^k z_3^k\}, \min\{S_4^k \widehat{z}_4^k\}\right\}}\right\}, \tag{3.34}$$

$$\vartheta^k > \max\left\{3, \frac{3\max\left\{\max\{S_1^k z_1^k\}, \max\{S_2^k \widehat{z}_2^k\}, \max\{S_3^k z_3^k\}, \max\{S_4^k \widehat{z}_4^k\}\right\}}{\mu^k}\right\}. \tag{3.35}$$

Yamashita and Yabe consider their procedure to compute α_D^k as a box constraint for the dual variables.

3.5 Reducing the Barrier Parameter

For general non-convex NLP problems, the choice of a good strategy to reduce μ^k is a very complex issue, often recognized as heuristic and problem dependent [51, 64]. Some procedures simply decrease μ^k by a fixed factor—usually $\mu^{k+1} = \mu^k/10$—up to a given lower bound. Researchers experience has shown that μ^k should not be decreased too fast because this may result in non-convergence.

When a primal-dual IP method is applied to a convex program, the deviation of $f(x(\mu^k))$ from optimality is always bounded by $p\mu^k$ (p is the number of inequality constraints). That is, $f(x(\mu^k)) - f(x^*) \leq p\mu^k$, independent of the particular problem functions [74]. Although the duality properties of convex programming cannot be fully extended to general NLP,

it is natural to ask whether successful schemes used for reducing μ^k in LP or convex QP could be extended to NLP. Such an extension has successfully been considered in [33, 77] and elsewhere, and is considered in this thesis as well.

3.5.1 Standard Procedure to Update μ

We may recall that the *complementarity gap* is computed at the current iterate from

$$\rho^k = (s_1^k)^T z_1^k + (s_2^k)^T \widehat{z}_2^k + (s_3^k)^T z_3^k + (s_4^k)^T \widehat{z}_4^k. \tag{3.36}$$

If the iterates converge to an optimum, then the sequence $\{\rho^k\}$ must converge to zero. The relationship between ρ^k and μ^k, that is implicit in the μ-complementarity equations (3.6a)–(3.6d) and in (3.36), suggests that μ^k could be reduced based on a predicted decrease of the complementarity gap. Most implementations of IP algorithms choose μ^{k+1} from

$$\mu^{k+1} = \sigma^k \frac{\rho^k}{2(p+q)}, \tag{3.37}$$

where σ^k is the expected, but not necessarily realized, decrease in the average (normalized) complementarity.

The parameter $\sigma^k \in [0, 1]$ is usually called a *centering parameter* and can be interpreted as follows [76, Chapter 1]. If $\sigma^k = 1$, the KKT system (3.6) defines a *centering direction*, a Newton step towards a point at the *barrier trajectory*—the *central path* in LP—a smooth trajectory converging to x^* as μ^k continuously goes to zero. Centering directions are usually biased strongly toward the interior of the nonnegative orthant and make little, if any, progress in reducing μ^k. However, by moving closer to the barrier trajectory, they set the scene for substantial progress in the next iteration. At the other extreme, the value $\sigma^k = 0$ gives the pure Newton step, sometimes known as the *affine-scaling direction*.

In computational practice, IP algorithms use intermediate values of σ^k from the open interval $(0, 1)$ to trade off between the twin goals of reducing μ^k and improving centrality. We dynamically choose σ^k as $\sigma^k = \max\{0.99\sigma^{k-1}, 0.1\}$, with $\sigma^0 = 0.2$.

3.5.2 Vanderbei-Shanno's Procedure to Update μ

Vanderbei and Shanno have proposed in [71] a scheme to update μ that takes into account the centrality of the current point. It is well known from theoretical analysis of IP methods that if the trajectory is close to the central path, a small μ may be chosen, whereas when

one is further away from the central path, a larger μ is preferable. Vanderbei and Shanno measure the distance from centrality by computing

$$\xi = \frac{\min\left\{ \min\{S_1 z_1\},\ \min\{S_2 \widehat{z}_2\},\ \min\{S_3 z_3\},\ \min\{S_4 \widehat{z}_4\}\right\}}{\rho/(2p+2q)}. \tag{3.38}$$

Clearly, $0 < \xi \leq 1$, and $\xi = 1$ if and only if the complementarity products $s_i z_i$ are a constant over all indices i. Consequently, they propose the following heuristic to update μ:

$$\mu^{k+1} = \zeta \min\left\{(1-\alpha_0)\frac{1-\xi^k}{\xi^k},\ 2\right\}^3 \frac{\rho^k}{2(p+q)}, \tag{3.39}$$

where $0 < \alpha_0 < 1$ denotes the step reduction factor described above, which, in [71], defaults to 0.95, and ζ is a settable scale factor, which defaults to 0.1.

3.6 Testing for Convergence

We consider the IP iterates terminated whenever an approximate local minimum has been obtained, in which case

$$\nu_1^k \leq \epsilon_1, \tag{3.40a}$$
$$\nu_2^k \leq \epsilon_1, \tag{3.40b}$$
$$\nu_3^k \leq \epsilon_2, \tag{3.40c}$$
$$\nu_4^k \leq \epsilon_2, \tag{3.40d}$$

or they are stuck at some point other than a local minimum (a possibility), in which case

$$\mu^k \leq \epsilon_\mu, \tag{3.41a}$$
$$\|\Delta x\|_\infty \leq \epsilon_2, \tag{3.41b}$$
$$\|g(x^k)\|_\infty \leq \epsilon_1, \tag{3.41c}$$
$$\nu_4^k \leq \epsilon_2, \tag{3.41d}$$

where

$$\nu_1^k = \max\left\{\|g(x^k)\|_\infty, \max\left\{\underline{h} - h(x^k)\right\}, \max\left\{h(x^k) - \overline{h}\right\}, \max\left\{\underline{x} - \widehat{x}^k\right\}, \max\left\{\widehat{x}^k - \overline{x}\right\}\right\}, \tag{3.42}$$

$$\nu_2^k = \frac{\left\|\nabla_x f(x^k) - \nabla_x g(x^k) y^k + \nabla_x h(x^k) z_2^k + \widehat{I}^T z_4^k\right\|_\infty}{1 + \|x^k\| + \|y^k\| + \|z_2^k\| + \|z_4^k\|}, \tag{3.43}$$

$$\nu_3^k = \frac{\rho^k}{1 + \|x^k\|}, \tag{3.44}$$

$$\nu_4^k = \frac{\left|f(x^k) - f(x^{k-1})\right|}{1 + \left|f(x^k)\right|}. \tag{3.45}$$

If the criteria $\nu_1^k \leq \epsilon_1$, $\nu_2^k \leq \epsilon_1$ and $\nu_3^k \leq \epsilon_2$ are satisfied, then the primal feasibility conditions (3.6e)–(3.6h) and (3.6j), the (scaled) dual feasibility condition (3.6i), and the (scaled) complementarity conditions (3.6a)–(3.6d), respectively, are satisfied. When the condition (3.40) is satisfied, the current iterate is a KKT point of accuracy ϵ_1.

When numerical problems prevent verifying the condition (3.40), the algorithm stops as soon as feasibility of the equality constraints is (hopefully) achieved, along with very small fractional change in the objective function value and negligible changes in the variables. That is, the undesirable condition (3.41) is verified.

Yet, the iterates should terminate if neither the condition (3.40) nor the condition (3.41) has been verified, and then either condition $k \geq k^{\max}$ (used maximum number of iterations) or condition $\max\{\alpha_P^k, \alpha_D^k\} < 10^{-12}$ (cannot progress further from w^k) does occur.

Typical convergence tolerance values are $\epsilon_1 = 10^{-4}$, $\epsilon_2 = 10^{-2}\epsilon_1$, and $\epsilon_\mu = 10^{-12}$.

3.7 Outline of the Primal-Dual IP Algorithm

An outline of the primal-dual IP algorithm that is described in this chapter is shown below. It remains to describe the algorithm initialization in STEP 0. Initialization heuristics are described in Chapter 6, along with various important implementation issues.

Algorithm 3.2 Primal-Dual Interior-Point Algorithm.

STEP 0: (Initialization)
Choose $\mu^0 > 0$, and a point w^0 that satisfy the strict positivity conditions $(s_1^0, s_2^0, s_3^0, s_4^0, z_1^0, \widehat{z}_2^0, z_3^0, \widehat{z}_4^0) > 0$; set $k \leftarrow 0$.

STEP 1: (Compute the Newton Direction)
Form the Newton system (3.13) (or the reduced system (3.23)) at the current point w^k and solve for the Newton direction Δw.

STEP 2: (Compute Step Length and Update Variables)
Compute the step length α^k in the direction Δw, and update primal and dual variables: $w^{k+1} \leftarrow w^k + \alpha^k \Delta w$.

STEP 3: (Test Convergence and Update the Barrier Parameter)
If the new point w^{k+1} satisfies the convergence criteria, stop. Otherwise, compute the barrier parameter $\mu^{k+1} < \mu^k$, set $k \leftarrow k+1$, and return to STEP 1.

3.8 Final Remarks

In this chapter, the mathematical development of a primal-dual IP algorithm for NLP has been described in detail. This IP algorithm is a direct extension of the IP method for LP that is described in [52]. Similar extensions have been proposed by Clements et al. [13], Wu et al. [77], Granville [33], Martínez et al. [54], and Irisarri et al. [39]. Although the OPF problem is non-convex and there is no guarantee regarding the convergence of these IP algorithms (as they are implemented) in solving non-convex problems, the successful results described in [13, 33, 39, 54, 77] have to some extent encouraged the current thesis research. In this thesis research, we have conducted the following studies:

- We have developed our primal-dual IP algorithm for NLP based on the NLP problem (1.1), which we consider as the standard form. Application of this IP algorithm to the OPF problems that are described in Chapter 2 is, therefore, straightforward. Towards this purpose, various issues in implementation are discussed in Chapter 6.

- We have studied the computational performance of the primal-dual IP algorithm as it employs different initialization heuristics, different schemes to compute the step lengths, and different updating formulae of the barrier parameter. Also, we have studied the influence of various parameters of the algorithm in the convergence process, as concerned with nonlinear OPF solution.

- We have thoroughly studied the performance of the primal-dual IP algorithm as it solves the RPD problem formulated both in rectangular and in polar coordinates. Such an analysis—rectangular coordinates versus polar coordinates—has not been performed in previous works. We have observed, among other particulars, that voltage bounds are more easily handled in polar coordinates, whereas the assembling of matrices is more efficiently done in rectangular coordinates.

- We have thoroughly described an alternative approach to obtain the Newton direction, called the *reduced system* approach. From such a reduced system, the implications of handling the voltage bounds as nonlinear functional bounds (voltages in rectangular coordinates) can be easily examined as we look at the extent the matrix $\nabla_x d(w)^T$ differs from $\nabla^2_{xx} L_\mu(w)$. Such an analysis is presented in Chapter 7.

Concerning future work with the IP algorithm that is described in this chapter, a study of the usefulness of inexact search directions sounds interesting to study. The idea is to

reduce overall computational time by reducing the effort of a single iteration, even at the expense of some increase in the iteration count. By an inexact search direction we mean that the vector Δw satisfies

$$\nabla^2_{ww} L_\mu(w^k)\Delta w = -\nabla_w L_\mu(w^k) + r^k. \tag{3.46}$$

for some suitable residual vector r^k. As usual in inexact Newton methods, the vector r^k is not fixed beforehand. Instead an iterative solver, such as a preconditioned conjugate gradient method, is used to solve the linear system (3.13); this method is stopped when the norm of the residual is smaller than a prefixed accuracy, that is, $\|r^k\|_2 \leq \epsilon^k$.

Another possibility of reducing the overall computational time is reducing the number of iterations, even at the expense of some increase in the cost of a single iteration. This is the approach followed by the *higher-order* IP variants for LP and convex QP. In the next chapter, we describe extensions to NLP of several higher-order IP methods for LP.

Chapter 4

Higher-Order Primal-Dual Interior-Point Algorithms

Evaluation of the Newton direction is usually the computationally most expensive task in a single iteration of a primal-dual IP algorithm. Concerning the computation of Δw in the IP algorithm that is described in Chapter 3, factorization of the coefficient matrix $\nabla^2_{ww} L_\mu(w^k)$ in (3.13) is much more expensive than the *forward* and *backward* solutions that follow factorization. Thus, we have reasons to think that is possible to improve the performance of the IP algorithm if we reduce the number of matrix factorizations to a necessary minimum, even at the expense of some increase in the cost of a single iteration.

It is likely that a reduction in the number of IP iterations will be accomplished through the incorporation of higher-order information into (3.13) to improve the order of accuracy to which the Newton direction approximates the nonlinear KKT equations. This is the central idea behind the *higher-order* IP variants, such as the *predictor-corrector* method introduced by Kojima, Mizuno and Yoshise [49, 1989] and later developed by Mehrotra [57, 1992]. What makes Mehrotra's method computationally very efficient is that a more successful search direction is obtained by solving two systems of linear equations in each iteration, in a way that allows for a higher-order approximation to the central path.

The two linear system solutions, known as the *predictor* and *corrector* steps, involve a single coefficient matrix with two different right-hand sides; therefore, only one matrix factorization is required. Accordingly, the direction Δw is decomposed into two parts: $\Delta w = \Delta w_{\text{aff}} + \Delta w_{\text{ccc}}$, where Δw_{aff} is called the *affine-scaling* (predictor) direction, and Δw_{ccc} is called the *centering* (corrector) direction. There is little additional work required

to compute the corrector step, if we reuse the matrix factorization required to compute the predictor step.

In LP and QP, additional savings in overall computational time also have been obtained by applying multiple corrector steps, as in the *perturbed composite Newton* (PCN) method of Tapia et al. [67, 1996], in the *multiple predictor-corrector* (MPC) method of Carpenter et al. [10, 1993], and in the *multiple centrality corrections* (MCC) method of Gondzio [31, 1996]. The basic idea behind Tapia's PCN method and Carpenter's MPC method is to perform more solves within each iteration with the intent of performing fewer IP iterations and, accordingly, fewer derivative evaluations and matrix factorizations overall. Gondzio's MCC method also attempts to reduce the number of iterations for convergence by adaptively adding one or more corrector steps to the predictor step; the MCC technique has been included as an option in various state-of-the-art IP software codes [62]. The predictor step used in the PCN, MPC and MCC approaches is the same predictor step used in Mehrotra's method. The use of multiple corrector steps, the engine behind these higher-order IP variants, is advantageous only if it reduces the overall number of derivative evaluations and matrix factorizations without performing an unreasonable number of extra solves.

In this chapter, we explore the usefulness of Mehrotra's predictor-corrector method [57], Tapia's PCN method [67], Carpenter's MPC method [10] and Gondzio's MCC method [31], in the context of nonlinear OPF solution. While the first approach was extended by Wu et al. [77, 1994] to an OPF algorithm in polar coordinates, the last two approaches have not been extended at all to nonlinear OPF solutions. In the next section, we describe an extension of the predictor-corrector method suitable for the OPF in rectangular coordinates. In Section 4.2, we describe extensions of the PCN and MPC methods for nonlinear OPF solution as well. In Section 4.3, we describe an extension of the MCC method. Final remarks and a summary of the contributions close the chapter in Section 4.4.

4.1 Predictor-Corrector Interior-Point Algorithm

To incorporate predictor and corrector steps into the (standard) primal-dual IP algorithm that is described in Chapter 3, we consider the perturbed KKT system (3.6) expressed in the form of (3.11). Such an arrangement provides us with a Newton system whose coefficient matrix is independent of μ^k. Consequently, the predictor and corrector steps involve the same coefficient matrix, thus requiring a single matrix factorization in each iteration. Next, rather than applying Newton's method to (3.11) to generate correction terms to the current

estimate, we substitute the new point $w^{k+1} = w^k + \Delta w$ directly into (3.11), to obtain the second-order approximation

$$\nabla^2_{ww} L_\mu(w^k)\Delta w = -\begin{pmatrix} S_1^k z_1^k \\ S_2^k \widehat{z}_2^k \\ S_3^k z_3^k \\ S_4^k \widehat{z}_4^k \\ s_1^k + s_2^k - \overline{h} + \underline{h} \\ h(x^k) + s_2^k - \overline{h} \\ s_3^k + s_4^k - \overline{x} + \underline{x} \\ \widehat{I} x^k + s_4^k - \overline{x} \\ \nabla_x L_\mu(w^k) \\ -g(x^k) \end{pmatrix} + \begin{pmatrix} \mu^k u \\ \mu^k u \\ \mu^k u \\ \mu^k u \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} \Delta S_1 \Delta z_1 \\ \Delta S_2 \Delta \widehat{z}_2 \\ \Delta S_3 \Delta z_3 \\ \Delta S_4 \Delta \widehat{z}_4 \\ 0 \\ h^q(\Delta x) \\ 0 \\ 0 \\ -\nabla_x g(\Delta x)\Delta y + \nabla_x h(\Delta x)\Delta z_2 \\ -g^q(\Delta x) \end{pmatrix} \tag{4.1}$$

where, as we drop most superscripts k,

$$\nabla^2_{ww} L_\mu(w) = \begin{bmatrix} Z_1 & 0 & 0 & 0 & S_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \widehat{Z}_2 & 0 & 0 & S_2 & S_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & Z_3 & 0 & 0 & 0 & S_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \widehat{Z}_4 & 0 & 0 & S_4 & S_4 & 0 & 0 \\ I_p & I_p & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I_p & 0 & 0 & 0 & 0 & 0 & 0 & \nabla_x h^T & 0 \\ 0 & 0 & I_q & I_q & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I_q & 0 & 0 & 0 & 0 & \widehat{I} & 0 \\ 0 & 0 & 0 & 0 & 0 & \nabla_x h & 0 & \widehat{I}^T & \nabla^2_{xx} L_\mu & -\nabla_x g \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\nabla_x g^T & 0 \end{bmatrix},$$

$Z_1 := \text{diag}(z_{1_i})$, $\widehat{Z}_2 := \text{diag}(\widehat{z}_{2_i})$, $Z_3 = \text{diag}(z_{3_i})$, $\widehat{Z}_4 := \text{diag}(\widehat{z}_{4_i})$, $\Delta S_1 := \text{diag}(\Delta s_{1_i})$, $\Delta S_2 := \text{diag}(\Delta s_{2_i})$, $\Delta S_3 := \text{diag}(\Delta s_{3_i})$, $\Delta S_4 := \text{diag}(\Delta s_{4_i})$, and $g^q(\cdot) \in I\!R^m$ and $h^q(\cdot) \in I\!R^p$ denote the quadratic terms of $g(\cdot)$ and $h(\cdot)$, respectively; we assume here that $g(x)$ and $h(x)$ are quadratic. The coefficient matrix in (4.1) can be made symmetric if we scale the linearized complementarity equations by S_1^{-1}, S_2^{-1}, S_3^{-1} and S_4^{-1}.

REMARK 4.1 *The major difference between the Newton systems* (4.1) *and* (3.13) *is the presence of the nonlinear "delta terms"* $\Delta S_1 \Delta z_1$, $\Delta S_2 \Delta \widehat{z}_2$, $\Delta S_3 \Delta z_3$, $\Delta S_4 \Delta \widehat{z}_4$, $g^q(\Delta x)$, $h^q(\Delta x)$, $\nabla_x g(\Delta x)\Delta y$ *and* $\nabla_x h(\Delta x)\Delta z_2$, *that appear on the right-hand side of* (4.1), *and cannot be solved directly. Moreover, the coefficient matrix in* (4.1) *is independent of* μ^k.

REMARK 4.2 *The second-order terms* $h^q(\Delta x)$, $g^q(\Delta x)$, $\nabla_x g(\Delta x)\Delta y$ *and* $\nabla_x h(\Delta x)\Delta z_2$ *are included only if* $h(x)$ *and* $g(x)$ *are quadratic, as occur in the rectangular coordinates OPF. Otherwise, it might be computationally expensive to include second-order terms. Although the computation of the incremental matrices* $\nabla_x g(\Delta x)$ *and* $\nabla_x h(\Delta x)$ *cost each the equivalent to a sparse matrix-vector product, we do not include the associated correction terms.*

Notice that the search direction obtained from the Newton system (4.1) consists of three components, say

$$\Delta w := \Delta w_{\text{aff}} + \Delta w_{\text{cen}} + \Delta w_{\text{cor}}, \tag{4.2}$$

where each of them is defined by one of the three terms on the right-hand side of (4.1). We can interpret these search direction components as follows [76, Chapter 10]:

- Δw_{aff} is an *affine-scaling direction*, the pure Newton direction that is obtained when we set $\mu^k = 0$ in the Newton systems (3.13) and (4.1). The affine-scaling direction is responsible for "optimization", that is, for reducing primal and dual infeasibility, and the complementarity gap; Δw_{aff} is provided by the first term on the right-hand side of (4.1).

- Δw_{cen} is a *centering direction*, whose size is governed by the adaptively chosen barrier parameter μ^k. The centering direction attempts to keep the current iterate away from the boundary of the feasible region and ideally close to the barrier trajectory, to improve the chances for a long step to be made in the next iteration; Δw_{cen} is provided by the second term on the right-hand side of (4.1).

- Δw_{cor} is a *corrector direction* that attempts to compensate for some of the nonlinearity in the affine-scaling direction; Δw_{cor} is provided by the last term on the right-hand side of (4.1).

The first two components—the affine-scaling direction and centering direction—combine to make up the standard direction computed from (3.13). In Mehrotra's algorithm [57], the affine-scaling direction is computed separately from, and prior to, the centering direction. This arrangement in computation provides us with the ability to choose μ^{k+1} adaptively rather than a priori, and to approximate the second-order *delta terms*. The "full" Newton direction is then computed as the combination of two directions: $\Delta w = \Delta w_{\text{aff}} + \Delta w_{\text{ccc}}$, where $\Delta w_{\text{ccc}} = \Delta w_{\text{cen}} + \Delta w_{\text{cor}}$ is the *combined centering-corrector direction*.

4.1.1 The Predictor Step

To determine a step that approximately satisfies (4.1), we first drop the μ *terms* and the *delta terms* on the right-hand side of (4.1), and compute the *affine-scaling direction* from

$$\nabla^2_{ww} L_\mu(w^k)\Delta w_{\text{aff}} = -\begin{pmatrix} S_1^k z_1^k \\ S_2^k \widehat{z}_2^k \\ S_3^k z_3^k \\ S_4^k \widehat{z}_4^k \\ s_1^k + s_2^k - \overline{h} + \underline{h} \\ h(x^k) + s_2^k - \overline{h} \\ s_3^k + s_4^k - \overline{x} + \underline{x} \\ \widehat{I} x^k + s_4^k - \overline{x} \\ \nabla_x f(x^k) - \nabla_x g(x^k) y^k + \nabla_x h(x^k) z_2^k + \widehat{I}^T z_4^k \\ -g(x^k) \end{pmatrix}. \tag{4.3}$$

The affine-scaling direction obtained from (4.3) is then used in two distinct ways [52]:

- to approximate the nonlinear *delta terms* on the right-hand side of (4.1); and
- to adaptively estimate the barrier parameter μ^{k+1}.

If, on one hand, the affine-scaling direction makes good progress in reducing the barrier parameter μ^k while holding the strict positivity conditions, we conclude that little centering is needed at this iteration, so we assign a small value to σ^k (the centering parameter in (3.37)). If, on the other hand, we can move only a short distance along the affine-scaling direction before violating the strict positivity conditions, we conclude that a significant amount of centering is needed, so we choose σ^k closer to 1.

An estimate to μ^{k+1} is computed as follows. First, we perform the standard ratio test (3.32) to determine the step length α^k_{aff} (we employ the step length Scheme–B) that would actually be taken if the affine-scaling direction given by (4.3) were used:

$$\begin{aligned} \alpha_s^{\text{aff}} = \min\Bigg\{ & \min_{i=1:p}\left\{\frac{-s_{1_i}^k}{\Delta s_{1_i}^{\text{aff}}}\middle| \Delta s_{1_i}^{\text{aff}} < 0\right\}, \min_{i=1:p}\left\{\frac{-s_{2_i}^k}{\Delta s_{2_i}^{\text{aff}}}\middle| \Delta s_{2_i}^{\text{aff}} < 0\right\}, \\ & \min_{i=1:q}\left\{\frac{-s_{3_i}^k}{\Delta s_{3_i}^{\text{aff}}}\middle| \Delta s_{3_i}^{\text{aff}} < 0\right\}, \min_{i=1:q}\left\{\frac{-s_{4_i}^k}{\Delta s_{4_i}^{\text{aff}}}\middle| \Delta s_{4_i}^{\text{aff}} < 0\right\}\Bigg\}, \\ \alpha_P^{\text{aff}} = \min\left\{1,\ \alpha_0 \alpha_s^{\text{aff}}\right\}, \end{aligned} \tag{4.4}$$

$$\alpha_z^{\text{aff}} = \min\left\{ \min_{i=1:\,p}\left\{ \frac{-z_{1_i}^k}{\Delta z_{1_i}^{\text{aff}}} \middle| \Delta z_{1_i}^{\text{aff}} < 0 \right\}, \min_{i=1:\,p}\left\{ \frac{-\widehat{z}_{2_i}^k}{\Delta \widehat{z}_{2_i}^{\text{aff}}} \middle| \Delta \widehat{z}_{2_i}^{\text{aff}} < 0 \right\}, \right.$$
$$\left. \min_{i=1:\,q}\left\{ \frac{-z_{3_i}^k}{\Delta z_{3_i}^{\text{aff}}} \middle| \Delta z_{3_i}^{\text{aff}} < 0 \right\}, \min_{i=1:\,q}\left\{ \frac{-\widehat{z}_{4_i}^k}{\Delta \widehat{z}_{4_i}^{\text{aff}}} \middle| \Delta \widehat{z}_{4_i}^{\text{aff}} < 0 \right\} \right\},$$
$$\alpha_D^{\text{aff}} = \min\left\{1,\ \alpha_0 \alpha_z^{\text{aff}}\right\}, \tag{4.5}$$
$$\alpha_{\text{aff}}^k = \min\left\{\alpha_P^{\text{aff}},\ \alpha_D^{\text{aff}}\right\}. \tag{4.6}$$

Second, an estimate of the complementarity gap is obtained from

$$\rho_{\text{aff}}^k = (s_1^k + \alpha_{\text{aff}}^k \Delta s_1^{\text{aff}})^T (z_1^k + \alpha_{\text{aff}}^k \Delta z_1^{\text{aff}}) + (s_2^k + \alpha_{\text{aff}}^k \Delta s_2^{\text{aff}})^T (\widehat{z}_2^k + \alpha_{\text{aff}}^k \Delta \widehat{z}_2^{\text{aff}})$$
$$+ (s_3^k + \alpha_{\text{aff}}^k \Delta s_3^{\text{aff}})^T (z_3^k + \alpha_{\text{aff}}^k \Delta z_3^{\text{aff}}) + (s_4^k + \alpha_{\text{aff}}^k \Delta s_4^{\text{aff}})^T (\widehat{z}_4^k + \alpha_{\text{aff}}^k \Delta \widehat{z}_4^{\text{aff}}). \tag{4.7}$$

Finally, an estimate of μ^{k+1}, which we call μ_{aff}^k, is obtained from

$$\mu_{\text{aff}}^k = \min\left\{ \left(\frac{\rho_{\text{aff}}^k}{\rho^k} \right)^2,\ 0.2 \right\} \frac{\rho_{\text{aff}}^k}{2(p+q)}. \tag{4.8}$$

where ρ^k is given by (3.36). This scheme chooses μ_{aff}^k to be small when Δw_{aff} produces a large decrease in complementarity, $\rho_{\text{aff}}^k \lll \rho^k$, and chooses μ_{aff}^k to be large otherwise.

4.1.2 The Corrector Step

Rather than computing the combined centering corrector direction Δw_{ccc} and adding it to Δw_{aff}, we compute the "full" Newton direction Δw at once from

$$\nabla_{ww}^2 L_\mu(w^k)\Delta w = -\begin{pmatrix} S_1^k z_1^k - \mu_{\text{aff}}^k u + \Delta S_1^{\text{aff}} \Delta z_1^{\text{aff}} \\ S_2^k \widehat{z}_2^k - \mu_{\text{aff}}^k u + \Delta S_2^{\text{aff}} \Delta \widehat{z}_2^{\text{aff}} \\ S_3^k z_3^k - \mu_{\text{aff}}^k u + \Delta S_3^{\text{aff}} \Delta z_3^{\text{aff}} \\ S_4^k \widehat{z}_4^k - \mu_{\text{aff}}^k u + \Delta S_3^{\text{aff}} \Delta \widehat{z}_4^{\text{aff}} \\ s_1^k + s_2^k - \overline{h} + \underline{h} \\ h(x^k) + s_2^k - \overline{h} + h^q(\Delta x^{\text{aff}}) \\ s_3^k + s_4^k - \overline{x} + \underline{x} \\ \widehat{I} x^k + s_4^k - \overline{x} \\ \nabla_x f(x^k) - \nabla_x g(x^k) y^k + \nabla_x h(x^k) z_2^k + \widehat{I}^T z_4^k \\ -g(x) - g^q(\Delta x^{\text{aff}}) \end{pmatrix}. \tag{4.9}$$

REMARK 4.3 *The variables are not actually updated between the* predictor *and* corrector *steps and, therefore, the linear systems* (4.3) *and* (4.9) *have the same coefficient matrix.*

4.1.3 Outline of the Predictor-Corrector IP Algorithm

The remaining steps in the predictor-corrector algorithm are the same as in the standard primal-dual IP algorithm. That is, we update variables, reduce the barrier parameter, and test for convergence precisely as described in Sections 3.4 through 3.6. An outline of the described predictor-corrector IP algorithm for NLP is shown below.

Algorithm 4.1 Predictor-Corrector Interior-Point Algorithm.

STEP 0: (Initialization)
Choose $\mu^0 > 0$, and a point $\boldsymbol{w}^0$ that satisfy the strict positivity conditions $(s_1^0, s_2^0, s_3^0, s_4^0, z_1^0, \widehat{z}_2^0, z_3^0, \widehat{z}_4^0) > 0$; set $k \leftarrow 0$.

STEP 1: (Compute the Newton Direction)
Compute the coefficient matrix in (4.1) and obtain its factorization. Then,

STEP 1.1: (Predictor Step)

a) compute the right-hand side vector of (4.3);

b) solve (4.3) for the affine-scaling direction $\Delta \boldsymbol{w}_{\text{aff}}$;

c) compute α_{aff}^k and obtain the estimate μ_{aff}^k.

STEP 1.2: (Corrector Step)

a) compute the right-hand side vector of (4.9) by adding the estimated μ_{aff}^k terms and the delta terms to the vector computed in STEP 1.1a;

b) solve (4.9) for the Newton direction $\Delta \boldsymbol{w}$.

STEP 2: (Compute Step Length and Update Variables)
Compute the step length α^k in the direction $\Delta \boldsymbol{w}$, and update primal and dual variables: $\boldsymbol{w}^{k+1} \leftarrow \boldsymbol{w}^k + \alpha^k \Delta \boldsymbol{w}$.

STEP 3: (Test Convergence and Update the Barrier Parameter)
If the new point $\boldsymbol{w}^{k+1}$ satisfies the convergence criteria, stop. Otherwise, compute the barrier parameter $\mu^{k+1} < \mu^k$, set $k \leftarrow k + 1$, and return to STEP 1.

The two linear system solutions in STEP 1.1b and STEP 1.2b of Algorithm 4.1 use the same matrix factorization. Therefore, the extra effort in the predictor-corrector IP algorithm, as compared with the standard primal-dual IP algorithm that is described in Chapter 3, is in the extra linear system solution to compute the affine-scaling direction

and the extra ratio test used to compute μ_{aff}^k. What is gained from this extra work—the additional forward and backward solution steps—is approximate second-order information concerning the trajectory from the current estimate to the optimal point as μ^k is varied continuously. This usually results in reduction in the number of iterations that, in general, translates into overall computational time savings.

The higher-order terms $\boldsymbol{h}(\Delta \boldsymbol{x})$ and $\boldsymbol{g}(\Delta \boldsymbol{x})$, which differ this procedure from that in [77] to solve the OPF in polar coordinates, are incorporated only in the case $\boldsymbol{h}(\boldsymbol{x})$ and $\boldsymbol{g}(\boldsymbol{x})$ are quadratic. Also, we have taken a *full* step in the delta terms of the corrector step but the damped step when predicting μ_{aff}^k.

4.2 Perturbed Composite Newton and Multiple Predictor-Corrector Algorithms

We may recall that Mehrotra's predictor-corrector method performs only one corrector step in obtaining the search direction in each iteration. The PCN method and the MPC method that we describe in this section perform one or more corrector steps within each iteration with the intent of performing less iterations than Mehrotra's method does. That is, the PCN and the MPC methods aim at exploiting the derivatives and the factorization that are required in the predictor step further in a sequence of solves of systems like (4.9) with different right-hand sides. As show in [10], the MPC method for LP and QP is equivalent to a perturbed level-M PCN method, where M is the number of corrector steps. The MPC and PCN algorithms for LP and QP are equivalents in the sense that they yield the same sequence of iterates when started from the same initial point. Such an equivalence of algorithms, however, is not observed for general NLP.

We begin this section describing the fundamental ideas behind the standard composite Newton method for solving nonlinear equations.

4.2.1 The Composite Newton Method: Fundamentals

Consider a nonlinear system of equations (such as the KKT equations) expressed as

$$\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}, \tag{4.10}$$

where $\boldsymbol{F} : I\!R^n \to I\!R^n$ is a vector of continuously differentiable functions. By a *damped Newton's method* we mean an iterative procedure that at each iteration solves the linear

system

$$\nabla_x F(x^k)^T \Delta x = -F(x^k) \tag{4.11}$$

for the direction Δx, and then moves to a new point

$$x^{k+1} = x^k + \alpha^k \Delta x, \tag{4.12}$$

where $\nabla_x F : \mathbb{R}^n \to \mathbb{R}^{n \times n}$ is the transposed Jacobian of $F(x)$, and $\alpha \in (0, 1]$ is the step length parameter. When the choice of step length is $\alpha^k = 1$ we drop the qualifier *damped.*

Since the computation and factorization of the Jacobian matrix $\nabla_x F(x^k)^T$ demand the greatest computational effort within an iteration, it may be advantageous to use the same derivative evaluation and matrix factorization in several solves. This is the idea behind the composite Newton method. At each iteration, the damped level-M composite Newton method first solves the system

$$\nabla_x F(x^k)^T \Delta x^0 = -F(x^k) \tag{4.13}$$

for the direction Δx^0. Afterwards, for $m_k = 1, 2, \ldots, M$, it solves the systems

$$\nabla_x F(x^k)^T \Delta x^{m_k} = -F\left(x^k + \sum_{j=0}^{m_k - 1} \Delta x^j\right) \tag{4.14}$$

for the directions Δx^{m_k}, and only then takes the step

$$x^{k+1} = x^k + \alpha^k \sum_{j=0}^{M} \Delta x^j. \tag{4.15}$$

Notice that the Jacobian matrix $\nabla_x F(x^k)^T$ is employed $M + 1$ times to iteratively obtain the search direction, before a step is actually taken.

