Optimal Power Flow on Networks with Tree Topology

Alex Shtof

Optimal Power Flow on Networks with Tree Topology

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

Alex Shtof

Submitted to the Senate of the Technion - Israel Institute of Technology
Adar bet 5779, March 2019

The Research Thesis Was Done Under The Supervision of Prof. Amir Beck and Prof. Shoham Sabach in the Faculty of Industrial Engineering and Management.

The generous financial help of the Technion is gratefully acknowledged.

Publications

[1] A. Beck, Y. Beck, Y. Levron, A. Shtof, L. Tetruashvili. Globally Solving a Class of Optimal Power Flow Problems in Radial Networks by Tree Reduction. Journal of Global Optimization, 72(3), pp. 373-402.

The initial idea was suggested by Prof. Yuval Levron, and an early prototype implementation was done by Prof. Amir Beck and Dr. Luba Tetruashvili. Dr. Yuval Beck provided additional expertise. A formal mathematical study of the idea, the theoretical justification for its correctness, the final algorithm's implementation, and writing the vast majority of the paper's contents are my own contributions. Chapter 4 of this thesis is almost identical to paper [1].

Contents

List of Tables

List of Figures

List of Algorithms

Abstract

In this thesis we study two classes of the so-called Optimal Power Flow (OPF) problem for networks with a tree topology, also known as radial networks. For each class we devise an algorithm for obtaining an approximate globally optimal solution. To the best of our knowledge, algorithms for both problem classes which produce globally optimal solution were not previously known.

Both algorithms share a similar structure. First, we establish a relationship between the optimal solution set of the OPF problem on a given network and the optimal solution set of the OPF problem on a smaller network. Then, we use the above relationship repeatedly to connect the optimal solution set of the OPF problem on the given network with the optimal solution set of a terminal problem, whose associated OPF problem is simple enough to be quickly and reliably solved. This connection results in an algorithm for solving a given OPF problem - we construct the terminal problem, find its optimal solution, and use the established connection to obtain an optimal solution of the original problem. However, the above-mentioned relationship consists of mathematical operations which cannot be computed exactly, and thus we resort to approximation. To evaluate the effect of the approximation on the quality of the algorithm's output in terms of feasibility and sub-optimality, we perform numerical experiments on several test problems, and analyze the convergence properties of the approximation scheme.

The first family comprises networks with one free generator, while the rest of the nodes are consumers and generators whose consumption and generation parameters are known. For this family we have developed a fast, robust, and exact method. This claim is backed up by a set of numerical experiments which demonstrate the method's effectiveness. The second family comprises networks with several free generators. The resulting algorithm is requires much more computational power and computer memory. However, because it can easily be implemented to run in parallel, it adapts well to the highly parallel nature of modern computers, whose parallel processors count is rapidly growing.

Chapter 1

Overview, Terminology and Notation

1.1 Introduction

Power systems are large networks which transport electrical power from generators to consumers. The optimal power flow problem is a mathematical optimization problem concerned with finding the state of the system that minimizes a certain objective, which might involve the total cost of generation, stability and reliability of the system, subject to the (nonconvex) constraints imposed by the laws of physics, and additional operational constraints. The decision variables, constraints and objective involve physical quantities which arise in the analysis of electrical circuits, such as voltage, current and power. Readers are referred to [28] for a survey on formulations and properties of OPF problems.

The nonconvexity of the feasible set makes the problem computationally intractable in general. In spite of this, the substantial economic and engineering importance of this problem led to the development of various methods for globally solving special instances of the problem, or even more common, to the derivation of approximation and heuristic techniques. For a comprehensive review of these methods, the reader is referred to the extensive review [10] as well as references therein.

In this work, we concentrate on *radial networks*, which are networks with a tree topology. Tree topology does not make the problem easier in general, and it remains computationally intractable. However, specific families of radial networks enjoyed some success in finding the global opimum in a computationally efficient manner, mostly via convex relaxations.

In this work we propose algorithms which compute a global optimum for two classes of the optimal power flow problem on radial networks. The first algorithm is fast and reliable, but is useful for a restricted class of problems. Thus, its usefulness in practice is limited to a small number of applications. However, it turns out to be useful as a computational step in our second method, which solves a much larger class of problems which have a much wider range of practical applications.

1.2 Notation

Either small, capital, or greek italics, e.g. a, I, α, are used to denote scalars or functions. Sets are denoted by italic capitals, either normal or calligraphic, e.g. C or $\mathcal{V}$. Whether a symbol denotes a scalar, a function or a set should be clear from the context. Column vectors and matrices are denoted by bold letters, e.g. $\mathbf{v}$ and $\mathbf{Y}$. Their components are denoted as scalars with index subscripts, e.g. v_k or Y_{ij}

The real number field is denoted by $\mathbb{R}$, and the complex number field is denoted by $\mathbb{C}$. A general space of coordinate vectors, such as tuples or matrices, is denoted by $\mathbb{E}$. The symbol $\mathbf{i}$ denotes the complex imaginary unit, i.e., $\mathbf{i}^2 = -1$. The symbol $\mathbf{i}$ will not be used for vectors or matrices. The complex conjugate of $z \in \mathbb{C}$ is denoted by z^*, its real and imaginary parts by $\mathrm{re}(z)$, $\mathrm{im}(z)$, and its magnitude and angle (in radians) by $|z|$, $\arg(z)$, respectively. For vectors and matrices $\mathrm{re}(\cdot)$, $\mathrm{im}(\cdot)$, $|\cdot|$, and $\arg(\cdot)$ are extended to denote component-wise operations.

An extended real-valued function f is a function which may take the value $+\infty$, that is, $f : \mathbb{E} \to (-\infty, +\infty]$. We denote by $\mathrm{dom}(f)$ its *effective domain*: $\mathrm{dom}(f) \equiv \{x \in \mathbb{E} : f(x) < +\infty\}$. Sub level sets are denoted by $[f \leq \alpha] \equiv \{\mathbf{x} \in \mathbb{E} : f(\mathbf{x}) \leq \alpha\}$. Naturally, we extend the above notation to level sets $[f = \alpha]$. The

indicator function of a set C, is an extended real-valued function defined by

$$\delta_C(\mathbf{x}) = \begin{cases} 0 & \mathbf{x} \in C, \\ +\infty & \mathbf{x} \notin C. \end{cases}$$

Vector components may be indexed arbitrarily, not necessarily by natural numbers. For a finite set $\mathcal{I}$ having some natural total order, we use the notation $S^{\mathcal{I}}$ to denote the vector space of objects, each of which is both a function taking elements of $\mathcal{I}$ and returning elements of S, and a vector whose components are the function values ordered by the natural order of $\mathcal{I}$. For $F \subseteq \mathcal{I}$, we denote by $\mathbf{w}_F$ the function $\mathbf{w}$ restricted to F, or the corresponding vector. For example, let $\mathcal{I} = \{a, b, c\}$ with the natural order being the English alphabet. The vector $\mathbf{w} \in \mathbb{R}^{\mathcal{I}}$ is a three-dimensional real vector in the vector space $\mathbb{R}^{\mathcal{I}}$, whose components are indexed by a, b, and c. In particular, for $\mathbf{w} = (3, -5, 4)^T$ we have $w_a = 3$, $w_b = -5$, and $w_c = 4$. We also have $\mathbf{w}_{\{a,b\}} = (w_a, w_b)^T = (3, -5)^T$ and $\mathbf{w}_{\{c,b\}} = (w_b, w_c)^T = (-5, 4)^T$. A general space of real coordinate vectors is denoted by $\mathbb{E}$. That is, $\mathbb{E}$ may be $\mathbb{R}^n$, $\mathbb{R}^{\mathcal{I}}$ or $\mathbb{R}^n \times \mathbb{R}^m$.

The following notation is defined for an undirected graph $G = (\mathcal{V}, \mathcal{E})$ with the node set $\mathcal{V}$ and the edge set $\mathcal{E}$. The set of edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a symmetric relation: $(u, v) \in \mathcal{E} \implies (v, u) \in \mathcal{E}$. Thus, each edge connecting i and j is represented by two pairs (i, j) and (j, i). For each $k \in \mathcal{V}$ we denote its neighbors by $\mathrm{N}(G, k) \equiv \{j : (k, j) \in \mathcal{E}\}$. A node k is a *leaf* of G if it has a unique neighbor. Throughout the paper we assume that $\mathcal{V}$ has some natural total order denoted by '$<$' and $\mathcal{E}$ is lexicographically ordered. In particular, it also means that nodes and edges can be used as vector indices.

A (free) tree $T = (\mathcal{V}, \mathcal{E})$ is a connected undirected acyclic graph. If the tree is rooted at some node, then the parent-child relationships are automatically defined. We denote by $\mathrm{root}(T)$ the root node of T, and by $\mathrm{ch}_T(j)$ the set of the (direct) children of the node $j \in \mathcal{V}$, by $\mathrm{parent}(T, j)$ its parent, by $\mathrm{des}_T(j)$ its set of descendants, which are all direct and indirect children of j, and by $\mathrm{sub}_T(j) = \{j\} \cup \mathrm{des}_T(j)$ the nodes in the sub-tree rooted at j. We also denote by $\mathrm{L}(T)$ the set of leaves of a rooted tree T, which are the nodes without children: $\mathrm{L}(T) = \{k \in \mathcal{V} : \mathrm{ch}_T(k) = \emptyset\}$. Note that the root of the tree may be a leaf in the graph-theoretic sense, but is usually not a leaf in the rooted tree sense. Specifically, $\mathrm{root}(T) \in \mathrm{L}(T)$ if and only if $\mathrm{root}(T)$ is the only node of T.

1.3 Power networks and general OPF problems

1.3.1 Power networks

In an attempt to make this work as self-contained as possible for readers with mainly mathematical background, we will review some facts which are well known to readers from the power systems community. Additional background material can be found in [10, 28, 51, 52, 34].

The term *power network* refers to a mathematical model which describes a system that conveys electrical power from generators to consumers, and includes both specified parameters and unknown decision variables. The model consists of a graph whose nodes model substations, each having the role of a generator or a load (consumer), while edges model power transmission lines. Both nodes and edges are associated with complex numbers that model the physical quantities describing the flow of power, and obey the laws of physics. We do not attempt to define those quantities, but rather describe the mathematical relationship which connects them to each other. The model also consists of an extended real-valued function which is used to assign a cost to the state of the network as described by the values of the decision variables, and whose effective domain is used to express additional constraints, which usually include information about the role of each node, and physical limits of the nodes and edges to produce, consume, or convey power.

The known quantities are *edge impedances*, which describe how well a transmission line conducts electricity. The decision variables are the *voltage* and *apparent power* associated with each node. In this work we concentrate on AC (Alternating Current) power networks in, in which the voltages associated with each node are sinusoidal signals, which vary in amplitude and phase, but share the same frequency. Such signals are modeled well by complex numbers called *phasors*, whose absolute value models the amplitude and angle models the phase. Consequently, to model the various physical interactions, edge impedances, voltages, and apparent powers are complex numbers. For apparent powers, the real and imaginary parts are called *active* and *reactive* powers, respectively. When active (or reactive) power is produced, the corresponding sign is positive, while when it is consumed the sign is negative. The active power models the amount of power produced or consumed by a component of the power network which

is manifested in the external world. At generators, the active power models the amount of energy the generator introduces into the network from an external source, such as fuel, while for consumers it models the amount of energy consumed from the electrical system and delivered back to the external world, such as the heat produced by a teapot. The reactive power models the amount of power 'stored' or 'delivered back' to the system by electromagnetic interactions.

Formally, a power network is a triple $(G, \mathbf{z}, F)$ where:

$G = (\mathcal{V}, \mathcal{E})$ is an undirected graph, which is called the network topology graph.

$\mathbf{z} \in \mathbb{C}^{\mathcal{E}}$ is a vector (or a function) which associates every edge with its impedance, and thus $(i, j) \in \mathcal{E} \implies z_{ij} = z_{ji}$.

$F : \mathbb{C}^{\mathcal{V}} \times \mathbb{C}^{\mathcal{V}} \to (-\infty, +\infty]$ is an extended real-valued function which assigns a cost to the state of the network described by the decision variables $(\mathbf{v}, \mathbf{s}) \in \mathbb{C}^{\mathcal{V}} \times \mathbb{C}^{\mathcal{V}}$. The vectors $\mathbf{v}$ and $\mathbf{s}$ contain the voltage and apparent power, respectively, associated with each node. Since our goal is to minimize the cost, the set $\mathrm{dom}(F)$ imposes constraints on the values of $\mathbf{v}$ and $\mathbf{s}$. The partition of the network's nodes to generators and consumers is embedded into $\mathrm{dom}(F)$. Examples are provided below.

Throughout this work, references to G and $\mathbf{z}$ without F are also common, and therefore a pair $(G, \mathbf{z})$ consisting of the topology graph and the impedance function also deserves a name - *network structure*. Since in this work we are concerned with globally solving these problems on classes of radial (tree topology) networks, after the introducing the basic concepts and terminology for general networks, radial networks will be given special attention.