4.2.2 The Perturbed Composite Newton Interior-Point Algorithm

The above procedure can easily be incorporated into the solution of the perturbed KKT equations (3.11). Let us assume that the predictor direction Δw_{aff} and the estimate of the barrier parameter μ^k_{aff} have been computed as in the predictor step of Mehrotra's method; let $\Delta w^0 = \Delta w_{\text{aff}}$. Then, for $m_k = 1, 2, \ldots, M$, solve the systems

$$\nabla^2_{ww} L_\mu(w^k) \Delta w^{m_k} = -\nabla_w L_\mu\left(w^k + \sum_{j=0}^{m_k - 1} \Delta w^j\right) + \mu^k_{\text{aff}} \widehat{u} \tag{4.16}$$

for the directions Δw^{m_k}, where $\widehat{u} = (u, 0)$ with $u \in \mathbb{R}^{2p+2q}$ and $0 \in \mathbb{R}^{r-2p-2q}$. Finally, define $\Delta w = \sum_{j=0}^{M} \Delta w^j$ and move to a new point, computing the step length and updating the variables precisely as in the primal-dual IP method and in Mehrotra's method.

4.2.3 Outline of the Perturbed Composite Newton IP Algorithm

An outline of the perturbed composite Newton IP algorithm for NLP is shown below. The outlined PCN algorithm considers the possibility of taking a varied number of composite Newton steps within the outer iterations. Defining how much correcting is advantageous is a critical issue. Below, we describe a procedure for dynamically choosing the maximum m_k in the MPC method, which may be used in the PCN method as well.

Algorithm 4.2 Perturbed Composite Newton Interior-Point Algorithm.

STEP 0: (Initialization)
Choose $\mu^0 > 0$, and a point w^0 that satisfy the strict positivity conditions $(s_1^0, s_2^0, s_3^0, s_4^0, z_1^0, \widehat{z}_2^0, z_3^0, \widehat{z}_4^0) > \mathbf{0}$; set $k \leftarrow 0$.

STEP 1: (Compute the Search Direction)
Compute the coefficient matrix in (4.1) and obtain its factorization. Then,

STEP 1.1: (Predictor Step)

a) compute the right-hand side of (4.3);

b) solve (4.3) for the predictor direction Δw_{aff};

c) compute α_{aff}^k and obtain the estimate μ_{aff}^k;

d) set the corrections counter $m_k \leftarrow 0$, and let $\Delta w^0 = \Delta w_{\text{aff}}$.

STEP 1.2: (Composite Newton Steps)

a) compute the right-hand side of (4.16);

b) solve (4.16) for the corrector direction Δw^{m_k};

c) test for improvement. If suitable, set $m_k \leftarrow m_k + 1$ and return to STEP 1.2a. Otherwise, set $\Delta w \leftarrow \sum_{j=0}^{m_k} \Delta w^j$.

STEP 2: (Compute Step Length and Update Variables)
Compute the step length α^k in the direction Δw, and update primal and dual variables: $w^{k+1} \leftarrow w^k + \alpha^k \Delta w$.

STEP 3: (Test Convergence and Update the Barrier Parameter)
If the new point w^{k+1} satisfies the convergence criteria, stop. Otherwise, compute the barrier parameter $\mu^{k+1} < \mu^k$, set $k \leftarrow k + 1$, and return to STEP 1.

Notice that we have used the same value of the predicted barrier parameter, μ_{aff}^k, in all corrector steps. Alternatively, as suggested by Tapia et al. in [67], the barrier parameter could be reset at every corrector step. In such a case, we would solve the systems

$$\nabla_{ww}^2 L_\mu(w^k)\Delta w^{m_k} = -\nabla_w L_\mu\left(w^k + \sum_{j=0}^{m_k-1} \Delta w^j\right) + \mu^{m_k}\widehat{u}, \tag{4.17}$$

for $m_k = 1, 2, \ldots, M$.

4.2.4 The Multiple Predictor-Corrector Interior-Point Algorithm

The predictor step in the MPC method computes Δw_{aff} and μ_{aff}^k in the same way as in the predictor step of Mehrotra's predictor-corrector method. Then, we let $\Delta w^0 = \Delta w_{\text{aff}}$ and compute the m_kth corrector term as follows:

$$\nabla_{ww}^2 L_\mu(w^k)\Delta w^{m_k} = -\begin{pmatrix} S_1^k z_1^k - \mu_{\text{aff}}^k u + \Delta S_1^{m_k-1}\Delta z_1^{m_k-1} \\ S_2^k \widehat{z}_2^k - \mu_{\text{aff}}^k u + \Delta S_2^{m_k-1}\Delta \widehat{z}_2^{m_k-1} \\ S_3^k z_3^k - \mu_{\text{aff}}^k u + \Delta S_3^{m_k-1}\Delta z_3^{m_k-1} \\ S_4^k \widehat{z}_4^k - \mu_{\text{aff}}^k u + \Delta S_4^{m_k-1}\Delta \widehat{z}_4^{m_k-1} \\ s_1^k + s_2^k - \overline{h} + \underline{h} \\ h(x^k) + s_2^k - \overline{h} + h^q(\Delta x^{m_k}) \\ s_3^k + s_4^k - \overline{x} + \underline{x} \\ \widehat{I}x^k + s_4^k - \overline{x} \\ \nabla_x f(x^k) - \nabla_x g(x^k) y^k + \nabla_x h(x^k) z_2^k + \widehat{I}^T z_4^k \\ -g(x^k) - g^q(\Delta x^{m_k}) \end{pmatrix}. \tag{4.18}$$

We allow for the number of corrections m_k to vary at each iteration by dynamically choosing m_k, as in Carpenter's MPC algorithm. Notice that when $m_k = 1$ for all k, the MPC method is the predictor-corrector method described above. Determining how much correcting is advantageous is a critical issue of the MPC algorithm. To address this issue in the context of LP and QP, Carpenter investigated two distinct and complementary cases: the feasible case and the infeasible ones.

In the feasible case, the sole motivation behind performing corrections from a feasible point is reducing complementarity. In such a case, Carpenter et al. consider performing the $(m_k + 1)$st correction only if $\rho^{m_k} < \rho^{m_k-1}$ and m_k is less than some maximum number of corrections (a prespecified parameter). If it is true that $\rho^{m_k} \geq \rho^{m_k-1}$, they stop correcting and use the search direction $\Delta w = \Delta w^{m_k-1}$. In the infeasible case, determining m_k must

integrate reducing complementarity with reducing infeasibility. In this case, an $(m_k + 1)$st correction is attempted only if m_k is less than the allowable maximum and $G^{m_k} < G^{m_k-1}$, where G is the norm of the residual of the KKT conditions.

4.2.5 Outline of the Multiple Predictor-Corrector IP Algorithm

An outline of the multiple predictor-corrector IP algorithm for NLP is shown below.

Algorithm 4.3 Multiple Predictor-Corrector Interior-Point Algorithm.

STEP 0: (Initialization)
Choose $\mu^0 > 0$, and a point w^0 that satisfy the strict positivity conditions $(s_1^0, s_2^0, s_3^0, s_4^0, z_1^0, \hat{z}_2^0, z_3^0, \hat{z}_4^0) > \mathbf{0}$; set $k \leftarrow 0$.

STEP 1: (Compute the Search Direction)
Compute the coefficient matrix in (4.1) and obtain its factorization. Then,

STEP 1.1: (Predictor Step)

a) compute the right-hand side of (4.3);

b) solve (4.3) for the predictor direction Δw_{aff};

c) compute α_{aff}^k and obtain the estimate μ_{aff}^k;

d) set the corrections counter $m_k \leftarrow 0$, and let $\Delta w^0 = \Delta w_{\text{aff}}$.

STEP 1.2: (Multiple Corrector Steps)

a) compute the right-hand side of (4.18);

b) solve (4.18) for the corrector direction Δw^{m_k};

c) compute the step length for the direction Δw^{m_k};

d) test for improvement. If suitable, set $m_k \leftarrow m_k + 1$ and return to STEP 1.2a. Otherwise, set $\Delta w \leftarrow \Delta w^{m_k}$.

STEP 2: (Compute Step Length and Update Variables)
Compute the step length α^k in the direction Δw, and update primal and dual variables: $w^{k+1} \leftarrow w^k + \alpha^k \Delta w$.

STEP 3: (Test Convergence and Update the Barrier Parameter)
If the new point w^{k+1} satisfies the convergence criteria, stop. Otherwise, compute the barrier parameter $\mu^{k+1} < \mu^k$, set $k \leftarrow k + 1$, and return to STEP 1.

4.3 Multiple Centrality Corrections Algorithm

Gondzio's MCC primal-dual IP algorithm for LP [31] uses the *predictor direction* that is obtained in Mehrotra's predictor-corrector method and then looks for one or more corrector terms aiming at two main goals: (i) improving the *centrality* of the next iterate, and (ii) increasing step lengths in the primal and dual spaces. The motivation for the first goal is to increase the chances for a long step to be taken in the next iteration, and, for the second goal, to obtain a faster reduction of primal and dual infeasibility; all together, to obtain acceleration of the convergence.

To achieve the goals (i) and (ii), the MCC method first enlarges the step lengths in both spaces—$\widetilde{\alpha}_P^k = \min\{\alpha_P^k + \delta_\alpha, 1\}$ and $\widetilde{\alpha}_D^k = \min\{\alpha_D^k + \delta_\alpha, 1\}$—and then makes a hypothetical further move along the predictor direction to the so-called *trial point*. This move shall violate, in general, the non-negativity conditions $(s_1, s_2, s_3, s_4) \in I\!R_+^p \times I\!R_+^p \times I\!R_+^q \times I\!R_+^q$ and $(z_1, \widehat{z}_2, z_3, \widehat{z}_4) \in I\!R_+^p \times I\!R_+^p \times I\!R_+^q \times I\!R_+^q$. Then, a corrector direction is defined to drive from this *trial point* towards some better centered *target*. This *target* is some point in a large neighborhood of the central path that is expected to be easier to reach and that allows for a long step to be made in the Newton direction.

The idea of following a sequence of *traceable targets*, called *weighted analytic centers*, as means of improving the centrality of subsequent iterates, was first proposed by Jansen et al. [43, 1993], and later translated into a successful computational practice by Gondzio [31, 1996] through his MCC technique. Below, we describe an extension to NLP of Gondzio's MCC technique for LP.

4.3.1 The Centrality Corrections

Gondzio discusses in [31] that, in theory, only the perfectly centered points allow for long steps to be made in the Newton direction, but, in practice, long steps are observed also for points that belong to a large neighborhood of the central path. Gondzio also observes, based on extensive computational experience, that what really reduces the efficiency of a primal-dual IP algorithm is a large discrepancy among the *complementarity products* $s_i z_i$. That is, we have $s_i z_i \lll s_j z_j$ for some indices i and j.

Complementarity products that are either too small or too large compared with their average $\mu_{\text{ave}} = \rho/2(p+q)$ are undesirable, with the former usually being more disastrous. An explanation for that is as follows. The step Δw of (4.1) aims at driving all complementarity products to the same value μ^k. However, to reduce the complementarity gap we need

$\mu^k < \mu_{\text{ave}}$. Thus, if the current iterate is badly centered, that is, the complementarity products differ in orders of magnitude, then the right-hand side of system (4.1) is very badly scaled. The Newton direction concentrates on reducing large products, but, due to the presence of smaller ones, only short steps are allowed, which slow down the convergence.

To define the centrality corrections, we first assume that the predictor direction Δw_{aff} has been determined and that the step length α^k_{aff} that would be taken if Δw_{aff} were used has been computed, as described in Section 4.2.1. Then, we look for a corrector direction Δw_{cor} such that a step length $\widetilde{\alpha}^k$ larger than α^k_{aff},

$$\widetilde{\alpha}^k := \min\left\{\alpha^k_{\text{aff}} + \delta_\alpha,\ 1\right\}, \tag{4.19}$$

and a composite direction

$$\Delta w := \Delta w_{\text{aff}} + \Delta w_{\text{cor}}, \tag{4.20}$$

can be taken without violating the non-negativity conditions. To make this possible, some requirements have to be imposed on Δw_{cor}. Notice from the definition of α^k_{aff} that, whenever $\alpha^k_{\text{aff}} < 1$, the trial point

$$\widetilde{w} = w^k + \widetilde{\alpha}^k \Delta w \tag{4.21}$$

may have components that violate the non-negativity conditions. Then, the corrector term Δw_{cor} shall compensate for those negative components and drive the trial point $\widetilde{w}$ back to the vicinity of the barrier trajectory (the central path in LP).

In the MCC approach, the effort of multiple corrections does not primarily concentrates on reducing the complementarity gap, that hopefully will be sufficiently reduced if a long step along the Newton direction is made. Nevertheless, to allow for long steps in the Newton direction the current point should be as close as possible to the central path. Towards this purpose, Gondzio suggests to define a sequence of *traceable targets* that goes from an arbitrary point such as $\widetilde{w}$ to a point close to the central path; Jansen et al. [43] suggest that these targets be defined in the space of complementarity products.

Below, we describe how the targets are defined. Given a small increase of step length δ_α, we define the enlarged step length $\widetilde{\alpha}^k$ in (4.19), obtain the trial point $\widetilde{w}$ in (4.21), and then compute the complementarity products for this trial point,

$$\widetilde{c}_i := \widetilde{s}_i \widetilde{z}_i, \qquad \text{for every } i. \tag{4.22}$$

Next, we identify the components $\widetilde{c}_i$ that do not belong to the interval $(\beta_{\min}\mu^k_{\text{aff}},\ \beta_{\max}\mu^k_{\text{aff}})$. These components are called the *outliers* complementarity products, and $\beta_{\min}$ and $\beta_{\max}$

are given relative threshold values to define these outliers products. The effort is focused on correcting only outliers complementarity products. To this end, the components $\widetilde{c}_i$ are projected on a hypercube $H := [\beta_{\min}\mu_{\text{aff}}^k, \beta_{\max}\mu_{\text{aff}}^k]^{2p+2q}$ to define the target

$$c_i := \begin{cases} \beta_{\min}\mu_{\text{aff}}^k, & \text{if } \widetilde{c}_i < \beta_{\min}\mu_{\text{aff}}^k, \\ \beta_{\max}\mu_{\text{aff}}^k, & \text{if } \widetilde{c}_i > \beta_{\max}\mu_{\text{aff}}^k, \\ \widetilde{c}_i, & \text{otherwise.} \end{cases} \tag{4.23}$$

Then, a corrector term Δw^{m_k} is obtained as the solution to the linear equations system

$$\nabla^2_{ww} L_\mu(w^k)\Delta w^{m_k} = \begin{pmatrix} c_1 - \widetilde{c}_1 \\ c_2 - \widetilde{c}_2 \\ c_3 - \widetilde{c}_3 \\ c_4 - \widetilde{c}_4 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{4.24}$$

The right-hand side vector of (4.24) has nonzero elements only in a subset of positions of $c_i - \widetilde{c}_i$ that refer to the complementarity products that do not belong to $(\beta_{\min}\mu_{\text{aff}}^k, \beta_{\max}\mu_{\text{aff}}^k)$. Furthermore, such a defined right-hand side vector can still remain badly scaled if there are very large complementarity products in $\widetilde{c}_i$. To prevent the undesirable effect of this bad scaling, all components of $c_i - \widetilde{c}_i$ smaller than $-\beta_{\max}\mu_{\text{aff}}^k$ are, in Gondzio's MCC implementation for LP, replaced with this value, which corresponds to limiting the expected decrease of very large complementarity products.

The modified centering direction Δw^{m_k} that solves (4.24) is used to correct the predictor direction, as follows:

$$\Delta w = \Delta w_{\text{aff}} + \Delta w^{m_k}. \tag{4.25}$$

A new step length in the direction Δw is determined, and new values for the primal and dual variables are computed, as previously described for the standard primal-dual and the predictor-corrector IP algorithms.

The correcting process can be repeated a desirable number of times. The direction Δw in (4.25) becomes in such a case a new predictor, $\Delta w_{\text{aff}} \leftarrow \Delta w$, for which a new trial point

is computed from (4.21). The point (4.22) in the complementarity product space is then used to define the new target (4.23). Next, a new modified centering direction Δw^{m_k} that solves (4.24) is computed and added to the predictor term, as in (4.25). In such a case, the corrector term added to Mehrotra's predictor direction is given by $\Delta w_{\text{ccc}} = \sum_{m_k} \Delta w^{m_k}$.

4.3.2 How Many Correction Steps Are Ideal?

Use of multiple centrality correction steps is of practical interest only if reduction in the iteration count translates into overall computational time savings. Then, it is essential to monitor the improvement resulting from the use of the modified centering directions Δw^{m_k} in (4.24). In Gondzio's MCC implementation to LP, correcting terminates when the step lengths in the primal and dual spaces—$\widehat{\alpha}_P$ and $\widehat{\alpha}_D$—determined for a composite direction Δw in (4.25) do not increase sufficiently compared with the step lengths found earlier for a predictor direction—$\widehat{\alpha}_P^{\text{aff}}$ and $\widehat{\alpha}_D^{\text{aff}}$. Following this procedure, we stop correcting if

$$\widehat{\alpha}_P < \alpha_P^{\text{aff}} + \gamma\delta_\alpha \qquad \text{or} \qquad \widehat{\alpha}_D < \alpha_D^{\text{aff}} + \gamma\delta_\alpha, \tag{4.26}$$

where γ is some prescribed tolerance.

Since multiple corrector steps reduce the number of iterations at the expense of extra effort per iteration, savings in overall computational time is influenced by two important factors: (i) decrease of the iterations count (matrix factorizations!), and (ii) the ratio of the costs of factorization and solution of the KKT system. Then, besides condition (4.26) it is necessary to limit the number of centrality corrections (M) per iteration.

Gondzio developed a heuristic to define M based on extensive computational experiments. He computes the ratio of the *factorization effort* to *solve effort*, $r_{f/s}$, and allows one centrality corrector if $r_{f/s} > 10$. If $r_{f/s} \leq 10$, then no centrality corrector is added, so the method reduces to the Mehrotra's predictor-corrector method. Due to the smaller expected savings with subsequent corrections, the use of the second centrality corrector is allowed only if $r_{f/s} > 30$. The third correction is allowed if $r_{f/s} > 50$. More than three correctors are allowed only for problems with very expensive factorizations, and M is never allowed to exceed 10.

We have not thoroughly addressed the choice of M in this research. In defining M, we should consider that forming and solving the KKT system in NLP demand much more effort than in LP; besides the computation of Jacobians and Hessians, the KKT system to be solved is indefinite. In order to define $r_{f/s}$ using actual computational times we could apply the predictor-corrector method in the first iteration and, then, proceed with

the MCC approach thereafter. In this way we would be able to consider other issues, such as intermediate matrix and vector computations, indirectly and directly accessed data, particularities of computer architecture, and so forth.

4.3.3 Outline of the Multiple Centrality Corrections IP Algorithm

An outline of the multiple centrality corrections IP algorithm for NLP is shown below.

Algorithm 4.4 Multiple Centrality Corrections Interior-Point Algorithm.

STEP 0: (Initialization)
Choose $\mu^0 > 0$, and a point w^0 that satisfy the strict positivity conditions $(s_1^0, s_2^0, s_3^0, s_4^0, z_1^0, \widehat{z}_2^0, z_3^0, \widehat{z}_4^0) > 0$; set $k \leftarrow 0$.

STEP 1: (Compute the Search Direction)
Compute the coefficient matrix in (4.1) and obtain its factorization. Then,

STEP 1.1: (Predictor Step)

a) compute the right-hand side of (4.3);

b) solve (4.3) for the predictor direction Δw_{aff};

c) compute α_{aff}^k and obtain the estimate μ_{aff}^k;

d) set the counter $m_k \leftarrow 0$. If $k = 1$, measure $r_{f/s}$ and define M.

STEP 1.2: (Multiple Corrector Steps)

a) compute the trial point (4.21) and the right-hand side of (4.24);

b) solve (4.24) for the corrector direction Δw^{m_k};

c) compute the step length for a composite direction (4.25);

d) test for improvement. If suitable, set $\Delta w_{\text{aff}} \leftarrow \Delta w$, $m_k \leftarrow m_k + 1$, and return to STEP 1.2a. Otherwise, set $\Delta w \leftarrow \Delta w_{\text{aff}}$.

STEP 2: (Compute Step Length and Update Variables)
Compute the step length α^k in the direction Δw, and update primal and dual variables: $w^{k+1} \leftarrow w^k + \alpha^k \Delta w$.

STEP 3: (Test Convergence and Update the Barrier Parameter)
If the new point w^{k+1} satisfies the convergence criteria, stop. Otherwise, compute the barrier parameter $\mu^{k+1} < \mu^k$, set $k \leftarrow k + 1$, and return to STEP 1.

4.4 Final Remarks

In this chapter, we have presented extensions to NLP of four successful higher-order IP methods for LP and QP, namely, (i) the *predictor-corrector* method developed by Mehrotra [57, 1992], (ii) the *perturbed composite Newton* method described by Tapia et al. [67, 1996], (iii) the *multiple predictor-corrector* method proposed by Carpenter et al. [10, 1993], and (iv) the *multiple centrality corrections* method developed by Gondzio [31, 1996]. The central idea behind all these techniques is to reduce the number of derivative evaluations and matrix factorizations to a necessary minimum, even at the expense of some increase in the cost of a single iteration.

The predictor-corrector technique was previously extended to nonlinear OPF solution by Wu et al. [77, 1994]. As far as we know, the PCN, MPC and MCC techniques have not been extended to power systems optimization. We have made a contribution by addressing the computational efficiency of these techniques in the context of nonlinear OPF solution. We make the following remarks:

- The corrector step of the predictor-corrector method that is described in [77] employs second-order terms ("delta terms") in the complementarity equations only. In our predictor-corrector method for solving the nonlinear OPF in rectangular coordinates, we are able to incorporate second-order terms in all KKT equations.

- We have extended to nonlinear OPF solution the perturbed composite Newton IP method for LP and QP, as developed by Tapia et al. [67]. This technique has not been considered in previous OPF algorithms.

- We have extended to nonlinear OPF solution the multiple predictor-corrector IP method for LP and QP, as developed by Carpenter et al. [10]. This technique has not been considered in previous OPF algorithms.

- Also, we have extended to nonlinear OPF solution the multiple centrality corrections technique for LP developed by Gondzio [31].

Extensive numerical experiments with the above higher-order IP methods are discussed in Chapter 7. Concerning future work with these higher-order IP variants, an issue deserving further research is how to dynamically choose the appropriate number of corrector steps within each iteration.

Chapter 5

Non-Interior Continuation Method for Nonlinear Programming

5.1 Introduction

We may recall that each point in a primal-dual IP algorithm is obtained by applying a single iteration of a damped Newton's method to a fixed set of nonlinear equations, parameterized by $\mu^k > 0$. Furthermore, the iterates start from a point w^0 that satisfies the strict positivity conditions, and follow a trajectory in the positive orthant of the complementarity product space in order to avoid spurious solutions, that is, points that satisfy

$$\begin{pmatrix} S_1 z_1 \\ S_2 \widehat{z}_2 \\ S_3 z_3 \\ S_4 \widehat{z}_4 \\ s_1 + s_2 - \overline{h} + \underline{h} \\ h(x) + s_2 - \overline{h} \\ s_3 + s_4 - \overline{x} + \underline{x} \\ \widehat{I} x + s_4 - \overline{x} \\ \nabla_x f(x) - \nabla_x g(x) y + \nabla_x h(x) z_2 + \widehat{I}^T z_4 \\ -g(x) \end{pmatrix} = 0 \tag{5.1}$$

but do not satisfy $(s_1, s_2, s_3, s_4) \geq 0$ and $(z_1, \widehat{z}_2, z_3, \widehat{z}_4) \geq 0$.

In this chapter, we present a new approach to solve the NLP problem (1.1), that handles the complementarity conditions, $s_i z_i = 0$, $s_i \geq 0$ and $z_i \geq 0$, without requiring that

the strict positivity conditions be satisfied at every iterate. Such a method originates from a reformulation of *linear complementarity problems* (LCP) as nonlinear systems of equations, which was developed by Chen and Harker [11, 1993], and Kanzow [46, 1996]. Given $M \in I\!R^{n\times n}$ and $q \in I\!R^n$, the LCP problem is to find $x \in I\!R^n$ and $y \in I\!R^n$ so that

$$Mx + q = y, \quad x \geq 0, \quad y \geq 0, \quad x^T y = 0. \tag{5.2}$$

Henceforth, we will refer to the LCP problem (5.2) simply as LCP(q, M).

Chen-Harker and Kanzow's reformulations of LCP(q, M) handle each complementarity condition, $x_i \geq 0$, $y_i \geq 0$ and $x_i y_i = 0$, by means of a function $\psi : I\!R^2 \to I\!R$ which is defined by the following characterization of its zeros:

$$\psi(a, b) = 0 \iff a \geq 0, \quad b \geq 0, \quad ab = 0. \tag{5.3}$$

Any function $\psi : I\!R^2 \to I\!R$ that have the property (5.3) is called an *NCP-function*, where the acronym NCP stands for *nonlinear complementarity problem.* In the last few years, various NCP-functions have been proposed. For example, the functions

$$\psi(a, b) := \frac{1}{2}\min^2\{a, b\}, \tag{5.4}$$

$$\psi(a, b) := \frac{1}{2}\left((ab)^2 + \min^2\{0, a\} + \min^2\{0, b\}\right), \tag{5.5}$$

$$\psi(a, b) := ab + \frac{1}{2}\alpha\left(\max^2\{0, a - \alpha b\} - a^2 + \max^2\{0, b - \alpha a\} - b^2\right), \quad \alpha > 1, \tag{5.6}$$

$$\psi(a, b) := \sqrt{a^2 + b^2} - (a + b), \tag{5.7}$$

$$\psi_\mu(a, b) := \sqrt{(a - b)^2 + \mu ab} - a - b, \quad \mu \in (0, 4), \tag{5.8}$$

$$\psi_\mu(a, b) := a + b - \sqrt{(a - b)^2 + 4\mu}, \quad \mu \to 0, \tag{5.9}$$

$$\psi_\mu(a, b) := a + b - \sqrt{a^2 + b^2 + 2\mu}, \quad \mu \to 0, \tag{5.10}$$

are NCP-functions [45]. We refer the interested reader to the works of Chen and Harker [11, 1993], Kanzow [44–46], Burke and Xu [6, 1996], and Hotta and Yoshise [37, 1996] for surveys on NCP-functions and their applications. Growing attention has been paid to the NCP-function (5.7) which was first introduced by Fischer [23] and further employed by several researchers. The NCP-functions (5.9) and (5.10) are of particular interest to our studies. They satisfy the property

$$\psi_\mu(a, b) = 0 \iff a > 0, \quad b > 0, \quad ab = \mu, \tag{5.11}$$

for any $\mu > 0$. Based on the NCP-functions (5.9) and (5.10), the Chen–Harker [11, 1993], and Kanzow [46, 1996] algorithms solve LCP(q, M) by *approximately* solving a sequence of

nonlinear systems of equations of the form

$$\Psi_\mu(x) := \begin{pmatrix} Mx + q - y \\ \psi_\mu(x_1, y_1) \\ \psi_\mu(x_2, y_2) \\ \vdots \\ \psi_\mu(x_n, y_n) \end{pmatrix} = 0, \tag{5.12}$$

where $\mu > 0$ is a *continuation parameter* that is forced to monotonically decrease to zero, as does the barrier parameter in an IP setting.

The most attractive feature of Chen–Harker and Kanzow algorithms for LCP is that they are *non-interior-point* (NIP) methods, in the sense that the attraction domain of the feasible path is all of $\mathbb{R}^n$ instead of the positive orthant $\mathbb{R}^n_+$. Since the non-negativity of any limit point is automatically assured by NCP-functions, without imposing additional conditions, the initial point and the iterates do not necessarily have to stay in the positive orthant, providing us with the freedom of choosing a starting point (x^0, y^0) that satisfy the condition $Mx^0 + q = y^0$.

A similar procedure has been used in the solution of the *nonlinear complementarity problem* (NCP) as well (see [16, 37, 44, 45]), which is to find an $x \in \mathbb{R}^n$ so that

$$x \geq 0, \quad F(x) \geq 0, \quad x^T F(x) = 0, \tag{5.13}$$

where $F : \mathbb{R}^n \to \mathbb{R}^n$ is a vector of continuously differentiable functions. The reformulation of NCP(F), as a parametrized sequence of nonlinear systems of equations, has the form

$$\Psi_\mu(x) := \begin{pmatrix} \psi_\mu(x_1, F_1(x)) \\ \psi_\mu(x_2, F_2(x)) \\ \vdots \\ \psi_\mu(x_n, F_n(x)) \end{pmatrix} = 0. \tag{5.14}$$

The systems of equations (5.12) and (5.14) are usually solved by Newton-type methods, which, however, are in general only locally convergent. In order to globalize the local method, an Armijo-type line search is performed to minimize a *merit function*, usually

$$\Phi_\mu(x) := \frac{1}{2}\Psi_\mu(x)^T \Psi_\mu(x). \tag{5.15}$$

The global minimizer of $\Phi_\mu(x)$ is a solution to $\Psi_\mu(x) = 0$.

In this chapter, we consider the KKT conditions for optimality of the OPF problem as a mixed NCP problem and, taking into consideration the NCP-function applications

mentioned above, propose a Newton-type method for its solution. The attractive feature of the proposed approach is the freedom of choosing a starting point with reduced primal and dual infeasibility. This freedom is possible because the initial point and subsequent iterates do not necessarily have to stay in the positive orthant of the complementarity product space, unlike IP methods. In the proposed procedure, the non-negativity conditions of any limit point are automatically assured by NCP-functions. To the best of our knowledge, the approach described in this chapter is the first application of NCP-functions to the solution of nonlinear OPF problems, which we see as an important contribution of this thesis.

The remainder of this chapter is organized as follows. In the next section, we present some properties of the NCP-functions (5.9) and (5.10) which will be used in the development of the proposed NIP algorithm for NLP. In Section 5.3, we describe in detail our NIP continuation method for solving nonlinear OPF problems. More specifically, we describe the reformulation of the KKT equations and related Newton system, describe an unconstrained minimization reformulation, discuss line search procedures to compute suitable step lengths along the search directions, describe a procedure to reduce the continuation parameter μ, and discuss the convergence test. An outline of the algorithm is presented in Section 5.4. Final remarks close the chapter in Section 5.5.

5.2 Some Properties of $\psi_\mu(a, b)$

In this section, we present some properties of the NCP-functions (5.9) and (5.10) which will be used in the development of our NIP continuation algorithm for solving problem (1.1).

LEMMA 5.1 (Kanzow [46, Lemma 2.1]) *The function* (5.9) *has the property* (5.11).

Proof. First assume that $a > 0$, $b > 0$, and $ab = \mu$. Then, we obtain

$$\begin{aligned}\psi_\mu(a,b) &= a + b - \sqrt{a^2 - 2ab + b^2 + 4ab} \\ &= a + b - \sqrt{(a+b)^2} \\ &= a + b - |a+b| \\ &= 0.\end{aligned}$$

To prove the converse result, assume that $\psi_\mu(a, b) = 0$, that is,

$$a + b = \sqrt{(a-b)^2 + 4\mu} > 0. \tag{5.16}$$

Squaring both sides of the equation in (5.16), we get $ab = \mu$. Therefore, $\text{sign}(a) = \text{sign}(b)$. Consequently, it follows from the inequality in (5.16) that $a > 0$ and $b > 0$. □

LEMMA 5.2 (Kanzow [46, Lemma 2.2]) *The function* (5.10) *has the property* (5.11).

Proof. If $a > 0$, $b > 0$, and $ab = \mu$, we get

$$\begin{aligned}\psi_\mu(a,b) &= a + b - \sqrt{(a+b)^2} \\ &= a + b - |a+b| \\ &= 0.\end{aligned}$$

On the other hand, the condition $\psi_\mu(a,b) = 0$ can be rewritten as

$$a + b = \sqrt{a^2 + b^2 + 2\mu} > 0, \tag{5.17}$$

from which $a > 0$, $b > 0$, and $ab = \mu$ follows in a similar way as in the proof of Lemma 5.1. □

LEMMA 5.3 (Burke and Xu [6, Lemma 4.5]) *Let* $\epsilon > 0$. *If* $|\psi_\mu(a,b)| \le \epsilon$, *then*

$$a \ge -\epsilon, \quad b \ge -\epsilon, \quad \textit{and} \quad \frac{|ab-\mu|}{|a| + |b| + \sqrt{\mu}} \le \epsilon.$$

Proof. If $|\psi_\mu(a,b)| \le \epsilon$, then

$$0 \le \sqrt{a^2 + b^2 + 2\mu} \le \epsilon + a + b. \tag{5.18}$$

If $\epsilon + a < 0$, then

$$\epsilon + a + b < b \le \sqrt{a^2 + b^2 + 2\mu},$$

which is a contradiction to (5.18). Hence, $a \ge -\epsilon$. Similarly, $b \ge -\epsilon$. Also

$$\begin{aligned}|\psi_\mu(a,b)| &= \frac{|(a+b)^2 - (a^2 + b^2 + 2\mu)|}{|(a+b) + \sqrt{(a^2 + b^2 + 2\mu)}|} \\ &\ge \frac{2|ab - \mu|}{(|a| + |b|) + (|a| + |b| + 2\mu)},\end{aligned}$$

which yields the result. □

Burke and Xu [6] also show that, if $\mu^k \le \epsilon$ and $\|\psi_\mu(x^k, y^k)\|_2 \le \beta\mu^k$, then

$$x_i^k \ge -\sqrt{\beta\epsilon}, \quad y_i^k \ge -\sqrt{\beta\epsilon}, \quad \text{and} \quad \frac{|x_i^k y_i^k - \mu^k|}{|x_i^k| + |y_i^k| + \sqrt{\mu^k}} \le \sqrt{\beta\epsilon}, \tag{5.19}$$

for $i = 1, 2, \ldots, n$. Furthermore, if $\mu^k \le \epsilon$ and $\|\psi_\mu(x^k, y^k)\|_\infty \le \epsilon$, then

$$x_i^k \ge -\epsilon, \quad y_i^k \ge -\epsilon, \quad \text{and} \quad \frac{|x_i^k y_i^k - \mu^k|}{|x_i^k| + |y_i^k| + \sqrt{\mu^k}} \le \epsilon, \tag{5.20}$$

for $i = 1, 2, \ldots, n$. The properties (5.19) and (5.20) are valuable in keeping track of the convergence; they induce termination when the relative error in the complementarity is small.