1.3.2 Power flow equations

The laws of physics are modeled by the following system of equations in $\mathbf{v}, \mathbf{s}$ and auxiliary variables $\mathbf{I} \in \mathbb{C}^{\mathcal{E}}$, which are known as the *power-flow equations*:

$$z_{ij} I_{ij} = v_i - v_j \qquad (i, j) \in \mathcal{E}, \tag{1.1}$$

$$I_{ij} = -I_{ji} \qquad (i, j) \in \mathcal{E}, \tag{1.2}$$

$$s_i = \sum_{j \in \mathrm{N}(G,i)} v_i I_{ij}^*, \qquad i \in \mathcal{V}. \tag{1.3}$$

Note that a zero impedance z_{ij} means an 'ideal' power transmission line which does not consume energy. Such cases do not happen in practice, and all entries of $\mathbf{z}$ are ubiquitously assumed to be non-zero in the existing literature. However, networks with zero impedances are used internally by the algorithms we describe in this work, and thus need to be addressed. Under the non-zero impedance assumption, we can directly express $\mathbf{I}$ in terms of $\mathbf{v}$, and the system of equations (1.1)-(1.3) has the following well-known form:

$$s_i = \sum_{j \in \mathcal{V}} v_i v_j^* Y_{ij}^*, \quad i \in \mathcal{V}, \tag{1.4}$$

where $Y \in \mathbb{C}^{\mathcal{V} \times \mathcal{V}}$ is defined by

$$Y_{ij} = \begin{cases} -\frac{1}{z_{ij}} & (i, j) \in \mathcal{E}, \\ \sum\limits_{k \in \mathrm{N}(G,i)} \frac{1}{z_{ik}} & i = j, \\ 0 & \text{otherwise.} \end{cases}$$

Note that the equations above depend only on the network structure $(G, \mathbf{z})$. We call a pair of vectors $(\mathbf{v}, \mathbf{s})$ an *admissible* pair of the structure $(G, \mathbf{z})$ if it satisfies the corresponding power-flow equations. The set of all admissible pairs is denoted by by $\mathrm{PFE}(G, \mathbf{z})$, that is:

$$\begin{aligned} \mathrm{PFE}(G, \mathbf{z}) \equiv \Big\{ (\mathbf{v}, \mathbf{s}) \in \mathbb{C}^{\mathcal{V}} \times \mathbb{C}^{\mathcal{V}} \,:\, \exists\, \mathbf{I} \in \mathbb{C}^{\mathcal{E}} \\ z_{ij} I_{ij} = v_i - v_j, \qquad (i, j) \in \mathcal{E}, \\ I_{ij} = -I_{ji}, \qquad (i, j) \in \mathcal{E}, \\ s_i = \sum_{j \in N(G,i)} v_i I_{ij}^*, \qquad i \in \mathcal{V} \Big\}. \end{aligned}$$

1.3.3 Edge power loss

Power is generated by the generators, but is consumed by both the desired consumers and by the transmission lines themselves. In most cases we desire to ensure that the transmission lines consume a very small amount of power for two practical reasons: we pay for the generation of this power, but our consumers do not enjoy it; if a transmission line consumes to much power it may overheat, and become damaged.

The *edge power loss* is the amount of power consumed by the transmission line which is modeled by a specific edge.

Definition (Edge power loss)**.** Given $(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(G, \mathbf{z})$, the power loss of the edge $(i, j) \in \mathcal{E}$ is defined by

$$\ell_{ji}(\mathbf{v}, \mathbf{s}) = \ell_{ij}(\mathbf{v}, \mathbf{s}) = \begin{cases} \frac{|v_i - v_j|^2}{z_{ij}^*} & z_{ij} \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

The origin of the definition above lies in the laws of physics, but can also be justified by pure mathematics. The apparent powers s_i represent how much power is produced or consumed at every node: positive values of $\mathrm{re}(s_i)$ or $\mathrm{im}(s_i)$ mean that active or reactive power is produced, while negative values mean that power is consumed. Thus, the difference between the total produced and consumed power at all network nodes is $\sum_{i \in \mathcal{V}} s_i$, which is exactly the power that remains to be consumed by the transmission lines. The following lemma coincides with the conclusion above, and suggests why the definition of ℓ_{ij} indeed makes sense.

Lemma 1 (Power preservation)**.** *Let $(G, \mathbf{z})$ be a network structure with $G = (\mathcal{V}, \mathcal{E})$, and let $(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(G, \mathbf{z})$. Then,*

$$\sum_{i \in \mathcal{V}} s_i = \sum_{\substack{(i,j) \in \mathcal{E} \\ i<j}} \ell_{ij}(\mathbf{v}, \mathbf{s})$$

Proof. Let $G, \mathbf{z}, \mathbf{v}$, and $\mathbf{s}$ satisfy the assumptions of the lemma. From (1.3) and (1.2) we obtain

$$\begin{aligned} \sum_{i \in \mathcal{V}} s_i &= \sum_{(i,j) \in \mathcal{E}} v_i I_{ij}^* \\ &= \sum_{\substack{(i,j) \in \mathcal{E} \\ i<j}} (v_i I_{ij}^* + v_j I_{ji}^*) \\ &= \sum_{\substack{(i,j) \in \mathcal{E} \\ i<j}} (v_i - v_j) I_{ij}^* \end{aligned}$$

If $z_{ij} = 0$ for some $(i, j) \in \mathcal{E}$, then from (1.1) we conclude that $v_i - v_j = 0$. If $z_{ij} \neq 0$, then from (1.1) we obtain

$$I_{ij} = \frac{v_i - v_j}{z_{ij}}.$$

Thus,

$$\begin{aligned} \sum_{i \in \mathcal{V}} s_i &= \sum_{\substack{(i,j) \in \mathcal{E} \\ i<j \\ z_{ij} \neq 0}} (v_i - v_j) \frac{(v_i - v_j)^*}{z_{ij}^*} \\ &= \sum_{\substack{(i,j) \in \mathcal{E} \\ i<j \\ z_{ij} \neq 0}} \frac{|v_i - v_j|^2}{z_{ij}^*} \\ &= \sum_{\substack{(i,j) \in \mathcal{E} \\ i<j}} \ell_{ij}(\mathbf{v}, \mathbf{s}) \end{aligned}$$

□

1.3.4 The Power Flow and the Optimal Power Flow problems

The Optimal Power Flow (OPF) problem on the network $(G, \mathbf{z}, F)$ is concerned with finding an admissible pair of the underlying structure which minimizes the network's cost, namely:

$$\begin{aligned} \min_{\mathbf{v},\mathbf{s}} \quad & F(\mathbf{v},\mathbf{s}), \\ \text{s.t.} \quad & (\mathbf{v},\mathbf{s}) \in \mathrm{PFE}(G,\mathbf{z}). \end{aligned} \tag{OPF}$$

The related Power Flow (PF) problem is obtained by defining F to be the indicator function of some set of constraints. In that case, the solution of the problem is just a vector satisfying all of the problem's constraints, namely, an element of the set

$$\{(\mathbf{v},\mathbf{s}) \in \mathrm{PFE}(G,\mathbf{z}) : (\mathbf{v},\mathbf{s}) \in \mathrm{dom}(F)\}.$$

It is typical, with both problems, that $\mathrm{dom}(F)$ imposes constraints for the quantities $|v_j|$ and s_j associated with each node j, meaning that F has the form

$$F(\mathbf{v},\mathbf{s}) = \hat{F}(\mathbf{v},\mathbf{s}) + \sum_{j\in\mathcal{V}} \delta_{\mathcal{C}_j}(|v_j|, s_j),$$

where $\mathcal{C}_j$ is the constraint associated with node j. The following are the common types of nodal constraints:

PQ constraint. If

$$\mathcal{C}_j = \mathrm{PQ}(\underline{u}_j, \overline{u}_k, \hat{s}_j) \equiv [\underline{u}_j, \overline{u}_j] \times \{\hat{s}_j\},$$

where $\hat{s}_j \in \mathbb{C}$, and $0 \le \underline{u}_j < \overline{u}_j \le +\infty$ are given constants, then $\mathcal{C}_j$ is called a *PQ constraint*. That is, with PQ constraints we exactly specify the apparent power, while constraining the voltage absolute value to lie in a given interval, which may be unbounded.

PV constraint. If

$$\mathcal{C}_j = \mathrm{PV}(\hat{u}_j, \hat{p}_j, \underline{q}_j, \overline{q}_j) \equiv \{\hat{u}_j\} \times \left(\{\hat{p}_j\} + \mathbf{i}[\underline{q}_j, \overline{q}_j]\right)$$

where $\hat{u}_j \in \mathbb{R}_+$, $\hat{p}_j \in \mathbb{R}$, and $-\infty \le \underline{q}_j < \overline{q}_j \le +\infty$ are given constants, then $\mathcal{C}_j$ is called a *PV constraint*. That is, with PV constraints we exactly specify the voltage absolute value and active power, while constraining the reactive power to lie in a given interval, which may be unbounded.

PV constraints typically model generators with a known power generation policy, which consists of the quantities which can be controlled: the voltage absolute value, and the amount of generated active power.

PQ constraints typically model consumers with a known power demand. Such constraints model a typical consumer (e.g. an apartment), which have a prescribed amount of both active and reactive power consumed, and whose electrical devices function properly when the voltage absolute value lies in some small interval around a specified value. In most cases, the interval is the same for almost all consumers due to national or international standards.

In typical instances of the Power Flow problem consumers have PQ constraints, while all generators, except for one, have PV constraints. The remaining generator, called the *slack node*, has a constraint which only specifies the voltage, while its apparent power remains free or constrained to an interval. However, looking at the power preservation lemma (Lemma 1), we conclude that the power generated by this generator is, in fact, not free at all. It must be exactly the amount of power required to sustain the balance between the generated and the consumed power, and hence its name.

In typical instances of the Optimal Power Flow problem, consumers have PQ constraints, while the quantities associated with the generators are free, or constrained to lie in some box or polyhedron. Indeed, when we solve the Optimal Power Flow problem, we are interested in finding the optimal power generation policy for every generator in the network, such that all of the network's constraints are satisfied.

Given our knowledge about the role of the edge power loss, a less common but yet useful type of constraint limits the amount of power consumed by every edge. These constraints are usually either of the form

$$|\ell_{ij}(\mathbf{v},\mathbf{s})| \le \overline{\ell}_{ij}$$

or

$$\mathrm{re}(\ell_{ij}(\mathbf{v},\mathbf{s})) \le \overline{\ell}_{ij},$$

and are typically called *thermal constraints*. When both nodal and thermal constraints are embedded into the function F, it may have the form

$$F(\mathbf{v},\mathbf{s}) = \hat{F}(\mathbf{v},\mathbf{s}) + \sum_{i\in\mathcal{V}} \delta_{C_j}(|v_j|, s_j) + \sum_{(i,j)\in\mathcal{E}} \delta_{[|\ell_{ij}|\le\overline{\ell}_{ij}]}(\mathbf{v},\mathbf{s})$$

The widely used MATPOWER software package [72] for solving both the Power Flow and the Optimal Power Flow problems supports all of the above constraint types.

1.3.5 Per unit system

The voltage absolute value constraints typically represent a small interval around some desired value. For convenience, the physical units of $\mathbf{v}$ are typically rescaled by a change of variables

$$\mathbf{v}' = \mathrm{diag}(\boldsymbol{\alpha})\mathbf{v}$$

such that the desired voltage absolute at each node is 1, and voltage constraints define an interval around 1. Similarly, the physical units of $\mathbf{s}$ are typically rescaled so that $\mathbf{s}$ is composed of small numbers which are easy to reason about from an engineering perspective. This change of variables can be viewed as a rescaling of the problem's data, including edge impedances and the other constants which participate in the constraints. For example, a PQ constraint might be of the form

$$s_j = 50 + 3\mathbf{i}, \quad 0.95 \le |v_j| \le 1.05.$$

Such a rescaled unit system is called the *per unit system*, and is pervasively used in power-flow analysis. The problem's data is usually given in the rescaled form, and the scaling coefficients are provided separately.

Chapter 2

Literature Review

In general, both the Power Flow and the Optimal Power Flow problems are NP-Hard, as was proved in [43]. In fact, it was shown that even feasibility checking under a very restrictive set of assumptions, which include tree topology, is still NP-Hard. This is, of course, not a surprise. The set PFE($G, \mathbf{z}$) is defined by a quadratic system of equations, and thus non-convex. The difficulty of these problems lead to the following approaches:

Heuristics Methods which attempt to solve a large class of problems with low computational cost, and backed by extensive numerical experiments which demonstrate that they indeed do so for a large percentage of practical problem instances. However, these methods are not backed by a theoretical analysis which proves that they indeed solve their intended problem, and they do occasionally fail on some problem instances. This category mostly comprises generic iterative methods, which hopefully converge to the desired solution.

Restricted reliable methods Efficient methods which solve the problems, or converge to the desired solution as the computational effort increases (e.g. the number of iteration tends to infinity). However, they are mathematically proven to work for problem instances which belong to some restricted class. The majority of these methods are convex relaxations, which are proven to be tight for some classes of networks.

Generic reliable methods Inefficient methods which attempt to solve some large and general classes of these problems. For example, a branch-and-bound method for solving Optimal Power Flow problems on a large class of networks.

Most existing methods, many of which are referenced below, belong to the first and the third families, while a small number of methods belong the second category of restricted and reliable methods, most of them are aimed at radial networks. The methods we present in this work also belong to this category, and aim at additional classes of networks with tree topology, which to the best of our knowledge are not covered by prior work. Since the focus of our work is on radial networks, we separate the discussion into methods useful for networks with arbitrary topology, and methods specialized for networks with a tree topology.

2.1 The Power Flow problem

Recall, that in this problem we are concerned with finding, given a power network $(G, \mathbf{z}, F)$ with $G = (\mathcal{V}, \mathcal{E})$, an element of its feasible set

$$\mathcal{F}(G, \mathbf{z}, F) = \{(\mathbf{v}, \mathbf{s}) \in \text{PFE}(G, \mathbf{z}) : (\mathbf{v}, \mathbf{s}) \in \text{dom}(F)\},$$

and that F is just the indicator of some set. Despite the fact that the power flow problem is not the center of this work, we conduct a review on the related litarature for two reasons. First, the simplicity of the problem is a good opportunity to make a detailed introduction into various concepts, properties, and terms related to power system analysis, without dealing with a complex problem at the same time. Second, one of our

algorithms can be used to efficiently and reliably solve the power flow problem on networks with tree topology.

In the detailed review below we introduce the classical power flow problem in which the equality constraints comprise a system of equations with no degrees of freedom. Then, we review several numerical methods for solving that system of equations, which include a variant of the Gauss-Seidel algorithm ([33]), Newton-Raphson method ([61]) along with several enhancements for exploiting sparsity ([62], [35], [69], [44]), approximation with a system of linear constraints ([34], p. 353), and specialized methods for networks with tree topology ([37], [19], [16]). Papers on the power-flow problem usually assume that $\mathrm{dom}(F)$ is defined by specific equality and inequality constraints, but do not deal with the challenge of enforcing the inequality constraints, and when they do, they provide several ad-hoc procedures for 'correcting' the iterates of the numerical methods. Readers who are familiar with these results are invited to skip this section and continue to Section 2.2.

In practical problems, the edge impedances are assumed to be non-zero, and thus the system of power-flow equations have the form (1.4). It is illuminating to look at the system of equations in terms of the *real* variables involved. We assume each apparent power s_j is represented in rectangular form, namely, the real variables $\mathrm{re}(s_j)$ and $\mathrm{im}(s_j)$. For the voltages v_j, we consider two cases - rectangular form, and polar form. When the rectangular form is considered, the system of power-flow equations (1.4) comprise a system of $2|\mathcal{V}|$ real quadratic equations in real $4|\mathcal{V}|$ variables. When the polar form is considered, denoting $v_j = u_j e^{\mathbf{i}\theta_j}$, the system is

$$s_j = \sum_{j \in \mathcal{V}} Y_{ij}^* u_i u_j e^{\mathbf{i}(\theta_i - \theta_j)},$$

leading to the conclusion that membership of $(\mathbf{v}, \mathbf{s})$ in $\mathrm{PFE}(G, \mathbf{z})$ does not depend on the actual voltage angles, but rather on their differences. In particular, the voltage angle of a node of our choice may be chosen arbitrarily, and the voltage angles at the rest of the network are relative to the voltage angle of that *reference node*.

With the conclusions above in mind, we look at the classical case, where all consumers have PQ constraints, all but one generators have PV constraints, and the remaining generator, termed the *slack node*, has prescribed voltage absolute value and bounds imposed on its active and reactive powers. In the classical setup, it is also typical to chose the slack as the reference node with zero angle, that is, its voltage is a real number. By denoting the set of consumers by $\mathcal{PQ}$, the set of non-slack generators by $\mathcal{PV}$, and observing that $\{r\} = \mathcal{V} \setminus (\mathcal{PQ} \cup \mathcal{PV})$ is a singleton containing only the slack node r, we can explicitly write the power-flow problem as the following system of constraints

$$\begin{aligned}
&(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(G, \mathbf{z}), \\
&v_r = \hat{u}_r \\
&\underline{p}_r \le \mathrm{re}(s_r) \le \overline{p}_r, \\
&\underline{q}_r \le \mathrm{im}(s_r) \le \overline{q}_r, \\
&s_j = \hat{s}_j, && j \in \mathcal{PQ}, \\
&\underline{u}_j \le |v_j| \le \overline{u}_j, && j \in \mathcal{PQ}, \\
&|v_j| = \hat{u}_j, && j \in \mathcal{PV}, \\
&\mathrm{re}(s_j) = \hat{p}_j, && j \in \mathcal{PV}, \\
&\underline{q}_j \le \mathrm{im}(s_j) \le \overline{q}_j, && j \in \mathcal{PV}.
\end{aligned} \tag{PF-C}$$

Assuming rectangular representation of v_j, the equality constraints in (PF-C), including power flow equations describing $(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(G, \mathbf{z})$, comprise $4|\mathcal{V}|$ polynomial equations of degree at most 2 in $4|\mathcal{V}|$ variables. Thus, assuming that the system is well behaved, it has a finite number of solutions. In theory, we could enumerate the solutions of the equation system (see [17] for a survey of such methods), and among them select one which satisfies the inequality constraints. However, according to the sharp bound of Bézout's theorem, this approach is intractable since the number of solutions is exponential in $|\mathcal{V}|$, and therefore iterative numerical methods are used instead. Because of the empirical effectiveness of state of the art numerical methods in solving the above classical power-flow problem, it is considered as 'solved' in the electrical engineering community. However, from a mathematical perspective, these methods still

remain heuristics. To the best of our knowledge, none of the iterative methods used to solve such problems have been endowed with a proof of convergence to a solution of the constraint system, and it is possible to find problem instances on which they fail.

To employ an iterative method, the constraints are first reformulated via algebraic manipulations into a more convenient system of equations and inequality constraints. Then, the special structure of the inequality constraints is used to compute an initial guess, which is fed to an iterative method which generates a sequence that, hopefully, converges to our desired solution. The various approaches we discuss below differ in the reformulation, in the method for computing the initial guess, the iterative method employed, and the constraint enforcement strategy.

Many existing works concentrate on the challenge of solving the system of equations, and either ignore or only superficially address the inequality constraint enforcement challenge. Constraint enforcement is usually done by an ad-hoc procedure to do one of the following: 'correct' quantities which go out of range as iterations progress; modify the problem data; ignore the inequality constraints and assume that the iterative equation solver will converge to a desired solution. A common voltage initialization strategy, known as 'flat start' is initializing the voltages of all PQ constrained nodes to 1, and all voltage angles to 0. Initialization of unspecified powers, such as the reactive power of a PV constrained node, is computed by substituting the initial voltages into the power-flow equations (1.4).

2.1.1 Methods for an arbitrary topology

One of the earliest methods for solving (PF-C), which was first described in [33], is to apply a variant of the well-known Gauss-Seidel method for linear equations to the system of power-flow equations. First, the equality constraints imposed by PQ, PV, and slack constraints are substituted into the power-flow equations. The remaining *m real* decision variables, which are obtained by either rectangular or polar complex representation, are denoted by $\mathbf{x} \in \mathbb{R}^m$, and the power-flow equations are represented in the form

$$\mathbf{x} = g(\mathbf{x})$$

by algebraic manipulations. Then, a Gauss-Seidel kind procedure for numerically computing a fixed-point of g can be applied: generate the sequence $\{\mathbf{x}^{(k)}\}_{k=1}^{\infty}$ by the following rule for computing $\mathbf{x}^{(k+1)}$ from $\mathbf{x}^{(k)}$:

$$\begin{aligned}&\text{for } i = 1 \text{ to } m\text{:}\\ &\quad x_i^{(k+1)} = g_i(x_1^{(k+1)}, \ldots, x_{i-1}^{(k+1)}, x_i^{(k)}, \ldots, x_m^{(k)}).\end{aligned}$$

Experimentally, in [33] it is reported that their reformulation into the form $\mathbf{x} = g(\mathbf{x})$ leads to an algorithm which converges on many practical problems. It is also reported that the method above converges slowly despite attempts to use some convergence acceleration strategies, and diverges when the matrix $\mathbf{Y}$ is ill-conditioned in some sense. Despite these drawbacks, the method is well suited for running in memory-constrained environments - at any given moment it operates on one equation, and thus requires only $O(|\mathcal{V}|)$ storage.

A natural approach to overcoming the drawbacks of the Gauss-Seidel method, for example in [61], is applying a multidimensional variant of the Newton-Raphson method for solving the system of power-flow equations. Typically, apparent powers are represented in rectangular form, while voltages are in polar form. After substituting the known quantities imposed by the PQ, PV, and slack constraints into the power-flow equations, the system re-arranged into the form $h(\mathbf{x}) = 0$, where the real vector $\mathbf{x}$ consists of the voltage absolute value and angle at each node $i \in \mathcal{PQ}$, and the voltage angle for each $i \in \mathcal{PV}$. Taking any solution $\mathbf{x}^\dagger$ of the equation system, for any $\mathbf{x}$ the first-order expansion of h yields

$$0 = h(\mathbf{x}^\dagger) \approx h(\mathbf{x}) + J_h(\mathbf{x})[\mathbf{x}^\dagger - \mathbf{x}],$$

where $J_h(\cdot)$ is the Jacobian matrix of h, whose rows are the gradients of the individual components of $h(\cdot)$. The above relationship is exploited to derive the recursive relation

$$0 = h(\mathbf{x}^{(k)}) + J_h(\mathbf{x}^{(k)})[\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}],$$

which under the assumption that $J_h(\cdot)$ is non-singular can be written as

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - [J_h(\mathbf{x}^{(k)})]^{-1} h(\mathbf{x}^{(k)}).$$

It is reported in [61] that without any intervention, e.g. attempts to enforce the inequality constraints, the method converges within 5 or 6 iterations to a solution from a 'flat start' initial point.

A major drawback of Newton's method is the need to solve a linear system of equations in each iterations, which requires $O(|\mathcal{V}|^3)$ work. However, on practical problems the Jacobians are usually sparse, and their sparsity patterns can be exploited to improve the matrix factorization procedures, and substantially accelerate the running times. For example, in [62] a heuristic for choosing node numbering such that the resulting Jacobian's LU decomposition remains sparse is presented. Additional methods for exploiting power network structure to ensure fast linear equation solving when applying Newton's method can be found in [35, 69, 44].

Another well-known and significantly faster modification of Newton's method is called the Fast Decoupled Load Flow, was introduced in [59]. Suppose that $\mathbf{x} = (\mathbf{u}, \boldsymbol{\theta})$, where $\mathbf{u}$ is the vector of node voltage absolute values, and $\boldsymbol{\theta}$ is the vector of voltage angles. Then, the Jacobian J_h can be written as

$$J_h(\mathbf{u}^{(k)}, \boldsymbol{\theta}^{(k)}) = \begin{bmatrix} J_{\mathbf{uu}}(\mathbf{u}^{(k)}, \boldsymbol{\theta}^{(k)}) & J_{\mathbf{u}\boldsymbol{\theta}}(\mathbf{u}^{(k)}, \boldsymbol{\theta}^{(k)}) \\ J_{\boldsymbol{\theta}\mathbf{u}}(\mathbf{u}^{(k)}, \boldsymbol{\theta}^{(k)}) & J_{\boldsymbol{\theta}\boldsymbol{\theta}}(\mathbf{u}^{(k)}, \boldsymbol{\theta}^{(k)}) \end{bmatrix}$$

The work points out that in practical problems, the 'coupling' terms $J_{\mathbf{u}\boldsymbol{\theta}}$ and $J_{\boldsymbol{\theta}\mathbf{u}}$, which contain the derivatives of the $\mathbf{u}$ components of h w.r.t $\boldsymbol{\theta}$, and vice versa, are close to zero. Thus, the Jacobian is first approximated by

$$\begin{bmatrix} J_{\mathbf{uu}}(\mathbf{u}^{(k)}, \boldsymbol{\theta}^{(k)}) & \mathbf{0} \\ \mathbf{0} & J_{\boldsymbol{\theta}\boldsymbol{\theta}}(\mathbf{u}^{(k)}, \boldsymbol{\theta}^{(k)}) \end{bmatrix}$$

Then, several physical properties of the system are exploited to discard various terms in the above matrices, and finally elementary algebraic manipulations lead to a system of equations whose matrix is constant, and can be factorized at the first iteration only. One such physical property is, for example, the observation that the difference between adjacent voltage angles as observed in practice is close to zero, and thus the cosine and sine of this difference $\theta_i - \theta_j$, which participates in the matrix above, can be approximated by 1 or $\theta_i - \theta_j$, respectively.