REMARK 5.1 *Let ψ_μ denote the function defined in* (5.9) *or* (5.10). *Then ψ_μ is continuously differentiable for all $(a, b) \in \mathbb{R}^2$, and the partial derivatives have the property*

$$\left.\frac{\partial \psi_\mu}{\partial a}\right|_{(a,b)} \in (0, 2) \qquad \text{and} \qquad \left.\frac{\partial \psi_\mu}{\partial b}\right|_{(a,b)} \in (0, 2) \qquad \text{for all} \quad a, b \in \mathbb{R}.$$

For example, if we consider $\psi_\mu(a, b)$ as defined in (5.10), *we get*

$$\frac{\partial \psi_\mu}{\partial a}(a, b) = 1 - \frac{a}{\sqrt{a^2 + b^2 + 2\mu}} \in (0, 2), \tag{5.21a}$$

and

$$\frac{\partial \psi_\mu}{\partial b}(a, b) = 1 - \frac{b}{\sqrt{a^2 + b^2 + 2\mu}} \in (0, 2). \tag{5.21b}$$

LEMMA 5.4 (Burke and Xu [6, Lemma 2.1]) *The function* (5.10) *has the following properties:*

- *For every $\mu \geq 0$, the function ψ_μ^2 is continuously differentiable on $\mathbb{R}^2$.*
- *For all $(a, b) \in \mathbb{R}^2$ when $\mu > 0$, and for all $(a, b) \in \mathbb{R}^2 \backslash \{(0, 0)\}$ when $\mu = 0$, it follows that*

$$\|\nabla^2 \psi_\mu^2(a, b)\| \leq 4(5 + \sqrt{2}). \tag{5.22}$$

- *For $\mu_1 \geq 0$, $\mu_2 \geq 0$ and $(a, b) \in \mathbb{R}^2$, we have*

$$|\psi_{\mu_1}^2(a, b) - \psi_{\mu_2}^2(a, b)| \leq (2 + 2\sqrt{2})|\mu_1 - \mu_2|. \tag{5.23}$$

Proof. See [6]. □

5.3 The Non-Interior-Point Continuation Algorithm

The approach for handling the complementarity conditions in the solution of nonlinear OPF problems by means of NCP-functions has recently been considered by Quintana and Torres [61, 1998]. As far as we know, the OPF algorithm introduced in [61] and described in detail below is the first one that uses NCP-functions. For the sake of presentation, we

assume that the complementarity conditions (3.6a)–(3.6d) are handled by means of the NCP-function (5.9). The following reformulation of the KKT system (3.6) is considered in the proposed method:

$$\Psi_\mu(w) := \begin{pmatrix} \psi_\mu(s_1, z_1) \\ \psi_\mu(s_2, \widehat{z}_2) \\ \psi_\mu(s_3, z_3) \\ \psi_\mu(s_4, \widehat{z}_4) \\ s_1 + s_2 - \overline{h} + \underline{h} \\ h(x) + s_2 - \overline{h} \\ s_3 + s_4 - \overline{x} + \underline{x} \\ \widehat{I}x + s_4 - \overline{x} \\ \nabla_x f(x) - \nabla_x g(x) y + \nabla_x h(x) z_2 + \widehat{I}^T z_4 \\ -g(x) \end{pmatrix} = 0. \tag{5.24}$$

A damped Newton-type method is used to solve (5.24), using the local approximation

$$\nabla_w \Psi_\mu(w^k)^T \Delta w = -\Psi_\mu(w^k), \tag{5.25}$$

where, as we drop the superscripts k,

$$\nabla_w \Psi_\mu(w)^T = \begin{bmatrix} \nabla_{s_1}\psi & 0 & 0 & 0 & \nabla_{z_1}\psi & 0 & 0 & 0 & 0 & 0 \\ 0 & \nabla_{s_2}\psi & 0 & 0 & \nabla_{z_2}\psi & \nabla_{z_2}\psi & 0 & 0 & 0 & 0 \\ 0 & 0 & \nabla_{s_3}\psi & 0 & 0 & 0 & \nabla_{z_3}\psi & 0 & 0 & 0 \\ 0 & 0 & 0 & \nabla_{s_4}\psi & 0 & 0 & \nabla_{z_4}\psi & \nabla_{z_4}\psi & 0 & 0 \\ I_p & I_p & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I_p & 0 & 0 & 0 & 0 & 0 & 0 & \nabla_x h^T & 0 \\ 0 & 0 & I_q & I_q & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I_q & 0 & 0 & 0 & 0 & \widehat{I} & 0 \\ 0 & 0 & 0 & 0 & 0 & \nabla_x h & 0 & \widehat{I}^T & \nabla^2_{xx} L_\mu & -\nabla_x g \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\nabla_x g^T & 0 \end{bmatrix}.$$

The coefficient matrix $\nabla_w \Psi_\mu(w)^T$ has the same nonzero pattern of $\nabla^2_{ww} L_\mu(w)$. The way to compute some of its diagonal matrices is the only change. The modified diagonal matrices are computed as follows:

$$\nabla_{s_1}\psi := \frac{\partial \psi_\mu(s_1, z_1)}{\partial s_1} = \operatorname{diag}\left(1 - \frac{s_{1_1}}{a_{1_1}}, \ldots, 1 - \frac{s_{1_p}}{a_{1_p}}\right), \tag{5.26a}$$

$$\nabla_{z_1}\psi := \frac{\partial \psi_\mu(s_1, z_1)}{\partial z_1} = \operatorname{diag}\left(1 - \frac{z_{1_1}}{a_{1_1}}, \ldots, 1 - \frac{z_{1_p}}{a_{1_p}}\right), \tag{5.26b}$$

where $a_{1_i} = (s_{1_i}^2 + z_{1_i}^2 + 2\mu^k)^{1/2}$,

$$\nabla_{s_2}\psi := \frac{\partial\psi_\mu(s_2,\widehat{z}_2)}{\partial s_2} = \operatorname{diag}\left(1-\frac{s_{2_1}}{a_{2_1}},\ldots,1-\frac{s_{2_p}}{a_{2_p}}\right), \tag{5.26c}$$

$$\nabla_{z_2}\psi := \frac{\partial\psi_\mu(s_2,\widehat{z}_2)}{\partial z_1} = \operatorname{diag}\left(1-\frac{\widehat{z}_{2_1}}{a_{2_1}},\ldots,1-\frac{\widehat{z}_{2_p}}{a_{2_p}}\right), \tag{5.26d}$$

$$= \frac{\partial\psi_\mu(s_2,\widehat{z}_2)}{\partial z_2}, \tag{5.26e}$$

where $a_{2_i} = (s_{2_i}^2 + \widehat{z}_{2_i}^2 + 2\mu^k)^{1/2}$,

$$\nabla_{s_3}\psi := \frac{\partial\psi_\mu(s_3,z_3)}{\partial s_3} = \operatorname{diag}\left(1-\frac{s_{3_1}}{a_{3_1}},\ldots,1-\frac{s_{3_q}}{a_{3_q}}\right), \tag{5.26f}$$

$$\nabla_{z_3}\psi := \frac{\partial\psi_\mu(s_3,z_3)}{\partial z_3} = \operatorname{diag}\left(1-\frac{z_{3_1}}{a_{3_1}},\ldots,1-\frac{z_{3_q}}{a_{3_q}}\right), \tag{5.26g}$$

where $a_{3_i} = (s_{3_i}^2 + z_{3_i}^2 + 2\mu^k)^{1/2}$,

$$\nabla_{s_4}\psi := \frac{\partial\psi_\mu(s_4,\widehat{z}_4)}{\partial s_4} = \operatorname{diag}\left(1-\frac{s_{4_1}}{a_{4_1}},\ldots,1-\frac{s_{4_q}}{a_{4_q}}\right), \tag{5.26h}$$

$$\nabla_{z_4}\psi := \frac{\partial\psi_\mu(s_4,\widehat{z}_4)}{\partial z_3} = \operatorname{diag}\left(1-\frac{\widehat{z}_{4_1}}{a_{4_1}},\ldots,1-\frac{\widehat{z}_{4_q}}{a_{4_q}}\right), \tag{5.26i}$$

$$= \frac{\partial\psi_\mu(s_4,\widehat{z}_4)}{\partial z_4}, \tag{5.26j}$$

where $a_{4_i} = (s_{4_i}^2 + \widehat{z}_{4_i}^2 + 2\mu^k)^{1/2}$. Notice from Remark 5.1 that these diagonal matrices are positive definite since all diagonal elements are in the interval $(0,2)$. The coefficient matrix in (5.25) can be made symmetric if the linearized complementarity equations are properly scaled by the diagonal matrices $\nabla_{z_1}\psi^{-1}$, $\nabla_{z_2}\psi^{-1}$, $\nabla_{z_3}\psi^{-1}$, and $\nabla_{z_4}\psi^{-1}$.

5.3.1 Unconstrained Minimization Reformulation

New estimates for the primal and dual variables are computed from

$$w^{k+1} := w^k + \alpha^k \Delta w, \tag{5.27}$$

where Δw is the search direction

$$\Delta w = -\left[\nabla_w \Psi_\mu(w^k)^T\right]^{-1}\Psi_\mu(w^k), \tag{5.28}$$

and α^k is the *step length* along Δw. A suitable step length α^k should be computed by performing a *line search* along the direction Δw, aiming at achieving a "sufficient" decrease in a *merit function* that is defined to measure progress towards a solution to (5.24). That is, the idea of a merit function is to assure that joint progress is made both towards a local minimizer and towards feasibility.

It is well known that for general non-convex NLP, the Newton's method may diverge for a poor initial estimate [24]. However, merit functions can be used to guide us in deciding on how much to shorten the step lengths so as to assure convergence. Since our intent is to find a solution of $\Psi_\mu(w) = 0$, an obvious merit function would be

$$\Phi_\mu(w) := \frac{1}{2}\Psi_\mu(w)^T\Psi_\mu(w), \tag{5.29}$$

which is known as the *natural merit function.* Thus, when solving the nonlinear equation system (5.24), we indeed turn our attention to an unconstrained minimization problem of the form

$$\min_{w\in\mathbb{R}^n} \Phi_\mu(w). \tag{5.30}$$

Solving the nonlinear system (5.24) is clearly equivalent to finding a *global minimum* point w^* of $\Phi_\mu(w)$, that gives $\Phi_\mu(w^*) = 0$.

Ideally, a suitable merit function has the property that an unconstrained minimizer of the merit function is a solution of the desired problem. However, while every solution to (5.24) is a solution to (5.30), there may be local minimizers of (5.30) that are not solutions to (5.24). Such a possibility is illustrated in Figure 5.1. Then, it only interests us to obtain a global minimum of (5.30). Furthermore, since the NLP problem (1.1) is non-convex, a solution of $\Psi_\mu(w) = 0$ is not necessarily a minimizer of (1.1) (maximizers and saddle points also satisfy this condition). Thus, reducing $\Phi_\mu(w)$ does not necessarily ensure minimization with respect to the original problem. On the other hand, Newton's method for solving the reformulation (5.24) can be combined with global methods for unconstrained optimization to produce global methods for solving (1.1), giving rise to a class of *quasi-Newton* methods.

We refer the interested reader to the Dennis Jr. and Schnabel's book [17] for a detailed treatment of the solution of nonlinear equation systems by unconstrained minimization techniques.

5.3.2 Backtracking Line Search Procedure

As discussed in [59] and elsewhere, in the early years of line search methods for unconstrained optimization, most implementations included an "accurate" line search. That is, α^k was chosen as a close approximate solution to $\min_{\alpha\in\mathbb{R}} \Phi_\mu(w^k + \alpha\Delta w)$. Nowadays, however, it is customary to perform an "inexact" line search. An inexact line search that is frequently used in the solution of LCP and NCP problems [6, 11], and in unconstrained minimization is the following Armijo-type procedure:

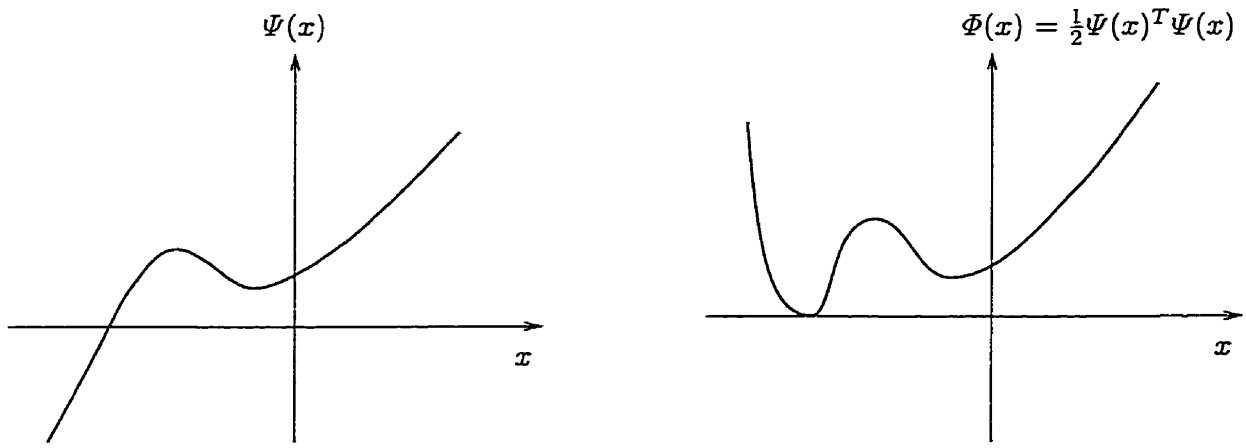

Figure 5.1: The nonlinear equations and a corresponding minimization problem, in one dimension [17, Figure 6.5.1].

- Given $\sigma_1 \in (0, 1]$ and $\alpha_1 \in (0, 1)$, find the smallest $m_k \in \{0, 1, 2, \dots\}$ that satisfies

$$\Phi_\mu(w^k + \alpha_1^{m_k}\Delta w) \leq \Phi_\mu(w^k) + \sigma_1\alpha_1^{m_k}\nabla_w\Phi_\mu(w^k)^T\Delta w \tag{5.31a}$$

  or, equivalently,

$$\Phi_\mu(w^k + \alpha_1^{m_k}\Delta w) \leq (1 - \sigma_1\alpha_1^{m_k})\Phi_\mu(w^k). \tag{5.31b}$$

- Then, let $\alpha^k = \alpha_1^{m_k}$ and set $w^{k+1} = w^k + \alpha^k\Delta w$.

The "inexactness" of this procedure is typified by a small value of σ_1 and a large value of α_1. Typical parameter values considered in [6, 11] are $\sigma_1 = 10^{-4}$ and $\alpha_1 = 0.99$. The amount of work associated with each trial point $\alpha_1^{m_k}$ is the evaluation of $\Phi_\mu(w^k + \alpha_1^{m_k}\Delta w)$. That is, the evaluation of $\Psi_\mu(w^k + \alpha_1^{m_k}\Delta w)$ followed by the inner-product with itself.

As we try to solve (5.30) using the direction Δw in the line search as computed from (5.25), we are taking advantage of the "structure" of the original problem, which is a desirable feature; see [17, Chapter 6]. The conditional tests in (5.31) can be satisfied for some m_k only if Δw is a *descent direction* for (5.30). A descent direction for (5.30) is any direction p^k for which $\nabla_w\Phi_\mu(w^k)^T p^k < 0$. Notice that,

$$\nabla_w\Phi_\mu(w^k) = \nabla_w\Psi_\mu(w^k)\Psi_\mu(w^k). \tag{5.32}$$

Hence, the *steepest-descent direction* for (5.30), $-\nabla_w\Phi_\mu(w^k)$, is along $-\nabla_w\Psi_\mu(w^k)\Psi_\mu(w^k)$. Consequently, the direction Δw from (5.28) is a descent direction for (5.30) since

$$\nabla_w\Phi_\mu(w)^T\Delta w = -\Psi_\mu(w)^T\left[\nabla_w\Psi_\mu(w)^T\right]\left[\nabla_w\Psi_\mu(w)^T\right]^{-1}\Psi_\mu(w) = -\Psi_\mu(w)^T\Psi_\mu(w) < 0,$$

as long as $\Psi_\mu(w^k) \neq 0$, and $\nabla_w \Phi_\mu(w^k)$ and Δw are not orthogonal. We can also infer from the above analysis that the procedures (5.31a) and (5.31b) are alike.

De Luca et al. in [16] comment that it has been often observed in the field of NCP algorithms that line search tests like (5.31) can lead to very small step lengths; in turn this can lead to very slow convergence and even to a numerical failure of the algorithm. To circumvent this problem many *non-monotone* line search tests have been used [6]. The non-monotone line search that we have implemented is as follows:

- Given $\sigma_1 \in (0, 1]$ and $\alpha_1 \in (0, 1)$, find the smallest $m_k \in \{0, 1, 2, \dots\}$ that satisfies

$$\Phi_\mu(w^k + \alpha_1^{m_k} \Delta w) \leq \max_{j=k-l+1:\, k} \Phi_\mu(w^j) - \sigma_1 \alpha_1^{m_k} \Phi_\mu(w^k). \tag{5.33}$$

- Then, let $\alpha^k = \alpha_1^{m_k}$ and set $w^{k+1} = w^k + \alpha^k \Delta w$.

The parameter l is typically given as $l = \min\{5, k\}$.

5.3.3 Reducing the Continuation Parameter

To reduce the parameter μ^k, some algorithms for LCP (see [6]) use the following procedure:

- If $\Phi_{\mu^k}(w^{k+1}) \geq \Phi_{\mu^k}(w^k)$, let $\mu^{k+1} = \mu^k$. Otherwise, given $\sigma_2 \in (0, 1]$ and $\alpha_2 \in (0, 1)$, find the smallest $m_k \in \{0, 1, 2, \dots\}$ that satisfies

$$\Phi_{(1-\sigma_2 \alpha_2^{m_k})\mu^k}(w^{k+1}) \leq \beta(1 - \sigma_2 \alpha_2^{m_k})\mu^k. \tag{5.34}$$

- Then, let $\mu^{k+1} = (1 - \sigma_2 \alpha_2^{m_k})\mu^k$.

The parameters used in (5.34) are typically $\sigma_2 = 0.9999$, $\alpha_2 = 0.99$, and $\beta := \Phi_{\mu^0}(w^0)/\mu^0$.

Notice that the complementarity equations in (5.24) are the only ones directly affected by μ. Then, to reduce computational effort, we could redefine the test (5.34) as

$$\widehat{\Phi}_{(1-\sigma_2 \alpha_2^{m_k})\mu^k}(w^{k+1}) \leq \beta(1 - \sigma_2 \alpha_2^{m_k})\mu^k, \tag{5.35}$$

where

$$\widehat{\Phi}_\mu(w) := \|\psi_\mu(s_1, z_1)\|^2 + \|\psi_\mu(s_2, \widehat{z}_2)\|^2 + \|\psi_\mu(s_3, z_3)\|^2 + \|\psi_\mu(s_4, \widehat{z}_4)\|^2. \tag{5.36}$$

We observed, in computational practice, that the conditional test (5.34) highly relies on the monotonicity of the line search to compute the step length α^k. Then, we propose to reduce μ^k in a way quite similar to that used in the IP method. First, we compute the measure

$$\varrho^k := |s_1^k|^T|z_1^k| + |s_2^k|^T|\widehat{z}_2^k| + |s_3^k|^T|z_3^k| + |s_4^k|^T|\widehat{z}_4^k|, \tag{5.37}$$

which "mimics" the complementarity gap ρ^k in the IP method. If the iterates converge to an optimum, then $\{\varrho^k\}$ must converge to zero. Then, we propose to compute μ^{k+1} from

$$\mu^{k+1} = \min\left\{\sigma^k \frac{\varrho^k}{2(p+q)},\ 0.9\mu^k\right\}, \tag{5.38}$$

where σ^k is as defined in the IP algorithm. This formula to update μ^k is computationally inexpensive and fairly effective in practice, as shown in Chapter 7.

5.3.4 Testing for Convergence

The converge test for the NIP algorithm differs only slightly from the convergence test for the IP algorithms, namely in the definition of criteria ν_3. We consider the NIP iterations terminated whenever an approximate local minimum has been obtained, in which case

$$\nu_1^k \leq \epsilon_1, \tag{5.39a}$$
$$\nu_2^k \leq \epsilon_1, \tag{5.39b}$$
$$\nu_3^k \leq \epsilon_2, \tag{5.39c}$$
$$\nu_4^k \leq \epsilon_2, \tag{5.39d}$$

or they are stuck at some point other than a local minimum (a possibility), in which case

$$\mu^k \leq \epsilon_\mu, \tag{5.40a}$$
$$\|\Delta x\|_\infty \leq \epsilon_2, \tag{5.40b}$$
$$\|g(x^k)\|_\infty \leq \epsilon_1, \tag{5.40c}$$
$$\nu_4^k \leq \epsilon_2, \tag{5.40d}$$

where

$$\nu_1^k = \max\left\{\|g(x^k)\|_\infty, \max\{\underline{h} - h(x^k)\}, \max\{h(x^k) - \overline{h}\}, \max\{\underline{x} - \widehat{x}^k\}, \max\{\widehat{x}^k - \overline{x}\}\right\}, \tag{5.41}$$

$$\nu_2^k = \frac{\|\nabla_x f(x^k) - \nabla_x g(x^k) y^k + \nabla_x h(x^k) z_2^k + \widehat{I}^T z_4^k\|_\infty}{1 + \|x^k\| + \|y^k\| + \|z_2^k\| + \|z_4^k\|}, \tag{5.42}$$

$$\nu_3^k = \max\{|\min\{s_1^k, z_1^k\}|,\ |\min\{s_2^k, \widehat{z}_2^k\}|,\ |\min\{s_3^k, z_3^k\}|,\ |\min\{s_4^k, \widehat{z}_4^k\}|\}, \tag{5.43}$$

$$\nu_4^k = \frac{|f(x^k) - f(x^{k-1})|}{1 + |f(x^k)|}. \tag{5.44}$$

Typical convergence tolerance values are $\epsilon_1 = 10^{-4}$, $\epsilon_2 = 10^{-2}\epsilon_1$ and $\epsilon_\mu = 10^{-10}$.

5.4 Outline of the Non-Interior-Point Algorithm

An outline of the NIP algorithm for NLP described in this chapter is shown below.

Algorithm 5.1 Non-Interior-Point Continuation Algorithm.

STEP 0: (Initialization)
Let $\mu^0 > 0$, $\sigma_1 \in (0, 1]$, $\alpha_1 \in (0, 1)$, and w^0 be given so that w^0 satisfies at least the primal feasibility equations. Set $k \leftarrow 0$.

STEP 1: (Compute the Newton Direction)
Given the current point w^k, let Δw be the solution to the Newton system

$$\nabla_w \Psi_\mu(w^k)^T \Delta w = -\Psi_\mu(w^k).$$

STEP 2: (Compute Step Length and Update Variables)
Find the smallest $m_k \in \{0, 1, 2, \dots\}$ that satisfies

$$\Phi_\mu(w^k + \alpha_1^{m_k} \Delta w) \leq \max_{j=k-l+1\,:\,k} \Phi_\mu(w^j) - \sigma_1 \alpha_1^{m_k} \Phi_\mu(w^k).$$

Let $\alpha^k = \alpha_1^{m_k}$ and set $w^{k+1} = w^k + \alpha^k \Delta w$.

STEP 3: (Update the Continuation Parameter)
If the new point w^{k+1} satisfies the convergence criteria, stop. Otherwise, compute

$$\mu^{k+1} = \min\left\{\sigma^k \frac{\varrho^k}{2(p+q)},\ 0.9\mu^k\right\},$$

set $k \leftarrow k + 1$, and return to STEP 1.

5.5 Final Remarks

In this chapter, a new algorithm for solving nonlinear OPF problems has been proposed. It is a NIP algorithm that handles the complementarity conditions by a recently introduced NCP-function. As far as we are aware, the proposed OPF algorithm is the first one based on NCP-functions. Distinctive features of this approach, as compared with IP methods, are that it can start from arbitrary points, and the iterates are not required to stay inside the

positive orthant of the complementarity product space. That is, the non-negativity conditions need be satisfied only at the solution point. By considering the optimality conditions for the OPF problem as a mixed nonlinear complementarity problem, we take advantage of recent mathematical development to solve complementarity problems. Particularly, we have used an NCP-function to transform entirely the KKT conditions into a system of nonlinear equations, that is solved afterwards by a Newton-type method.

Tognola and Bacher recently proposed an OPF algorithm [68, 1997] that also eliminates the need for the iterates to stay within the positive orthant of the complementarity product space. This is the reason why they call their approach an *unlimited point algorithm*. Their unlimited point algorithm, which is based on a transformation of the nonnegative variables, share many similarities with IP algorithms, as does our technique, but is quite different from the NIP approach proposed in this chapter; no transformation of variables is involved in our NIP approach.

Concerning future work with the approach proposed in this chapter, we consider as potential directions for research the following topics:

- Since the NIP algorithm can start from arbitrary points and all matrices in the IP and NIP algorithms have the same nonzero pattern, an improved OPF algorithm most likely can be developed if we combine together the two algorithms. Notice that switching from one algorithm to the other demands no changes of the linear algebra kernel, the core of both techniques.

- The robustness of the NIP algorithm likely will be improved if we consider a *Levenberg-Marquardt-type method* for the solution of (5.25). Instead of solving (5.28) for the direction Δw, this method computes a search direction d^k as the solution of the modified linear system

$$\left[\nabla_w \Psi_\mu(w^k)^T + \sigma^k I\right] d^k = -\Psi_\mu(w^k) \tag{5.45}$$

  where $\sigma^k \geq 0$ here is the Levenberg-Marquardt parameter. If the direction generated by (5.45) is not a "good" descent direction, according to the test

$$\nabla_w \Phi_\mu(w^k)^T d^k \leq -\rho \|d^k\|^p, \quad \rho > 0, \quad p > 2, \tag{5.46}$$

  we resort to the *steepest descent direction*, that is, let $d^k = -\nabla_w \Phi_\mu(w^k)$.

- As suggested for the primal-dual IP algorithm, an issue worthy of investigation is the usefulness of inexact search directions. By this we mean that the vector Δw satisfies

$$\nabla_w \Psi_\mu(w^k)^T \Delta w = -\Psi_\mu(w^k) + r^k. \tag{5.47}$$

where r^k here is the vector of residuals and measures how inexactly system (5.28) is solved. An iterative solver is used to solve the linear system (5.28), and this method is stopped when the norm of the residual is smaller than a prefixed accuracy, that is, $\|r^k\|_2 \leq \epsilon^k$. Facchinei and Kanzow [20] have proposed an *inexact Levenberg-Marquardt-type algorithm* to solve large NCP problems that employs a test of the form $\|r^k\| \leq (0.1/(k+1))\|\Phi_\mu(w^k)\|$.

- Since the computation and factorization of matrix $\nabla_w \Psi_\mu(w^k)^T$ demand the greatest computational effort within an iteration of the NIP algorithm, it may be advantageous to use the same derivative evaluation and matrix factorization in several solves. Then, the composite Newton method that is described in Chapter 4 could be extended to the NIP algorithm as well.

- The implementation of other NCP-functions is another interesting topic of research. For instance, many recently developed algorithms to solve complementarity problems (see [16, 20]) employ non-smooth reformulations, involving the computation of the generalized Jacobian of Clarke [12] within a Newton-type algorithm.

In the next chapter, we describe many issues that are directly related to the efficient implementation of the IP and NIP algorithms that we have described so far.

Chapter 6

Practical Implementation Issues

In this chapter, we discuss many points and issues that are directly related to an efficient implementation of the standard primal-dual IP algorithm, the higher-order primal-dual IP algorithms, and the NIP algorithm for NLP previously described in Chapters 3, 4 and 5. Emphasis, however, is on the solution of the OPF problems that are described in Chapter 2. In the next section, we describe procedures for starting point choices in all algorithms. In Section 6.2, we describe the assembling of matrices for the OPF in rectangular coordinates. We derive explicit formulae for cheap computation of the Hessian $\nabla^2_{xx} L_\mu(w^k)$ through a proposed map of Lagrange multipliers—$(y, z_2) \mapsto (\lambda^p, \lambda^q, \lambda^v)$. In Section 6.3, we present a similar analysis for the OPF in polar coordinates. In Section 6.4, we discuss data structures and present some important code fragments. In Section 6.5, we discuss how the linear systems are solved. Final remarks in Section 6.6 close the chapter.

6.1 Initialization of Algorithms

We may recall that neither a starting point nor subsequent iterates are mandatory to be feasible points for primal-dual IP methods. Since feasibility is attained during the iterative process, as optimality is reached, the denomination *infeasible* primal-dual IP method is quite common in the literature; the strict positivity conditions, however, must be satisfied at every point. Cases where such a single initialization criterion—strict positivity—perform poorly are very common. For initial points that yield inequality constraints close to binding, it is advantageous to sacrifice initial feasibility of those inequalities to avoid too small slack values; too small initial slacks may result in too small step lengths at early stages, seriously

slowing down global improvement of the variables and hence the convergence. Besides the strict positivity conditions, a good starting point should also satisfy two other conditions (see [76]). First, the point should be well centered, that is, the perturbed complementarity conditions (3.6a)–(3.6d) should hold at $k = 0$ so that the complementarity products $s_i^0 z_i^0$ are similar for every index i. Second, the point should not be too infeasible, that is, the ratio of the infeasibility to complementarity gap, $(\nu_1^0 + \nu_2^0)/\nu_3^0$, should not be too large.

Below, we describe four initialization heuristics; only the first two conform to the IP algorithms, whereas all of them conform to the proposed NIP algorithm.

6.1.1 Heuristic–A: IP and NIP Algorithms

- Estimate the primal variables x^0 by one of the following four approaches (listed in preference order): (i) as given by a converged ac load flow solution, (ii) as given by the first 2 or 3 iterations of a Gauss-Seidel's method that is applied to the load flow equations, (iii) as given by a dc load flow solution, or (iv) as a flat start using the middle point between the upper and lower limits for the bounded variables.

- Then, the primal slack variables are initialized as

$$s_1^0 = \min\left\{\max\left\{\gamma(\overline{h} - \underline{h}),\, h(x^0) - \underline{h}\right\},\, (1-\gamma)(\overline{h} - \underline{h})\right\}, \tag{6.1a}$$

$$s_2^0 = \overline{h} - \underline{h} - s_1^0, \tag{6.1b}$$

$$s_3^0 = \min\left\{\max\left\{\gamma(\overline{x} - \underline{x}),\, \widehat{x}^0 - \underline{x}\right\},\, (1-\gamma)(\overline{x} - \underline{x})\right\}, \tag{6.1c}$$

$$s_4^0 = \overline{x} - \underline{x} - s_3^0, \tag{6.1d}$$

  where γ is the relative distance of the slacks to the boundary of the positive orthant. In our implementations γ defaults to 0.35.

- Assuming that $\mu^0 > 0$ is given, the dual variables z_1^0, z_2^0, z_3^0 and z_4^0 are obtained from

$$z_1^0 = \mu^0 (S_1^0)^{-1} u, \tag{6.2a}$$

$$z_2^0 = \mu^0 (S_2^0)^{-1} u - z_1^0, \tag{6.2b}$$

$$z_3^0 = \mu^0 (S_3^0)^{-1} u, \tag{6.2c}$$

$$z_4^0 = \mu^0 (S_4^0)^{-1} u - z_3^0. \tag{6.2d}$$

- The dual variable y_i^0 is set to -1 if it is related to the active power balance constraint (2.12a), and set to zero if it is related to the reactive power balance constraint (2.12b).

6.1.2 Heuristic–B: IP and NIP Algorithms

- Choose x^0, s_1^0, s_2^0, s_3^0, s_4^0 and y^0 in the same way as in the Heuristic–A above.
- Then, obtain the remaining dual variables from

$$z_{1_i}^0 = \begin{cases} 1-\gamma, & \text{if } s_{1_i}^0 = \gamma(\overline{h}_i - \underline{h}_i), \\ 1/2, & \text{if } s_{1_i}^0 = h_i(x^0) - \underline{h}_i, \\ \gamma, & \text{otherwise.} \end{cases} \tag{6.3a}$$

$$z_{2_i}^0 = 1 - z_{1_i}^0, \tag{6.3b}$$

$$z_{3_i}^0 = \begin{cases} 1-\gamma, & \text{if } s_{3_i}^0 = \gamma(\overline{x}_i - \underline{x}_i), \\ 1/2, & \text{if } s_{3_i}^0 = x_i^0 - \underline{x}_i, \\ \gamma, & \text{otherwise.} \end{cases} \tag{6.3c}$$

$$z_{4_i}^0 = 1 - z_{3_i}^0. \tag{6.3d}$$

- Finally, obtain the initial barrier parameter from

$$\mu^0 = \sigma^0 \frac{(s_1^0)^T z_1^0 + (s_2^0)^T \widehat{z}_2^0 + (s_3^0)^T z_3^0 + (s_4^0)^T \widehat{z}_4^0}{2(p+q)}. \tag{6.4}$$

6.1.3 Heuristic–C: NIP Algorithm

- Choose x^0 and y^0 in the same way as in the Heuristic–A above, and then obtain the primal slacks from

$$s_1^0 = h(x^0) - \underline{h}, \tag{6.5a}$$

$$s_2^0 = \overline{h} - h(x^0), \tag{6.5b}$$

$$s_3^0 = \widehat{x}^0 - \underline{x}, \tag{6.5c}$$

$$s_4^0 = \overline{x} - \widehat{x}^0. \tag{6.5d}$$

- Then, obtain the remaining dual variables from

$$z_1^0 = u, \tag{6.6a}$$

$$z_2^0 = 0, \tag{6.6b}$$

$$z_3^0 = u, \tag{6.6c}$$

$$z_4^0 = 0. \tag{6.6d}$$

6.1.4 Heuristic–D: NIP Algorithm

- Choose x^0 and y^0 in the same way as in the Heuristic–A above, and then obtain the primal slacks from

$$s_1^0 = \min\left\{\max\left\{0,\, h(x^0) - \underline{h}\right\},\, \overline{h} - \underline{h}\right\}, \tag{6.7a}$$

$$s_2^0 = \overline{h} - \underline{h} - s_1^0, \tag{6.7b}$$

$$s_3^0 = \min\left\{\max\left\{0,\, \widehat{x}^0 - \underline{x}\right\},\, \overline{x} - \underline{x}\right\}, \tag{6.7c}$$

$$s_4^0 = \overline{x} - \underline{x} - s_3^0. \tag{6.7d}$$

- Then, obtain the remaining dual variables from

$$z_{1_i}^0 = \begin{cases} \gamma, & \text{if } \; s_{1_i}^0 = 0, \\ \gamma/2, & \text{if } \; s_{1_i}^0 = h_i(x^0) - \underline{h}_i, \\ 0, & \text{otherwise.} \end{cases} \tag{6.8a}$$

$$z_{2_i}^0 = \gamma - z_{1_i}^0, \tag{6.8b}$$

$$z_{3_i}^0 = \begin{cases} \gamma, & \text{if } \; s_{3_i}^0 = 0, \\ \gamma/2, & \text{if } \; s_{3_i}^0 = x_i^0 - \underline{x}_i, \\ 0, & \text{otherwise.} \end{cases} \tag{6.8c}$$

$$z_{4_i}^0 = \gamma - z_{3_i}^0. \tag{6.8d}$$

The following comments are pertinent to the above initialization heuristics:

- It is well known that a Lagrange multiplier at an optimal solution measures the sensitivity of the objective function with respect to small perturbations in the related constraint; see [24]. Thus, in the transmission power losses minimization problem, the Lagrange multipliers related to constraints (2.12a) and (2.12b) are expected to be about -1 and zero, respectively. This justifies the proposed choice of y^0.

- In the Heuristic–A, we first choose $\mu^0 > 0$ and then compute z_1^0, z_2^0, z_3^0 and z_4^0 so that all complementarity products $s_i^0 z_i^0$ have the same value μ^0. In Heuristic–B, on the contrary, we first estimate z_1^0, z_2^0, z_3^0 and z_4^0 and, afterwards, use these estimates to define $\mu^0 > 0$. The centrality of w^0 as obtained by Heuristic–B may be poor.

- The Heuristic–C for the NIP algorithm relaxes the non-negativity conditions, whereas the Heuristic–D relaxes the strict positivity conditions; the non-negativity of any limit point will be automatically ensured by the NCP-functions.

6.2 Gradients and Hessians: Rectangular Coordinates

The gradient vector $\nabla_x f(x)$ and the Jacobian matrices $\nabla_x g(x)^T$ and $\nabla_x h(x)^T$ are obtained from Equations (A.1) through (A.14) in Appendix A.