The method is reported to converge slower then Newton's method, but the substantial reduction of the iteration cost and its effectiveness when solving practical problems made it popular. Of course, the algorithm is not suitable for problems for which the simplifying physical and mathematical assumptions do not hold, as pointed out, for example, in [6]. As a result, an improved version of the algorithm was introduced in [63], which requires weaker approximating assumptions. The empirical success of the method led to a theoretical analysis, for example in [53], which strives to provide an alternative mathematical justification for the remarkable convergence properties observed in practice.

An additional approach is to use a continuation method. The idea is to reformulate a system of equations $h(\mathbf{x}) = 0$ by inserting an additional parameter $\lambda \in [0, 1]$ and obtaining the system $H(\mathbf{x}, \lambda) = 0$, such that by substituting $\lambda = 0$ the solution for $\mathbf{x}$ of is 'easy', for example it is trivially satisfied by the 'flat start' initialization, and such that $H(\mathbf{x}, 1) = h(\mathbf{x})$. Then, a sequence $\lambda_1 = 0 < \lambda_2 < \cdots < \lambda_N = 1$ is chosen, and a solution of $H(\mathbf{x}, \lambda_k) = 0$ is used as an initial point for some iterative scheme to compute a solution of $H(\mathbf{x}, \lambda_{k+1}) = 0$. The idea is based on the intuition that a solution for λ_k is close to the desired solution for λ_{k+1}, and thus is a good initial point for the iterative algorithm. See [3, 49, 70, 38] for examples of continuation methods for the Power Flow problem.

The problem can also be approximated by a system of linear constraints, by again exploiting the properties of the physical quantities which are observed in practice. Such approximations are known as 'DC power flow' or 'DC load flow' formulation, and are described in several textbooks, such as [34, 55, 20], and extended in a recent paper [60]. These approaches will be discussed in the next section, since they are useful for the optimal power flow problem as well.

A wide range of commercial software products for power system analysis include solvers for the power-flow problem. See [7] for a comprehensive review. Well-known open source packages include MATPOWER[72], PSAT[50], and PSASP[71]. Most software packages use Gauss-Seidel, Newton-Raphson, or Fast Decoupled Load Flow as their default.

2.1.2 Methods for a tree topology

On the large scale, e.g. a state or a large region, power networks typically have cycles to ensure redundancy. Such networks are termed *transmission networks*, since their objective is to transmit power from

the generators in the power stations to the consumers. On the small scale, e.g. a small neighborhood, networks typically do have a tree topology, and are termed *distribution networks*. On the one hand, the special structure imposed by the tree topology can be exploited to construct specialized algorithms. On the other hand, the problem data for such networks typically make classical power-flow algorithms diverge, as pointed out by [22]. Thus, specialized algorithms for distribution networks $(T, \mathbf{z}, F)$ with tree topology $T = (\mathcal{V}, \mathcal{E})$ were developed.

The forward-backward sweep methods are iterative methods in which each iteration propagates information in using traversals (sweeps) of the topology tree. The slack node is designated as the root of the tree, each edge connecting node j with its parent is augmented additional vector of auxiliary variables $\mathbf{t}_j$, and the power-flow equations are reformulated into a system of equations of the following form:

$$\begin{aligned} \phi_i(s_i, \mathbf{t}_i, \mathbf{t}_{j_1}, \dots, \mathbf{t}_{j_{n_i}}) = 0, \qquad & i \in \mathcal{V},\ \{j_1, \dots, j_{n_i}\} = \mathrm{ch}_T(i), \\ \psi_i(|v_i|, |v_j|, \mathbf{t}_j) = 0, \qquad & i \in \mathcal{V},\ j \in \mathrm{ch}_T(i), \\ \omega_i(v_i, v_j, \mathbf{t}_j) = 0, \qquad & i \in \mathcal{V},\ j \in \mathrm{ch}_T(i). \end{aligned}$$

Then, each iterate is computed from the previous one by scanning the tree twice, and solving the system above w.r.t some of the variables, while keeping the others constant. In fact, the algorithms we present in this work are also based on a similar transformation, e.g. Lemma 3, but are not iterative in nature. The earliest forward-backward method described in [8] is one such example: the 'forward' sweep advances from the leaves toward the root, and solves equations involving ϕ_i, ψ_i for $|v_i|$ and $\mathbf{t}_i$ by substituting the previously computed quantities in place of the variables associated with the children; then, the 'backward' sweep advances from the root to the leaves and solves equations involving ω_i for $\arg(v_j)$ by substituting $\mathbf{t}_j$ from 'forward' sweep and v_i which was computed when the parent of i was scanned. A variety of such methods, which differ in the reformulation used and the variables solved for in each sweep, is reviewed in [22].

Another family of methods are convex relaxations. Various reformulations of the power-flow equations into a system of quadratic equations are relaxed to either SOCP or SDP constraints, and the conditions ensuring that the relaxation is tight for tree networks are analyzed. Since the same convex relaxations are used for the Optimal Power Flow problem, these methods are reviewed in the next section.

2.2 The Optimal Power Flow problem

In the classical setup of the OPF problem, consumers have PQ constraints while generators have box constraints. Moreover, the objective function is typically separable along the nodes, depends on the voltage absolute value and apparent power associated with each node, and has some 'simple' form, e.g. a polynomial or piecewise linear function. Mathematically, the objective is of the form

$$F(\mathbf{v}, \mathbf{s}) = \sum_{j \in \mathcal{V}} \left[\delta_{C_j}(|v_j|, s_j) + \phi_j(|v_j|, s_j) \right],$$

where C_j is the constraint associated with node j, which for a consumer j is

$$C_j = \mathrm{PQ}(\underline{u}_j, \overline{u}_j, \hat{s}_j),$$

while for a generator j is

$$C_j = \mathrm{BX}(\underline{u}_j, \overline{u}_j, \underline{p}_j, \overline{p}_j, \underline{q}_j, \overline{q}_j) \equiv [\underline{u}_j, \overline{u}_j] \times \left([\underline{p}_j, \overline{p}_j] + \mathbf{i}[\underline{q}_j, \overline{q}_j] \right),$$

and ϕ_j is the cost associated with node j which has some 'simple' form. Denoting the set of generators by $\mathcal{G} \subseteq \mathcal{V}$ and the set of consumers by $\mathcal{PQ} = \mathcal{V} \setminus \mathcal{G}$, the classical OPF problem can be explicitly written as

$$\begin{aligned} \min_{\mathbf{v},\mathbf{s}\in\mathbb{C}^{\mathcal{V}}} \quad & \sum_{j\in\mathcal{V}} \phi_j(|v_j|, s_j), \\ \text{s.t.} \quad & (\mathbf{v},\mathbf{s}) \in \mathrm{PFE}(G,\mathbf{z}), \\ & \underline{u}_j \le |v_j| \le \overline{u}_j, && j \in \mathcal{V} \\ & s_j = \hat{s}_j, && j \in \mathcal{PQ} \\ & \underline{p}_j \le \mathrm{re}(s_j) \le \overline{p}_j, && j \in \mathcal{G} \\ & \underline{q}_j \le \mathrm{im}(s_j) \le \overline{q}_j. && j \in \mathcal{G} \end{aligned} \tag{OPF-C}$$

The first example of ϕ_j, which is commonly used, is minimizing the total cost of generation. Recall, that $\mathrm{re}(s_j)$ is the amount of power produced (or consumed) by node j which is manifested in the external world outside the power network. With generators, the cost of generation, such as the amount of fuel required, is directly dependent on this quantity. Thus, when minimizing the total cost of generation, for each $j \in \mathcal{PQ}$ we assign $\phi_j \equiv 0$, while for each $j \in \mathcal{G}$ we assign $\phi_j(|v_j|, s_j) = c_j(\mathrm{re}(s_j))$, where $c_j(t)$ represents the cost of generating t units of power at generator j. In fact, this formulation is so common, that it is often referred to as 'the' Optimal Power Flow problem.

Another example is attempting to minimize the deviation of each voltage absolute value from its prescribed value at each consumer node. Recall, that $[\underline{u}_j, \overline{u}_j]$ is a small interval around the prescribed voltage absolute value which is required for the consumer's devices to function properly, and that the voltages are rescaled such that the prescribed magnitude is 1. In this case, for each $j \in \mathcal{G}$ we set $\phi_j \equiv 0$, while for $j \in \mathcal{PQ}$ we set $\phi_j(u, s) = |u - 1|$.

Additional variants of the classical optimal power-flow problem (OPF-C) exist in the literature, which include additional useful objective functions, and additional constraint types, such as thermal constraints we introduced in Section 1.3.4, constraints on voltage angle difference of adjacent nodes in the network, or the use of binary variables to decide, for example, wheather we should or should not use a certain generator. A recent review of various common formulations, including solution methods, is available in [10, 27].

2.2.1 Methods for arbitrary topology

A widely used approach, including commercial software, is an approximation of the non-linear constraints by linear constraints, exploiting various physical properties of the voltages and angles in real-world networks. This approach is called 'DC power flow' in the literature. A thorough review of these models was conducted in [60, 68], including experimental testing of the approximation accuracy. In these approximations, powers are represented in rectangular form, while voltages in polar form. This approach is used mainly for problems with large networks, since linear constraints are well handled by modern state of the art linear programming solvers. However, they are not useful when high accuracy is required.

Here, we provide one example of such transformation, which is introduced in textbooks on power system analysis, such as [34, 55, 20]. Recall that in polar voltage form $v_i = u_i e^{\mathbf{i}\theta_i}$, the power flow equations for each $i \in \mathcal{V}$ are

$$s_i = \sum_{j\in\mathcal{V}} u_i u_j e^{\mathbf{i}(\theta_i-\theta_j)} Y_{ij}^* = u_i^2 Y_{ii}^* + \sum_{\substack{j\in\mathcal{V}\\ j\neq i}} u_i u_j (\cos(\theta_i-\theta_j) + \mathbf{i}\sin(\theta_i-\theta_j)) Y_{ij}^*.$$

By decomposing Y into $A \equiv \mathrm{re}(Y)$ and $B \equiv \mathrm{im}(Y)$, we obtain for each $i \in \mathcal{V}$

$$\begin{aligned} \mathrm{re}(s_i) &= u_i^2 A_{ii} + \sum_{\substack{j\in\mathcal{V}\\ j\neq i}} u_i u_j (A_{ij}\cos(\theta_i-\theta_j) + B_{ij}\sin(\theta_i-\theta_j)), \\ \mathrm{im}(s_i) &= -u_i^2 B_{ii} + \sum_{\substack{j\in\mathcal{V}\\ j\neq i}} u_i u_j (A_{ij}\sin(\theta_i-\theta_j) - B_{ij}\cos(\theta_i-\theta_j)). \end{aligned}$$

The first simplifying assumption is that in practical networks, edge impedances are 'almost imaginary', that is, the ratio $|\mathrm{re}(z_{ij})|/|\mathrm{im}(z_{ij})|$ is small for each edge, and thus the components of Y are 'almost

imaginary' as well. This assumption is mathematically realized by the approximation $A_{ij} \approx 0$, and the approximate power flow equations become

$$\begin{aligned}\mathrm{re}(s_i) &\approx \sum_{\substack{j\in\mathcal{V}\\ j\neq i}} u_i u_j B_{ij} \sin(\theta_i - \theta_j),\\ \mathrm{im}(s_i) &\approx -u_i^2 B_{ii} + \sum_{\substack{j\in\mathcal{V}\\ j\neq i}} u_i u_j B_{ij} \cos(\theta_i - \theta_j).\end{aligned}$$

The second simplifying assumption is that in practice, the angle differences $\theta_i - \theta_j$ are very small. This assumption is mathematically realized by the approximations $\cos(\theta_i - \theta_j) \approx 1$ and $\sin(\theta_i - \theta_j) \approx \theta_i - \theta_j$, resulting in

$$\begin{aligned}\mathrm{re}(s_i) &\approx \sum_{\substack{j\in\mathcal{V}\\ j\neq i}} u_i u_j B_{ij} (\theta_i - \theta_j),\\ \mathrm{im}(s_i) &\approx -u_i^2 B_{ii} + \sum_{\substack{j\in\mathcal{V}\\ j\neq i}} u_i u_j B_{ij}.\end{aligned}$$

The third and final assumption is that in practice for each node we often have $u_i \approx 1$, and the effect of $\theta_i - \theta_j$ dominates the equations with $\mathrm{re}(s_i)$, meaning that

$$\begin{aligned}\mathrm{re}(s_i) &\approx \sum_{\substack{j\in\mathcal{V}\\ j\neq i}} B_{ij} (\theta_i - \theta_j),\\ \mathrm{im}(s_i) &\approx -u_i^2 B_{ii} + \sum_{\substack{j\in\mathcal{V}\\ j\neq i}} u_i u_j B_{ij}.\end{aligned}$$

In optimal power flow problems whose decision variables of interest do *not* include $\mathrm{im}(s_i)$ and u_i, including the classical (OPF-C) variant whose objective function depends on $\mathrm{re}(s_i)$ only, this approximation produces a linear system of equations. When these quantities are of interest, first order Taylor approximations of u_i^2 and $u_i u_j$ around $u_i \approx 1$ can be used, such as in [32]. As pointed out by [60], several error bounds for such linear approximations were derived, but they are of little practical value, and the accuracy was mainly studied experimentally. Moreover, additional linear approximations and their uses in power system analysis are available in [60, 68] and references therein.

Instead of approximating the problem, another approach is to use a standard non-linear local optimization solver, such as the primal dual interior-point solver described in [66] and used in MATPOWER, the KNITRO [13] package, or the IPOPT [65] package. The real decision variables used by these methods may either come from the rectangular or the polar representation. These methods produce accurate results *when* they converge to an optimal solution, and are useful mainly for small to medium sized problems, because of the need to invert matrices whose size is proportional to the number of nodes. A review of such methods, including their numerical performance on practical problems, was done in [39].

For example, an interior point method, such as the one described in [66], is constructed from a reformulation of the optimal power flow problem as

$$\begin{aligned}\min_{\mathbf{x},\mathbf{z}} \quad & f(\mathbf{x}),\\ \text{s.t.} \quad & g(\mathbf{x}) = 0,\\ & h(\mathbf{x}) + \mathbf{z} = 0,\\ & \mathbf{z} \geq 0,\end{aligned}$$

where $\mathbf{x}$ is the real decision variables vector, and $\mathbf{z}$ is the slack variables vector associated with the constraints $h(\mathbf{x}) \leq 0$. Then, a variant of Newton's non-linear equation solving method is applied to the

following system of equations which approximates the equations implied by the KKT conditions:

$$\begin{aligned}
&\nabla f(\mathbf{x}) + \sum_i \lambda_i \nabla g_i(\mathbf{x}) + \sum_j \mu_j \nabla h_j(\mathbf{x}) = 0, && \leftarrow \nabla_{\mathbf{x}} L = 0,\\
&\boldsymbol{\mu} - \boldsymbol{\eta} = 0, && \leftarrow \nabla_{\mathbf{z}} L = 0,\\
&g(\mathbf{x}) = 0, && \leftarrow \text{primal feasibility}\\
&h(\mathbf{x}) + \mathbf{z} = 0, && \leftarrow \text{dual feasibility}\\
&\eta_i z_i = \frac{1}{t}. && \leftarrow \text{approximate complementarity}
\end{aligned}$$

Larger values of t better approximate the exact complementarity conditions $\eta_i z_i = 0$. The value of t is increased every one or more Newton iterations. One variant of this method is described in detail in the paper [66], which presents the interior point algorithm employed by the MIPS solver, which is the default in MATPOWER.

Local optimization methods occasionally fail to converge to an optimal solution, as demonstrated by the numerical experiments performed in [14, 39]. Motivated by the above, [5, 42] started a stream of research on convex relaxations, and the conditions ensuring that the relaxation is tight. Since the vast majority of these results are about networks with tree topology, we defer the details to the next subsection.

Additional approaches include: other local optimization algorithms, such as ADMM in [47], and sequential convex minimization in [67]; global, but slow, optimization algorithms, such as branch and bound in [36], and an algorithm based on the theory of polynomial optimization in [31]; heuristic global optimization algorithms, such as tabu search in [1], particle swarm optimization in [2], and genetic algorithms in [41].

2.2.2 Methods for networks with tree topology

Tree topology on its own does not make the problem easier, since according to [43] even the feasibility problem is NP-Hard. However, when augmented with additional assumptions on the problem's structure, convex relaxations are tight, and thus allow efficiently and reliably computing global optima.

Suppose w.l.o.g that $\mathcal{V} = \{1, \dots, n\}$, meaning in particular that the decision variables are $\mathbf{v}, \mathbf{s} \in \mathbb{C}^n$. The power-flow equations (1.4) are a quadratic system of equations in the real variables $\mathbf{x} = (\mathrm{re}(\mathbf{v}), \mathrm{im}(\mathbf{v}))$ and $\mathbf{y} = (\mathrm{re}(\mathbf{s}), \mathrm{im}(\mathbf{s}))$, which can be written as

$$\mathbf{y} = \mathrm{diag}(\mathbf{x}\mathbf{x}^T \mathbf{W}),$$

where the matrix $\mathbf{W} \in \mathbb{R}^{2n \times 2n}$ is derived from the matrix $\mathbf{Y}$ in the power-flow equations (1.4). A relaxation of the classical problem (OPF-C) is obtained, as described in [42, 12], by denoting $\mathbf{X} = \mathbf{x}\mathbf{x}^T$, and remembering that $\mathbf{X}$ is a symmetric positive-semidefinite matrix of rank one, and that $\mathbf{x}$ can be uniquely, up to the sign, recovered from such matrices. Assuming also that ϕ_j are functions of the real variables $\mathrm{re}(s_j), \mathrm{im}(s_j)$, the (OPF-C) problem can be written as

$$\begin{aligned}
\min_{\mathbf{X},\mathbf{y}} \quad & \sum_{j=1}^{n} \phi_j(y_j, y_{j+n}),\\
\text{s.t.} \quad & \mathbf{y} = \mathrm{diag}(\mathbf{X}\mathbf{W}),\\
& \underline{y}_j \le y_j \le \overline{y}_j, && j \in \{1, \dots, 2n\}\\
& \underline{u}_j^2 \le X_{j,j} + X_{j+n,j+n} \le \overline{u}_j^2, && j \in \{1, \dots, n\}\\
& \mathbf{X} \succeq 0,\\
& \mathrm{rank}(\mathbf{X}) = 1
\end{aligned}$$

Note, that the objective is convex by assumption, and all but the last constraint are convex. When the last constraint is discarded, a convex SDP relaxation is obtained. Additional conditions are required for the relaxation to be exact. For example, in [12], it is proved that when there are no lower bounds on the active and reactive powers, that is, when $\underline{y}_j = -\infty$ for all $j \in \{1, \dots, 2n\}$, and when ϕ_j are linear functions, then the relaxation is tight.

Variants of the above convex relaxations, as well as additional relaxations to SOCP problems, can be found in [29, 25, 30, 11, 42]. A comprehensive survey of this stream of research can be found in [45, 46]. Various conditions for the tightness of convex relaxations were studied in the above-mentioned papers and references therein, but they eventually come down to the lack of upper or lower bounds, either on the voltage absolute values or the active and reactive powers.

Despite the success of convex relaxations, there are many optimal power flow instances on tree networks for which convex relaxations are incapable of producing the global optimum. For example, in contrast to popular works on convex relaxations, the authors of [40] study the conditions which ensure that SDP relaxations will *not* produce the global optimum, and create a library of such problem instances. This prompts the need for different approaches, which compute global optima for networks under assumptions which are not covered by convex relaxations. The focus of our work is on two such problem classes.

Chapter 3

Tree Reduction Lemmas

In this short chapter we present two technical results, which serve as the foundation for the algorithms we present in this work. Both results facilitate a reformulation of the power-flow equations which turns out to be useful in solving the problem classes we aim to address. The proofs are of little mathematical depth, and rely on elementary properties of complex numbers, and thus readers may wish to skip the proofs.

The first result presents a relationship between an admissible pair of some network structure with an admissible pair of a smaller structure, obtained by removing a single leaf. The result uses the sub-vector notation $\mathbf{x}_A$, denoting the vector obtained by taking the components of $\mathbf{x}$ whose indices are in A.

Lemma 2 (Leaf reduction lemma)**.** *Let $(G, \mathbf{z}')$ be a network structure with $G = (\mathcal{V}, \mathcal{E})$. Let $k \in \mathcal{V}$ be a leaf of G with $\mathrm{N}(G, k) = \{m\}$. Let $(G', \mathbf{z}')$ be the network structure obtained by removing the node k and its associated edge, that is, $G' = (\mathcal{V}', \mathcal{E}')$ where*

$$\mathcal{V}' = \mathcal{V} \setminus \{k\},\ \mathcal{E}' = \mathcal{E} \setminus \{(k,m),(m,k)\},\ \mathbf{z}' = \mathbf{z}_{\mathcal{E}'}.$$

Let $(\mathbf{v}, \mathbf{s}) \in \mathbb{C}^{\mathcal{V}} \times \mathbb{C}^{\mathcal{V}}$ be such that $v_k \neq 0$. Then, $(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(G, \mathbf{z})$ if and only if there exists $\mathbf{s}' \in \mathbb{C}^{\mathcal{V}'}$ such that the following system holds:

$$\begin{aligned}
&(\mathbf{v}_{\mathcal{V}'}, \mathbf{s}') \in \mathrm{PFE}(G', \mathbf{z}'),\\
&\quad s'_i = s_i, && i \neq m,\\
&\quad s'_m = s_m + \tilde{s}_k,\\
&\quad |v_m| = \left| |v_k| - \frac{z_{km} s_k^*}{|v_k|} \right|, && (3.1)\\
&\quad v_k = v_m + \frac{z_{km} \tilde{s}_k^*}{v_m^*}, && (3.2)
\end{aligned}$$

where

$$\tilde{s}_k = s_k - z_{km} \frac{|s_k|^2}{|v_k|^2}. \tag{3.3}$$

Proof. By definition, $(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(G, \mathbf{z})$ if and only if the power-flow equations hold. We isolate the equations which involve the indices k, m:

$$\begin{aligned}
z_{km} I_{km} &= v_k - v_m\\
I_{km} &= -I_{mk} && (3.4)\\
z_{mk} I_{mk} &= v_m - v_k && (3.5)\\
I_{mk} &= -I_{km}, && (3.6)\\
s_k &= v_k I_{km}^*\\
s_m &= v_m I_{mk}^* + \sum_{\substack{j \in \mathrm{N}(G,k)\\ j \neq k}} v_m I_{mj}^*
\end{aligned}$$

Since $z_{km} = z_{mk}$, equations (3.4), (3.5), and (3.6) can be eliminated, and substituting $I_{mk} = -I_{km}$, we obtain the following equivalent system:

$$\begin{aligned} z_{km} I_{km} &= v_k - v_m \\ s_k &= v_k I_{km}^* \qquad (3.7) \\ s_m &= -v_m I_{km}^* + \sum_{\substack{j \in \mathrm{N}(G,k) \\ j \neq k}} v_m I_{mj} \qquad (3.8) \end{aligned}$$

By adding (3.7) and (3.8), and replacing (3.8) with the resulting equation, we obtain the following equivalent system:

$$\begin{aligned} z_{km} I_{km} &= v_k - v_m \\ s_k &= v_k I_{km}^* \\ s_m + s_k - (v_k - v_m) I_{km}^* &= \sum_{\substack{j \in \mathrm{N}(G,k) \\ j \neq k}} v_m I_{mj} \qquad (3.9) \end{aligned}$$

Defining

$$\tilde{s}_k = s_k - (v_k - v_m) I_{km}^*, \qquad (3.10)$$

we can re-write equation (3.9) as

$$s_m + \tilde{s}_k = \sum_{\substack{j \in \mathrm{N}(G,k) \\ j \neq k}} v_m I_{mj}^*.$$

The equation above, together with the power-flow equations which do not involve the indices k, m are exactly the power-flow equations associated with $(G', \mathbf{z}')$, with s_m replaced by $s_m + \tilde{s}_k$. Thus, we conclude that $(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(G, \mathbf{z})$ if and only if there exists $\mathbf{s}' \in \mathbb{C}^{\mathcal{V}'}$ such that:

$$\begin{aligned} &(\mathbf{v}_{\mathcal{V}'}, \mathbf{s}') \in \mathrm{PFE}(G', \mathbf{z}') \\ &s'_i = s_i, && i \neq m \\ &s'_m = s_m + \tilde{s}_k, \\ &z_{km} I_{km} = v_k - v_m \qquad (3.11) \\ &s_k = v_k I_{km}^* \qquad (3.12) \end{aligned}$$