The Hessian matrix of the term $P_i(e, f, t)$ that appears in the constraint (2.12a), has the form

$$\nabla^2_{xx} P_i(x) = \begin{bmatrix} \nabla^2_{ee} P_i & \nabla^2_{fe} P_i^T & \nabla^2_{te} P_i^T \\ \nabla^2_{fe} P_i & \nabla^2_{ff} P_i & \nabla^2_{tf} P_i^T \\ \nabla^2_{te} P_i & \nabla^2_{tf} P_i & \nabla^2_{tt} P_i \end{bmatrix}. \tag{6.9}$$

The Hessian sub-matrices $\nabla^2_{ee} P_i$, $\nabla^2_{fe} P_i$ and $\nabla^2_{ff} P_i$ are given by (A.15), (A.16) and (A.17), respectively. For each $(i, \cdot) \in \mathcal{T}$, two non-zeros of $\nabla^2_{te} P_i$ are computed from (A.21) and (A.25), two non-zeros of $\nabla^2_{tf} P_i$ are computed from (A.29) and (A.33), and one nonzero of $\nabla^2_{tf} P_i$ is computed from (A.37). For each $(\cdot, i) \in \mathcal{T}$, two non-zeros of $\nabla^2_{te} P_i$ are computed from (A.22) and (A.26), and two non-zeros of $\nabla^2_{tf} P_i$ are computed from (A.30) and (A.34).

The Hessian matrix of the term $Q_i(e, f, t)$ that appears in the constraints (2.12b), (2.13a) and (2.13b), has the form

$$\nabla^2_{xx} Q_i(x) = \begin{bmatrix} \nabla^2_{ee} Q_i & \nabla^2_{fe} Q_i^T & \nabla^2_{te} Q_i^T \\ \nabla^2_{fe} Q_i & \nabla^2_{ff} Q_i & \nabla^2_{tf} Q_i^T \\ \nabla^2_{te} Q_i & \nabla^2_{tf} Q_i & \nabla^2_{tt} Q_i \end{bmatrix}. \tag{6.10}$$

The Hessian sub-matrices $\nabla^2_{ee} Q_i$, $\nabla^2_{fe} Q_i$ and $\nabla^2_{ff} Q_i$ are given by (A.18), (A.19) and (A.20), respectively. For each $(i, \cdot) \in \mathcal{T}$, two non-zeros of $\nabla^2_{te} Q_i$ are computed from (A.23) and (A.27), two non-zeros of $\nabla^2_{tf} Q_i$ are computed from (A.31) and (A.35), and one nonzero of $\nabla^2_{tf} Q_i$ is computed from (A.38). For each $(\cdot, i) \in \mathcal{T}$, two non-zeros of $\nabla^2_{te} Q_i$ are computed from (A.24) and (A.28), and two non-zeros of $\nabla^2_{tf} Q_i$ are computed from (A.32) and (A.36).

The Hessian of the voltage bound constraint (2.13c) has two non-zeros only; the two non-zeros are diagonal elements, and have value 2.

REMARK 6.1 *The Hessian sub-matrices* (A.15) *through* (A.20) *are constant and can be computed by direct reference to G_{ij} and B_{ij}, for all $j \in \mathcal{N}_i$. Moreover, if $\nabla^2_{ee} P_i$ ($\nabla^2_{ee} Q_i$) is known then $\nabla^2_{ff} P_i$ ($\nabla^2_{ff} Q_i$) is readily available. In practice, none of these Hessians need be individually formed; the Hessian matrix $\nabla^2_{xx} L_\mu(w)$ can be directly obtained.*

6.2.1 Lagrangian Hessian

Notice from (3.15) that each bus contributes with three Hessians in the composition of $\nabla^2_{xx}L_\mu(w)$: (i) the Hessian (6.9) that is associated with the constraint (2.12a); (ii) the Hessian (6.10) that is associated with one of the constraints (2.12b), (2.13a) or (2.13b); and (iii) the Hessian $\nabla^2_{xx}(V_i^2)$ that is associated with the constraint (2.13c). Given the special structure of these constraint Hessians, the Lagrangian Hessian $\nabla^2_{xx}L_\mu(w)$ can be computed by direct reference to the nonzero elements of $\boldsymbol{G}$ and $\boldsymbol{B}$. The computation is as follows:

- For $j = 1, 2, \ldots, |\mathcal{N}|$, compute the elements

$$\nabla^2_{e_je_j}L_\mu = 2(G_{jj}\lambda_j^p - B_{jj}\lambda_j^q + \lambda_j^v), \tag{6.11a}$$

$$\nabla^2_{e_ie_j}L_\mu = G_{ij}(\lambda_i^p + \lambda_j^p) - B_{ij}(\lambda_i^q + \lambda_j^q), \quad \text{for } i \in \mathcal{N}_j, \text{ with } i > j, \tag{6.11b}$$

$$\nabla^2_{f_ie_j}L_\mu = B_{ij}(\lambda_i^p - \lambda_j^p) + G_{ij}(\lambda_i^q - \lambda_j^q), \quad \text{for } i \in \mathcal{N}_j, \text{ with } i > j, \tag{6.12a}$$

$$\nabla^2_{f_ie_j}L_\mu = -\nabla^2_{f_je_i}L_\mu, \quad \text{for } i \in \mathcal{N}_j, \text{ with } i < j, \tag{6.12b}$$

$$\nabla^2_{f_jf_j}L_\mu = \nabla^2_{e_je_j}L_\mu, \quad \text{if } j \neq 1, \tag{6.13a}$$

$$\nabla^2_{f_if_j}L_\mu = \nabla^2_{e_ie_j}L_\mu, \quad \text{for } i \in \mathcal{N}_j, \text{ with } i > j > 1. \tag{6.13b}$$

  These formulae are displayed in Tables 6.1, 6.2 and 6.3.

- For each transformer $(i, j) \in \mathcal{T}$, compute the elements

$$\nabla^2_{t_{ij}e_i}L_\mu = \lambda_i^p(\nabla^2_{t_{ij}e_i}P_i) + \lambda_j^p(\nabla^2_{t_{ij}e_i}P_j) + \lambda_i^q(\nabla^2_{t_{ij}e_i}Q_i) + \lambda_j^q(\nabla^2_{t_{ij}e_i}Q_j), \tag{6.14a}$$

$$\nabla^2_{t_{ij}e_j}L_\mu = \lambda_i^p(\nabla^2_{t_{ij}e_j}P_i) + \lambda_j^p(\nabla^2_{t_{ij}e_j}P_j) + \lambda_i^q(\nabla^2_{t_{ij}e_j}Q_i) + \lambda_j^q(\nabla^2_{t_{ij}e_j}Q_j), \tag{6.14b}$$

$$\nabla^2_{t_{ij}f_i}L_\mu = \lambda_i^p(\nabla^2_{t_{ij}f_i}P_i) + \lambda_j^p(\nabla^2_{t_{ij}f_i}P_j) + \lambda_i^q(\nabla^2_{t_{ij}f_i}Q_i) + \lambda_j^q(\nabla^2_{t_{ij}f_i}Q_j), \tag{6.14c}$$

$$\nabla^2_{t_{ij}f_j}L_\mu = \lambda_i^p(\nabla^2_{t_{ij}f_j}P_i) + \lambda_j^p(\nabla^2_{t_{ij}f_j}P_j) + \lambda_i^q(\nabla^2_{t_{ij}f_j}Q_i) + \lambda_j^q(\nabla^2_{t_{ij}f_j}Q_j), \tag{6.14d}$$

$$\nabla^2_{t_{ij}t_{ij}}L_\mu = \lambda_i^p(\nabla^2_{t_{ij}t_{ij}}P_i) + \lambda_i^q(\nabla^2_{t_{ij}t_{ij}}Q_i), \tag{6.14e}$$

where λ_j^p is the negative of the Lagrange multiplier $y_{(\cdot)}$ related to bus j's constraint in (2.12a) ($\lambda_1^p := 1$); λ_j^q is either the negative of the Lagrange multiplier $y_{(\cdot)}$ related to bus j's constraint in (2.12b), or the Lagrange multiplier $z_{2_{(\cdot)}}$ related to bus j's constraint in (2.13a) or (2.13b); and λ_j^v is the Lagrange multiplier $z_{2_{(\cdot)}}$ related to bus j's constraint in (2.13c).

Making a mapping from *constraint multipliers* to *bus multipliers*, the bus multipliers $\boldsymbol{\lambda}^p$, $\boldsymbol{\lambda}^q$ and $\boldsymbol{\lambda}^v$ are obtained: $(y, z_2) \mapsto (\boldsymbol{\lambda}^p, \boldsymbol{\lambda}^q, \boldsymbol{\lambda}^v)$. This mapping plays an important role in the efficient implementation of the OPF algorithm; it not only considerably reduces the number of logical operations in the evaluation of $\nabla_x L_\mu(w^k)$ and $\nabla^2_{xx}L_\mu(w^k)$ but also allows for efficient data structure and reduced computer memory usage, as shown below.

Table 6.1: Computation of $\nabla^2_{ee} L_\mu(w)$, using the Equations (6.11) and symmetry.

	e_1	e_2	$\cdots$	e_n
e_1	$2(G_{11}\lambda_1^p - B_{11}\lambda_1^q + \lambda_1^v)$	$G_{21}(\lambda_2^p + \lambda_1^p)$ $-B_{21}(\lambda_2^q + \lambda_1^q)$	$\cdots$	$G_{n1}(\lambda_n^p + \lambda_1^p)$ $-B_{n1}(\lambda_n^q + \lambda_1^q)$
e_2	$G_{21}(\lambda_2^p + \lambda_1^p)$ $-B_{21}(\lambda_2^q + \lambda_1^q)$	$2(G_{22}\lambda_2^p - B_{22}\lambda_2^q + \lambda_2^v)$	$\cdots$	$G_{n2}(\lambda_n^p + \lambda_2^p)$ $-B_{n2}(\lambda_n^q + \lambda_2^q)$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
e_n	$G_{n1}(\lambda_n^p + \lambda_1^p)$ $-B_{n1}(\lambda_n^q + \lambda_1^q)$	$G_{n2}(\lambda_n^p + \lambda_2^p)$ $-B_{n2}(\lambda_n^q + \lambda_2^q)$	$\cdots$	$2(G_{nn}\lambda_n^p - B_{nn}\lambda_n^q + \lambda_n^v)$

Table 6.2: Computation of $\nabla^2_{fe} L_\mu(w)$, using the Equations (6.11) and "some symmetry".

	e_1	e_2	e_3	$\cdots$	e_n
f_2	$B_{21}(\lambda_2^p + \lambda_1^p)$ $+G_{21}(\lambda_2^q - \lambda_1^q)$	0	$-B_{32}(\lambda_3^p - \lambda_2^p)$ $-G_{32}(\lambda_3^q - \lambda_2^q)$	$\cdots$	$-B_{n2}(\lambda_n^p - \lambda_2^p)$ $-G_{n2}(\lambda_n^q - \lambda_2^q)$
f_3	$B_{31}(\lambda_3^p + \lambda_1^p)$ $+G_{31}(\lambda_3^q - \lambda_1^q)$	$B_{32}(\lambda_3^p - \lambda_2^p)$ $+G_{32}(\lambda_3^q - \lambda_2^q)$	0	$\cdots$	$-B_{n3}(\lambda_n^p - \lambda_3^p)$ $-G_{n3}(\lambda_n^q - \lambda_3^q)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
f_n	$B_{n1}(\lambda_n^p + \lambda_1^p)$ $+G_{n1}(\lambda_n^q - \lambda_1^q)$	$B_{n2}(\lambda_n^p - \lambda_2^p)$ $+G_{n2}(\lambda_n^q - \lambda_2^q)$	$B_{n3}(\lambda_n^p - \lambda_3^p)$ $+G_{n3}(\lambda_n^q - \lambda_3^q)$	$\cdots$	0

Table 6.3: Computation of $\nabla^2_{ff} L_\mu(w)$, discarding the row/column one from Table 6.1.

	f_2	f_3	$\cdots$	f_n
f_2	$2(G_{22}\lambda_2^p - B_{22}\lambda_2^q + \lambda_2^v)$	$G_{32}(\lambda_3^p + \lambda_2^p)$ $-B_{32}(\lambda_3^q + \lambda_2^q)$	$\cdots$	$G_{n2}(\lambda_n^p + \lambda_2^p)$ $-B_{n2}(\lambda_n^q + \lambda_2^q)$
f_3	$G_{32}(\lambda_3^p + \lambda_2^p)$ $-B_{32}(\lambda_3^q + \lambda_2^q)$	$2(G_{33}\lambda_3^p - B_{33}\lambda_3^q + \lambda_3^v)$	$\cdots$	$G_{n3}(\lambda_n^p + \lambda_3^p)$ $-B_{n2}(\lambda_n^q + \lambda_2^q)$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
f_n	$G_{n2}(\lambda_n^p + \lambda_2^p)$ $-B_{n2}(\lambda_n^q + \lambda_2^q)$	$G_{n3}(\lambda_n^p + \lambda_3^p)$ $-B_{n2}(\lambda_n^q + \lambda_2^q)$	$\cdots$	$2(G_{nn}\lambda_n^p - B_{nn}\lambda_n^q + \lambda_n^v)$

REMARK 6.2 *The Hessian* $\nabla^2_{xx} L_\mu(w)$ *is symmetric and highly sparse, with a (block) sparse structure that is similar to that of the load flow Jacobian. The areas* ①, ②, *and* ③ *in the Figure 6.1 are locations of the elements evaluated by* (6.11a), (6.11b) *and* (6.12a), *respectively; and the area* ④, *location of the elements evaluated by* (6.14). *The number of elements in* ①, ② *and* ③ *is* $|\mathcal{N}| + 2|\mathcal{B}|$, *the same number of elements in the matrix* $\boldsymbol{B}$. *The number of elements in* ④ *is* $5|\mathcal{T}|$.

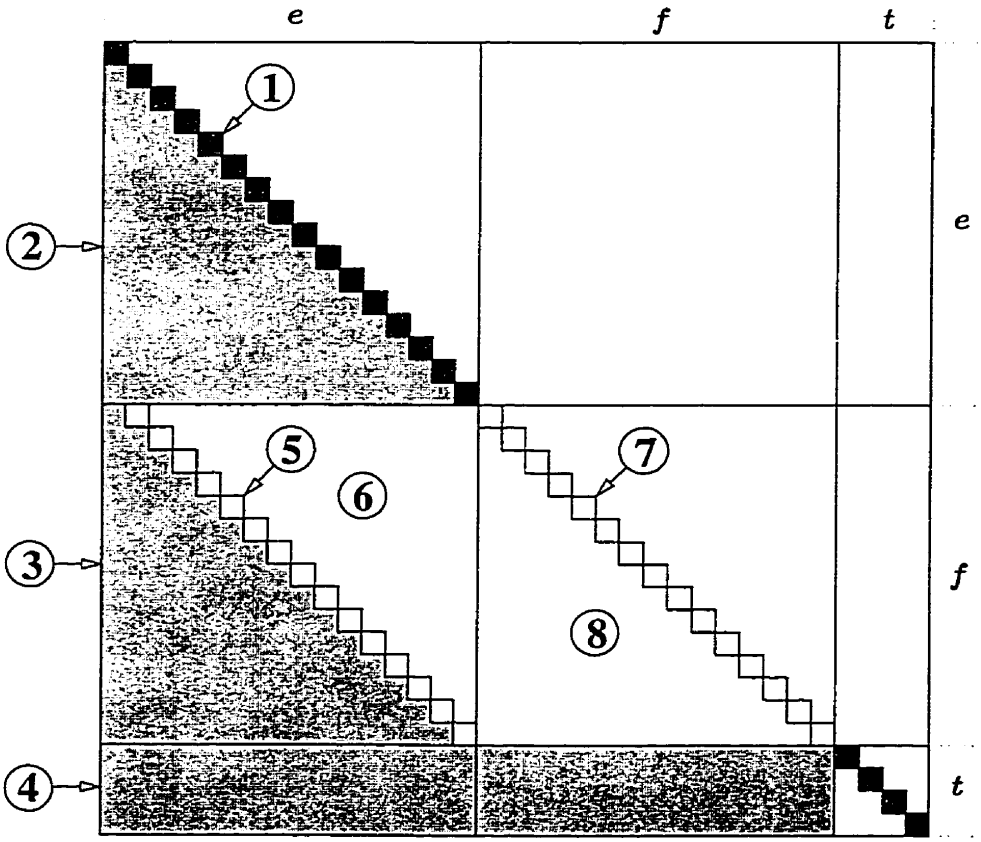

Figure 6.1: Locations of the computed non-zeros to obtain the Hessian $\nabla^2_{xx} L_\mu(w)$.

REMARK 6.3 *The number of* floating point operations (flops) *to compute* $\nabla^2_{xx} L_\mu(w)$ *can be easily estimated as a function of the number of buses* $|\mathcal{N}|$, *number of branches* $|\mathcal{B}|$, *and number of transformers* $|\mathcal{T}|$ *in the system. We need to compute* $|\mathcal{N}|$ *non-zeros in* ①, $|\mathcal{B}|$ *non-zeros in* ②, $|\mathcal{B}|$ *non-zeros in* ③, *and* $5|\mathcal{T}|$ *non-zeros in* ④. *The cost to compute each of the* $|\mathcal{N}| + 2|\mathcal{B}|$ *non-zeros in* ①, ② *and* ③ *is* 5 *flops. The cost to compute each set of* 5 *non-zeros in* ④ *is* 96 *flops. The total number of flops is, therefore,* $5|\mathcal{N}| + 10|\mathcal{B}| + 96|\mathcal{T}|$.

In the derivation of the above formulae for estimating the number of flops to compute $\nabla^2_{xx} L_\mu(w)$ we assume that for each pair of connected buses i and j there corresponds a single element $(i,j) \in \mathcal{B}$. Therefore, we have $|\mathcal{B}| = \frac{1}{2}\sum_{j\in\mathcal{N}} |\mathcal{N}_j|$. The above formulae for estimating the number of flops can be generalized if we simply take $|\mathcal{B}|$ as the number of elements below the diagonal of the susceptance matrix $\boldsymbol{B}$.

6.3 Gradients and Hessians: Polar Coordinates

In this section, we deal with the polar representation of (complex) voltages. The gradient vector $\nabla_x f(x)$ and the Jacobian matrices $\nabla_x g(x)^T$ and $\nabla_x h(x)^T$ are computed from Equations (A.39) through (A.50) in Appendix A.

The Hessian of the term $P_i(v,\theta,t)$ that appears in the constraint (2.20a), has the form

$$\nabla^2_{xx} P_i(x) = \begin{bmatrix} \nabla^2_{vv} P_i & \nabla^2_{\theta v} P_i^T & \nabla^2_{tv} P_i^T \\ \nabla^2_{\theta v} P_i & \nabla^2_{\theta\theta} P_i & \nabla^2_{t\theta} P_i^T \\ \nabla^2_{tv} P_i & \nabla^2_{t\theta} P_i & \nabla^2_{tt} P_i \end{bmatrix}. \tag{6.15}$$

The Hessian sub-matrices $\nabla^2_{vv} P_i$, $\nabla^2_{\theta v} P_i$ and $\nabla^2_{\theta\theta} P_i$ are given by (A.51), (A.52) and (A.53), respectively. For each $(i,\cdot) \in \mathcal{T}$, two non-zeros of $\nabla^2_{tv} P_i$ are computed from (A.57) and (A.61), two non-zeros of $\nabla^2_{t\theta} P_i$ are computed from (A.65) and (A.69), and one nonzero of $\nabla^2_{tt} P_i$ is computed from (A.73). For each $(\cdot,i) \in \mathcal{T}$, two non-zeros of $\nabla^2_{tv} P_i$ are computed from (A.58) and (A.62), and two non-zeros of $\nabla^2_{t\theta} P_i$ are computed from (A.66) and (A.70) (with proper switch of index i by index j, and vice versa).

The Hessian of the term $Q_i(v,\theta,t)$ that appears in the constraints (2.20b), (2.21a) and (2.21b), has the form

$$\nabla^2_{xx} Q_i(x) = \begin{bmatrix} \nabla^2_{vv} Q_i & \nabla^2_{\theta v} Q_i^T & \nabla^2_{tv} Q_i^T \\ \nabla^2_{\theta v} Q_i & \nabla^2_{\theta\theta} Q_i & \nabla^2_{t\theta} Q_i^T \\ \nabla^2_{tv} Q_i & \nabla^2_{t\theta} Q_i & \nabla^2_{tt} Q_i \end{bmatrix}. \tag{6.16}$$

The Hessian sub-matrices $\nabla^2_{vv} Q_i$, $\nabla^2_{\theta v} Q_i$ and $\nabla^2_{\theta\theta} Q_i$ are given by (A.54), (A.55) and (A.56), respectively. For each $(i,\cdot) \in \mathcal{T}$, two non-zeros of $\nabla^2_{tv} Q_i$ are computed from (A.59) and (A.63), two non-zeros of $\nabla^2_{t\theta} Q_i$ are computed from (A.67) and (A.71), and one nonzero of $\nabla^2_{tt} Q_i$ is computed from (A.74). For each $(\cdot,i) \in \mathcal{T}$, two non-zeros of $\nabla^2_{tv} Q_i$ are computed from (A.60) and (A.64), and two non-zeros of $\nabla^2_{t\theta} Q_i$ are computed from (A.68) and (A.72).

6.3.1 Lagrangian Hessian

Each bus now contributes with two Hessians in the composition of $\nabla^2_{xx}L_\mu(w)$: (i) the Hessian (6.15) that is associated with the constraint (2.20a); and (ii) the Hessian (6.16) that is associated with one of the constraints (2.20b), (2.21a) or (2.21b). The evaluation of $\nabla^2_{xx}L_\mu(w)$ is computationally less expensive if we use information from the Jacobian matrices $\nabla_x g(x)^T$ and $\nabla_x h(x)^T$, that is, if we compute the Hessian elements in terms of the elements H_{ij}, L_{ij}, M_{ij} and N_{ij} of the load flow Jacobian, which are defined by Equations (A.39) through (A.46). The computation of $\nabla^2_{xx}L_\mu(w)$ is as follows:

- For $j = 1, 2, \ldots, |\mathcal{N}|$, compute the elements

$$\nabla^2_{V_jV_j}L_\mu = 2(G_{jj}\lambda^p_j - B_{jj}\lambda^q_j), \tag{6.17a}$$

$$\nabla^2_{V_iV_j}L_\mu = \frac{N_{ij}\lambda^p_i + L_{ij}\lambda^q_i}{V_i} + \frac{N_{ji}\lambda^p_j + L_{ji}\lambda^q_j}{V_j}, \quad \text{for } i \in \mathcal{N}_j,\ i > j, \tag{6.17b}$$

$$\nabla^2_{\theta_jV_j}L_\mu = \frac{H_{jj}\lambda^p_j + M_{jj}\lambda^q_j + \sum_{l\in\mathcal{N}_j}(H_{lj}\lambda^p_l + M_{lj}\lambda^q_l)}{V_j}, \quad \text{if } j \neq 1, \tag{6.18a}$$

$$\nabla^2_{\theta_iV_j}L_\mu = -\frac{H_{ij}\lambda^p_i + M_{ij}\lambda^q_i - H_{ji}\lambda^p_j - M_{ji}\lambda^q_j}{V_j}, \quad \text{for } i \in \mathcal{N}_j,\ i > j, \tag{6.18b}$$

$$\nabla^2_{\theta_j\theta_j}L_\mu = -M_{jj}\lambda^p_j + H_{jj}\lambda^q_j + \sum_{l\in\mathcal{N}_j}(M_{lj}\lambda^p_l - H_{lj}\lambda^q_l), \quad \text{if } j \neq 1, \tag{6.19a}$$

$$\nabla^2_{\theta_i\theta_j}L_\mu = -M_{ij}\lambda^p_i + H_{ij}\lambda^q_i - M_{ji}\lambda^p_j + H_{ji}\lambda^q_j, \quad \text{for } i \in \mathcal{N}_j,\ i > j. \tag{6.19b}$$

  These formulae are displayed in Tables 6.4, 6.5 and 6.6.

- For each transformer $(i, j) \in \mathcal{T}$, compute the elements

$$\nabla^2_{t_{ij}V_i}L_\mu = \lambda^p_i(\nabla^2_{t_{ij}V_i}P_i) + \lambda^p_j(\nabla^2_{t_{ij}V_i}P_j) + \lambda^q_i(\nabla^2_{t_{ij}V_i}Q_i) + \lambda^q_j(\nabla^2_{t_{ij}V_i}Q_j), \tag{6.20a}$$

$$\nabla^2_{t_{ij}V_j}L_\mu = \lambda^p_i(\nabla^2_{t_{ij}V_j}P_i) + \lambda^p_j(\nabla^2_{t_{ij}V_j}P_j) + \lambda^q_i(\nabla^2_{t_{ij}V_j}Q_i) + \lambda^q_j(\nabla^2_{t_{ij}V_j}Q_j), \tag{6.20b}$$

$$\nabla^2_{t_{ij}\theta_i}L_\mu = \lambda^p_i(\nabla^2_{t_{ij}\theta_i}P_i) + \lambda^p_j(\nabla^2_{t_{ij}\theta_i}P_j) + \lambda^q_i(\nabla^2_{t_{ij}\theta_i}Q_i) + \lambda^q_j(\nabla^2_{t_{ij}\theta_i}Q_j), \tag{6.20c}$$

$$\nabla^2_{t_{ij}\theta_j}L_\mu = \lambda^p_i(\nabla^2_{t_{ij}\theta_j}P_i) + \lambda^p_j(\nabla^2_{t_{ij}\theta_j}P_j) + \lambda^q_i(\nabla^2_{t_{ij}\theta_j}Q_i) + \lambda^q_j(\nabla^2_{t_{ij}\theta_j}Q_j), \tag{6.20d}$$

$$\nabla^2_{t_{ij}t_{ij}}L_\mu = \lambda^p_i(\nabla^2_{t_{ij}t_{ij}}P_i) + \lambda^q_i(\nabla^2_{t_{ij}t_{ij}}Q_i). \tag{6.20e}$$

REMARK 6.4 *The block sparse structures of $\nabla^2_{xx}L_\mu(w)$ for the polar and rectangular voltage representations are alike. Area ① in Figure 6.1 indicates locations of the elements evaluated by* (6.17a)*; area ②, the elements evaluated by* (6.17b)*; area ⑤, the elements evaluated by*

Table 6.4: Computation of $\nabla^2_{vv} L_\mu(\boldsymbol{w})$, using the Equations (6.17) and symmetry.

	V_1	V_2	$\cdots$	V_n
V_1	$2(G_{11}\lambda_1^p - B_{11}\lambda_1^q)$	$(N_{21}\lambda_2^p + L_{21}\lambda_2^q)/V_2$ $+(N_{12}\lambda_1^p + L_{12}\lambda_1^q)/V_1$	$\cdots$	$(N_{n1}\lambda_n^p + L_{n1}\lambda_n^q)/V_n$ $+(N_{1n}\lambda_1^p + L_{1n}\lambda_1^q)/V_1$
V_2	$(N_{12}\lambda_1^p + L_{12}\lambda_1^q)/V_1$ $+(N_{21}\lambda_2^p + L_{21}\lambda_2^q)/V_2$	$2(G_{22}\lambda_2^p - B_{22}\lambda_2^q)$	$\cdots$	$(N_{n2}\lambda_n^p + L_{n2}\lambda_n^q)/V_n$ $+(N_{2n}\lambda_2^p + L_{2n}\lambda_2^q)/V_2$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
V_n	$(N_{1n}\lambda_1^p + L_{1n}\lambda_1^q)/V_1$ $+(N_{n1}\lambda_n^p + L_{n1}\lambda_n^q)/V_n$	$(N_{2n}\lambda_2^p + L_{2n}\lambda_2^q)/V_2$ $+(N_{n2}\lambda_n^p + L_{n2}\lambda_n^q)/V_n$	$\cdots$	$2(G_{nn}\lambda_n^p - B_{nn}\lambda_n^q)$

Table 6.5: Computation of $\nabla^2_{\theta v} L_\mu(\boldsymbol{w})$, using the Equations (6.17).

	V_1	V_2	$\cdots$	V_n
θ_2	$(H_{12}\lambda_1^p + M_{12}\lambda_1^q -$ $H_{21}\lambda_2^p - M_{21}\lambda_2^q)/V_1$	$(\sum_{l\in\mathcal{N}_2}(H_{l2}\lambda_l^p + M_{l2}\lambda_l^q)$ $+H_{22}\lambda_2^p + M_{22}\lambda_2^q)/V_2$	$\cdots$	$(H_{n2}\lambda_n^p + M_{n2}\lambda_n^q -$ $H_{2n}\lambda_2^p - M_{2n}\lambda_2^q)/V_n$
θ_3	$(H_{13}\lambda_1^p + M_{13}\lambda_1^q -$ $H_{31}\lambda_3^p - M_{31}\lambda_3^q)/V_1$	$(H_{23}\lambda_2^p + M_{23}\lambda_2^q -$ $H_{32}\lambda_3^p - M_{32}\lambda_3^q)/V_2$	$\cdots$	$(H_{n3}\lambda_n^p + M_{n3}\lambda_n^q -$ $H_{3n}\lambda_3^p - M_{3n}\lambda_3^q)/V_n$
$\vdots$	$\vdots$	$\vdots$		$\vdots$
θ_n	$(H_{1n}\lambda_1^p + M_{1n}\lambda_1^q -$ $H_{n1}\lambda_n^p - M_{n1}\lambda_n^q)/V_1$	$(H_{2n}\lambda_2^p + M_{2n}\lambda_2^q -$ $H_{n2}\lambda_2^p - M_{n2}\lambda_2^q)/V_2$	$\cdots$	$(\sum_{l\in\mathcal{N}_n}(H_{ln}\lambda_l^p + M_{ln}\lambda_l^q)$ $+M_{nn}\lambda_n^p + M_{nn}\lambda_n^q)/V_n$

Table 6.6: Computation of $\nabla^2_{\theta\theta} L_\mu(\boldsymbol{w})$, using the Equations (6.18) and symmetry.

	θ_2	θ_3	$\cdots$	θ_n
θ_2	$-M_{22}\lambda_2^p + H_{22}\lambda_2^q +$ $\sum_{l\in\mathcal{N}_2}(M_{l2}\lambda_l^p - H_{l2}\lambda_l^q)$	$-M_{32}\lambda_3^p + H_{32}\lambda_3^q$ $-M_{23}\lambda_2^p + H_{23}\lambda_2^q$	$\cdots$	$-M_{n2}\lambda_n^p + H_{n2}\lambda_n^q$ $-M_{2n}\lambda_2^p + H_{2n}\lambda_2^q$
θ_3	$-M_{23}\lambda_2^p + H_{23}\lambda_2^q$ $-M_{32}\lambda_3^p + H_{32}\lambda_3^q$	$-M_{33}\lambda_3^p + H_{33}\lambda_3^q +$ $\sum_{l\in\mathcal{N}_3}(M_{l3}\lambda_l^p - H_{l3}\lambda_l^q)$	$\cdots$	$-M_{n3}\lambda_n^p + H_{n3}\lambda_n^q$ $-M_{3n}\lambda_3^p + H_{3n}\lambda_3^q$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
θ_n	$-M_{2n}\lambda_2^p + H_{2n}\lambda_2^q$ $-M_{n2}\lambda_n^p + H_{n2}\lambda_n^q$	$-M_{3n}\lambda_3^p + H_{3n}\lambda_3^q$ $-M_{n3}\lambda_n^p + H_{n3}\lambda_n^q$	$\cdots$	$-M_{nn}\lambda_n^p + H_{nn}\lambda_n^q +$ $\sum_{l\in\mathcal{N}_n}(M_{ln}\lambda_l^p - H_{ln}\lambda_l^q)$

(6.17c); *areas* ③ *and* ⑥, *the elements evaluated by* (6.18b); *area* ⑦, *the elements evaluated by* (6.18c); *and area* ⑧, *the elements evaluated by* (6.19b). *The number of elements in areas* ①, ②, ③, ⑤, ⑥, ⑦, *and* ⑧ *is* $3|\mathcal{N}|+4|\mathcal{B}|-2|\mathcal{N}_1|-2$, *that is,* $2(|\mathcal{N}|+|\mathcal{B}|-|\mathcal{N}_1|-1)$ *additional elements evaluated in comparison with the number of elements for the rectangular voltage representation. Area* ④ *indicates locations of the elements that are evaluated by* (6.20); *the number of these elements is* $5|\mathcal{T}|$.

REMARK 6.5 *The number of flops to compute* $\nabla^2_{xx}L_\mu(w)$ *with the polar representation can also be easily estimated. Besides the non-zeros in common with the rectangular representation, we need to compute* $|\mathcal{N}|-1$ *non-zeros in* ⑤, $|\mathcal{B}|-|\mathcal{N}_1|$ *non-zeros in* ⑥, $|\mathcal{N}|-1$ *non-zeros in* ⑦, *and* $|\mathcal{B}|-|\mathcal{N}_1|$ *non-zeros in* ⑧. *The cost to compute each nonzero in* ① *is 4 flops, in* ② *is 9 flops, in* ③ *is 8 flops, in* ⑤ *is* $4|\mathcal{N}_j|+4$ *flops, in* ⑥ *is 2 flops (using terms from* ③*), in* ⑦ *is* $4|\mathcal{N}_j|+3$ *flops, and in* ⑧ *is 7 flops. Each set of five non-zeros in* ④ *requires* 61 *flops. The total number of flops is, therefore,* $11|\mathcal{N}|+42|\mathcal{B}|+61|\mathcal{T}|-17|\mathcal{N}_1|-7$.

6.4 Data Structures and Major Code Fragments

In this section, we describe data structures suitable for holding the sparse Jacobian and Hessian matrices; we assume that the voltages are given in rectangular coordinates. To take advantage of the mapping $(y, z_2) \mapsto (\lambda^p, \lambda^q, \lambda^v)$, we rewrite the Lagrangian Hessian

$$\nabla^2_{xx}L_\mu(w) = \nabla^2_{xx}f(x) - \sum_{j=1}^{m} y_j \nabla^2_{xx}g_j(x) + \sum_{j=1}^{p} z_{2_j} \nabla^2_{xx}h_j(x)$$

in the form

$$\nabla^2_{xx}L_\mu(w) = \sum_{i=1}^{|\mathcal{N}|} \Big[\lambda^p_i \nabla^2_{xx}P_i(x) + \lambda^q_i \nabla^2_{xx}Q_i(x) + \lambda^v_i \nabla^2_{xx}V_i^2(x)\Big]. \tag{6.21}$$

Similarly, we rewrite the Lagrangian gradient

$$\nabla_x L_\mu(w) = \nabla_x f(x) - \nabla_x g(x)y + \nabla_x h(x)z_2 + \widehat{I}^T z_4$$

in the form

$$\nabla_x L_\mu(w) = \nabla_x p(x)\lambda^p + \nabla_x q(x)\lambda^q + \nabla_x v(x)\lambda^v + \widehat{I}^T z_4, \tag{6.22}$$

where

$$\nabla_x p(x) := \big[\nabla_x P_1(x), \nabla_x P_2(x), \ldots, \nabla_x P_{|\mathcal{N}|}(x)\big], \tag{6.23}$$

$$\nabla_x q(x) := \big[\nabla_x Q_1(x), \nabla_x Q_2(x), \ldots, \nabla_x Q_{|\mathcal{N}|}(x)\big]. \tag{6.24}$$

The last two terms in the right-hand side of (6.22) yield the vector

$$\begin{pmatrix} 2e\lambda^v \\ 2f\lambda^v \\ z_4 \end{pmatrix}.$$

The vector products $e\lambda^v$ and $f\lambda^v$ are to be interpreted componentwise.