We conclude that it is sufficient to show that the system of equations (3.10), (3.11), and (3.12) above are equivalent to the equations (3.1), (3.2), and (3.3) in the lemma. From (3.12), recalling the non-zero voltage assumption, we conclude that

$$I_{km}^* = \frac{s_k}{v_k}.$$

Substituting the above into (3.11) and (3.10) we obtain the following equivalent system:

$$\tilde{s}_k = s_k - (v_k - v_m) \frac{s_k}{v_k} \qquad (3.13)$$

$$z_{km} \frac{s_k^*}{v_k^*} = v_k - v_m \qquad (3.14)$$

Isolating $v_k - v_m$ from (3.14) and substituting into (3.13) yields

$$\tilde{s}_k = s_k - z_{km} \frac{|s_k|^2}{|v_k|^2}. \qquad (3.15)$$

Isolating s_k from (3.13) yields

$$s_k = \tilde{s}_k \frac{v_k}{v_m},$$

and substituting into (3.14) yields

$$v_k = v_m + \frac{z_{km} \tilde{s}_k^*}{v_m^*}. \qquad (3.16)$$

The system of equations (3.15) and (3.16) is thus equivalent to (3.13) and (3.14), and hence together with (3.14) we obtain the following system with the first equation being redundant:

$$\begin{aligned} z_{km}\frac{s_k^*}{v_k^*} &= v_k - v_m, \\ v_k &= v_m + \frac{z_{km}\tilde{s}_k^*}{v_m^*}, \\ \tilde{s}_k &= s_k - z_{km}\frac{|s_k|^2}{|v_k|^2}. \end{aligned} \tag{$*$}$$

Re-arranging the first equation in $(*)$ yields

$$v_m = v_k\left(1 - \frac{z_{km}s_k^*}{|v_k|^2}\right),$$

and taking the absolute value on both sides yields

$$|v_m| = |v_k|\left|1 - \frac{z_{km}s_k^*}{|v_k|^2}\right| = \left||v_k| - \frac{z_{km}s_k^*}{|v_k|}\right| \tag{3.17}$$

Remembering that the first equation in $(*)$ was redundant, we can replace it with the implication (3.17) without changing the solution set of the system, and obtain the following equivalent system:

$$\begin{aligned} |v_m| &= \left||v_k| - \frac{z_{km}s_k^*}{|v_k|}\right|, \\ v_k &= v_m + \frac{z_{km}\tilde{s}_k^*}{v_m^*}, \\ \tilde{s}_k &= s_k - z_{km}\frac{|s_k|^2}{|v_k|^2}, \end{aligned}$$

concluding the proof. □

Applying the first result repeatedly, removing one leaf at a time until only one node is left, we obtain the following result providing a reformulation of the power-flow equations for network structures with tree topology having an arbitrarily chosen root node.

Lemma 3 (Full reduction lemma)**.** *Let $(T, \mathbf{z})$ be a network structure with a rooted topology tree $T = (\mathcal{V}, \mathcal{E})$, and let $r = \mathrm{root}(T)$. Let $(\mathbf{v}, \mathbf{s}) \in \mathbb{C}^{\mathcal{V}} \times \mathbb{C}^{\mathcal{V}}$ be such that $v_j \neq 0$ for all $j \in \mathcal{V}$. Then $(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(T, \mathbf{z})$ if and only if the following system in $\mathbf{v}$, $\mathbf{s}$, and auxiliary vector $\mathbf{s}' \in \mathbb{C}^{\mathcal{V}}$ holds:*

$$s_r' = 0 \tag{3.18a}$$

$$s_i' = s_i + \sum_{j \in \mathrm{ch}_T(i)} \tilde{s}_j, \qquad i \in \mathcal{V}, \tag{3.18b}$$

$$|v_i| = \left||v_j| - \frac{z_{ij}s_j'^*}{|v_j|}\right| \qquad i \in \mathcal{V},\ j \in \mathrm{ch}_T(i), \tag{3.18c}$$

$$v_j = v_i + \frac{z_{ij}\tilde{s}_j^*}{v_i^*}, \qquad i \in \mathcal{V},\ j \in \mathrm{ch}_T(i), \tag{3.18d}$$

with $\tilde{s}_j$ defined by

$$\tilde{s}_j \equiv s_j' - z_{ij}\frac{|s_j'|^2}{|v_j|^2}, \quad i \in \mathcal{V}, j \in \mathrm{ch}_T(i). \tag{3.19}$$

Proof. By induction on the number of nodes of the tree. Let $T, \mathbf{z}, \mathcal{V}, \mathcal{E}, r$ satisfy the assumptions of the lemma.

Base: Assume $|\mathcal{V}| = 1$, that is $\mathcal{V} = \{r\}$ and $\mathcal{E} = \emptyset$. In that case, by definition $(v_r, s_r) \in \mathrm{PFE}(T, \mathbf{z})$ if and only if

$$s_r = 0,$$

which is equivalent to the system

$$s'_r = 0, \quad s'_r = s_r.$$

The above is exactly the system (3.18) for this case, since $\mathrm{ch}_T(r) = \emptyset$.

Hypothesis: Assume the lemma is true for network structures with rooted topology trees having $|\mathcal{V}| - 1$ nodes.

Step: Assume $|\mathcal{V}| > 1$. Let $k \in \mathrm{L}(T)$, $m = \mathrm{parent}(k)$, $T' = (\mathcal{V}', \mathcal{E}')$ with

$$\mathcal{V}' = \mathcal{V} \setminus \{k\} \quad \mathcal{E}' = \mathcal{E} \setminus \{\{k, m\}\},$$

and $\mathbf{z}' = \mathbf{z}_{\mathcal{E}'}$. By Lemma 2, $(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(T, \mathbf{z})$ if and only if there exists $\mathbf{s}' \in \mathbb{C}^{\mathcal{E}'}$ such that

$$\begin{aligned}
&(\mathbf{v}_{\mathcal{V}'}, \mathbf{s}') \in \mathrm{PFE}(T', \mathbf{z}'), \\
&s'_i = s_i, && i \neq m, \\
&s'_m = s_m + \tilde{s}_k, \\
&|v_m| = \left| |v_k| - \frac{z_{km} s_k^*}{|v_k|} \right|, \\
&v_k = v_m + \frac{z_{km} \tilde{s}_k^*}{v_m^*},
\end{aligned}$$

with $\tilde{s}_k$ defined in equation (3.19) (with $j = k$). By the induction hypothesis applied to $(T', \mathbf{z}')$, the above is true if and only if there exists $\mathbf{s}'' \in \mathbb{C}^{\mathcal{E}'}$ such that

$$\begin{aligned}
&s''_r = 0, \\
&s''_i = s'_i + \sum_{j \in \mathrm{ch}_{T'}(i)} \tilde{s}_j && i \in \mathcal{V}', && (*) \\
&|v_i| = \left| |v_j| - \frac{z_{ij} s''^*_j}{|v_j|} \right|, && i \in \mathcal{V}',\ j \in \mathrm{ch}_{T'}(i), \\
&v_j = v_i + \frac{z_{ij} \tilde{s}_j^*}{v_i^*}, && i \in \mathcal{V}',\ j \in \mathrm{ch}_{T'}(i), \\
&s'_i = s_i, && i \neq m, && (**) \\
&s'_m = s_m + \tilde{s}_k, && && (***) \\
&|v_m| = \left| |v_k| - \frac{z_{km} s_k^*}{|v_k|} \right|, \\
&v_k = v_m + \frac{z_{km} \tilde{s}_k^*}{v_m^*},
\end{aligned}$$

where

$$\tilde{s}_j = s''_j - z_{ij} \frac{|v_j|^2}{|s''_j|^2} \quad i \in \mathcal{V}',\ j \in \mathrm{ch}_{T'}(i).$$

For $i \neq m$, substituting $(**)$ into $(*)$ and noting that $\mathrm{ch}_T(i) = \mathrm{ch}_{T'}(i)$ results in

$$s''_i = s_i + \sum_{j \in \mathrm{ch}_T(i)} \tilde{s}_j.$$

For $i = m$, substituting $(***)$ into $(*)$ results in

$$s''_i = s_i + \tilde{s}_k + \sum_{j \in \mathrm{ch}_{T'}(i)} \tilde{s}_j = s_i + \sum_{j \in \mathrm{ch}_T(i)} \tilde{s}_j$$

From both cases we eliminate the variables s'_i, and by letting $s''_k = s_k$ we obtain the following equivalent system:

$$
\begin{aligned}
s''_r &= 0, \\
s''_i &= s_i + \sum_{j \in \mathrm{ch}_T(i)} \tilde{s}_j && i \in \mathcal{V} \\
|v_i| &= \left| |v_j| - \frac{z_{ij} s''^*_j}{|v_j|} \right|, && i \in \mathcal{V}',\ j \in \mathrm{ch}_{T'}(i), \\
|v_m| &= \left| |v_k| - \frac{z_{km} s^*_k}{|v_k|} \right|, \\
v_j &= v_i + \frac{z_{ij} \tilde{s}_i^*}{v_i^*}, && i \in \mathcal{V}',\ j \in \mathrm{ch}_{T'}(i), \\
v_k &= v_m + \frac{z_{km} \tilde{s}^*_k}{v^*_m},
\end{aligned}
$$

Finally, in the system above the 3rd and 4th equations collapse into

$$
|v_i| = \left| |v_j| - \frac{z_{ij} s''^*_j}{|v_j|} \right|, \quad i \in \mathcal{V},\ j \in \mathrm{ch}_T(i),
$$

while the 5th and 6th equations collapse into

$$
v_j = v_j + \frac{z_{ij} s''^*_j}{v^*_j}, \quad i \in \mathcal{V},\ j \in \mathrm{ch}_T(i).
$$

Renaming s''_i to s'_i for all $i \in \mathcal{V}$ concludes the proof.

□

Chapter 4

Networks with One Generator

In this chapter, we will show how to solve the optimal power flow problem for a specific class of power networks with a tree topology which we call *restricted radial networks*.

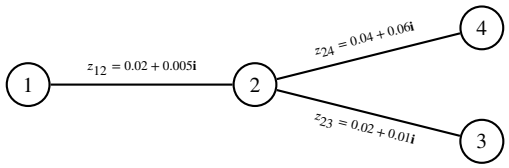
Definition. A power network $P = (T, \mathbf{z}, F)$ is called a **restricted radial network** if the following conditions hold:

- $T = (\mathcal{V}, \mathcal{E})$ is a rooted tree with $|T| \geq 2$ and $r = \mathrm{root}(T)$;
- F is of the form $F(\mathbf{v}, \mathbf{s}) = f(\mathbf{v}, \mathbf{s}) + \delta_{[\arg=0]}(v_r) + \sum_{j \in \mathcal{V}} \delta_{C_j}(|v_j|, s_j)$ such that
  - if $k \in \mathcal{V} \setminus \mathrm{L}(T),\ k \neq \mathrm{root}(T)$ then C_k is a PQ constraint;
  - if $k \in \mathrm{L}(T)$ then C_k is compact, and is either a PQ or a PV constraint;
  - if $k = \mathrm{root}(T)$ we have $C_k = [\underline{u}_k, \overline{u}_k] \times \mathbb{C}$, where $\underline{u}_k \in \mathbb{R}_+$ and $\overline{u}_k \in \mathbb{R}_{++} \cup \{+\infty\}$ are given constants satisfying $\underline{u}_k < \overline{u}_k$. That is, the root voltage is real, and is constrained to lie in a (possibly unbounded) interval.
  - Voltages are bounded away from zero: for all $k \in \mathcal{V}$ we have $\min\{u : (u, s) \in C_k\} > 0$.

At this stage we impose no restrictions on f. However, when we analyze the conditions which ensure the correctness of the algorithm, some assumptions on f will have to be made. In addition, the assumption that PV constrained nodes cannot be leaves was introducted to simplify the presentation of our algorithm as a concept, and indeed in Section 4.1.9 we show how this assumption can be eliminated. The illustration in Figure 4.1 is an example of a restricted radial network, showing all problem data except for f.

We also implicitly assume the following:

Assumption. *There exists a function $\mathcal{Q}_P$ such that for any feasible pair $(\mathbf{v}, \mathbf{s})$ of the OPF problem on a restricted radial network, we have $\mathbf{s} = \mathcal{Q}_P(\mathbf{v})$.*



$C_1 = [0.97, \infty] \times \mathbb{C}$
↖Root constraint: $\underline{u}_1 = 0.97,\ \overline{u}_1 = \infty$
$C_2 = [0.9, 1.1] \times \{-0.2 - 0.1\mathbf{i}\} = \mathrm{PQ}(0.9, 1.1, -0.2 - 0.1\mathbf{i})$
$C_3 = [0.9, 1.1] \times \{-0.4 - 0.3\mathbf{i}\} = \mathrm{PQ}(0.9, 1.1, -0.4 - 0.3\mathbf{i})$
$C_4 = \{1\} \times \{s \in \mathbb{C} : \mathrm{re}(s) = 0.25,\ \mathrm{im}(s) \in [-1, 1]\} = \mathrm{PV}(0.25, -1, 1)$

Figure 4.1: Example of a restricted radial network.

Such a function, for example, always exists when all edge impedances are non-zero, and is defined by the power-flow equations (1.4). Thus, for the networks encountered in practice, this assumption always holds.

We point out that the OPF problem on a restricted radial network bears close relationship with the power flow problem. Indeed, counting all equality constraints, we have $4|\mathcal{V}| - 1$ equations in $4|\mathcal{V}|$ variables, making the OPF problem essentially one dimensional. In fact, if $\mathrm{dom}(f)$ imposes an equality constraint on the root voltage u_r, we recover the classical power flow problem (PF-C) with the root node being the slack.