The Jacobian of the active power balance constraints, $\nabla_x p(x)^T$, and the Jacobian of the reactive power balance constraints, $\nabla_x q(x)^T$, have the same nonzero pattern. Therefore, they share the same pointers (adjacency structure) in the compact storage. The single common data structure for $\nabla_x p(x)^T$ and $\nabla_x q(x)^T$ is as follows:

`IROJ(I)` : pointer to the beginning of the non-zeros in the row `I` of $\nabla_x p(x)^T$ and $\nabla_x q(x)^T$.

`NCOJ(K)` : column number of the nonzero stored at the position `K`.

`PJAC(K)` : value of the nonzero element $\nabla_{x_j} P_i$ stored at the position `K`.

`QJAC(K)` : value of the nonzero element $\nabla_{x_j} Q_i$ stored at the position `K`.

A single data structure (adjacency structure) is considered for the bus-conductance and bus-susceptance matrices as well, as follows:

`IROY(I)` : pointer to the beginning of the non-zeros in the row `I` of $\boldsymbol{B}$ and $\boldsymbol{G}$.

`NCOY(K)` : column number of the nonzero stored at the position `K`.

`BBUS(K)` : value of the nonzero element B_{ij} stored at the position `K`.

`GBUS(K)` : value of the nonzero element G_{ij} stored at the position `K`.

The adjacency structures (`IROY`, `NCOY`) and (`IROJ`, `NCOJ`) are defined only once, before the iterative process starts. The following relations are considered in the Fortran 77 code fragments presented below:

Fortran	Formulae	Fortran	Formulae
`NBUS`	$\lvert\mathcal{N}\rvert$	`ZFOU(I)`	z_{4_i}
`NTAP`	$\lvert\mathcal{T}\rvert$	`PLAM(I)`	λ_i^p
`EVOL(I)`	e_i	`QLAM(I)`	λ_i^q
`FVOL(I)`	f_i	`A(.)`	$\nabla^2_{ww} L_\mu$
`ZTWO(I)`	z_{2_i}	`B(.)`	$\nabla_x L_\mu$

The Fortran 77 code fragment that implements the Hessian equations (6.11a) through (6.12a) is displayed below. The line `g:` implements (6.11a), the line `k:` implements (6.11b), the line `m:` implements (6.12a), and the empty line `n:` stands for the implementation of (6.12b) through (6.13b).

```
a:       DO 1010 J = 1 , NBUS
b:          PMUL = PLAM(J)
c:          QMUL = QLAM(J)
d:          KBEG = IROD(J)
e:          KEND = IROY(J+1) - 1
f:          L    = L + 1
g:          A(L) = 2.0 * (PMUL * GBUS(KBEG) - QMUL * BBUS(KBEG) + ZTWO(J))
h:          DO 1000 K = KBEG + 1 , KEND
i:             I    = NCOY(K)
j:             L    = L + 1
k:             A(L) = GBUS(K) * (PMUL + PLAM(I)) - BBUS(K) * (QMUL + QLAM(I))
l:             L    = L + 1
m:             A(L) = BBUS(K) * (PMUL - PLAM(I)) + GBUS(K) * (QMUL - QLAM(I))
n:             ...
o: 1000     CONTINUE
p: 1010  CONTINUE
```

The code fragment that implements the Lagrangian gradient equation (6.22) is displayed below. The line `j:` implements the first two terms in (6.22), whereas the lines `m:`, `n:` and `r:` implement the last two terms in (6.22).

```
a:       L    = NBUS - 1
b:       KEND = 0
c:       DO 1010 I = 1 , NBUS
d:          PMUL = PLAM(I)
e:          QMUL = QLAM(I)
f:          KBEG = KEND + 1
g:          KEND = IROJ(I+1) - 1
h:          DO 1000 K = KBEG , KEND
i:             J    = NCOJ(K)
j:             B(J) = B(J) + PMUL * PJAC(K) + QMUL * QJAC(K)
k: 1000     CONTINUE
l:          L    = L + 1
m:          B(I) = B(I) + 2.0 * ZTWO(I) * EVOL(I)
n:          B(L) = B(L) + 2.0 * ZTWO(I) * FVOL(I)
o: 1010  CONTINUE
p:       DO 1020 I = 1 , NTAP
q:          L    = L + 1
r:          B(L) = B(L) + ZFOU(I)
s: 1020  CONTINUE
```

6.4.1 A Block-Data Structure

The primal variables $\boldsymbol{x}$ have been, so far, arranged in one of two forms: $\boldsymbol{x} = (\boldsymbol{e}, \boldsymbol{f}, \boldsymbol{t})^T$ or $\boldsymbol{x} = (\boldsymbol{v}, \boldsymbol{\theta}, \boldsymbol{t})^T$. In computational practice, the variables associated with the sub-matrix

$$\boldsymbol{R} := \begin{bmatrix} \mathbf{0} & \nabla_{\boldsymbol{x}} \boldsymbol{h}^T & \mathbf{0} \\ \nabla_{\boldsymbol{x}} \boldsymbol{h} & \nabla^2_{\boldsymbol{xx}} L_\mu & -\nabla_{\boldsymbol{x}} \boldsymbol{g} \\ \mathbf{0} & -\nabla_{\boldsymbol{x}} \boldsymbol{g}^T & \mathbf{0} \end{bmatrix} \tag{6.25}$$

should be arranged in a way to form 5×5 blocks, when using rectangular coordinates

	λ_j^v	λ_j^q	λ_j^p	f_j	e_j		λ_i^v	λ_i^q	λ_i^p	f_i	e_i
λ_j^v	0	0	0	$2f_j$	$2e_j$		0	0	0	0	0
λ_j^q	0	0	0	$\widehat{M}_{jj}$	$\widehat{L}_{jj}$		0	0	0	$\widehat{M}_{ii}$	$\widehat{L}_{ii}$
λ_j^p	0	0	0	$\widehat{H}_{jj}$	$\widehat{N}_{jj}$	$\cdots$	0	0	0	$\widehat{H}_{ii}$	$\widehat{N}_{ii}$
f_j	$2f_j$	$\widehat{M}_{jj}$	$\widehat{H}_{jj}$	$\nabla^2_{f_j f_j} L_\mu$	0		0	$\widehat{M}_{ij}$	$\widehat{H}_{ij}$	$\nabla^2_{f_i f_j} L_\mu$	$\nabla^2_{f_i e_j} L_\mu$
e_j	$2e_j$	$\widehat{L}_{jj}$	$\widehat{N}_{jj}$	0	$\nabla^2_{e_j e_j} L_\mu$		0	$\widehat{L}_{ij}$	$\widehat{N}_{ij}$	$\nabla^2_{f_j e_i} L_\mu$	$\nabla^2_{e_i e_j} L_\mu$
				$\vdots$		$\ddots$				$\vdots$	
λ_i^v	0	0	0	0	0		0	0	0	$2f_i$	$2e_i$
λ_i^q	0	0	0	$\widehat{M}_{ij}$	$\widehat{L}_{ij}$		0	0	0	$\widehat{M}_{ii}$	$\widehat{L}_{ii}$
λ_i^p	0	0	0	$\widehat{H}_{ij}$	$\widehat{N}_{ij}$	$\cdots$	0	0	0	$\widehat{H}_{ii}$	$\widehat{N}_{ii}$
f_i	0	$\widehat{M}_{ii}$	$\widehat{H}_{ii}$	$\nabla^2_{f_i f_j} L_\mu$	$\nabla^2_{f_i e_j} L_\mu$		$2f_i$	$\widehat{M}_{ii}$	$\widehat{H}_{ii}$	$\nabla^2_{f_i f_i} L_\mu$	0
e_i	0	$\widehat{L}_{ii}$	$\widehat{N}_{ii}$	$\nabla^2_{f_j e_i} L_\mu$	$\nabla^2_{e_i e_j} L_\mu$		$2e_i$	$\widehat{L}_{ii}$	$\widehat{N}_{ii}$	0	$\nabla^2_{e_i e_i} L_\mu$

and to form 4×4 blocks, when using polar coordinates

	λ_j^q	λ_j^p	θ_j	V_j		λ_i^q	λ_i^p	θ_i	V_i
λ_j^q	0	0	M_{jj}	L_{jj}		0	0	M_{ii}	L_{ii}
λ_j^p	0	0	H_{jj}	N_{jj}	$\cdots$	0	0	H_{ii}	N_{ii}
θ_j	M_{jj}	H_{jj}	$\nabla^2_{\theta_j \theta_j} L_\mu$	$\nabla^2_{\theta_j V_j} L_\mu$		M_{ij}	H_{ij}	$\nabla^2_{\theta_i \theta_j} L_\mu$	$\nabla^2_{\theta_i V_j} L_\mu$
V_j	L_{jj}	N_{jj}	$\nabla^2_{\theta_j V_j} L_\mu$	$\nabla^2_{V_j V_j} L_\mu$		L_{ij}	N_{ij}	$\nabla^2_{\theta_i V_j} L_\mu$	$\nabla^2_{V_i V_j} L_\mu$
			$\vdots$		$\ddots$			$\vdots$	
λ_i^q	0	0	M_{ij}	L_{ij}		0	0	M_{ii}	L_{ii}
λ_i^p	0	0	H_{ij}	N_{ij}	$\cdots$	0	0	H_{ii}	N_{ii}
θ_i	M_{ii}	H_{ii}	$\nabla^2_{\theta_i \theta_j} L_\mu$	$\nabla^2_{\theta_i V_j} L_\mu$		M_{ii}	H_{ii}	$\nabla^2_{\theta_i \theta_i} L_\mu$	$\nabla^2_{\theta_i V_i} L_\mu$
V_i	L_{ii}	N_{ii}	$\nabla^2_{\theta_i V_j} L_\mu$	$\nabla^2_{V_i V_j} L_\mu$		L_{ii}	N_{ii}	$\nabla^2_{\theta_i V_i} L_\mu$	$\nabla^2_{V_i V_i} L_\mu$

The idea of arranging the OPF variables in blocks was first proposed in [66], in the context of a Newton's OPF. This arrangement allows for efficient ordering and block factorization.

Rather than examining fill ins in the elimination of $12|\mathcal{B}|+6|\mathcal{N}|$ elements below the diagonal of $\boldsymbol{R}$, we examine block fill ins in the elimination of $|\mathcal{B}|$ off-diagonal blocks.

6.5 Solving the Linear Systems

Primal-dual codes devote the greatest part of their computational effort to solving large, sparse, structured linear equation systems of the form

$$\begin{bmatrix} \boldsymbol{D}^2 & \boldsymbol{A}^T \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{x} \end{pmatrix} = \begin{pmatrix} \boldsymbol{v} \\ \boldsymbol{w} \end{pmatrix}. \tag{6.26}$$

For instance, the reduced system (3.23) defines $\boldsymbol{D}^2 = \nabla_{\boldsymbol{x}} \boldsymbol{d}^T$ and $\boldsymbol{A} = -\nabla_{\boldsymbol{x}} \boldsymbol{g}^T$. The linear system (6.26) is usually solved in one of two forms: (i) the *normal-equations* form, involving the symmetric positive definite matrix $\boldsymbol{A}\boldsymbol{D}^2\boldsymbol{A}^T$, or (ii) the *augmented-system* form, exploiting the specially structured symmetric indefinite matrix in (6.26).

A major advantage of the normal-equations form is that all elimination pivot orders for finding the LU factors L_{ij} are stable [29]. Unfortunately, in NLP, the diagonal block matrix $\boldsymbol{D}^2$ is a general sparse symmetric matrix rather than a diagonal matrix, and, despite $\boldsymbol{D}^2$ and $\boldsymbol{A}$ being sparse, the matrix $\boldsymbol{A}\boldsymbol{D}^2\boldsymbol{A}^T$ is generally dense. Sparsity preserving requirements make the augmented-system approach the natural choice to solving (6.26), as concerned with IP methods for NLP. The major difficulty, in this case, is that it cannot be guaranteed that all pivoting orders are numerically stable. An ordering/symbolic factorization phase attempts to choose a pivot ordering based on sparsity that will lead to low fill-ins. When the factorization is computed with the actual numerical values, interchanges that alter the predicted pivot sequence may be required to retain numerical stability.

6.5.1 Factorization of Symmetric Indefinite Matrices

Most factorizations of a symmetric indefinite matrix $\boldsymbol{T}$ have the form [25, 76]

$$\boldsymbol{P}\boldsymbol{T}\boldsymbol{P}^T = \boldsymbol{L}\widehat{\boldsymbol{D}}\boldsymbol{L}^T, \tag{6.27}$$

where $\boldsymbol{P}$ is a permutation matrix, $\boldsymbol{L}$ is a unit lower triangular matrix, and $\widehat{\boldsymbol{D}}$ is a block diagonal matrix in which each block is either 1×1 or symmetric 2×2. A stable elimination procedure is due to Bunch and Parlett [5], known as the *Bunch-Parlett procedure*. The Bunch-Parlett procedure eliminate either one or two columns and rows of the matrix at a time. Each elimination step is performed with a diagonal pivot block selected from

the remaining matrix $\widetilde{T}$—either a single diagonal element of $\widetilde{T}$, or a 2×2 pivot block whose diagonal entries are also diagonal elements of $\widetilde{T}$. A row/column permutation is then performed to move the pivot block to the upper left of the remaining matrix, to obtain

$$\widetilde{P}\widetilde{T}\widetilde{P}^T = \begin{bmatrix} \widetilde{E} & \widetilde{C}^T \\ \widetilde{C} & \widehat{T} \end{bmatrix}, \tag{6.28}$$

where $\widetilde{E}$ is the 1×1 or 2×2 pivot block. Performing the actual numerical operations yields

$$\widetilde{P}\widetilde{T}\widetilde{P}^T = \begin{bmatrix} I & 0 \\ \widetilde{C}\widetilde{E}^{-1} & I \end{bmatrix} \begin{bmatrix} \widetilde{E} & 0 \\ 0 & \widehat{T} - \widetilde{C}\widetilde{E}^{-1}\widetilde{C}^T \end{bmatrix} \begin{bmatrix} I & 0 \\ \widetilde{C}\widetilde{E}^{-1} & I \end{bmatrix}^T. \tag{6.29}$$

The algorithm includes the column(s) $\widetilde{C}\widetilde{E}^{-1}$ in the L factor, includes the pivot block $\widetilde{E}$ in the $\widehat{D}$ factor, and updates the matrix $\widetilde{T} \leftarrow \widehat{T} - \widetilde{C}\widetilde{E}^{-1}\widetilde{C}^T$. The entire process is now repeated on the remaining matrix $\widehat{T} - \widetilde{C}\widetilde{E}^{-1}\widetilde{C}^T$.

The key criterion for choosing the pivot block $\widetilde{E}$ is that the growth of elements in the remaining matrix $\widehat{T} - \widetilde{C}\widetilde{E}^{-1}\widetilde{C}^T$ is not too great. The Bunch-Parlett procedure examines all elements in the remaining matrix and identifies the largest diagonal and largest off-diagonal elements, whose magnitudes are denoted by χ_{diag} and χ_{off}, respectively. The ratio $\chi_{\text{off}}/\chi_{\text{diag}}$ indicates the growth that we can expect in the remaining matrix if the largest diagonal is selected as a 1×1 pivot. If the ratio is not too large, this diagonal element is selected as the pivot. Otherwise, a 2×2 pivot of the form

$$\widetilde{E} = \begin{bmatrix} \widetilde{T}_{ii} & \widetilde{T}_{ij} \\ \widetilde{T}_{ij} & \widetilde{T}_{jj} \end{bmatrix} \tag{6.30}$$

is chosen, where $\widetilde{T}_{ij}$ is the off-diagonal element that achieves the largest magnitude χ_{off}.

For sparse matrices, a second criterion is used in selecting a pivot: the update step $\widetilde{T} \leftarrow \widehat{T} - \widetilde{C}\widetilde{E}^{-1}\widetilde{C}^T$ should not create too much fill-in. Fourer and Mehrotra [25] describe a modification of the Bunch-Parlett strategy that combines stability and sparsity consideration. Their strategy determines the number of non-zeros in the update matrices $\widetilde{C}\widetilde{E}^{-1}\widetilde{C}^T$, where $\widetilde{E}$ is the set of all possible 1×1 and 2×2 pivots in the remaining matrix. It first examines all pivots for which the update matrices have the minimum number of non-zeros and checks to see whether any of them satisfies a stability condition that prevents excessive growth. For a prospective 1×1 pivot, this condition is

$$|\widetilde{T}_{ii}^{-1}| \|\widetilde{T}_{*i}\|_\infty \leq \delta^{-1}, \tag{6.31}$$

whereas the 2×2 stability condition is

$$\left| \begin{bmatrix} \widetilde{T}_{ii} & \widetilde{T}_{ij} \\ \widetilde{T}_{ij} & \widetilde{T}_{jj} \end{bmatrix}^{-1} \right| \begin{bmatrix} \|\widetilde{\boldsymbol{T}}_{*i}\|_\infty \\ \|\widetilde{\boldsymbol{T}}_{*j}\|_\infty \end{bmatrix} \leq \begin{bmatrix} \delta^{-1} \\ \delta^{-1} \end{bmatrix}, \tag{6.32}$$

where δ is a small parameter—according to experiments in [25], values of δ in the range of 16^{-4} to 16^{-6} give consistently good factorization. If any of the eligible pivots satisfy one of these conditions, they are selected, with a preference given to 1×1 pivots. Otherwise, the procedure examines pivots for which the update matrix $\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{E}}^{-1}\widetilde{\boldsymbol{C}}^T$ is successively more dense, stopping when it finds a pivot that satisfies one of the stability conditions above.

In contrast to the positive definite case, there is no general way to find a numerically stable pivot order for an indefinite matrix using only symbolic information—the ordering process cannot be dissociated from the numerical factorization. However, once a stable pivot order has been determined, we can reuse it at subsequent iterations as long as it continues to give satisfactory factorizations and solutions. For each system of the form $\boldsymbol{T}\boldsymbol{z} = \boldsymbol{r}$ that is solved using that factorization, the solution is deemed satisfactory if

$$\|\boldsymbol{T}\boldsymbol{z} - \boldsymbol{r}\|_\infty \leq 10^{-5}. \tag{6.33}$$

If a reused pivot order is found to give an unsatisfactory factorization or solution then we perform another dynamic ordering/factorization step and saves the new pivot order for reuse on subsequent iterations.

6.5.2 UMFPACK: A Public Domain Linear System Solver

In our algorithms implementations, we have utilized the public domain linear system solver UMFPACK Version 2.2, of Timothy Davis and Iain Duff, University of Florida. The acronym UMFPACK stands for *Unsymmetric-pattern MultiFrontal PACKage*. The source code and technical reports describing UMFPACK are available via the World Wide Web at `http://www.cise.ufl.edu/~davis`, or by anonymous ftp at `ftp.cise.ufl.edu:pub/faculty/davis`. About features of UMFPACK, we quote the following from the code documentation:

> UMFPACK Version 2.2 is a package for solving systems of sparse linear systems, $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, where $\boldsymbol{A}$ is sparse and can be unsymmetric. It is written in ANSI Fortran 77. There are options for choosing a good pivot order, factorizing a subsequent matrix with the same pivot order and nonzero pattern as a previously factorized matrix, and solving systems of linear equations with the factors (with $\boldsymbol{A}$, $\boldsymbol{L}$, or $\boldsymbol{U}$; or with their transposes in the single/double precision

> versions). Iterative refinement, with sparse backward error estimates, can be performed. Single and double precision, complex, and complex double precision (complex*16) routines are available.

About the method implemented in UMFPACK, we quote the following from the code documentation:

> The multifrontal method factorizes a large sparse matrix using a sequence of small dense frontal matrices. The square frontal matrices are factorized efficiently using dense matrix kernels. Classical multifrontal methods assume a symmetric nonzero pattern. The unsymmetric-pattern multifrontal method (UMFPACK), relaxes this assumption by using rectangular frontal matrices. High performance is achieved by using dense matrix kernels to factorize these rectangular frontal matrices, and also through an approximate degree update algorithm that is much faster (asymptotically and in practice) than computing the exact degrees. Since a general sparse code must select pivots based on both numerical and symbolic (fill-reducing) criteria, the analysis phase (pivot selection and symbolic factorization) and the numerical factorization are combined. The rectangular frontal matrices are constructed dynamically, since the structure is not known prior to factorization.
>
> Version 2.2 of UMFPACK combines features of both unifrontal and multifrontal methods. In the multifrontal method, in contrast with a (uni-)frontal method, several frontal matrices are used. Each is used for one or more pivot steps, and the resulting Schur complement is summed with other Schur complements to generate another frontal matrix. Although this means that arbitrary sparsity patterns can be handled efficiently, extra work is required to add the Schur complements together and can be costly because indirect addressing is required. The frontal method avoids this extra work by factorizing the matrix with a single frontal matrix. Rows and columns are added to the frontal matrix, and pivot rows and columns are removed. Data movement is simpler, but higher fill-in can result if the matrix cannot be permuted into a variable-band form with small profile. UMFPACK Version 2.2 is based on a combined unifrontal/multifrontal algorithm that enables a general fill-in reduction ordering to be applied but avoiding the data movement of previous multifrontal approaches.

For more information on UMFPACK, see [14, 15].

6.6 Final Remarks

In this chapter, we have discussed many issues that are directly related to an effective implementation of the algorithms proposed in this thesis. A few remarks are as follows:

- We have presented four initialization heuristics. Two of the heuristics conform with the whole set of IP algorithms, and the four of them conform with the NIP algorithm. The numerical performance of each initialization heuristic has been numerically tested, as discussed in Chapter 7.

- We have derived explicit formulae to efficiently assemble the Hessian matrices, both in rectangular and in polar coordinates. We show that this task is more efficiently done in rectangular coordinates, where the functions Hessians are constant.

- We have proposed a mapping from constraint multipliers to bus multipliers: $(y, z_2) \mapsto (\lambda^p, \lambda^q, \lambda^v)$. Such a mapping considerably reduces the number of logical operations in the evaluation of the Lagrangian gradient and Hessian, and allows for efficient data structure and savings in computer memory usage. An efficient evaluation of the Lagrangian gradient is crucial for the line search procedure wherein this task is repeatedly performed.

- We have derived explicit formulae to estimate the number of flops required to obtain the Lagrangian Hessian—the major effort in forming the Newton system—both in rectangular and in polar coordinates.

- We have presented two code fragments that emphasize some of the advantages of the proposed mapping of Lagrange multipliers.

- We also have discussed a block-data structure and the solution of symmetric indefinite systems. However, such a block-data structure have not been implemented in our computer codes; we have used a public domain indefinite system solver that does not take advantage of block-data structures.

Chapter 7

Computational Experiments

In this chapter, we present numerical experiments with the IP and NIP algorithms for NLP previously described in Chapters 3, 4 and 5. These experiments relate to the solution of the nonlinear *reactive power dispatch* (RPD) problems (2.18) and (2.23). The main purposes of the computational tests are: (i) to gain insight into the various methods, (ii) to compare competing methods, (iii) to tune parameters in the algorithms, and (iv) to compare the performance of the proposed algorithms in the solution of the rectangular and polar "versions" of the RPD problem. To this end, we have initially developed prototype codes under the MATLAB® (The MathWorks, Inc.) environment and tested on the well known IEEE test systems with 6, 30, 57, 118 and 300 buses. To be able to solve larger problems, and fairly compare competing methods, we also have coded all algorithms in Fortran 77 (in double precision arithmetic) and compiled with the -O2 option. All the numerical results presented have been produced on a Pentium Pro 200 MHz with 64 MBytes of RAM, running the Linux operating system.

This chapter is organized as follows. In the next section, we display the names and the main features of the developed codes. In Section 7.2, we describe the set of test power systems. Several statistics are given related to the dimension of these problems and an indication of the degree of difficulty (the number of binding constraints) to solve them. In Section 7.3, we discuss numerical results with the set of IP algorithms. A default set of parameters, initialization heuristic, step length scheme, and updating formula of the barrier parameter has been defined and effects of changing this set in different ways are examined. In Section 7.4, we discuss numerical results with the NIP algorithm. In Section 7.5, we make some further comparisons of the rectangular and polar variants of the RPD problem. Final remarks in Section 7.6 close the chapter.

7.1 The Developed OPF Codes

We have written a set of twelve computer codes in ANSI Fortran 77. The compiler used is the GNU Fortran 77 compiler that is included in common Linux distributions. The set of twelve OPF codes associates a variant of the OPF problem—minimum transmission active power losses—that is formulated either in rectangular (R) or in polar (P) coordinates, with the five IP algorithms that are described in Chapters 3 and 4, and the NIP algorithm that is described in Chapter 5. The name of the codes, as we address them in the discussions in this chapter, along with their major distinguishing features are displayed in Table 7.1.

Table 7.1: The names and some distinguishing features of the developed OPF codes.

Code Name	Coordinates	Optimization Technique	Description
R-IPD P-IPD	Rectangular Polar	Infeasible Primal-Dual IP Method	Algorithm 3.2
R-PCM P-PCM	Rectangular Polar	Predictor-Corrector IP Method	Algorithm 4.1
R-PCN P-PCN	Rectangular Polar	Perturbed Composite Newton IP Method	Algorithm 4.2
R-MPC P-MPC	Rectangular Polar	Multiple Predictor-Corrector IP Method	Algorithm 4.3
R-MCC P-MCC	Rectangular Polar	Multiple Centrality Corrections IP Method	Algorithm 4.4
R-NIP P-NIP	Rectangular Polar	Non-Interior-Point Continuation Method	Algorithm 5.1

7.2 The Test Power Systems

The performance of the OPF codes is tested on a set of eleven power systems that range in size from 14 to 2098 buses. Some statistics for the test power systems are displayed in Table 7.2, where, for each power system, we give the total number of buses ($|\mathcal{N}|$), the number of *generator* buses ($|\mathcal{G}|$), the number of load buses *eligible* for shunt var control ($|\mathcal{E}|$), the number of load buses with *fixed* var sources ($|\mathcal{F}|$), the total number of *branches* ($|\mathcal{B}|$), and the number of *transformers* with LTC device ($|\mathcal{T}|$). Table 7.2 also displays the initial active power losses in MW and in percentage of the total system load.

Table 7.2: Statistics for the test power systems.

Test System	Number of Buses and Branches						Initial P_{loss}	
	$\lvert\mathcal{N}\rvert$	$\lvert\mathcal{G}\rvert$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{F}\rvert$	$\lvert\mathcal{B}\rvert$	$\lvert\mathcal{T}\rvert$	(MW)	(%)
IEEE-14	14	5	1	8	20	3	13.38	5.16
IEEE-24	24	11	1	12	38	5	51.55	1.95
IEEE-30	30	6	5	19	41	4	17.62	6.22
IEEE-57	57	7	5	45	80	10	27.99	2.24
IEEE-118	118	54	12	52	186	9	129.88	3.54
IEEE-300	300	69	23	208	411	35	408.43	1.76
MEXI-256	256	58	23	175	376	50	210.18	2.02
IEMX-555	555	126	46	383	787	85	617.79	1.84
BRAS-340	340	59	52	229	684	12	1802.91	4.25
BRAS-810	810	114	185	511	1340	166	1767.43	4.89
BRAS-2098	2098	169	426	1503	3283	239	1173.61	6.00

The basic load flow data for the IEEE test systems is available by anonymous ftp at `wahoo.ee.uwashington.edu`. The basic load flow data for the MEXI-256 test system, a *longitudinal* system that is derived from the Mexican power network, is obtained from [63]. The test system IEMX-555 has been obtained as a combination of the IEEE-300 and MEXI-256 test systems. The power systems BRAS-340, BRAS-810 and BRAS-2098 are modified reduced systems derived from the actual Brazilian power network.

The physical and the operational limits for MEXI-256, IEMX-555 and the set of IEEE test systems have been defined as $\pm 5\%$ off nominal value for the load bus voltages (V_i, for $i \in \mathcal{F} \cup \mathcal{E}$), as $\pm 2\%$ off specified values for the generator bus voltages (V_i, for $i \in \mathcal{G}$), and as $\pm 10\%$ off nominal value for the transformer tap settings (t_{ij}). To test algorithm efficiency and robustness, solution difficulty has been increased by choosing small sets of buses eligible for shunt var control (a small set $\mathcal{E}$) and setting fairly narrow limits for the reactive power outputs of generators. Such a specification of the OPF problem shrinks its feasible region and, therefore, increases the chances of a large number of binding constraints to occur in its solution.

Some statistics for the solved NLP problems are displayed in Table 7.3, where, for each problem, we give the number of primal variables (n), the number of equality constraints (m), the number of nonlinear functional inequalities (p), and the number of simple bound constraints (q). Notice that the number of nonlinear functional inequality constraints and of simple bound constraints are indeed double the figures displayed in Table 7.3 since each inequality is subject to both lower and upper bounds. Also displayed in Table 7.3 are the

final active power losses in MW and in percentage of the total load, as well as the number of reactive power limits (Q), of voltage limits (V) and of transformer tap setting limits (t) that are activated in the optimal point.

Table 7.3: Sizes of the NLP problem (1.1), final losses and number of active limits.

Problem	Test System	NLP Problem Sizes				Final P_{loss}		Active Limits		
		n	m	p	q	(MW)	(%)	V	Q	t
1	IEEE-14	30	21	20	3	13.13	5.07	3	5	0
2	IEEE-24	52	35	36	5	49.05	1.72	9	3	0
3	IEEE-30	63	48	41	4	17.78	6.27	3	2	1
4	IEEE-57	123	101	69	10	26.56	2.12	9	2	1
5	IEEE-118	244	169	184	9	119.23	3.25	14	23	0
6	IEEE-300	634	507	392	35	378.51	1.63	62	28	1
7	MEXI-256	561	430	337	50	198.24	1.91	52	13	0
8	IEMX-555	1194	937	727	85	577.80	1.72	110	45	1
9	BRAS-340	691	568	451	12	1734.60	4.09	55	40	0
10	BRAS-810	1785	1320	1109	166	1667.91	4.61	145	163	6
11	BRAS-2098	4434	3600	2693	239	1110.00	5.67	176	329	24

7.3 Experiments with the Interior-Point Algorithms

In this section, we describe extensive computational experiments with the IP algorithms. To avoid testing an unreasonable large number of combinations of parameters, initialization heuristics, step length computation schemes, and updating formulae of the barrier parameter, a "reference" set of parameters and formulae has been defined and the effect of changing this set in different ways is examined. Initially, we employ the initialization Heuristic–A (see Section 6.1.1), the computation of the step lengths by Scheme–B (see Section 3.4.2), and the computation of the barrier parameter by the standard procedure (see Section 3.5.1). The parameters default to

Parameter	μ^0	σ^0	α_0	γ	$\beta_{\min}$	$\beta_{\max}$	M	ϵ_1	ϵ_2	ϵ_μ
Value	1.0	0.2	0.99995	0.35	0.1	10.0	1	10^{-4}	10^{-6}	10^{-12}

For the test runs in which the OPF algorithms have converged, the tables below mostly display the number of iterations (iters) and the elapsed CPU times (time). All reported CPU times are defined in *seconds*, and do not include the time for data I/O. In the case

an algorithm has failed to converge, we then give an indication of the main reason for that failure. For example, by α^8 we mean that after the 8th iteration the step lengths are nearly zero and, consequently, the algorithm fails to progress further. By ν_2^{50} we mean that the convergence criteria ν_2 has not been satisfied after 50 iterations. At the bottom of each table, we display the total number of iterations and CPU time to solve the whole set of eleven problems.

7.3.1 Performance with Default Parameters

The results of experiments carried out with the default algorithms settings (parameters and formulae) are displayed in Table 7.4 (voltages in rectangular coordinates) and Table 7.5 (voltages in polar coordinates). In these experiments, the IP algorithms have employed the same settings for all problems. That is, the IP algorithms have not been "tuned" to particular problems. In doing so, it is likely that for some problems some algorithms will fail to converge to an optimal solution. Nevertheless, we consider instructive to present and discuss non-converged cases, and, whenever possible, to identify modifications of the algorithms that might restore convergence of the iterative process. As a result, we expect to identify some strength and weakness of the proposed algorithms.

The numerical results reported in Table 7.4 show that the R-IPD code has failed to solve the Problems 6, 8, 9 and 10. Similar convergence difficulty has been observed with the P-IPD code, as shown in Table 7.5. The iterative process for Problem 6, up to iteration 9, is illustrated in Table 7.6. We can infer from Table 7.6 that the R-IPD code using the default parameters and formulae has failed to solve Problem 6 because the step length α^k prematurely becomes too small (nearly zero). Since $\alpha^9 \approx 0$, the variables remain practically unchanged after the 9th iteration. Consequently, the primal-dual IP algorithm fails to progress further in reducing the primal and dual infeasibility, and the complementarity gap.

It is clear from the ratio tests (3.28) and (3.29) that the step lengths may adversely be set to very small values whenever one or more variable with strict positivity conditions prematurely approaches zero. This may happen for a number of reasons such as the starting point w^0 being a badly centered one, the barrier parameter μ^k being reduced too fast, and so forth. In the case of Problem 6, we manage to restore convergence of the primal-dual IP iterations in two simple ways: (i) reducing the step length safety factor from $\alpha_0 = 0.99995$ to $\alpha_0 = 0.9$ (in which case iters = 19 and time = 6.84), and (ii) increasing the centering parameter from $\sigma^0 = 0.2$ to $\sigma^0 = 0.7$ (in which case iters = 31 and time = 11.40).

Table 7.4: Runs with default parameters: IP codes using Rectangular Coordinates.

Problem	R-IPD		R-PCM		R-PCN		R-MPC		R-MCC	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	14	1.68	13	1.58	8	0.95	15	1.78	11	1.32
2	15	1.92	8	1.05	9	1.16	8	1.11	11	1.41
3	14	1.82	9	1.20	8	1.05	9	1.17	10	1.31
4	17	2.56	10	1.54	10	1.54	10	1.56	12	1.84
5	18	3.61	11	2.30	10	2.11	11	2.29	14	2.90
6	α^9		11	4.25	11	4.30	11	4.23	16	6.23
7	19	5.81	11	3.48	11	3.50	11	3.46	14	4.46
8	α^9		13	8.66	12	8.08	13	8.63	17	11.32
9	α^8		11	4.77	11	4.81	11	4.77	16	6.90
10	α^5		12	11.89	13	13.36	12	11.88	21	20.98
11	26	61.19	25	61.57	21	52.57	19	47.76	27	66.56
Total	—		134	102.29	124	93.43	130	88.64	169	125.23

Table 7.5: Runs with default parameters: IP codes using Polar Coordinates.