Such networks with one 'free' generator (the slack) are rarely encountered in practice. However, the problem is still challenging despite its apparent simplicity, and our algorithm for solving it has two important applications. First, as pointed out above, our algorithm can be used as a *provably correct non-heuristic* method to solve the power flow problem on radial networks. Second, our algorithm can be used as a computational step in an algorithm for solving optimal power flow problems on more general networks, and that is exactly how we utilize it in Chapter 5. Recall that the problem is NP-Hard, and thus we do not claim to do the improbable, and develop a polynomial-time algorithm for that problem. Rather, we claim that our algorithm runs quickly in practice since networks which lead to exponential running time are rare to encounter in practice, and we back our claim by the numerical experiments in Section 4.4.

4.1 The Tree Reduction/Expansion Method

Explicitly written, the optimal power flow problem on restricted radial networks is

$$\begin{aligned} \min_{\mathbf{v},\mathbf{s}} \quad & f(\mathbf{v},\mathbf{s}), \\ \text{s.t.} \quad & (\mathbf{v},\mathbf{s}) \in \mathrm{PFE}(T,\mathbf{z}), \\ & \arg(v_r) = 0, \\ & (|v_j|, u_j) \in C_j. \end{aligned}$$

In contrast to popular methods, our algorithm for solving the OPF problem on a restricted radial network does not, by nature, generate a recurrent sequence which converges to the optimal solution. Instead, our algorithm is based on a technique that essentially fully characterizes the set defined by the constraints in the 'subject to' section of the problem above, and then picks out of this set a vector (or vectors) corresponding to the minimal value of f. Thus, in this section we often ignore f (or assume that $f \equiv 0$) and refer to the tuple $P = (T, \mathbf{z}, \{C_j\}_{j\in\mathcal{V}})$ as the restricted radial network. The 'subject to' constraints are formally defined by the set of *feasible pairs* of P

$$\mathcal{F}_{vs}(P) \equiv \{(\mathbf{v},\mathbf{s}) \in \mathrm{PFE}(T,\mathbf{z}) : \arg(v_r) = 0,\ (|v_j|, s_j) \in C_j\}.$$

Our main computational tool is, however, the set of *feasible voltages* of P, defined by

$$\mathcal{F}_v(P) \equiv \{\mathbf{v} : \exists \mathbf{s} \text{ s.t. } (\mathbf{v},\mathbf{s}) \in \mathcal{F}_{vs}(P)\}.$$

The set $\mathcal{F}_v(P)$ fully characterizes $\mathcal{F}_{vs}(P)$. Recalling our assumption that $\mathbf{s} = \mathcal{Q}_P(\mathbf{v})$, we have the relationship

$$\mathcal{F}_{vs}(P) = \{(\mathbf{v}, \mathcal{Q}_P(\mathbf{v})) : \mathbf{v} \in \mathcal{F}_v(P)\}.$$

The method comprises two stages. The aim of the first stage is to find a representation of $\mathcal{F}_v(P)$ that will enable us to evaluate all the points in $\mathcal{F}_v(P)$ up to some discretization. This is done by a sequence of reductions of the power network up to the point that we reach an elementary network with a single node. This sequence of reductions is done by the *tree reduction method*. In the second stage, we use the outcomes of the first stage in order to find all points in $\mathcal{F}_v(P)$ up to some discretization, using a method that we call *the tree expansion method*, and pick the optimal solution out of these points. The combination of the two methods is the *tree reduction/expansion method*.

This section is devoted to the derivation of the tree reduction/expansion method and its practical implementation. We derive our method under a technical assumption on the network, which somewhat limits the applicability of our algorithm. Then, we show how this assumption can be substantially weakened, making our algorithm applicable to almost any restricted radial network.

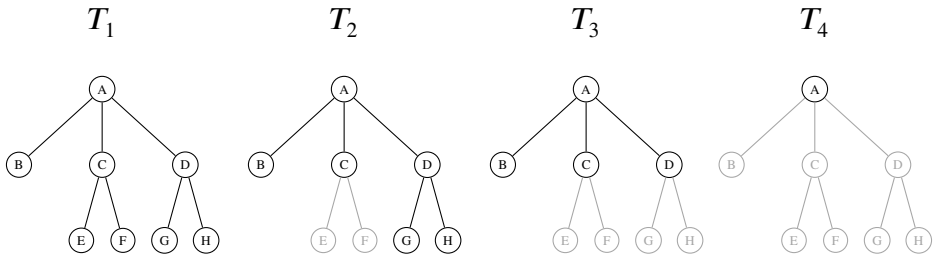


Figure 4.2: A sequence of reductions. Grayed nodes and edges were discarded by the reductions. T_1 consists of the nodes $\{A, B, C, D, E, F, G, H\}$ and rooted at A. T_2 is a reduction of T_1 via C. T_3 is a reduction of T_2 via D. Finally, T_4 is a reduction of T_3 via A. Also, for any $k < j$ the tree T_j is an indirect reduction of T_k

4.1.1 Tree Reductions

We begin with the definition of a *tree reduction*, illustrated in Figure 4.2.

Definition (Tree reduction)**.**

1. A node $j \in \mathcal{V}$ in a rooted tree $T = (\mathcal{V}, \mathcal{E})$ is **reducible** if $\mathrm{ch}_T(j) \subseteq \mathrm{L}(T)$, and $\mathrm{ch}_T(j) \neq \emptyset$. That is, j's children form a non-empty set of leaves.

2. A rooted tree T is a **reduction of a rooted tree S via the node j**, if j is reducible in S and T was constructed by removing $\mathrm{ch}_S(j)$ and all their associated edges from S. A tree T is a reduction of S if it is a reduction of S via some reducible node of S.

3. A tree T is an **indirect reduction of a rooted tree** S, if there exists a sequence of rooted trees $S = T_1, T_2, \ldots, T_m = T$ such that T_{k+1} is a reduction of T_k for any $k = 1, 2, \ldots, m-1$.

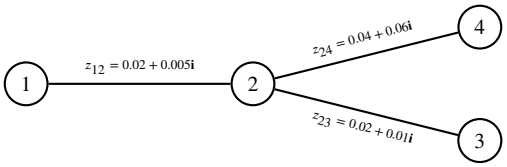
4.1.2 Curved Radial Networks (CRNs)

Another step that is required to derive the tree reduction method is to define a slight generalization of restricted radial networks with $f \equiv 0$.

Definition (Curved Radial Network)**.** A power network $(T, \mathbf{z}, F)$, described by the tuple $P = (T, \mathbf{z}, \{\mathcal{C}_k\}_{k\in\mathcal{V}})$, is a **curved radial network** (or a CRN) if the following conditions hold:

- $T = (\mathcal{V}, \mathcal{E})$ is a rooted tree with $\mathrm{root}(T) = r$.
- F is of the form $F(\mathbf{v}, \mathbf{s}) = \delta_{[\arg=0]}(v_r) + \sum_{j\in\mathcal{V}} \delta_{\mathcal{C}_j}(|v_j|, s_j)$.
- For any $k \neq r,\ k \notin \mathrm{L}(T)$ the set $\mathcal{C}_k$ corresponds to a PQ constraint.
- For any $k \in \mathrm{L}(T)$ we have
$$\mathcal{C}_k = \mathrm{image}(\nu_k, \sigma_k),$$
where $\nu_k : [0,1] \to \mathbb{R}_{++}$ and $\sigma_k : [0,1] \to \mathbb{C}$ are given continuous functions. That is, for any leaf k, the set $\mathcal{C}_k$ is a continuous one-dimensional curve in $\mathbb{R}_{++} \times \mathbb{C}$.
- If $r \notin \mathrm{L}(T)$, the root constraint is $\mathcal{C}_r = [\underline{u}_r, \overline{u}_r] \times \mathbb{C}$, where $\underline{u}_r \in \mathbb{R}_+$ and $\overline{u}_r \in \mathbb{R}_{++} \cup \{+\infty\}$ are given constants satisfyng $\underline{u}_r < \overline{u}_r$. That is, the root voltage is real because of the definition of F, and constrained to lie in a (possibly unbounded) interval.
- Voltages are bounded away from zero: for all $k \in \mathcal{V}$ we have $\min\{u : (u, s) \in \mathcal{C}_k\} > 0$.

The definition of feasible pairs $\mathcal{F}_{vs}(P)$ and feasible voltages $\mathcal{F}_v(P)$ naturally extends to curved radial networks. To see why a CRN is indeed a generalization of a restricted radial network with $f \equiv 0$, we point out two observations. First, in a CRN we allow $|\mathcal{V}| = 1$, while in a restricted radial network we require $|\mathcal{V}| \geq 2$. Second, when $|\mathcal{V}| \geq 2$, the only difference between the two network types is in the



$$
\begin{aligned}
&\mathcal{C}_1 = [0.97, \infty] \times \mathbb{C} \\
&\mathcal{C}_2 = [0.9, 1.1] \times \{-0.2 - 0.1\mathbf{i}\} \\
&\mathcal{C}_3 = \text{image}(\nu_3, \sigma_3) \qquad \nu_3(t) = 0.9 + 0.2t,\ \sigma_3(t) = -0.4 - 0.3\mathbf{i} \\
&\mathcal{C}_4 = \text{image}(\nu_4, \sigma_4) \qquad \nu_4(t) = 1,\ \sigma_4(t) = 0.25 + \mathbf{i} \cdot [-1 + 2t]
\end{aligned}
$$

Figure 4.3: A CRN equivalent to the network in Figure 4.1.

definition of $\mathcal{C}_k$ for $k \in \mathrm{L}(T)$ as one-dimensional curves. However, these sets in a restricted radial network are essentially one-dimensional line segments, and thus can be represented as one-dimensional curves. Indeed, if $(T, \mathbf{z}, \{\mathcal{C}_k\}_{k\in\mathcal{V}})$ is a restricted radial network, then for any $k \in \mathrm{L}(T)$, we can represent the constraint set $\mathcal{C}_k$ as $\mathcal{C}_k = \text{image}(\nu_k, \sigma_k)$ with the following functions for PQ constraints (recalling that $\underline{u}_k, \overline{u}_k$ are finite real numbers):

$$\nu_k(t) = (1-t)\underline{u}_k + t\overline{u}_k, \quad \sigma_k(t) = \hat{s}_k,$$

and the following functions for PV constraints (recalling that $\underline{q}_k, \overline{q}_k$ are finite real numbers):

$$\nu_k(t) = \hat{u}_k, \quad \sigma_k(t) = \hat{p}_k + \mathbf{i}\left[(1-t)\underline{q}_k + t\overline{q}_k\right].$$

For example, applying the transformation described above to the network in Figure 4.1 results in the CRN illustrated in Figure 4.3.

4.1.3 The CRN Reduction Theorem

The next theorem is the main result in the chapter and constitutes the theoretical justification for the tree reduction/expansion method. The result establishes a relationship between a CRN with some topology tree T, and a smaller CRN whose topology graph is a reduction of T. The theorem uses the sub-vector notation $\mathbf{x}_A$ designating the vector composed of the entries of $\mathbf{x}$ whose indices are in A.