Problem	P-IPD		P-PCM		P-PCN		P-MPC		P-MCC	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	14	1.70	13	1.56	8	0.95	15	1.76	11	1.27
2	15	1.94	8	1.06	9	1.14	8	1.03	11	1.44
3	13	1.72	9	1.19	8	1.06	9	1.15	10	1.29
4	16	2.41	10	1.51	10	1.57	10	1.54	12	1.84
5	17	3.42	11	2.28	11	2.29	11	2.31	15	3.05
6	α^{10}		11	4.09	12	4.53	11	4.11	15	5.63
7	17	5.21	11	3.41	11	3.49	11	3.49	14	4.43
8	α^9		13	8.44	13	8.53	13	8.42	17	11.15
9	α^8		11	4.67	11	4.69	11	4.66	15	6.37
10	α^6		13	12.22	13	12.46	13	12.28	19	17.77
11	24	54.00	24	57.67	26	62.81	22	54.07	28	66.83
Total	—		134	98.10	132	103.52	134	94.82	167	121.07

As we reduce the safety factor α_0 we aim at keeping the variables with non-negativity conditions a bit further away from the boundary of the positive orthant. As we increase the centering parameter σ^0 we aim at decreasing the barrier parameter μ^k at more modest rates. By means of these two simple algorithm modifications, we also are able to restore convergence for the other cases displayed in Tables 7.4 and 7.5 in which the R-IPD and

Table 7.6: Non-converged iterative process for Problem 6: R-IPD code.

k	α^k	ν_1^k	ν_2^k	ν_3^k	ν_4^k
0	—	$0.330 \times 10^{+0}$	$0.133 \times 10^{+2}$	$0.854 \times 10^{+3}$	—
1	0.61825	$0.218 \times 10^{+0}$	$0.648 \times 10^{+1}$	$0.217 \times 10^{+2}$	0.207×10^{-1}
2	0.59557	0.869×10^{-1}	$0.323 \times 10^{+1}$	$0.112 \times 10^{+2}$	0.814×10^{-3}
3	0.77363	0.196×10^{-1}	$0.848 \times 10^{+0}$	$0.417 \times 10^{+1}$	0.767×10^{-3}
4	1.00000	0.462×10^{-2}	0.361×10^{-2}	$0.800 \times 10^{+0}$	0.315×10^{-2}
5	1.00000	0.422×10^{-1}	0.538×10^{-3}	$0.150 \times 10^{+0}$	0.142×10^{-1}
6	0.54681	0.189×10^{-1}	0.367×10^{-3}	0.817×10^{-1}	0.993×10^{-2}
7	0.75333	0.109×10^{-1}	0.155×10^{-3}	0.308×10^{-1}	0.467×10^{-2}
8	0.00011	0.109×10^{-1}	0.155×10^{-3}	0.308×10^{-1}	0.390×10^{-6}
9	0.00000	0.109×10^{-1}	0.155×10^{-3}	0.308×10^{-1}	0.227×10^{-10}

P-IPD codes have failed to solve.

The numerical results displayed in Tables 7.4 and 7.5 suggest that the IP algorithms are practically insensitive to the dimension of the NLP problems, as far as concerned with the number of iterations, which is a typical feature of Newton-type methods. If we define the size of the NLP problem (1.1) as $n + m + p + q$, then we can infer from Tables 7.3, 7.4 and 7.5 that while the NLP problem sizes have increased from 74 to 10566 (factor of 142), the number of iterations have increased from 8 to 28 (factor of 3.5).

The results displayed in Tables 7.4 and 7.5 also suggest the superiority of the higher-order IP variants over the plain primal-dual IP method (R-IPD and P-IPD), not only in terms of number of iterations and CPU times but also in terms of robustness; all eleven problems have been solved by the higher-order IP methods using the default parameters. As far as concerned with the total number of iterations to solve the eleven problems, the best computational performance has been that of the R-PCN code, with 124 iterations. As far as concerned with the total CPU time, the best performance has been that of the R-MPC code, with 88.64 seconds. Apparently, the worse performance has been that of the R-MCC and P-MCC codes. We show below that these MCC codes indeed perform very well when they use other parameter settings.

The higher-order IP variants employ predictor and corrector steps to improve the order of accuracy to which the Newton directions approximate the nonlinear KKT equations. In these IP variants, the centrality of the iterates are improved by solving two or more linear systems within each iteration. This allows for larger steps to be taken towards a solution, as illustrated in Figures 7.1 and 7.2, such as better convergence rates are usually achieved.

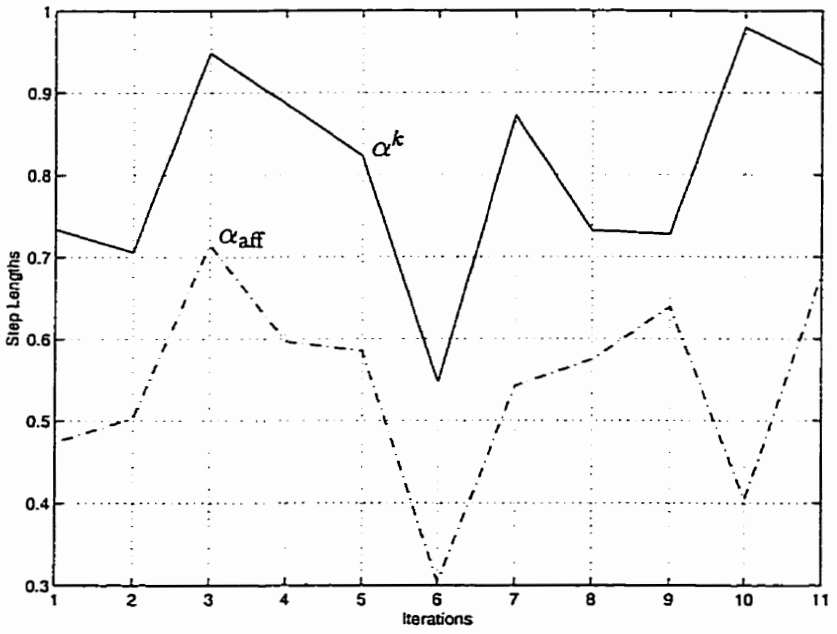

Figure 7.1: Increase of the step lengths through predictor-corrector steps: Problem 6 solved by R-PCM code using the default parameters and formulae.

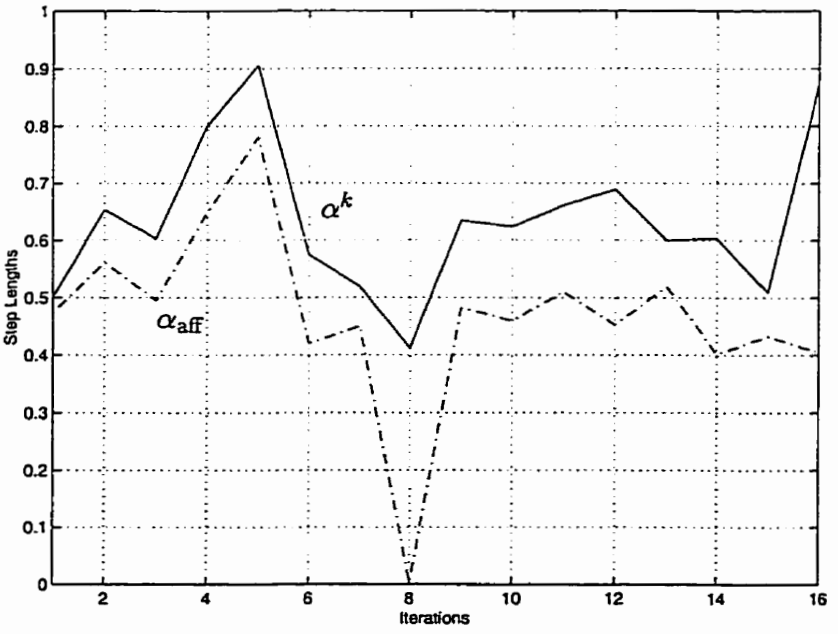

Figure 7.2: Increase of the step lengths through centrality correction steps: Problem 6 solved by R-MCC code using the default parameters and formulae.

In Table 7.7, we display the CPU times (in seconds and in percentage of the total solution time) elapsed in the major steps of the predictor-corrector IP algorithm, when it solves Problem 11. The first three lines in Table 7.7 relate to set up of the data structures to hold the matrices, and the numerical evaluation of the bus admittance matrices G and B; these tasks need to be performed only once, before the iterative process begins. Notice that the symbolic/numerical factorization of the coefficient matrix $\nabla^2_{ww} L_\mu(w)$ demands much more computational effort than the linear system solutions that use this factorization (85.46% versus 3.90%). As the higher-order IP methods aim at reducing the number of matrix factorizations to a necessary minimum, they usually requires less CPU times than the plain primal-dual IP method.

Table 7.7: Elapsed CPU times in each major step of the R-PCM code: Problem 11.

Set the nonzero structure and compute the bus matrices G and B :	0.77	1.25
Set the nonzero structure for the Jacobians $\nabla_x p(x)^T$ and $\nabla_x q(x)^T$:	0.03	0.05
Set the nonzero structure for the Hessian matrix $\nabla^2_{ww} L_\mu(w)$:	0.04	0.07
Form the Newton system $\nabla^2_{ww} L_\mu(w^k)\Delta w = -\nabla_w L_\mu(w^k)$:	3.35	5.44
Perform the symbolic/numerical factorization of $\nabla^2_{ww} L_\mu(w^k)$:	52.62	85.46
Linear system solutions :	2.40	3.90
Compute the primal and dual step lengths :	0.12	0.20
Update the primal and dual variables :	0.22	0.36
Evaluate the nonlinear function-vectors $g(x^k)$ and $h(x^k)$:	0.12	0.20
Compute the Jacobian matrices $\nabla_x p(x^k)^T$ and $\nabla_x q(x^k)^T$:	0.20	0.32
Test for convergence :	0.25	0.41
Update the barrier parameter μ^k :	0.04	0.07
Other :	1.40	2.27
Total (secs and %) :	61.57	100.00

7.3.2 Influence of Initialization Heuristics

In Tables 7.8 and 7.9, we display the results of experiments performed with the IP algorithms using the initialization Heuristic–B (see Section 6.1.2). The results for the initialization Heuristic–A are displayed in Tables 7.4 and 7.5. We may infer from Tables 7.8 and 7.9 that the numerical performance for the initialization Heuristic–B is also good; however, such a performance is inferior to the performance for the initialization Heuristic–A.

We remark that the most aggravated performance relates to the MCC method. This is

Table 7.8: Influence of initialization heuristics: Heuristic–B, Rectangular Coordinates.

Problem	R-IPD		R-PCM		R-PCN		R-MPC		R-MCC	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	13	1.54	11	1.33	8	0.96	13	1.54	11	1.33
2	14	1.81	8	1.04	9	1.17	8	1.03	12	1.59
3	11	1.42	8	1.03	8	1.03	9	1.15	9	1.16
4	15	2.23	9	1.36	9	1.32	9	1.34	12	1.82
5	17	3.36	11	2.23	11	2.24	11	2.21	19	3.75
6	α^{10}		12	4.50	12	4.53	12	4.50	20	7.42
7	18	5.44	11	3.41	12	3.72	11	3.45	16	5.00
8	α^{10}		13	8.39	13	8.45	13	8.51	23	14.88
9	17	6.78	13	5.46	13	5.54	13	5.44	26	10.66
10	α^{7}		13	12.21	13	12.46	15	14.12	27	25.28
11	α^{14}		25	59.89	ν_1^{50}		22	51.60	41	97.03
Total	—		134	100.85	—		136	94.89	216	169.92

Table 7.9: Influence of initialization heuristics: Heuristic–B, Polar Coordinates.

Problem	P-IPD		P-PCM		P-PCN		P-MPC		P-MCC	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	12	1.40	10	1.20	9	1.07	10	1.23	11	1.30
2	14	1.76	8	1.06	9	1.19	8	1.03	12	1.52
3	11	1.39	8	1.04	7	0.90	8	1.02	10	1.30
4	13	1.92	10	1.49	11	1.63	10	1.46	12	1.79
5	17	3.36	11	2.22	11	2.23	11	2.19	19	3.78
6	α^{6}		13	4.73	13	4.74	14	5.06	23	8.31
7	α^{5}		13	3.98	13	4.03	13	4.03	18	5.51
8	α^{7}		14	8.77	14	8.86	16	9.93	25	15.66
9	17	6.85	13	5.32	13	5.27	14	5.67	29	11.65
10	α^{6}		17	15.25	15	13.85	34	29.56	36	32.14
11	α^{13}		26	59.81	28	66.04	20	46.10	49	111.13
Total	—		143	104.87	143	109.81	158	107.28	244	194.09

due, in part, to the fact that the starting points obtained by Heuristic–A are usually better centered than the starting points obtained by Heuristic–B. We may recall from Section 6.1 that, in the Heuristic–A, we first choose $\mu^0 > 0$ and then compute z_1^0, z_2^0, z_3^0 and z_4^0; hence, all complementarity products $s_i^0 z_i^0$ have the same value μ^0, which is beneficial for the MCC algorithm. In the Heuristic–B, on the contrary, we first estimate z_1^0, z_2^0, z_3^0 and

z_4^0 and, afterwards, we use these estimates to define a $\mu^0 > 0$. Thus, there may be large discrepancies between the complementarity products; since only one centrality correction has been allowed within each IP iteration ($M = 1$) the MCC algorithm has been unable to properly correct the centrality of the iterates.

Once more, the R-IPD and P-IPD codes have failed to solve the most significant problems, for the same reason they have failed to solve these problems using the Heuristic–A. Among the higher-order IP algorithms, the only failure that is reported has occurred with the R-PCN code, which has failed to satisfy the primal feasibility condition for the Problem 11 (BRAS-2098) after 50 iterations. In spite of such a failure, on the whole the codes using rectangular coordinates (R-) have shown a computational performance slightly superior to the performance of the codes using polar coordinates (P-).

7.3.3 Influence of Step Length Procedures

The results displayed in Tables 7.10 through 7.15 relate to performance evaluation of the step length rules. More specifically, in Tables 7.10 and 7.11, we display the results of experiments performed with the IP algorithms using the step length Scheme–A (see Section 3.4.1). The results for the Scheme–B are displayed in Tables 7.4 and 7.5. In Tables 7.12 and 7.13, we display the results for the step length Scheme–C (see Section 3.4.3). Finally, in Tables 7.14 and 7.15, we display the results of experiments with the R-PCM and P-PCM codes using Scheme–B with different values of the safety factor α_0.

With respect to the results for the Scheme–A, we observe that the higher-order IP methods have once more performed well. Their performance with Scheme–A indeed has been slightly superior to their performance with Scheme–B, as far as concerned with the number of iterations and CPU times. All the higher-order IP algorithms have solved the eleven problems. We remark, however, that the R-PCM code required a reasonable large number of iterations to solve Problem 11, as compared with the number of iterations required by the other IP codes as well as by the same code using the Scheme–B. In such a case, we have observed that for several iterations of the R-PCM code using Scheme–A either the primal step length or the dual step length has been set too close to zero; consequently, the convergence of the primal and the dual feasibility has occurred to be quite slow.

Still related to the Scheme–A, we observe that the R-IPD and P-IPD codes using this step length rule have been able to solve most of the problems which they had failed to solve using the Scheme–B (see Tables 7.4 and 7.5). We may recall that the Scheme–A

Table 7.10: Influence of step length procedures: Scheme–A, Rectangular Coordinates.

Problem	R-IPD		R-PCM		R-PCN		R-MPC		R-MCC	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	19	2.24	9	1.07	8	0.95	9	1.09	9	1.07
2	28	3.57	8	1.02	8	1.04	8	1.04	8	0.99
3	20	2.55	8	1.05	8	1.04	8	1.06	10	1.32
4	25	3.71	9	1.41	9	1.38	9	1.38	10	1.49
5	33	6.56	10	2.08	10	2.09	10	2.08	14	2.88
6	23	8.49	10	3.88	11	4.32	10	3.88	13	5.02
7	32	9.88	10	3.14	10	3.18	11	3.52	13	4.15
8	35	22.23	11	7.37	11	7.34	11	7.35	15	9.90
9	29	11.93	11	4.67	11	4.75	11	4.84	16	6.85
10	ν_1^{50}		11	11.00	12	12.19	11	10.99	19	19.00
11	ν_3^{50}		50	126.70	25	62.65	19	47.05	28	68.67
Total	—		147	163.39	123	100.93	117	84.28	155	121.34

Table 7.11: Influence of step length procedures: Scheme–A, Polar Coordinates.

Problem	P-IPD		P-PCM		P-PCN		P-MPC		P-MCC	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	18	2.10	9	1.10	8	0.95	9	1.11	9	1.09
2	29	3.64	8	1.01	8	1.01	8	1.01	8	1.05
3	21	2.71	8	1.00	8	1.04	8	1.04	10	1.24
4	25	3.69	10	1.51	9	1.37	10	1.49	10	1.49
5	34	6.70	11	2.28	10	2.09	11	2.20	14	2.85
6	22	7.77	10	3.76	11	4.22	10	3.83	13	4.93
7	33	10.03	11	3.41	10	3.18	11	3.42	14	4.43
8	37	22.86	11	7.17	12	7.95	11	7.16	15	9.75
9	17	6.61	10	4.22	10	4.26	10	4.21	16	6.66
10	ν_1^{50}		11	10.65	12	11.69	11	10.61	17	15.99
11	50	110.70	24	57.13	26	62.81	20	47.17	23	54.31
Total	—		123	93.24	124	100.57	119	83.25	149	103.79

takes separate step lengths in the primal and dual spaces. Since in these experiments it has never occurred to both the primal and the dual step lengths being set close to zero, the primal-dual IP algorithm has been able to recover from adverse situations in which one of the steps has been set close to zero.

We remark that, among the various IP algorithms, the best numerical performance

Table 7.12: Influence of step length procedures: Scheme–C, Rectangular Coordinates.

Problem	R-IPD		R-PCM		R-PCN		R-MPC		R-MCC	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	24	2.89	9	1.07	8	0.98	9	1.09	49	5.87
2	21	2.68	8	1.02	8	1.04	8	1.04	39	5.04
3	20	2.57	8	1.05	8	1.04	8	1.06	41	5.30
4	25	3.71	9	1.35	11	1.66	9	1.38	38	5.61
5	33	6.58	10	2.08	10	2.09	10	2.08	ν_3^{50}	
6	23	8.49	10	3.88	11	4.32	10	3.88	38	14.34
7	22	6.75	10	3.21	10	3.18	10	3.14	ν_2^{50}, ν_3^{50}	
8	36	22.75	11	7.44	12	8.03	11	7.46	40	26.32
9	18	7.14	10	4.33	10	4.28	10	4.36	43	18.18
10	23	21.73	11	11.00	11	11.40	11	10.88	48	48.46
11	$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$		$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$		$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$		$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$		$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$	
Total	—		—		—		—		—	

Table 7.13: Influence of step length procedures: Scheme–C, Polar Coordinates.

Problem	P-IPD		P-PCM		P-PCN		P-MPC		P-MCC	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	16	1.88	9	1.07	8	0.98	11	1.28	ν_2^{50}	
2	29	3.66	8	1.02	8	1.04	10	1.22	37	4.58
3	21	2.69	8	1.05	8	1.04	ν_2^{50}		40	5.06
4	25	3.67	9	1.41	9	1.38	10	1.48	ν_2^{50}, ν_3^{50}	
5	34	6.73	11	2.25	10	2.09	ν_2^{50}, ν_3^{50}		ν_2^{50}, ν_3^{50}	
6	22	7.82	ν_2^{50}, ν_3^{50}		11	4.27	$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$		$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$	
7	ν_3^{50}		$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$		11	3.51	$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$		ν_3^{50}	
8	37	22.79	12	7.83	13	8.58	$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$		$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$	
9	17	6.82	11	4.67	10	4.26	$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$		47	19.35
10	ν_1^{50}, ν_2^{50}		11	10.53	11	10.69	$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$		$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$	
11	$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$		24	56.79	ν_2^{50}, ν_3^{50}		$\nu_1^{50}, \nu_2^{50}, \nu_3^{50}$		ν_2^{50}, ν_3^{50}	
Total	—		—		—		—		—	

with the Scheme–A has been that of the MPC algorithm, both in rectangular and in polar coordinates.

With respect to the results for the Scheme–C, we can observe from Tables 7.12 and 7.13 that none of the IP codes have performed satisfactorily with this step length rule, mainly the "versions" in polar coordinates (Table 7.13). For instance, all codes but P-PCM have failed to

Table 7.14: Influence of the size of safety factor α_0: R-PCM code.

Problem	$\alpha_0 = 0.95$		$\alpha_0 = 0.97$		$\alpha_0 = 0.98$		$\alpha_0 = 0.995$		$\alpha_0 = 0.9995$	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	10	1.19	10	1.19	10	1.19	10	1.19	10	1.19
2	10	1.28	9	1.16	9	1.16	8	1.03	8	1.03
3	10	1.30	9	1.16	9	1.16	9	1.16	9	1.16
4	11	1.66	11	1.66	11	1.66	10	1.51	10	1.51
5	11	2.26	11	2.26	11	2.29	11	2.29	11	2.27
6	12	4.67	12	4.65	11	4.21	12	4.59	11	4.27
7	12	3.75	12	3.79	11	3.43	11	3.47	11	3.47
8	13	8.63	13	8.61	13	8.67	13	8.63	13	8.70
9	12	5.11	12	5.16	12	5.18	13	5.60	11	4.77
10	12	11.89	12	12.05	12	11.89	11	10.86	11	10.86
11	25	61.84	24	59.09	22	53.42	23	57.26	24	58.59
Total	138	103.58	135	100.78	131	94.26	131	97.59	129	97.82

Table 7.15: Influence of the size of safety factor α_0: P-PCM code.

Problem	$\alpha_0 = 0.95$		$\alpha_0 = 0.97$		$\alpha_0 = 0.98$		$\alpha_0 = 0.995$		$\alpha_0 = 0.9995$	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	11	1.33	10	1.21	10	1.21	9	1.09	10	1.19
2	10	1.28	9	1.16	9	1.16	9	1.16	9	1.16
3	10	1.30	9	1.16	9	1.16	9	1.16	9	1.16
4	11	1.66	11	1.66	11	1.66	10	1.51	10	1.51
5	12	2.48	11	2.26	11	2.29	11	2.29	11	2.27
6	12	4.65	12	4.65	12	4.55	11	4.15	11	4.27
7	13	4.14	12	3.80	12	3.75	11	3.47	11	3.47
8	14	9.20	13	8.57	13	8.57	13	8.51	13	8.50
9	12	5.19	12	5.16	11	4.71	12	5.14	11	4.75
10	13	12.46	12	11.50	12	11.43	12	11.40	12	11.36
11	27	65.05	26	63.40	24	58.38	35	79.75	25	60.28
Total	145	108.74	137	104.53	134	98.87	142	119.63	132	99.92

solve the largest problem (at least in the case we allow them to perform only one corrector step per iteration). The reason is that in most of the iterations the dual step length is set too close to zero; consequently, the primal and dual feasibility, and the complementarity conditions simultaneously fail to be satisfied, as can be inferred from Tables 7.12 and 7.13.

We remark, however, that our IP algorithms do not incorporate other major features of

the Yamashita–Yabe IP algorithm, for which the step length Scheme–C has been designed; the lack of these features may have deteriorated the practical performance of the step length Scheme–C as this step length rule is implemented in our IP algorithms.

In Tables 7.14 and 7.15, we display the results of experiments performed with several values of the step length safety factor α_0. We have carried out such an analysis only in the context of the predictor-corrector IP method; we believe that the conclusions drawn for the predictor-corrector IP method may be extended to the other higher-order IP methods. Notice from Tables 7.14 and 7.15 that the best performance has been obtained with safety factors in the interval $0.98 \leq \alpha_0 \leq 0.9995$. We also can observe that the closer α_0 is to unit the lower is the number of IP iterations. On the other hand, as we have discussed above, a safety factor of $\alpha_0 = 0.9$, or even lower, may be required as a means of restoring convergence for non-converged runs with the plain primal-dual IP method.

With respect to the high number of iterations to solve Problem 11, that is required by the P-PCM code using $\alpha_0 = 0.995$, we remark that in four IP iterations the algorithm made steps $\alpha^k < 10^{-4}$, such that convergence of the equality constraints turned to be very slow.

7.3.4 Influence of μ^0 and Updating Formulae of μ^k

In Tables 7.16 through 7.19, we deal with the influence of μ^0 in the convergence process of the plain primal-dual IP algorithm and the predictor-corrector IP algorithm. We begin our analysis with a discussion of the non-converged cases with the plain primal-dual IP algorithm. We infer from Table 7.17 that the iterative process for Problem 3 (IEEE-30) has failed to converge with $\mu^0 = 0.001$ because the step lengths are prematurely (iteration 5) set too close to zero. However, we have observed that the P-IPD code solves this problem with $\mu^0 = 0.001$ if we simply reduce the safety factor from $\alpha_0 = 0.99995$ to $\alpha_0 = 0.9$ (in which case iters = 12 and time = 1.60). Similarly, the iterative process for Problem 4 (IEEE-57), which has failed to converge with $\mu^0 = 0.001$, converges if we set $\alpha_0 = 0.9$ (in which case iters = 18 and time = 2.70).

With reference to Table 7.16, the iterative process for Problem 6 (IEEE-300), which has failed to converge with $\mu^0 = 10$, converges if we set $\alpha_0 = 0.9$ (iters = 20 and time = 7.38) or increase the centering parameter from $\sigma^0 = 0.2$ to $\sigma^0 = 0.7$ (iters = 35 and time = 13.04). The iterative process for Problem 6, which has failed to converge with $\mu^0 = 1$, converges if we set $\alpha_0 = 0.9$ (iters = 19 and time = 6.84) or set $\sigma^0 = 0.7$ (iters = 31 and time = 11.40); and converges with $\mu^0 = 0.1$ and $\mu^0 = 0.001$ if we set $\sigma^0 = 0.7$ (iters = 21 and time = 7.72).

Table 7.16: Influence of μ^0 in the plain primal-dual IP method: R-IPD code.

Problem	$\mu^0 = 10$		$\mu^0 = 1$		$\mu^0 = 0.1$		$\mu^0 = 0.01$		$\mu^0 = 0.001$	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	15	1.82	14	1.68	12	1.44	11	1.32	12	1.44
2	15	1.92	15	1.92	13	1.69	12	1.57	11	1.43
3	15	1.98	14	1.82	11	1.45	10	1.30	11	1.44
4	16	2.43	17	2.56	15	2.23	13	1.97	α^5	
5	18	3.87	18	3.61	18	3.61	15	3.00	14	2.82
6	α^{12}		α^9		α^8		14	5.13	α^6	
7	20	6.19	19	5.81	17	5.20	16	4.89	α^5	
8	α^{11}		α^9		α^7		16	10.05	α^9	
9	α^{10}		α^8		18	7.21	α^7		α^{10}	
10	α^5		α^5		α^4		15	13.95	α^{10}	
11	28	68.89	25	58.71	22	50.12	α^6		α^7	
Total	—		—		—		—		—	

Table 7.17: Influence of μ^0 in the plain primal-dual IP method: P-IPD code.

Problem	$\mu^0 = 10$		$\mu^0 = 1$		$\mu^0 = 0.1$		$\mu^0 = 0.01$		$\mu^0 = 0.001$	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	15	1.81	14	1.70	13	1.59	11	1.34	12	1.46
2	17	2.19	15	1.94	13	1.71	11	1.47	12	1.56
3	15	1.97	13	1.72	11	1.42	10	1.31	α^5	
4	17	2.58	16	2.41	15	2.28	13	1.95	α^4	
5	18	3.67	17	3.42	17	3.40	15	3.03	15	3.00
6	α^{12}		α^{10}		α^9		14	5.09	α^6	
7	20	6.19	17	5.21	17	5.19	16	4.88	α^6	
8	α^{12}		α^9		α^8		16	9.89	α^8	
9	α^{10}		α^8		17	6.82	α^7		α^5	
10	α^5		α^6		α^4		α^6		α^4	
11	29	69.50	24	55.19	22	49.06	23	50.07	α^8	
Total	—		—		—		—		—	

Except for minor changes in the number of iterations and CPU times, the same has been observed for Problem 6 in Table 7.17. We also have observed that the convergence process with higher values of σ^0 has been characterized by slow decrease of the complementarity gap (criterion ν_3) and by step lengths close to unit.

Still related to Tables 7.16 and 7.17, the iterative process for Problem 7 (MEXI-256),

which has failed to converge with $\mu^0 = 0.001$, converges if we set $\sigma^0 = 0.8$ (iters = 29 and time = 8.85).

The iterative process for Problem 8 (TEST-555), which has failed to converge with $\mu^0 = 10$, converges if we set $\sigma^0 = 0.7$ (iters = 35 and time = 22.53). It converges with $\mu^0 = 1$ if we set $\alpha_0 = 0.9$ (iters = 20 and time = 12.78) or set $\sigma^0 = 0.7$ (iters = 32 and time = 20.44); it converges with $\mu^0 = 0.1$ if we set $\alpha_0 = 0.9$ (iters = 18 and time = 11.24) or set $\sigma^0 = 0.7$ (iters = 35 and time = 21.86); and it converges with $\mu^0 = 0.001$ if we set $\alpha_0 = 0.9$ (iters = 20 and time = 12.78) or set $\sigma^0 = 0.7$ (iters = 27 and time = 16.99).

The iterative process for Problem 9 (BRAS-340), which has failed to converge with $\mu^0 = 10.0$, converges if we set $\alpha_0 = 0.9$ (iters = 22 and time = 9.29) or set $\sigma^0 = 0.7$ (iters = 34 and time = 14.71); it converges with $\mu^0 = 1.0$ if we set $\alpha_0 = 0.9$ (iters = 20 and time = 8.10) or set $\sigma^0 = 0.7$ (iters = 31 and time = 12.81); and it converges with $\mu^0 = 0.01$ if we set $\sigma_0 = 0.7$ (iters = 24 and time = 10.15). The same has been observed for the non-converged tests with the P-IPD code.

The iterative process for Problem 10 (BRAS-810), which has failed to converge with $\mu^0 = 10$, converges if we set $\sigma^0 = 0.4$ (iters = 21 and time = 20.42); converges with $\mu^0 = 1$ if we set $\sigma^0 = 0.4$ (iters = 18 and time = 17.06); converges with $\mu^0 = 0.1$ if we set $\sigma^0 = 0.4$ (iters = 15 and time = 14.02); and converges with $\mu^0 = 0.001$ if we set both $\alpha_0 = 0.9$ and $\sigma_0 = 0.8$ (iters = 24 and time = 22.70). The same has been observed for the non-converged tests with the P-IPD code.

The iterative process for Problem 11 (BRAS-2098), which has failed to converge with $\mu^0 = 0.01$, converges if we set $\alpha_0 = 0.9$ and $\sigma^0 = 0.6$ (iters = 35 and time = 78.92). Convergence with $\mu^0 = 0.001$ could not be restored by means of such a simple modification of parameters. The same has been observed for the non-converged tests with the P-IPD code.

In Table 7.18, the iterative process for Problem 9, which has failed to converge with $\mu^0 = 0.001$, involves very small affine steps ($\alpha_{\text{aff}}^k \ll 1$) and the convergence is very slow.

In summary, the presented numerical results recommend choosing $\mu^0 \in [10^{-2}, 1.0]$ with the best choice for this test set being $\mu^0 = 10^{-2}$. The convergence range for the barrier parameter is relatively large, as long as the centering parameter σ is properly chosen. The major limitation in the choice of μ^0 is that it should not be chosen too small. From the above analysis for the non-converged cases, we can conclude that the plain primal-dual IP method is also capable of solving all the eleven problems, as long as we properly set the

Table 7.18: Influence of μ^0 in the predictor-corrector IP method: R-PCM code.

Problem	$\mu^0 = 10$		$\mu^0 = 1$		$\mu^0 = 0.1$		$\mu^0 = 0.01$		$\mu^0 = 0.001$	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	10	1.23	13	1.58	12	1.46	8	0.97	7	0.87
2	9	1.19	8	1.05	8	1.05	6	0.79	6	0.80
3	9	1.23	9	1.16	8	1.07	7	0.93	6	0.80
4	10	1.55	10	1.57	9	1.40	8	1.25	10	1.51
5	11	2.35	11	2.30	10	2.09	9	1.83	8	1.66
6	12	4.81	11	4.26	10	3.80	9	3.34	10	3.81
7	12	3.86	11	3.48	10	3.16	9	2.80	10	3.16
8	13	8.82	13	8.70	11	7.26	10	6.53	10	6.65
9	12	5.34	11	4.82	12	5.05	14	5.76	21	8.82
10	12	12.27	11	11.06	10	9.77	9	8.54	10	9.49
11	26	66.89	25	61.77	23	55.22	19	45.13	20	48.13
Total	136	109.54	133	101.75	123	91.33	108	77.87	118	85.70

Table 7.19: Influence of μ^0 in the predictor-corrector IP method: P-PCM code.

Problem	$\mu^0 = 10$		$\mu^0 = 1$		$\mu^0 = 0.1$		$\mu^0 = 0.01$		$\mu^0 = 0.001$	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	7	0.86	13	1.61	12	1.42	8	0.96	7	0.86
2	10	1.30	8	1.03	8	1.05	7	0.90	6	0.78
3	9	1.22	9	1.17	8	1.04	7	0.95	6	0.81
4	10	1.55	10	1.56	9	1.38	8	1.22	10	1.49
5	11	2.33	11	2.33	10	2.02	9	1.85	8	1.62
6	12	4.67	11	4.20	10	3.70	9	3.33	10	3.62
7	12	3.85	11	3.48	11	3.43	10	3.16	10	3.07
8	14	9.37	13	8.55	12	7.76	11	7.01	11	6.95
9	12	5.40	11	4.71	12	4.91	14	5.74	21	8.51
10	12	11.98	11	10.60	10	9.46	9	8.35	11	10.40
11	24	62.13	24	59.09	22	51.21	22	50.11	22	49.50
Total	133	104.66	132	98.33	124	87.38	114	83.58	122	87.61

safety factor α_0 and the centering parameter σ^0.

In Tables 7.20 and 7.21, we display the results of experiments in which we have updated μ^k using the standard procedure (see Section 3.5.1) and have chosen $\mu^0 = 0.01$. We may recall that in Tables 7.4 and 7.5 we have used $\mu^0 = 1$. We may infer from Tables 7.20 and 7.21 that, on the whole, the IP codes have shown their best performance so far. Surprisingly,

Table 7.20: Influence of updating formula of μ: Standard Procedure, Rectangular Form.

Problem	R-IPD		R-PCM		R-PCN		R-MPC		R-MCC	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	11	1.32	8	0.97	7	0.83	8	0.95	8	0.98
2	12	1.57	6	0.79	7	0.93	6	0.80	7	0.89
3	10	1.30	7	0.93	7	0.97	7	0.94	8	1.06
4	13	1.97	8	1.25	8	1.21	8	1.21	10	1.51
5	15	3.00	9	1.83	9	1.84	10	2.02	12	2.47
6	14	5.13	9	3.34	9	3.40	9	3.40	12	4.59
7	16	4.89	9	2.80	10	3.21	10	3.16	12	3.77
8	16	10.05	10	6.53	10	6.59	11	7.10	14	9.10
9	α^7		14	5.76	14	5.87	18	7.40	22	9.13
10	15	13.95	9	8.54	9	8.55	10	9.51	14	13.18
11	α^6		19	45.13	14	33.07	19	44.76	22	51.66
Total	—		108	77.87	104	66.47	116	81.25	141	98.34

Table 7.21: Influence of updating formula of μ: Standard Procedure, Polar Form.

Problem	P-IPD		P-PCM		P-PCN		P-MPC		P-MCC	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	11	1.34	8	0.96	7	0.84	8	0.97	8	0.96
2	11	1.47	7	0.90	7	0.90	7	0.93	8	1.04
3	10	1.31	7	0.95	7	0.93	7	0.90	8	1.03
4	13	1.95	8	1.22	8	1.20	8	1.23	10	1.51
5	15	3.03	9	1.85	9	1.86	10	2.04	12	2.44
6	14	5.09	9	3.33	9	3.31	9	3.32	12	4.44
7	16	4.88	10	3.16	10	3.19	10	3.13	12	3.77
8	16	9.89	11	7.01	10	6.40	11	7.01	15	9.61
9	α^7		14	5.74	14	5.92	17	6.86	23	9.30
10	α^6		9	8.35	10	9.50	10	9.44	14	13.01
11	23	50.07	22	50.11	20	46.08	17	38.29	22	49.81
Total	—		114	83.58	111	80.13	114	74.12	144	96.92

the R-PCN code have solved the largest problem in 14 iterations and 33.07 seconds. We also can observe a significant improvement in the performance of the R-MCC and P-MCC codes.

The results of experiments performed with the Vanderbei–Shanno's procedure to update μ^k are displayed in Tables 7.22 and 7.23. Once more, the higher-order IP methods have performed very well. We remark, however, that the computational performance with

the Standard Procedure to update μ^k is slightly superior to the performance with the Vanderbei–Shanno's procedure.

Table 7.22: Influence of updating formula of μ: Vanderbei–Shanno, Rectangular Form.

Problem	R-IPD		R-PCM		R-PCN		R-MPC		R-MCC	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	16	1.93	8	0.97	7	0.83	8	0.95	9	1.10
2	17	2.23	7	0.91	8	1.05	7	0.90	9	1.16
3	13	1.69	8	1.06	7	0.91	8	1.08	9	1.22
4	18	2.70	9	1.38	9	1.40	9	1.33	11	1.68
5	19	3.86	9	1.81	9	1.84	10	2.02	13	2.69
6	20	7.26	10	3.77	10	3.78	10	3.75	13	4.91
7	20	6.19	10	3.14	11	3.50	10	3.16	13	4.13
8	α^{28}		11	7.06	11	7.12	11	7.04	15	9.72
9	α^{12}		14	5.75	14	5.80	18	7.51	25	10.29
10	α^{12}		10	9.44	10	9.60	10	9.41	15	14.14
11	α^{11}		21	50.07	19	45.87	18	42.49	24	56.39
Total	—		117	85.36	115	81.70	119	79.64	143	107.43

Table 7.23: Influence of updating formula of μ: Vanderbei–Shanno, Polar Form.

Problem	P-IPD		P-PCM		P-PCN		P-MPC		P-MCC	
	iters	time	iters	time	iters	time	iters	time	iters	time
1	15	1.78	8	0.97	7	0.83	8	0.97	9	1.15
2	17	2.17	8	1.01	8	1.03	8	1.06	9	1.14
3	13	1.69	8	1.06	7	0.93	8	1.10	9	1.20
4	18	2.65	9	1.34	9	1.40	9	1.36	11	1.66
5	19	3.80	9	1.83	9	1.89	10	2.04	13	2.65
6	17	5.97	10	3.64	10	3.74	10	3.70	13	4.74
7	21	6.43	10	3.14	11	3.50	10	3.11	13	4.07
8	24	14.69	12	7.75	11	7.12	12	7.69	16	10.26
9	α^{10}		14	5.65	15	6.16	21	8.56	25	10.15
10	33	20.40	10	9.28	10	9.55	11	10.28	15	13.75
11	α^{9}		21	47.26	19	43.73	19	43.14	26	58.02
Total	—		119	82.93	116	79.88	126	83.01	159	108.79

7.3.5 Influence of the Maximum Number of Corrector Steps

In Tables 7.24 and 7.25, we report on numerical results of experiments in which we allow the MCC algorithm to perform more than one centrality correction step per iteration. We have considered the cases where the MCC algorithm performs exactly 2 centrality corrections (2-Fixed), 3 centrality corrections (3-Fixed), 4 centrality corrections (4-Fixed), 5 centrality corrections (5-Fixed), and a variable number of centrality corrections within each IP iteration, limited by a maximum of 5 centrality corrections ($M = 5$).

The MCC technique has shown its best performance as it performs exactly 5 centrality corrections (in rectangular coordinates) and 4 centrality corrections (in polar coordinates) per iteration. However, we believe that if an efficient heuristic to dynamically define the number of corrections per iteration is implemented the MCC technique with variable number of centrality corrections per iteration may outperform any of the IP algorithms.

Table 7.24: Performance of R-MCC code using Scheme–B and $\mu^0 = 0.01$.

Problem	2-Fixed		3-Fixed		4-Fixed		5-Fixed		$M = 5$	
	iters	time	iters	time	iters	time	iters	time	iters	time
5	10	2.09	9	1.90	9	1.96	9	2.00	9	2.00
6	10	3.88	9	3.68	9	3.78	9	3.87	9	3.87
7	11	3.57	10	3.36	10	3.43	10	3.48	10	3.53
8	11	7.47	10	6.97	9	6.50	10	7.52	10	7.67
9	12	5.35	10	4.54	9	4.32	9	4.55	10	4.76
10	11	10.88	10	10.29	10	10.88	10	11.14	10	11.32
11	19	46.34	17	43.30	17	44.97	14	37.83	14	36.46
Total	84	79.58	75	74.04	73	75.84	71	70.39	72	69.61

Table 7.25: Performance of P-MCC code using Scheme–B and $\mu^0 = 0.01$.

Problem	2-Fixed		3-Fixed		4-Fixed		5-Fixed		$M = 5$	
	iters	time	iters	time	iters	time	iters	time	iters	time
5	10	2.05	9	1.94	9	1.97	9	2.00	9	2.01
6	10	3.90	9	3.55	9	3.64	9	3.79	9	3.84
7	11	3.64	10	3.33	10	3.44	10	3.55	10	3.50
8	12	8.12	11	7.53	10	7.20	10	7.21	10	7.69
9	13	5.54	10	4.40	10	4.55	9	4.24	10	4.65
10	11	10.76	12	12.08	9	9.41	10	10.92	10	10.78
11	19	44.24	17	41.40	15	37.75	15	38.66	15	37.24
Total	86	78.25	78	74.23	72	67.96	72	70.37	73	69.71

In Tables 7.26 and 7.27, we report on numerical results of experiments in which we allow the MPC algorithm to perform more than one corrector step per iteration. We have considered the cases where the MPC algorithm performs a maximum of 2 corrector steps ($M = 2$), a maximum of 3 corrector steps ($M = 3$), a maximum of 4 corrector steps ($M = 4$), a maximum of 5 corrector steps ($M = 5$), or a fixed number of exactly 3 corrector steps within each IP iteration. We can observe that the P-MPC code has performed considerably better than the R-MPC code.

The PCN method have shown some convergence instability when it performs more then one corrector step per iteration in the solution of the larger problems. Such an instability is related to the step lengths being adversely set to too small values for more than one corrector steps, despite the outstanding performance shown by this algorithm when we consider one corrector step.

Table 7.26: Performance of R-MPC code using Scheme–B and $\mu^0 = 0.01$.

Problem	$M = 2$		$M = 3$		$M = 4$		$M = 5$		3-Fixed	
	iters	time	iters	time	iters	time	iters	time	iters	time
5	9	1.86	9	1.90	8	1.70	8	1.76	7	1.54
6	8	3.08	9	3.55	9	3.58	9	3.58	8	3.26
7	9	2.91	9	2.91	9	2.97	9	3.03	7	2.41
8	10	6.64	10	6.70	9	6.21	9	6.15	8	5.68
9	10	4.40	9	4.06	9	4.26	8	3.72	8	3.68
10	10	9.83	10	9.92	10	9.95	10	9.97	7	7.44
11	19	45.38	19	45.38	19	45.38	19	45.38	α^2	
Total	75	74.10	75	74.42	73	74.05	72	73.59	—	

Table 7.27: Performance of P-MPC code using Scheme–B and $\mu^0 = 0.01$.

Problem	$M = 2$		$M = 3$		$M = 4$		$M = 5$		3-Fixed	
	iters	time	iters	time	iters	time	iters	time	iters	time
5	9	1.89	8	1.70	8	1.70	8	1.73	7	1.50
6	9	3.43	8	3.11	7	2.76	7	2.75	8	3.16
7	10	3.23	9	2.93	9	2.97	9	3.01	9	2.96
8	10	6.53	10	6.76	9	6.10	9	6.24	9	6.23
9	10	4.34	10	4.29	9	4.04	8	3.67	8	3.53
10	10	9.74	10	9.74	10	9.74	10	9.74	7	7.28
11	16	36.79	16	36.88	16	36.88	16	36.88	α^2	
Total	74	65.95	71	65.41	68	64.19	67	64.02	—	

7.4 Experiments with the Non-Interior-Point Algorithm

In this section, we discuss the numerical results of experiments performed with the proposed NIP algorithm, as related to nonlinear OPF solution both in rectangular and in polar coordinates. Initially, in Tables 7.28, 7.29 and 7.30, we present some preliminary results that we have been obtained with a simplified implementation of the NIP algorithm that we coded in MATLAB®. In this MATLAB® prototype implementation, we have used a fixed step length of $\alpha^k = 0.95$, and have chosen $\mu^0 = 0.01$, with the simple updating rule $\mu^{k+1} = \max\{0.1\mu^k, 10^{-10}\}$. We have chosen the starting points as follows:

$$
\begin{aligned}
s_1^0 &= h(x^0) - \underline{h}, \\
s_2^0 &= \overline{h} - h(x^0), \\
s_3^0 &= \widehat{x}^0 - \underline{x}, \\
s_4^0 &= \overline{x} - \widehat{x}^0, \\
z_1^0 &= u, \\
z_2^0 &= Z, \\
z_3^0 &= u, \\
z_4^0 &= Z.
\end{aligned}
$$

The convergence processes that are displayed in Tables 7.28 through 7.30 show that the NIP algorithm have performed extremely well using the initialization heuristic described above.

Table 7.28: The convergence process for Problem 3 solved by R-NIP code.

k	ν_1^k	ν_2^k	ν_3^k	ν_4^k	$\Phi_\mu(w^k)$
0	1.462×10^{-1}	8.115×10^{-2}	8.041×10^{-1}	$1.340 \times 10^{+0}$	$1.488 \times 10^{+0}$
1	2.638×10^{-3}	8.839×10^{-2}	$5.798 \times 10^{+0}$	$1.407 \times 10^{+1}$	$5.377 \times 10^{+3}$
2	1.638×10^{-3}	6.476×10^{-2}	3.497×10^{-1}	$1.003 \times 10^{+0}$	$1.530 \times 10^{+1}$
3	5.467×10^{-4}	2.742×10^{-3}	6.449×10^{-2}	1.765×10^{-1}	2.404×10^{-1}
4	5.962×10^{-2}	8.844×10^{-4}	5.852×10^{-2}	3.340×10^{-2}	3.634×10^{-2}
5	4.719×10^{-2}	2.441×10^{-4}	4.528×10^{-2}	4.556×10^{-3}	1.381×10^{-2}
6	1.287×10^{-2}	5.811×10^{-5}	6.219×10^{-2}	1.483×10^{-3}	1.663×10^{-2}
7	9.833×10^{-2}	1.261×10^{-4}	9.827×10^{-2}	4.397×10^{-4}	5.950×10^{-2}
8	5.912×10^{-3}	4.753×10^{-5}	1.001×10^{-2}	2.339×10^{-4}	2.893×10^{-4}
9	3.138×10^{-4}	3.182×10^{-6}	2.281×10^{-3}	2.670×10^{-5}	7.313×10^{-6}
10	1.671×10^{-5}	2.292×10^{-7}	5.879×10^{-4}	3.121×10^{-6}	4.856×10^{-7}

Table 7.29: The convergence process for Problem 4 solved by R-NIP code.

k	ν_1^k	ν_2^k	ν_3^k	ν_4^k	$\Phi_\mu(w^k)$
0	7.064×10^{-1}	1.246×10^{-1}	$1.000 \times 10^{+0}$	$1.249 \times 10^{+0}$	$4.819 \times 10^{+0}$
1	3.235×10^{-1}	2.326×10^{-1}	$6.064 \times 10^{+0}$	$1.622 \times 10^{+1}$	$9.049 \times 10^{+3}$
2	1.620×10^{-2}	1.624×10^{-1}	4.222×10^{-1}	$1.171 \times 10^{+0}$	$2.641 \times 10^{+1}$
3	1.267×10^{-2}	7.913×10^{-3}	9.031×10^{-2}	2.057×10^{-1}	4.558×10^{-1}
4	2.109×10^{-2}	1.743×10^{-3}	2.096×10^{-2}	3.140×10^{-2}	1.135×10^{-2}
5	9.684×10^{-2}	4.381×10^{-4}	9.294×10^{-2}	3.118×10^{-3}	1.057×10^{-1}
6	4.350×10^{-2}	1.419×10^{-4}	7.499×10^{-2}	2.994×10^{-3}	3.256×10^{-2}
7	1.502×10^{-2}	1.206×10^{-4}	7.424×10^{-2}	1.848×10^{-3}	2.419×10^{-2}
8	3.845×10^{-3}	6.489×10^{-5}	1.962×10^{-2}	5.591×10^{-4}	8.300×10^{-4}
9	4.852×10^{-4}	1.009×10^{-5}	3.985×10^{-3}	8.975×10^{-5}	2.700×10^{-5}
10	3.289×10^{-5}	7.957×10^{-7}	6.663×10^{-4}	7.484×10^{-6}	8.774×10^{-7}

Table 7.30: The convergence process for Problem 5 solved by R-NIP code.

k	ν_1^k	ν_2^k	ν_3^k	ν_4^k	$\Phi_\mu(w^k)$
0	4.494×10^{-1}	2.218×10^{-1}	$1.000 \times 10^{+0}$	$1.084 \times 10^{+0}$	$1.499 \times 10^{+1}$
1	2.840×10^{-2}	4.304×10^{-2}	$5.808 \times 10^{+0}$	$3.108 \times 10^{+1}$	$1.726 \times 10^{+4}$
2	3.672×10^{-2}	8.817×10^{-2}	3.793×10^{-1}	$2.534 \times 10^{+0}$	$5.196 \times 10^{+1}$
3	8.029×10^{-2}	1.766×10^{-2}	8.928×10^{-2}	4.672×10^{-1}	9.464×10^{-1}
4	1.034×10^{-1}	8.768×10^{-3}	9.949×10^{-2}	8.750×10^{-2}	2.268×10^{-1}
5	2.821×10^{-1}	7.703×10^{-4}	2.875×10^{-1}	1.371×10^{-2}	9.424×10^{-1}
6	1.732×10^{-1}	1.699×10^{-4}	1.737×10^{-1}	4.283×10^{-3}	3.685×10^{-1}
7	1.298×10^{-2}	1.145×10^{-4}	2.288×10^{-2}	7.737×10^{-4}	3.787×10^{-3}
8	7.688×10^{-4}	1.117×10^{-5}	4.717×10^{-3}	1.505×10^{-4}	1.102×10^{-4}
9	3.563×10^{-2}	5.085×10^{-6}	3.567×10^{-2}	2.054×10^{-5}	5.085×10^{-3}
10	1.825×10^{-3}	1.459×10^{-6}	1.832×10^{-3}	4.307×10^{-6}	1.292×10^{-5}
11	1.370×10^{-4}	9.536×10^{-7}	1.373×10^{-4}	8.556×10^{-7}	4.400×10^{-8}
12	2.808×10^{-5}	5.131×10^{-8}	2.809×10^{-5}	3.993×10^{-8}	1.253×10^{-9}

In Table 7.31, we display the numerical results of experiments performed with a more elaborated implementation in Fortran 77 of the NIP algorithm. For each test run, we give the number of iterations (iters), the total number of merit function evaluations (Φ-eval), the total number of full/damped Newton steps taken (full/damp), and the CPU times (time). In these simulations, for performance evaluation and comparison purposes, the NIP algorithm use the initialization Heuristic–A of the IP algorithm, the non-monotone line search test to obtain the step length, and the proposed scheme to update the continuation parameter.

We remark that while the merit function used for the NIP method—the natural merit function $\Phi = \frac{1}{2}\Psi_\mu(w)^T\Psi_\mu(w)$—only guarantees convergence to a stationary point of the Lagrangian function, not necessarily a local minimizer of the NLP problem, in all test runs the documented minimizer was obtained. Moreover, the R-NIP code have outperformed all IP codes in the solution of the largest problem.

Table 7.31: Non-interior-point method using initialization Heuristic–A.

Problem	R-NIP				P-NIP			
	iters	Φ-eval	full/damp	time	iters	Φ-eval	full/damp	time
1	9	9	9/0	1.09	8	8	8/0	0.97
2	7	7	7/0	0.92	8	8	8/0	1.04
3	9	9	9/0	1.18	8	8	8/0	1.04
4	9	9	9/0	1.34	10	10	10/0	1.50
5	10	10	10/0	2.01	12	12	12/0	2.41
6	11	11	11/0	4.06	12	12	12/0	4.36
7	13	13	13/0	4.05	17	31	16/1	5.27
8	12	12	12/0	7.80	17	17	17/0	10.65
9	16	20	14/2	6.68	18	32	17/1	7.27
10	13	13	13/0	12.64	16	33	14/2	15.18
11	13	13	13/0	31.75	20	50	18/2	48.40
Total	122	126	120/2	73.52	146	270	140/6	98.09

7.5 Polar vs. Rectangular: The Voltage Bound Issue

In Table 7.32, we display the number of flops required to compute the Lagrangian Hessian $\nabla^2_{xx}L_\mu(w)$, the number of non-zeros in $\nabla^2_{xx}L_\mu(w)$, and the number of non-zeros in the matrix $\nabla_x d(w)^T$ of the reduced system, when using rectangular and polar coordinates. Even though the computation of $\nabla^2_{xx}L_\mu(w)$ in polar coordinates profits from the computation of the Jacobians $\nabla_x g(x)^T$ and $\nabla_x h(x)^T$, this computation still requires nearly double the flops the computation in rectangular coordinates requires. Therefore, the assembling of matrices is more efficiently done in rectangular coordinates.

A pitfall of the OPF formulation in rectangular coordinates is the necessity to handle the voltage bounds as nonlinear functional bounds; in polar coordinates, voltage bounds are simple variable bounds. The computational implications can be easily examined in the reduced system (3.24), as we look at the extent the matrix $\nabla_x d(x)^T$ differs from the Hessian $\nabla^2_{xx}L_\mu(w)$. The expression for these matrices are repeated below for convenience

Table 7.32: Number of non-zeros and flops.

Problem	flops to obtain $\nabla^2_{xx}L_\mu$		Non-zeros in $\nabla^2_{xx}L_\mu$		Non-zeros in $\nabla_x d^T$	
	Rectangular	Polar	Rectangular	Polar	Rectangular	Polar
IEEE-30	944	1823	415	613	467	661
IEEE-57	2525	3973	856	1278	968	1392
IEEE-118	3314	7590	1718	4572	1952	5592
IEEE-300	10410	19154	4315	6301	4913	7455

of reference:

$$\nabla^2_{xx}L_\mu(w) = \nabla^2_{xx}f(x) - \sum_{j=1}^{m} y_j \nabla^2_{xx}g_j(x) + \sum_{j=1}^{p} z_{2_j} \nabla^2_{xx}h_j(x), \tag{7.1}$$

and

$$\nabla_x d(w)^T = \nabla^2_{xx}L_\mu(w) + \mu^k \nabla_x h(x)\big(S_1^{-2} + S_2^{-2}\big)\nabla_x h(x)^T + \mu^k \widehat{I}^T\big(S_3^{-2} + S_4^{-2}\big)\widehat{I}. \tag{7.2}$$

Notice that the voltage bounds in polar coordinates, being part of $\widehat{x}$, simply affect the diagonal of $\nabla^2_{xx}L_\mu(w)$ through the third term in the right-hand side of (7.2), whereas the voltage bounds in rectangular coordinates, being part of $\nabla_x h(x)^T$, contribute with new non-zeros to $\nabla_x d(x)^T$ whenever two connected buses have neighbor buses in common. Nevertheless, some of the fill ins that are caused by the voltage functional bounds co-occur with fill ins caused by the reactive power constraints. Hence, on the whole, functional voltage bounds have little effect on the factorization cost, as evidenced in Table 7.32.

The handling of branch flow constraints in rectangular and in polar coordinates are alike since the related Hessian matrices have exactly the same nonzero structure.

7.6 Final Remarks

In this chapter, we have discussed the results of extensive experiments performed with the IP and NIP algorithms that are proposed in Chapters 3 through 5. To evaluate the effectiveness of the proposed algorithms, we have performed over one thousand experiments based on a test set of eleven problems, including actual power networks of up to 2098 buses. In these experiments, the algorithms have not been tuned to particular problems; the same parameter setting is used for all problems. Initially, we tested the performance of the IP

algorithms as they use the default parameters and formulae. The higher-order IP algorithms were able to solve the whole set of problems, whereas the plain primal-dual IP algorithm failed to solve the larger problems. However, by means of simple adjustment of parameters this algorithm was able to solve the whole set of problems too.

With respect to initialization of the algorithms, we have observed that the higher-order IP algorithms are less sensitive to the choice of the initial point than the plain primal-dual IP algorithm. Nevertheless, a slightly better performance is achieved with the initialization Heuristic–A, as the points obtained by this heuristic are, in general, better centered than the points obtained by Heuristic–B.

With respect to the step length rules, we have observed that the higher-order IP algorithms perform well with the Scheme–A (separate step lengths in the primal and dual spaces) and the Scheme–B (single common step length), and poorly with the Scheme–C (box constraint on the dual step). The plain primal-dual IP algorithm has performed better with the Scheme–A. With respect to the choice of the safety factor α_0, the best performance has been obtained with safety factors in the interval $0.98 \leq \alpha_0 \leq 0.9995$. However, a safety factor of $\alpha_0 = 0.9$, or even lower, may be used as a means of restoring convergence for non-converged runs with the plain primal-dual IP method.

With respect to the choice of μ^0, we have observed that the convergence range for the barrier parameter is relatively large, as long as the centering parameter σ^0 is properly chosen. The major limitation in the choice of μ^0 is that it should not be chosen too small; the numerical results recommend choosing $\mu^0 \in [10^{-2}, 1]$, with the best choice for the test set considered being $\mu^0 = 10^{-2}$. We believe that the proper choice of μ^0 is directly related to the initialization heuristic used. With respect to the updating formula of μ^k, the IP algorithms have performed well with both formulae.

With respect to the number of corrector steps in the higher-order IP algorithms, we have observed that the MPC and MCC techniques may outperform the predictor-corrector IP method if they are allowed to perform more than one corrector step per iteration. The MCC technique, in particular, has great potential and is our favorite.

The computational experiments with the NIP algorithm, though not quite extensive as the experiments with the IP algorithms, have shown that this new approach is very promising. We can observe that the best performance overall in the solution of the largest problem was achieved by the `R-NIP` code.

Chapter 8

Conclusions

8.1 Summary and Contributions

In the daily operation of a power system, deciding on an optimal control action, aiming at the economic and reliable operation of a system, is an extremely difficult task. However, such a task has been successfully performed by OPF procedures at power system control centers. The OPF problem is inevitably a very large non-convex NLP problem. Although local approximation-based optimization techniques such as SLP and SQP have been widely used to solve OPF problems, recently there has been an increasing need to speed up solutions which can be accomplished by solving the OPF problems in a nonlinear manner. Due to the size and special feature of these problems, IP methods have computationally proven to be a viable alternative for their solution. This thesis research has concentrated on the solution of large-scale OPF problems, in a nonlinear manner, by IP and NIP methods.

In Chapter 2, three variants of the broad class of OPF problems are described, namely, (i) the reactive power dispatch problem, (ii) the maximum loadability problem, and (iii) the minimum load shedding problem. The OPF problem (i) has been formulated, in this thesis, using voltages either in rectangular or in polar coordinates. Advantages of using bus voltages in rectangular coordinates, as explored in the thesis, are ease of matrix setup and incorporation of second-order information in higher-order IP methods; a minor difficulty related to the rectangular coordinates, as compared with the polar coordinates, is the need to handle simple voltage bounds as (simple) functional bounds. Mainly through the OPF formulations (ii) and (iii), we emphasize the increasing need to solve OPF problems in a nonlinear manner.

In Chapter 3, the mathematical development of a primal-dual IP algorithm for NLP is described in details. This IP algorithm development is a direct extension of the IP method for LP that is described in [52]. In this thesis research, we have conducted the following studies:

- We have developed in detail our infeasible primal-dual IP algorithm for NLP based on the NLP problem (1.1), which is a suitable form to formulate most OPF problems.
- We have studied the performance of the primal-dual IP algorithm as it employs several initialization heuristics, schemes to compute the step lengths, and updating formulae of the barrier parameter. Also, we have studied the influence of various parameters of the algorithm in the convergence process, as concerned with nonlinear OPF solution.
- We have thoroughly studied the performance of the primal-dual IP algorithm as it solves the RPD problem formulated in both rectangular and polar coordinates. Such an analysis—rectangular coordinates versus polar coordinates—has not been performed in previous works.
- We have described an alternative approach to compute the Newton direction, called the *reduced system* approach. From such a reduced system, the implications of handling the voltage bounds as functional bounds (in rectangular coordinates) instead of simple bounds (in polar coordinates) can be easily examined as we look at the extent the matrix $\nabla_x d(w)^T$ differs from the matrix $\nabla^2_{xx} L_\mu(w)$. Such an analysis has been presented in Chapter 7.

In Chapter 4, we present extensions to NLP of four successful higher-order IP methods for LP and QP, namely, (i) the *predictor-corrector* method, (ii) the *perturbed composite Newton* method, (iii) the *multiple predictor-corrector* method, and (iv) the *multiple centrality corrections* method. The central idea behind all these techniques is to reduce the number of derivative evaluations and matrix factorizations to a necessary minimum, even at the expense of some increase in the cost of a single iteration. The approach (i) was first extended to nonlinear OPF solution in [77]. Concerning the higher-order IP methods, the contributions made in this thesis are as follows:

- We have extended to nonlinear OPF solution the perturbed composite Newton IP method for LP and QP, as developed by Tapia et al. [67].
- We have extended to nonlinear OPF solution the multiple predictor-corrector IP method for LP and QP, as developed by Carpenter et al. [10].

- Also, we have extended to nonlinear OPF solution the multiple centrality corrections technique for LP developed by Gondzio [31].

None of the techniques (ii), (iii) and (iv) was previously considered for nonlinear OPF solution.

In Chapter 5, a new algorithm to solve nonlinear OPF problems is proposed. It is a NIP algorithm that handles the complementarity conditions by a recently introduced NCP-function. As far as we are aware, the proposed OPF algorithm is the first one based on NCP-functions. Distinctive features of this approach, as compared with IP methods, are that it can start from arbitrary points, and the iterates are not required to stay inside the positive orthant of the complementarity product space. That is, the non-negativity conditions need be satisfied only at the solution point.

In Chapter 6, we discuss many issues that are directly related to the efficient implementation of the IP and NIP algorithms, as concerned with nonlinear OPF solution. The contributions made in this chapter are as follows:

- We have presented four initialization heuristics. Two of the heuristics conform with the whole set of IP algorithms whereas all four conform with the NIP algorithm. The numerical performance of each initialization heuristic is discussed in Chapter 7.
- We have derived explicit formulae to efficiently assemble the Hessian matrices, both in rectangular and in polar coordinates. We have shown that this task is more efficiently done in rectangular coordinates, where the function Hessians are constant.
- We have proposed a mapping from constraint multipliers to bus multipliers: $(y, z_2) \mapsto (\lambda^p, \lambda^q, \lambda^v)$. Such a mapping considerably reduces the number of logical operations in the evaluation of the Lagrangian gradient and Hessian, and allows for efficient data structure and savings in computer memory usage.
- We have derived explicit formulae to estimate the number of flops required to obtain the Lagrangian Hessian—the major effort in forming the Newton system—both in rectangular and in polar coordinates.
- We have presented some code fragments to emphasize the advantages of the proposed mapping of Lagrange multipliers. Also, we have discussed a block-data structure and the solution of symmetric indefinite systems.

In Chapter 7, for performance evaluation and comparison purposes, we discuss extensive computational experiments with the IP and NIP algorithms. Results of over one thousand tests are presented. The conclusions derived from this chapter are as follows:

- Initially, we tested the performance of the IP algorithms for the default parameters and formulae. The higher-order IP algorithms were able to solve all the problems, whereas the plain primal-dual IP algorithm failed to solve the larger ones. By means of parameter adjustments, this algorithm solved all the problems too.

- With respect to initialization of the algorithms, we have observed that the higher-order IP algorithms are less sensitive to the choice of the initial point than the plain primal-dual IP algorithm. However, better performance is achieved with the initialization Heuristic–A, as the points obtained by this heuristic are, in general, better centered than the points obtained by Heuristic–B.

- With respect to the step length rules, we have observed that the higher-order IP algorithms perform satisfactorily with the Scheme–A (separate step lengths in the primal and dual spaces) and the Scheme–B (single common step length), and poorly with the Scheme–C (box constraint on the dual step).

- With respect to the choice of the safety factor α_0, the best performance has been obtained with safety factors in the interval $0.98 \leq \alpha_0 \leq 0.9995$. However, a safety factor of $\alpha_0 = 0.9$, or even lower, may be used as a means of restoring convergence for non-converged runs with the plain primal-dual IP method.

- With respect to the choice of μ^0, we have observed that the convergence range for the barrier parameter is relatively large, as long as the centering parameter σ^0 is properly chosen. The major limitation in the choice of μ^0 is that it should not be chosen too small; the numerical results recommend choosing $\mu^0 \in [10^{-2}, 1]$, with the best choice for the test set considered being $\mu^0 = 10^{-2}$.

- With respect to the updating formula of μ^k, the IP algorithms have performed satisfactorily with both formulae.

- With respect to the number of corrector step in the higher-order IP algorithms, we have observed that the MPC and MCC techniques may outperform the predictor-corrector IP method if they are allowed to perform more than one corrector step per iteration.

- The computational experiments with the NIP algorithm, though not quite extensive as the experiments with the IP algorithms, have shown that this new approach is very promising. We have observed that the best performance overall in the solution of the largest problem has been achieved by the R-NIP code.
- Related to the issue Rectangular Coordinates versus Polar Coordinates, the computational experiments have revealed that their performance are alike, as far as concerned with the number of iterations and CPU times.
- Interestingly enough, despite the non-convexity of the nonlinear OPF problem and the reasonable large number of experiments performed, using different initialization, parameters, formulae, and so forth, the IP and NIP algorithms have obtained the same local optimum for the problems.

On the whole, the results discussed in Chapter 7 have illustrated the viability of the proposed IP and NIP algorithms to solve large scale OPF problems in a nonlinear manner.

8.2 Directions for Future Research

Concerning future work with the infeasible primal-dual IP algorithm that is described in Chapter 3, a study of the usefulness of inexact search directions sounds interesting. The idea is to reduce the overall computational time by reducing the effort of a single iteration, even at the expense of some increase in the iteration count. By an inexact search direction we mean that the vector Δw satisfies

$$\nabla^2_{ww} L_\mu(w^k)\Delta w = -\nabla_w L_\mu(w^k) + r^k. \tag{8.1}$$

for some suitable residual vector r^k. An iterative solver, such as a preconditioned conjugate gradient method, can be used to solve the linear system (3.13); this method is stopped when the norm of the residual is smaller than a prefixed accuracy, that is, $\|r^k\|_2 \leq \epsilon^k$.

Concerning future work with the higher-order IP methods described in Chapter 4, an issue deserving further research, in our most immediate vision, is how to dynamically choose the appropriate number of corrector steps within each iteration of the higher-order IP methods. Also, it sounds interesting to study a combination of the various methods in a single algorithm.

Concerning future work with the NIP approach described in Chapter 5, we consider as potential directions for research the following topics:

- Since the NIP algorithm can start from arbitrary points and all matrices in the IP and NIP algorithms have the same nonzero pattern, an improved OPF algorithm most likely can be developed if we combine together the two algorithms. Notice that switching from one algorithm to the other demands no changes of the linear algebra kernel, the core of both techniques.

- The robustness of the NIP algorithm can be improved if we consider a *Levenberg-Marquardt-type method* for the solution of (5.25). Instead of solving (5.28) for the direction Δw, this method computes a search direction d^k as the solution of the modified linear system

$$\left[\nabla_w \Psi_\mu(w^k)^T + \sigma^k I\right] d^k = -\Psi_\mu(w^k) \tag{8.2}$$

where $\sigma^k \geq 0$ here is the Levenberg-Marquardt parameter. If the direction generated by (5.45) is not a "good" descent direction, according to the test

$$\nabla_w \Phi_\mu(w^k)^T d^k \leq -\rho \|d^k\|^p, \quad \rho > 0, \quad p > 2, \tag{8.3}$$

we resort to the *steepest descent direction*, that is, let $d^k = -\nabla_w \Phi_\mu(w^k)$.

- As suggested for the primal-dual IP algorithm, an issue worth of investigation is the usefulness of inexact search directions. By this we mean that the vector Δw satisfies

$$\nabla_w \Psi_\mu(w^k)^T \Delta w = -\Psi_\mu(w^k) + r^k. \tag{8.4}$$

where r^k here is the vector of residuals and measures how inexactly system (5.28) is solved. An iterative solver is used to solve the linear system (5.28); this method is stopped when the norm of the residual is smaller than a prefixed accuracy, that is, $\|r^k\|_2 \leq \epsilon^k$. Facchinei and Kanzow [20] have proposed an *inexact Levenberg-Marquardt-type algorithm* to solve large NCP problems that uses a test like $\|r^k\| \leq (0.1/(k+1))\|\Phi_\mu(w^k)\|$..

- Since the computation and factorization of matrix $\nabla_w \Psi_\mu(w^k)^T$ demand the greatest computational effort within an iteration of the NIP algorithm, it may be advantageous to use the same derivative evaluation and matrix factorization in several solves. Then, the composite Newton method could be extended to the NIP algorithm.

- The implementation of other NCP-functions is another interesting topic of research. For instance, many recently developed algorithms to solve complementarity problems (see [16, 20]) employ non-smooth reformulation, involving the computation of the generalized Jacobian of Clarke [12] within a Newton-type algorithm.

Appendix A

Derivatives: Rectangular and Polar Coordinates

A.1 First-Order Derivatives: Rectangular Coordinates

$$\frac{\partial P_i}{\partial e_i} = 2G_{ii}e_i + \sum_{j\in\mathcal{N}_i}(G_{ij}e_j - B_{ij}f_j) \qquad =: \widehat{N}_{ii} \qquad \text{(A.1)}$$

$$\frac{\partial P_i}{\partial e_j} = G_{ij}e_i + B_{ij}f_i \qquad =: \widehat{N}_{ij} \qquad \text{(A.2)}$$

$$\frac{\partial P_i}{\partial f_i} = 2G_{ii}f_i + \sum_{j\in\mathcal{N}_i}(G_{ij}f_j + B_{ij}e_j) \qquad =: \widehat{H}_{ii} \qquad \text{(A.3)}$$

$$\frac{\partial P_i}{\partial f_j} = G_{ij}f_i - B_{ij}e_i \qquad =: \widehat{H}_{ij} \qquad \text{(A.4)}$$

$$\frac{\partial P_i}{\partial t_{ij}} = -g_{ij}(e_ie_j + f_if_j) - b_{ij}(f_ie_j - e_if_j) + 2t_{ij}g_{ij}(e_i^2 + f_i^2) \qquad \text{(A.5)}$$

$$\frac{\partial P_j}{\partial t_{ij}} = -g_{ij}(e_je_i + f_jf_i) - b_{ij}(f_je_i - e_jf_i) \qquad \text{(A.6)}$$

$$\frac{\partial Q_i}{\partial e_i} = -2B_{ii}e_i - \sum_{j\in\mathcal{N}_i}(G_{ij}f_j + B_{ij}e_j) \qquad =: \widehat{L}_{ii} \qquad \text{(A.7)}$$

$$\frac{\partial Q_i}{\partial e_j} = G_{ij}f_i - B_{ij}e_i \qquad =: \widehat{L}_{ij} \qquad \text{(A.8)}$$

$$\frac{\partial Q_i}{\partial f_i} = -2B_{ii}f_i + \sum_{j\in\mathcal{N}_i}(G_{ij}e_j - B_{ij}f_j) \qquad =: \widehat{M}_{ii} \qquad \text{(A.9)}$$

$$\frac{\partial Q_i}{\partial f_j} = -G_{ij}e_i - B_{ij}f_i \qquad =: \widehat{M}_{ij} \qquad \text{(A.10)}$$

$$\frac{\partial Q_i}{\partial t_{ij}} = -g_{ij}(f_i e_j - e_i f_j) + b_{ij}(e_i e_j + f_i f_j) - 2t_{ij} b_{ij}(e_i^2 + f_i^2) \tag{A.11}$$

$$\frac{\partial Q_j}{\partial t_{ij}} = -g_{ij}(f_j e_i - e_j f_i) + b_{ij}(e_j e_i + f_j f_i) \tag{A.12}$$

$$\frac{\partial V_i^2}{\partial e_i} = 2e_i \tag{A.13}$$

$$\frac{\partial V_i^2}{\partial f_i} = 2f_i \tag{A.14}$$

A.2 Second-Order Derivatives: Rectangular Coordinates

$$\nabla^2_{ee} P_i = \begin{bmatrix} 0 & 0 & \cdots & 0 & G_{i1} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & G_{i2} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & G_{ii-1} & 0 & \cdots & 0 \\ G_{i1} & G_{i2} & \cdots & G_{ii-1} & 2G_{ii} & G_{ii+1} & \cdots & G_{in} \\ 0 & 0 & \cdots & 0 & G_{ii+1} & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & G_{in} & 0 & \cdots & 0 \end{bmatrix} \tag{A.15}$$

$$\nabla^2_{fe} P_i = \begin{bmatrix} 0 & 0 & \cdots & 0 & -B_{i2} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & -B_{ii-1} & 0 & \cdots & 0 \\ B_{i1} & B_{i2} & \cdots & B_{ii-1} & 0 & B_{ii+1} & \cdots & B_{in} \\ 0 & 0 & \cdots & 0 & -B_{ii+1} & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & -B_{in} & 0 & \cdots & 0 \end{bmatrix} \tag{A.16}$$

$$\nabla^2_{ff} P_i = \begin{bmatrix} 0 & 0 & \cdots & 0 & G_{i2} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & G_{i3} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & G_{ii-1} & 0 & \cdots & 0 \\ G_{i2} & G_{i3} & \cdots & G_{ii-1} & 2G_{ii} & G_{ii+1} & \cdots & G_{in} \\ 0 & 0 & \cdots & 0 & G_{ii+1} & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & G_{in} & 0 & \cdots & 0 \end{bmatrix}. \tag{A.17}$$

$$\nabla^2_{ee}Q_i = -\begin{bmatrix} 0 & 0 & \cdots & 0 & B_{i1} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & B_{i2} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & B_{ii-1} & 0 & \cdots & 0 \\ B_{i1} & B_{i2} & \cdots & B_{ii-1} & 2B_{ii} & B_{ii+1} & \cdots & B_{in} \\ 0 & 0 & \cdots & 0 & B_{ii+1} & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & B_{in} & 0 & \cdots & 0 \end{bmatrix} \tag{A.18}$$

$$\nabla^2_{fe}Q_i = \begin{bmatrix} 0 & 0 & \cdots & 0 & -G_{i2} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & -G_{ii-1} & 0 & \cdots & 0 \\ G_{i1} & G_{i2} & \cdots & G_{ii-1} & 0 & G_{ii+1} & \cdots & G_{in} \\ 0 & 0 & \cdots & 0 & -G_{ii+1} & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & -G_{in} & 0 & \cdots & 0 \end{bmatrix} \tag{A.19}$$

$$\nabla^2_{ff}Q_i = -\begin{bmatrix} 0 & 0 & \cdots & 0 & B_{i2} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & B_{i3} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & B_{ii-1} & 0 & \cdots & 0 \\ B_{i2} & B_{i3} & \cdots & B_{ii-1} & 2B_{ii} & B_{ii+1} & \cdots & B_{in} \\ 0 & 0 & \cdots & 0 & B_{ii+1} & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & B_{in} & 0 & \cdots & 0 \end{bmatrix}. \tag{A.20}$$

$$\frac{\partial^2 P_i}{\partial t_{ij}\partial e_i} = -g_{ij}e_j + b_{ij}f_j + 4t_{ij}g_{ij}e_i \tag{A.21}$$

$$\frac{\partial^2 P_j}{\partial t_{ij}\partial e_i} = -g_{ij}e_j - b_{ij}f_j \tag{A.22}$$

$$\frac{\partial^2 Q_i}{\partial t_{ij}\partial e_i} = g_{ij}f_j + b_{ij}e_j - 4t_{ij}b_{ij}e_i \tag{A.23}$$

$$\frac{\partial^2 Q_j}{\partial t_{ij}\partial e_i} = -g_{ij}f_j + b_{ij}e_j \tag{A.24}$$

$$\frac{\partial^2 P_i}{\partial t_{ij}\partial e_j} = -g_{ij}e_i - b_{ij}f_i \tag{A.25}$$

$$\frac{\partial^2 P_j}{\partial t_{ij}\partial e_j} = -g_{ij}e_i + b_{ij}f_i, \tag{A.26}$$

$$\frac{\partial^2 Q_i}{\partial t_{ij}\partial e_j} = -g_{ij}f_i + b_{ij}e_i \tag{A.27}$$

$$\frac{\partial^2 Q_j}{\partial t_{ij}\partial e_j} = g_{ij}f_i + b_{ij}e_i \tag{A.28}$$

$$\frac{\partial^2 P_i}{\partial t_{ij}\partial f_i} = -g_{ij}f_j - b_{ij}e_j + 4t_{ij}g_{ij}f_i \tag{A.29}$$

$$\frac{\partial^2 P_j}{\partial t_{ij}\partial f_i} = -g_{ij}f_j + b_{ij}e_j \tag{A.30}$$

$$\frac{\partial^2 Q_i}{\partial t_{ij}\partial f_i} = -g_{ij}e_j + b_{ij}f_j - 4t_{ij}b_{ij}f_i \tag{A.31}$$

$$\frac{\partial^2 Q_j}{\partial t_{ij}\partial f_i} = g_{ij}e_j + b_{ij}f_j \tag{A.32}$$

$$\frac{\partial^2 P_i}{\partial t_{ij}\partial f_j} = -g_{ij}f_i + b_{ij}e_i \tag{A.33}$$

$$\frac{\partial^2 P_j}{\partial t_{ij}\partial f_j} = -g_{ij}f_i - b_{ij}e_i, \tag{A.34}$$

$$\frac{\partial^2 Q_i}{\partial t_{ij}\partial f_j} = g_{ij}e_i + b_{ij}f_i \tag{A.35}$$

$$\frac{\partial^2 Q_j}{\partial t_{ij}\partial f_j} = -g_{ij}e_i + b_{ij}f_i \tag{A.36}$$

$$\frac{\partial^2 P_i}{\partial t_{ij}^2} = 2g_{ij}(e_i^2 + f_i^2) \tag{A.37}$$

$$\frac{\partial^2 Q_i}{\partial t_{ij}^2} = -2b_{ij}(e_i^2 + f_i^2) \tag{A.38}$$

A.3 First-Order Derivatives: Polar Coordinates

$$\frac{\partial P_i}{\partial V_i} = \frac{P_i}{V_i} + G_{ii}V_i \qquad =: N_{ii} \tag{A.39}$$

$$\frac{\partial P_i}{\partial V_j} = V_i(G_{ij}\cos(\theta_i - \theta_j) + B_{ij}\sin(\theta_i - \theta_j)) \qquad =: N_{ij} \tag{A.40}$$

$$\frac{\partial P_i}{\partial \theta_i} = -Q_i - B_{ii}V_i^2 \qquad =: H_{ii} \tag{A.41}$$

$$\frac{\partial P_i}{\partial \theta_j} = V_i V_j (G_{ij} \sin(\theta_i - \theta_j) - B_{ij} \cos(\theta_i - \theta_j)) \qquad =: H_{ij} \qquad \text{(A.42)}$$

$$\frac{\partial Q_i}{\partial V_i} = \frac{Q_i}{V_i} - B_{ii} V_i \qquad =: L_{ii} \qquad \text{(A.43)}$$

$$\frac{\partial Q_i}{\partial V_j} = V_i (G_{ij} \sin(\theta_i - \theta_j) - B_{ij} \cos(\theta_i - \theta_j)) \qquad =: L_{ij} \qquad \text{(A.44)}$$

$$\frac{\partial Q_i}{\partial \theta_i} = P_i - G_{ii} V_i^2 \qquad =: M_{ii} \qquad \text{(A.45)}$$

$$\frac{\partial Q_i}{\partial \theta_j} = -V_i V_j (G_{ij} \cos(\theta_i - \theta_j) + B_{ij} \sin(\theta_i - \theta_j)) \qquad =: M_{ij} \qquad \text{(A.46)}$$

$$\frac{\partial P_i}{\partial t_{ij}} = -V_i V_j (g_{ij} \cos(\theta_i - \theta_j) + b_{ij} \sin(\theta_i - \theta_j)) + 2 t_{ij} g_{ij} V_i^2 \qquad \text{(A.47)}$$

$$\frac{\partial P_j}{\partial t_{ij}} = -V_i V_j (g_{ij} \cos(\theta_i - \theta_j) - b_{ij} \sin(\theta_i - \theta_j)) \qquad \text{(A.48)}$$

$$\frac{\partial Q_i}{\partial t_{ij}} = -V_i V_j (g_{ij} \sin(\theta_i - \theta_j) - b_{ij} \cos(\theta_i - \theta_j)) - 2 t_{ij} b_{ij} V_i^2 \qquad \text{(A.49)}$$

$$\frac{\partial Q_j}{\partial t_{ij}} = V_i V_j (g_{ij} \sin(\theta_i - \theta_j) + b_{ij} \cos(\theta_i - \theta_j)) \qquad \text{(A.50)}$$

A.4 Second-Order Derivatives: Polar Coordinates

$$\nabla^2_{vv} P_i = \begin{bmatrix} 0 & 0 & \cdots & 0 & \frac{N_{i1}}{V_i} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \frac{N_{i2}}{V_i} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & \frac{N_{ii-1}}{V_i} & 0 & \cdots & 0 \\ \frac{N_{i1}}{V_i} & \frac{N_{i2}}{V_i} & \cdots & \frac{N_{ii-1}}{V_i} & 2G_{ii} & \frac{N_{ii+1}}{V_i} & \cdots & \frac{N_{in}}{V_i} \\ 0 & 0 & \cdots & 0 & \frac{N_{ii+1}}{V_i} & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \frac{N_{in}}{V_i} & 0 & \cdots & 0 \end{bmatrix} \qquad \text{(A.51)}$$

$$\nabla^2_{\theta v} P_i = \begin{bmatrix} 0 & \frac{H_{i2}}{V_2} & \cdots & 0 & \frac{H_{i2}}{V_i} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \frac{H_{ii-1}}{V_{i-1}} & \frac{H_{ii-1}}{V_i} & 0 & \cdots & 0 \\ -\frac{H_{i1}}{V_1} & -\frac{H_{i2}}{V_2} & \cdots & -\frac{H_{ii-1}}{V_{i-1}} & \frac{H_{ii}}{V_i} & -\frac{H_{ii+1}}{V_{i+1}} & \cdots & -\frac{H_{in}}{V_n} \\ 0 & 0 & \cdots & 0 & \frac{H_{ii+1}}{V_i} & \frac{H_{ii+1}}{V_{i+1}} & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \frac{H_{in}}{V_i} & 0 & \cdots & \frac{H_{in}}{V_n} \end{bmatrix} \qquad \text{(A.52)}$$

$$\nabla^2_{\theta\theta} P_i = \begin{bmatrix} M_{i2} & 0 & \cdots & 0 & -M_{i2} & 0 & \cdots & 0 \\ 0 & M_{i3} & \cdots & 0 & -M_{i3} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & M_{ii-1} & -M_{ii-1} & 0 & \cdots & 0 \\ -M_{i2} & -M_{i3} & \cdots & -M_{ii-1} & -M_{ii} & -M_{ii+1} & \cdots & -M_{in} \\ 0 & 0 & \cdots & 0 & -M_{ii+1} & M_{ii+1} & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & -M_{in} & 0 & \cdots & M_{in} \end{bmatrix} \tag{A.53}$$

$$\nabla^2_{vv} Q_i = \begin{bmatrix} 0 & 0 & \cdots & 0 & \frac{L_{i1}}{V_i} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \frac{L_{i2}}{V_i} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & \frac{L_{ii-1}}{V_i} & 0 & \cdots & 0 \\ \frac{L_{i1}}{V_i} & \frac{L_{i2}}{V_i} & \cdots & \frac{L_{ii-1}}{V_i} & -2B_{ii} & \frac{L_{ii+1}}{V_i} & \cdots & \frac{L_{in}}{V_i} \\ 0 & 0 & \cdots & 0 & \frac{L_{ii+1}}{V_i} & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \frac{L_{in}}{V_i} & 0 & \cdots & 0 \end{bmatrix} \tag{A.54}$$

$$\nabla^2_{\theta v} Q_i = \begin{bmatrix} 0 & \frac{M_{i2}}{V_2} & \cdots & 0 & \frac{M_{i2}}{V_i} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \frac{M_{ii-1}}{V_{i-1}} & \frac{M_{ii-1}}{V_i} & 0 & \cdots & 0 \\ -\frac{M_{i1}}{V_1} & -\frac{M_{i2}}{V_2} & \cdots & -\frac{M_{ii-1}}{V_{i-1}} & \frac{M_{ii}}{V_i} & -\frac{M_{ii+1}}{V_{i+1}} & \cdots & -\frac{M_{in}}{V_n} \\ 0 & 0 & \cdots & 0 & \frac{M_{ii+1}}{V_i} & \frac{M_{ii+1}}{V_{i+1}} & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \frac{M_{in}}{V_i} & 0 & \cdots & \frac{M_{in}}{V_n} \end{bmatrix} \tag{A.55}$$

$$\nabla^2_{\theta\theta} Q_i = \begin{bmatrix} -H_{i2} & 0 & \cdots & 0 & H_{i2} & 0 & \cdots & 0 \\ 0 & -H_{i3} & \cdots & 0 & H_{i3} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & -H_{ii-1} & H_{ii-1} & 0 & \cdots & 0 \\ H_{i2} & H_{i3} & \cdots & H_{ii-1} & H_{ii} & H_{ii+1} & \cdots & H_{in} \\ 0 & 0 & \cdots & 0 & H_{ii+1} & -H_{ii+1} & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & H_{in} & 0 & \cdots & -H_{in} \end{bmatrix}. \tag{A.56}$$

$$\frac{\partial^2 P_i}{\partial t_{ij} \partial V_i} = 4 t_{ij} g_{ij} V_i - V_j (g_{ij} \cos(\theta_i - \theta_j) + b_{ij} \sin(\theta_i - \theta_j)) \tag{A.57}$$

$$\frac{\partial^2 P_j}{\partial t_{ij} \partial V_i} = -V_j (g_{ij} \cos(\theta_i - \theta_j) - b_{ij} \sin(\theta_i - \theta_j)) \tag{A.58}$$

$$\frac{\partial^2 Q_i}{\partial t_{ij} \partial V_i} = -4 t_{ij} b_{ij} V_i - V_j (g_{ij} \sin(\theta_i - \theta_j) - b_{ij} \cos(\theta_i - \theta_j)) \tag{A.59}$$

$$\frac{\partial^2 Q_j}{\partial t_{ij} \partial V_i} = V_j (g_{ij} \sin(\theta_i - \theta_j) + b_{ij} \cos(\theta_i - \theta_j)) \tag{A.60}$$

$$\frac{\partial^2 P_i}{\partial t_{ij} \partial V_j} = -V_i (g_{ij} \cos(\theta_i - \theta_j) + b_{ij} \sin(\theta_i - \theta_j)) \tag{A.61}$$

$$\frac{\partial^2 P_j}{\partial t_{ij} \partial V_j} = -V_i (g_{ij} \cos(\theta_i - \theta_j) - b_{ij} \sin(\theta_i - \theta_j)) \tag{A.62}$$

$$\frac{\partial^2 Q_i}{\partial t_{ij} \partial V_j} = -V_i (g_{ij} \sin(\theta_i - \theta_j) - b_{ij} \cos(\theta_i - \theta_j)) \tag{A.63}$$

$$\frac{\partial^2 Q_j}{\partial t_{ij} \partial V_j} = V_i (g_{ij} \sin(\theta_i - \theta_j) + b_{ij} \cos(\theta_i - \theta_j)) \tag{A.64}$$

$$\frac{\partial^2 P_i}{\partial t_{ij} \partial \theta_i} = V_i V_j (g_{ij} \sin(\theta_i - \theta_j) - b_{ij} \cos(\theta_i - \theta_j)) \tag{A.65}$$

$$\frac{\partial^2 P_j}{\partial t_{ij} \partial \theta_i} = V_i V_j (g_{ij} \sin(\theta_i - \theta_j) + b_{ij} \cos(\theta_i - \theta_j)) \tag{A.66}$$

$$\frac{\partial^2 Q_i}{\partial t_{ij} \partial \theta_i} = -V_i V_j (g_{ij} \cos(\theta_i - \theta_j) + b_{ij} \sin(\theta_i - \theta_j)) \tag{A.67}$$

$$\frac{\partial^2 Q_j}{\partial t_{ij} \partial \theta_i} = V_i V_j (g_{ij} \cos(\theta_i - \theta_j) - b_{ij} \sin(\theta_i - \theta_j)) \tag{A.68}$$

$$\frac{\partial^2 P_i}{\partial t_{ij} \partial \theta_j} = -V_i V_j (g_{ij} \sin(\theta_i - \theta_j) - b_{ij} \cos(\theta_i - \theta_j)) \tag{A.69}$$

$$\frac{\partial^2 P_j}{\partial t_{ij} \partial \theta_j} = -V_i V_j (g_{ij} \sin(\theta_i - \theta_j) + b_{ij} \cos(\theta_i - \theta_j)) \tag{A.70}$$

$$\frac{\partial^2 Q_i}{\partial t_{ij} \partial \theta_j} = V_i V_j (g_{ij} \cos(\theta_i - \theta_j) + b_{ij} \sin(\theta_i - \theta_j)) \tag{A.71}$$

$$\frac{\partial^2 Q_j}{\partial t_{ij} \partial \theta_j} = -V_i V_j (g_{ij} \cos(\theta_i - \theta_j) - b_{ij} \sin(\theta_i - \theta_j)) \tag{A.72}$$

$$\frac{\partial^2 P_i}{\partial t_{ij}^2} = 2 g_{ij} V_i^2 \tag{A.73}$$

$$\frac{\partial^2 Q_i}{\partial t_{ij}^2} = -2 b_{ij} V_i^2 \tag{A.74}$$

Bibliography

[1] I. AKROTIRIANAKIS AND B. RUSTEM, *A globally convergent interior point algorithm for general non-linear programming problems*, Technical Report 97/14, Department of Computing, Imperial College of Science, Technology and Medicine, London, UK, 1997.

[2] K. C. ALMEIDA AND F. D. GALIANA, *Critical cases in the optimal power flow*, IEEE Trans. on Power Systems, 11 (1996), pp. 1509–1518.

[3] M. G. BREITFELD AND D. F. SHANNO, *Preliminary computational experience with modified log-barrier functions for large-scale nonlinear programming*, Technical Report RRR 08-93, Rutgers University, New Brunswick, NJ, 1993.

[4] M. G. BREITFELD AND D. F. SHANNO, *A globally convergent penalty-barrier algorithm for nonlinear programming and its computational performance*, Technical Report RRR 12-94, Rutgers University, New Brunswick, NJ, 1994.

[5] J. R. BUNCH AND B. N. PARLETT, *Direct methods for solving symmetric indefinite systems of linear equations*, SIAM Journal on Numerical Analysis, 8 (1971), pp. 639–655.

[6] J. V. BURKE AND S. XU, *The global linear convergence of a non-interior path-following algorithm for linear complementarity problems*, Technical Report, Department of Mathematics, University of Washington, Seattle, WA, Sept. 1996.

[7] R. H. BYRD, J. C. GILBERT, AND J. NOCEDAL, *A trust region method based on interior point techniques for nonlinear programming*, Technical Report, Computer Science Department, University of Colorado, Boulder, CO, May 1996.

[8] R. H. BYRD, M. E. HRIBAR, AND J. NOCEDAL, *An interior point algorithm for large scale nonlinear programming*, Technical Report, Computer Science Department, University of Colorado, Boulder, CO, July 1997.

[9] C. A. CAÑIZARES, *Applications of optimization to voltage collapse analysis*, (1998). Panel Session: "Optimization Techniques in Voltage Collapse Analysis," IEEE/PES Summer Meeting, San Diego.

[10] T. J. CARPENTER, I. J. LUSTIG, J. M. MULVEY, AND D. F. SHANNO, *Higher-order predictor-corrector interior point methods with applications to quadratic objectives*, SIAM Journal on Optimization, 3 (1993), pp. 696–725.

[11] B. CHEN AND P. T. HARKER, *A non-interior-point continuation method for linear complementarity problems*, SIAM Journal on Matrix Analysis and Applications, 14 (1993), pp. 1168–1190.

[12] F. H. CLARKE, *Optimization and Nonsmooth Analysis*, SIAM, 1990.

[13] K. A. CLEMENTS, P. W. DAVIS, AND K. D. FREY, *Treatment of inequality constraints in power system state estimation*, IEEE Trans. on Power Systems, 10 (1995), pp. 567–573.

[14] T. A. DAVIS AND I. S. DUFF, *An unsymmetric-pattern multifrontal method for sparse LU factorization.* To appear in SIAM Journal on Matrix Analysis and Applications.

[15] ——, *A combined unifrontal/multifrontal method for unsymmetric sparse matrices*, Technical Report TR-97-016, Computer and Information Science and Engineering Department, University of Florida, Sept. 1997.

[16] T. DE LUCA, F. FACCHINEI, AND C. KANZOW, *A semismooth equation approach to the solution of nonlinear complementarity problems*, Mathematical Programming, 75 (1996), pp. 407–439.

[17] J. E. DENNIS JR. AND R. B. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM Classics In Applied Mathematics, Philadelphia, 1996.

[18] H. W. DOMMEL AND W. F. TINNEY, *Optimal power flow solution*, IEEE Trans. on Power Apparatus and Systems, PAS-87 (1968), pp. 1866–1876.

[19] A. EL-BAKRY, R. TAPIA, T. TSUCHIYA, AND Y. ZHANG, *On the formulation and theory of the newton interior-point method for nonlinear programming*, Journal of Optimization Theory and Applications, 89 (1996), pp. 507–541.

[20] F. FACCHINEI AND C. KANZOW, *A nonsmooth inexact Newton method for the solution of large-scale nonlinear complementarity problems*, Mathematical Programming, 76 (1997), pp. 493–512.

[21] A. V. FIACCO AND G. P. MCCORMICK, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, John Wiley & Sons, 1968.

[22] X. V. FILHO, S. GRANVILLE, B. G. GORENSTIN, D. S. CARVALHO JR., L. A. S. PILLOTO, E. S. SOBRINHO, AND J. C. O. MELLO, *Optimization methods in power systems: Application and perspectives*, in Proc. of the VI SEPOPE, Salvador, Bahia, Brasil, May 1998. Paper IP-63 in CD-ROM.

[23] A. FISCHER, *A special Newton-type optimization method*, Optimization, 24 (1992), pp. 269–284.

[24] R. FLETCHER, *Practical Methods of Optimization*, John Wiley & Sons, 1987.

[25] R. FOURER AND S. MEHROTRA, *Solving symmetric indefinite systems in an interior-point method for linear programming*, Mathematical Programming, 62 (1993), pp. 15–39.

[26] K. R. FRISCH, *The logarithmic potential method of convex programming*, Manuscript, University Institute of Economics, Oslo, Norway, 1955.

[27] F. D. GALIANA AND Z.-C. ZENG, *Analysis of the load flow behaviour near a Jacobian singularity*, IEEE Trans. on Power Systems, 7 (1992), pp. 1362–1369.

[28] D. M. GAY, M. L. OVERTON, AND M. H. WRIGHT, *A primal-dual interior method for nonconvex nonlinear programming*, Technical Report 97-4-08, Computer Sciences Research Center, Bell Laboratories, Murray Hill, NJ, July 1997.

[29] A. GEORGE AND J. W.-H. LIU, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall, 1981.

[30] P. E. GILL, W. MURRAY, D. B. PONCELEÓN, AND M. A. SAUNDERS, *Primal-dual methods for linear programming*, Mathematical Programming, 70 (1995), pp. 251–277.

[31] J. GONDZIO, *Multiple centrality corrections in a primal-dual method for linear programming*, Computational Optimization and Applications, 6 (1996), pp. 137–156.

[32] C. C. GONZAGA, *Path following methods for linear programming*, SIAM Review, 34 (1992), pp. 167–224.

[33] S. GRANVILLE, *Optimal reactive dispatch through interior point methods*, IEEE Trans. on Power Systems, 9 (1994), pp. 136–146.

[34] S. GRANVILLE, J. C. O. MELLO, AND A. C. G. MELO, *Application of interior point methods to power flow unsolvability*, IEEE Trans. on Power Systems, 11 (1996), pp. 1096–1103.

[35] W. W. HAGER, ed., *Large Scale Optimization: State of the Art*, Kluwer Academic Publishers, 1994.

[36] I. A. HISKENS, *Analysis tools for power systems: Contending with nonlinearities*, Proceedings of the IEEE, 83 (1995), pp. 1573–1587.

[37] K. HOTTA AND A. YOSHISE, *Global convergence of a class of non-interior-point algorithms using Chen-Harker-Kanzow functions for nonlienar complementarity problems*, Discussion Papers Series 708, Institute of Policy and Planning Sciences, University of Tsukuba, Japan, 1996.

[38] M. HUNEAULT AND F. D. GALIANA, *A survey of the optimal power flow literature*, IEEE Trans. on Power Systems, 6 (1991), pp. 762–770.

[39] G. D. IRISARRI, L. M. KIMBALL, K. A. CLEMENTS, A. BAGCHI, AND P. W. DAVIS, *Economic dispatch with network and ramping constraints via interior point methods*, 1997 IEEE/PES Summer Meeting. Paper no. PE-448-PWRS-0-04-1997.

[40] G. D. IRISARRI, X. WANG, J. TONG, AND S. MOKHTARI, *Maximum loadability of power systems using interior point non-linear optimization method*, IEEE Trans. on Power Systems, 12 (1997), pp. 162–172.

[41] S. IWAMOTO, M. KUSANO, AND V. H. QUINTANA, *Hierarchical state estimation using a fast rectangular coordinates method*, 1989 IEEE/PES Winter Meeting. Paper no. 89 WM 160-3 PWRS.

[42] S. IWAMOTO AND Y. TAMURA, *A load flow calculation method for ill-conditioned power systems*, IEEE Trans. on Power Apparatus and Systems, PAS-100 (1981), pp. 1736–1743.

[43] B. JANSEN, C. ROOS, T. TERLAKY, AND J.-P. VIAL, *Primal-dual target following algorithms for linear programming*, Technical Report 93-107, Faculty of Technical Mathematics and Informatics, Technical University of Delft, Delft, The Netherlands, 1993.

[44] C. KANZOW, *Some equation-based methods for the nonlinear complementarity problem*, Optimization Methods and Software, 3 (1994), pp. 327–340.

[45] ——, *Nonlinear complementarity as unconstrained optimization*, Journal of Optimization Theory and Applications, 88 (1996), pp. 139–155.

[46] ——, *Some noninterior continuation methods for linear complementarity problems*, SIAM Journal on Matrix Analysis and Applications, 17 (1996), pp. 851–868.

[47] N. KARMARKAR, *A new polynomial-time algorithm for linear programming*, Combinatorica, 4 (1984), pp. 373–395.

[48] L. G. KHACHIYAN, *A polynomial algorithm in linear programming*, Doklady Akademii Nauk SSSR, 224 (1979), pp. 1093–1096. English version in Soviet Methematics Doklady, 20 (1979), pp. 191-194.

[49] M. KOJIMA, S. MIZUNO, AND A. YOSHISE, *A primal-dual interior-point method for linear programming.* in Progress in Mathematical Programming, Interior Point and Related Methods, N. Megiddo, ed., Springer-Verlag, New York, 1989, pp. 29–47.

[50] P. KUNDUR, *Power System Stability and Control*, McGraw-Hill, Inc., 1994.

[51] L. S. LASDON, J. PLUMMER, AND G. YU, *Primal-dual and primal interior point algorithms for general nonlinear programs*, ORSA Journal on Computing, 7 (1995), pp. 321–332.

[52] I. J. LUSTIG, R. E. MARSTEN, AND D. F. SHANNO, *Computational experience with a primal-dual interior point method for linear programming*, Linear Algebra and Its Applications, 152 (1991), pp. 191–222.

[53] R. E. MARSTEN AND M. J. SALTZMAN, *Implementation of a dual affine interior point algorithm for linear programming*, ORSA Journal on Computing, 1 (1989), pp. 287–297.

[54] J. L. MARTÍNEZ, A. GÓMEZ, AND V. H. QUINTANA, *Reactive-power optimization by interior point methods: Implementation issues*, in Proc. of the 12th Power Systems Computation Conference, Dresden, Germany, Aug. 1996, pp. 844–850.

[55] J. MEDINA, V. H. QUINTANA, AND A. J. CONEJO, *A clipping-off interior-point technique for medium-term hydro-thermal coordination.* To appear in IEEE Trans. on Power Systems.

[56] N. MEGIDDO, *Pathways to the optimal set in linear programming*, Technical Report RJ 5295, IBM Almaden Research Center, San Jose, CA, 1986.

[57] S. MEHROTRA, *On the implementation of a primal-dual interior point method*, SIAM Journal on Optimization, 2 (1992), pp. 575–601.

[58] S. MIZUNO AND A. NAGASAWA, *A primal-dual affine-scaling potential-reduction algorithm for linear programming*, Mathematical Programming, 62 (1993), pp. 119–131.

[59] W. MURRAY AND M. H. WRIGHT, *Line search procedures for the logarithmic barrier function*, Technical Report, Department of Operations Research, Stanford University, 1997.

[60] T. J. OVERBYE AND R. P. KLUMP, *Effective calculation of power system low-voltage solutions*, IEEE Trans. on Power Systems, 11 (1996), pp. 75–82.

[61] V. H. QUINTANA AND G. L. TORRES, *Optimal power flow by interior and non-interior point methods for nonlinear programming*, in Proc. of the VI SEPOPE, Salvador, Bahia, Brasil, May 1998. Paper IP-63 in CD-ROM.

[62] V. H. QUINTANA, G. L. TORRES, AND J. MEDINA-PALOMO, *Interior-point methods and their applications to power systems: A classification of publications and software codes.* To appear in IEEE Trans. on Power Systems, Paper No. 98 SM 244.

[63] M. SANTOS-NIETO AND V. H. QUINTANA, *Reactive power control for real power loss minimization via a non-conventional lp algorithm*, Technical Report UW E&CE 86-02, Dept. of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, Canada, 1986.

[64] D. F. SHANNO, M. G. BREITFELD, AND E. M. SIMANTIRAKI, *Implementing barrier methods for nonlinear programming*, Technical Report RRR 39-95, Rutgers University, New Brunswick, NJ, 1995.

[65] B. STOTT, O. ALSAC, AND A. J. MONTICELLI, *Security analysis and optimization*, Proceedings of the IEEE, 75 (1987), pp. 1623–1644.

[66] D. I. SUN, B. ASHLEY, B. BREWER, A. HUGHES, AND W. F. TINNEY, *Optimal power flow by Newton approach*, IEEE Trans. on Power Apparatus and Systems, PAS-103 (1984), pp. 2864–2880.

[67] R. TAPIA, Y. ZHANG, M. SALTZMAN, AND A. WEISER, *The mehrotra predictor-corrector interior-point method as a perturbed composite newton method*, SIAM Journal on Optimization, (1996), pp. 47–56.

[68] G. TOGNOLA AND R. BACHER, *Unlimited point algorithm for OPF problems.* To appear in IEEE Trans. on Power Systems, Paper no. PE-167PWRS-16-09-1997.

[69] G. L. TORRES AND V. H. QUINTANA, *An interior point method for nonlinear optimal power flow using voltage rectangular coordinates*, IEEE Trans. on Power Systems, 13 (1998), pp. 1211–1218.

[70] G. L. TORRES, V. H. QUINTANA, AND G. LAMBERT-TORRES, *Optimal power flow in rectangular form via an interior point method*, in Proc. of the 1996 IEEE North American Power Symposium, Massachusetts Institute of Technology, Cambridge, MA, Nov. 1996, pp. 481–488.

[71] R. J. VANDERBEI AND D. F. SHANNO, *An interior point algorithm for nonconvex nonlinear programming*, Technical Report SOR-97-21, School of Engineering and Operations Research, Princeton University, Princeton, NJ, Nov. 1997.

[72] H. WEI, H. SASAKI, AND R. YOKOYAMA, *An application of interior point quadratic programming algorithm to power system optimization problems*, IEEE Trans. on Power Systems, 11 (1996), pp. 260–266.

[73] A. J. WOOD AND B. F. WOLLENBERG, *Power Generation, Operation, and Control*, John Wiley & Sons, second ed., 1996.

[74] M. H. WRIGHT, *Interior methods for constrained optimization.* pp. 341–407, in W. W. Hager (ed.), *Acta Numerica*, Kluwer Academic Publishers, 1991.

[75] M. H. WRIGHT, *Some properties of the hessian of the logarithmic barrier function*, Mathematical Programming, 67 (1994), pp. 265–295.

[76] S. J. WRIGHT, *Primal-Dual Interior-Point Methods*, SIAM, 1997.

[77] Y. WU, A. S. DEBS, AND R. E. MARSTEN, *A direct nonlinear predictor-corrector primal-dual interior point algorithm for optimal power flow*, IEEE Trans. on Power Systems, 9 (1994), pp. 876–883.

[78] H. YAMASHITA AND H. YABE, *Superlinear and quadratic convergence of some primal-dual interior point methods for constrained optimization*, Mathematical Programming, 75 (1996), pp. 377–397.

[79] X. YAN AND V. H. QUINTANA, *An efficient predictor-corrector interior point algorithm for security-constrained economic dispatch*, IEEE Trans. on Power Systems, 12 (1997), pp. 803–810.

[80] Y. ZHANG, *Solving large-scale linear programs by interior-point methods under the MATLAB environment*, Technical Report TR96-01, Department of Mathematics and Statistics, Univeristy of Maryland Baltimore County, Maryland, Feb. 1996.