Theorem 1 (The CRN reduction theorem)**.** *Let $P = (T, \mathbf{z}, \{C_k\}_{k\in\mathcal{V}})$ be a CRN with $|\mathcal{V}| \geq 2$. Let j be a reducible node in T. Let:*

- *$T' = (\mathcal{V}', \mathcal{E}')$ be the reduction of T via j;*
- *$\{\nu_k, \sigma_k\}_{k\in\mathrm{ch}_T(j)}$ be the functions describing the curves $\mathcal{C}_k$, $k \in \mathrm{ch}_T(j)$;*
- *$\underline{u}_j, \overline{u}_j, \hat{s}_j$ be the scalars for which $\mathcal{C}_j = \mathrm{PQ}(\underline{u}_j, \overline{u}_j, \hat{s}_j)$.*

Define the functions $\tilde{\nu}_k : [0,1] \to \mathbb{R}_+$ and $\tilde{\sigma}_k : [0,1] \to \mathbb{C}$, and the set U_j by:

$$\tilde{\nu}_k(t) = \left|\nu_k(t) - \frac{(\sigma_k(t))^* z_{kj}}{\nu_k(t)}\right|, \qquad k \in \mathrm{ch}_T(j), \tag{4.1}$$

$$\tilde{\sigma}_k(t) = \sigma_k(t) - z_{kj}\frac{|\sigma_k(t)|^2}{(\nu_k(t))^2}, \qquad k \in \mathrm{ch}_T(j), \tag{4.2}$$

$$U_j = [\underline{u}_j, \overline{u}_j] \cap \left(\bigcap_{k\in\mathrm{ch}_T(j)} \text{image}(\tilde{\nu}_k)\right). \tag{4.3}$$

Assume that the functions $\tilde{\nu}_k$ are invertible, and let

$$\phi_k \equiv \tilde{\sigma}_k \circ \tilde{\nu}_k^{-1}. \tag{4.4}$$

If $U_j = \emptyset$, then $\mathcal{F}_v(P) = \emptyset$. Otherwise, set $[v_j^{min}, v_j^{max}] = U_j$ and we have that $\mathbf{v} \in \mathcal{F}_v(P)$ if and only if all of the following hold:

1. $\mathbf{v}_{\mathcal{V}'} \in \mathcal{F}_v(P')$, *where* $P' = (T', \mathbf{z}', \{C'_k\}_{k\in\mathcal{V}'})$ *is a CRN with with* $\mathbf{z}' \equiv \mathbf{z}_{\mathcal{E}'}$, $C'_k = C_k$ *for* $k \neq j$, *and* C'_j *is defined by the following pair of functions:*

$$v_j(t) = (1-t)\cdot v_j^{min} + t \cdot v_j^{max}, \tag{4.5}$$

$$\sigma_j(t) = \begin{cases} \hat{s}_j + \sum_{k\in \mathrm{ch}_T(j)} (\phi_k \circ v_j)(t), & j \neq \mathrm{root}(T), \\ 0, & j = \mathrm{root}(T); \end{cases} \tag{4.6}$$

2. *for any* $k \in \mathrm{ch}_T(j)$ *we have*

$$v_k = v_j + z_{kj}\left(\frac{\phi_k(|v_j|)}{v_j}\right)^*. \tag{4.7}$$

The tree reduction theorem essentially states that the problem of finding the feasible voltages of a given CRN $P = (T, \mathbf{z}, \{C_k\}_{k\in\mathcal{V}})$ can be reduced into the problem of finding the feasible vectors of a smaller CRN $P' = (T', \mathbf{z}', \{C'_k\}_{k\in\mathcal{V}'})$. Assuming that we found the set of feasible vectors of P', we can find the voltages of the larger CRN by using the functions ϕ_k given in (4.4). These functions are called the *voltage transfer functions* since they allow us, using equation (4.7), to map the voltage of the node through which the reduction was performed to the voltages of its children nodes.

The proof of the theorem is technical in nature - readers who are not interested are invited to skip it. The proof relies on the leaf reduction lemma (Lemma 2) we proved earlier. First, we require the following simple result.

Lemma 4. *Let* $(T, \mathbf{z})$ *be a network structure with* $T = (\mathcal{V}, \mathcal{E})$ *being the topology tree. Let* $T' = (\mathcal{V}', \mathcal{E}')$ *be the reduction of* T *via the reducible node* $j \in \mathcal{V}$, *and let* $(\mathbf{v}, \mathbf{s}) \in \mathbb{C}^{\mathcal{V}} \times \mathbb{C}^{\mathcal{V}}$ *be such that* $v_k \neq 0$ *for all* $k \in \mathrm{ch}_T(j)$. *Then* $(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(T, \mathbf{z})$ *if and only if all of the following hold:*

$$(\mathbf{v}_{\mathcal{V}'}, \mathbf{s}') \in PFE(T', \mathbf{z}_{\mathcal{E}'}), \qquad \mathbf{s}'_{\mathcal{V}'\setminus\{j\}} = \mathbf{s}_{\mathcal{V}'\setminus\{j\}}, \tag{4.8}$$

$$|v_j| = \left||v_k| - \frac{s_k^* z_{kj}}{|v_k|}\right|, \qquad k \in \mathrm{ch}_T(j), \tag{4.9}$$

$$s'_j = s_j + \sum_{k\in\mathrm{ch}_T(j)} \tilde{s}_k, \tag{4.10}$$

$$\tilde{s}_k = s_k - z_{kj}\frac{|s_k|^2}{|v_k|^2}, \qquad k \in \mathrm{ch}_T(j), \tag{4.11}$$

$$v_k = v_j + \frac{\tilde{s}_k^* z_{kj}}{v_j^*}, \qquad k \in \mathrm{ch}_T(j), \tag{4.12}$$

Proof. Follows directly by repeatadly applying Lemma 2 with $k \in \mathrm{ch}_T(j)$ and $m = j$. □

Proof of Theorem 1. Let $r = \mathrm{root}(T)$ and let $\mathcal{V}'' = \mathcal{V}' \setminus \{j\}$. The statement $\mathbf{v} \in \mathcal{F}_v(P)$, by definition, holds if and only if there exists $\mathbf{s}$ such that

$$\begin{aligned} &(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(T, \mathbf{z}), \\ &(|v_k|, s_k) \in C_k, && k \in \mathcal{V}, \\ &\arg(v_r) = 0. \end{aligned}$$

Using the definition of C_k for $k \in \mathrm{ch}_T(j) \cup \{j\}$, the system above is equivalent to the following system in the variables $\mathbf{v}, \mathbf{s}$ and $\mathbf{t} \in \mathbb{R}^{\mathrm{ch}_T(j)}$:

$$\begin{aligned} &(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(T, \mathbf{z}), \\ &|v_k| = v_k(t_k),\ s_k = \sigma_k(t_k),\ t_k \in [0,1], && k \in \mathrm{ch}_T(j), \\ &s_j = \hat{s}_j, && \text{if } j \neq r, \\ &\underline{u}_j \leq |v_j| \leq \overline{u}_j, && \text{if } j \neq r, \\ &(|v_k|, s_k) \in C_k, && k \in \mathcal{V}'' \\ &\arg(v_r) = 0. \end{aligned}$$

Recalling that $|v_j| > 0$ for all j and applying Lemma 4 to the constraint $(\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(T, \mathbf{z})$, while utilizing the identities $|v_k| = \nu_k(t_k),\ s_k = \sigma_k(t_k),\ s_j = \hat{s}_j$, we obtain equivalence to the following system in the variables $\mathbf{v}, \mathbf{s}' \in \mathbb{C}^{\mathcal{V}'}, \mathbf{s}_{\mathcal{V}''} \in \mathbb{C}^{\mathcal{V}''}, \mathbf{t} \in \mathbb{R}^{\mathrm{ch}_T(j)}, \tilde{\mathbf{s}} \in \mathbb{C}^{\mathrm{ch}_T(j)}$

$$
\begin{aligned}
&(\mathbf{v}_{\mathcal{V}'}, \mathbf{s}') \in \mathrm{PFE}(T', \mathbf{z}_{\mathcal{E}'}), && \mathbf{s}'_{\mathcal{V}''} = \mathbf{s}_{\mathcal{V}''},\\
&|v_j| = \left| \nu_k(t_k) - \frac{(\sigma_k(t_k))^* z_{kj}}{\nu_k(t_k)} \right|, && k \in \mathrm{ch}_T(j),\\
&s'_j = \hat{s}_j + \sum_{k \in \mathrm{ch}_T(j)} \tilde{s}_k, && \text{if } j \neq r,\\
&v_k = v_j + \frac{\tilde{s}_k^* z_{kj}}{v_j^*}, && k \in \mathrm{ch}_T(j),\\
&\tilde{s}_k = \sigma_k(t_k) - z_{kj} \frac{|\sigma_k(t_k)|^2}{(\nu_k(t_k))^2}, && k \in \mathrm{ch}_T(j),\\
&t_k \in [0,1], && k \in \mathrm{ch}_T(j),\\
&\underline{u}_j \leq |v_j| \leq \overline{u}_j, && \text{if } j \neq r,\\
&(|v_k|, s_k) \in C_k, && k \in \mathcal{V}'',\\
&\arg(v_r) = 0.
\end{aligned}
$$

Substituting the definition of $\tilde{\nu}_k, \tilde{\sigma}_k$, and observing that $\tilde{s}_k = \tilde{\sigma}_k(t_k)$, we conclude that the above system is equivalent to the following system in the variables $\mathbf{v}, \mathbf{s}' \in \mathbb{C}^{\mathcal{V}'}, \mathbf{s}_{\mathcal{V}''} \in \mathbb{C}^{\mathcal{V}''}, \mathbf{t} \in \mathbb{R}^{\mathrm{ch}_T(j)}$:

$$
\begin{aligned}
&(\mathbf{v}_{\mathcal{V}'}, \mathbf{s}') \in \mathrm{PFE}(T', \mathbf{z}_{\mathcal{E}'}), && \mathbf{s}'_{\mathcal{V}''} = \mathbf{s}_{\mathcal{V}''},\\
&|v_j| = \tilde{\nu}_k(t_k), && k \in \mathrm{ch}_T(j),\\
&s'_j = \hat{s}_j + \sum_{k \in \mathrm{ch}_T(j)} \tilde{\sigma}_k(t_k), && \text{if } j \neq r,\\
&v_k = v_j + \frac{(\tilde{\sigma}_k(t_k))^* z_{kj}}{v_j^*}, && k \in \mathrm{ch}_T(j),\\
&t_k \in [0,1], && k \in \mathrm{ch}_T(j),\\
&\underline{u}_j \leq |v_j| \leq \overline{u}_j, && \text{if } j \neq r,\\
&(|v_k|, s_k) \in C_k, && k \in \mathcal{V}'',\\
&\arg(v_r) = 0.
\end{aligned}
$$

Using the assumed invertibility of $\tilde{\nu}_k$, we can conclude that $|v_j| = \tilde{\nu}_k(t_k)$ if and only if $|v_j| \in \mathrm{image}(\tilde{\nu}_k)$ and $t_k = \tilde{\nu}_k^{-1}(|v_j|)$, and obtain equivalence to the following system in the variables $\mathbf{v}, \mathbf{s}' \in \mathbb{C}^{\mathcal{V}'}, \mathbf{s}_{\mathcal{V}''} \in \mathbb{C}^{\mathcal{V}''}$:

$$
\begin{aligned}
&(\mathbf{v}_{\mathcal{V}'}, \mathbf{s}') \in \mathrm{PFE}(T', \mathbf{z}_{\mathcal{E}'}), && \mathbf{s}'_{\mathcal{V}''} = \mathbf{s}_{\mathcal{V}''},\\
&s'_j = \hat{s}_j + \sum_{k \in \mathrm{ch}_T(j)} \tilde{\sigma}_k(\tilde{\nu}_k^{-1}(|v_j|)), && \text{if } j \neq r,\\
&v_k = v_j + \frac{(\tilde{\sigma}_k(\tilde{\nu}_k^{-1}(|v_j|)))^* z_{kj}}{v_j^*}, && k \in \mathrm{ch}_T(j),\\
&|v_j| \in \mathrm{image}(\tilde{\nu}_k), && k \in \mathrm{ch}_T(j), && (*)\\
&\underline{u}_j \leq |v_j| \leq \overline{u}_j, && \text{if } j \neq r, && (**)\\
&(|v_k|, s_k) \in C_k, && k \in \mathcal{V}'',\\
&\arg(v_r) = 0.
\end{aligned}
$$

Since $\tilde{\nu}_k$ is continuous on the domain $[0,1]$, the set $\mathrm{image}(\tilde{\nu}_k)$ is a bounded closed interval. Recalling the definition of U_j, we conclude that if $U_j = \emptyset$, then $\mathcal{F}_v(P) = \emptyset$. If $U_j \neq \emptyset$, then it is a closed bounded interval with a positive lower bound. Denoting $U_j = [\underline{\nu}_j, \overline{\nu}_j]$, the constraints $(*)$ and $(**)$ above can be

replaced with the inequality $0 < \underline{v}_j \leq |v_j| \leq \overline{v}_j$. Thus, by letting

$$\nu_j : [0,1] \to \mathbb{R} \quad \text{s.t.} \quad \nu_j(t) = \overline{v}_j \cdot t + \underline{v}_j \cdot (1-t),$$

the system can be equivalently written as

$$\begin{aligned}
&(\mathbf{v}_{\mathcal{V}'}, \mathbf{s}') \in \mathrm{PFE}(T', \mathbf{z}_{\mathcal{E}'}), && \mathbf{s}'_{\mathcal{V}''} = \mathbf{s}_{\mathcal{V}''},\\
&s'_j = \hat{s}_j + \sum_{k \in \mathrm{ch}_T(j)} \tilde{\sigma}_k(\tilde{v}_k^{-1}(|v_j|)), && \text{if } j \neq r,\\
&v_k = v_j + \frac{(\tilde{\sigma}_k(\tilde{v}_k^{-1}(|v_j|)))^* z_{kj}}{v_j^*}, && k \in \mathrm{ch}_T(j),\\
&|v_j| = \nu_j(t),\\
&(|v_k|, s_k) \in C_k, && k \in \mathcal{V}''\\
&t \in [0,1],\\
&\arg(v_r) = 0.
\end{aligned}$$

Note that when $j = r$, the constraint $(\mathbf{v}_{\mathcal{V}'}, \mathbf{s}') \in \mathrm{PFE}(T', \mathbf{z}_{\mathcal{E}'})$ amounts to $s'_r = 0$. Thus, by defining σ_j as in (4.6), letting

$$C'_j = \mathrm{image}(\nu_j, \sigma_j),$$

reordering, and recalling the definition of ϕ_k, the system can be rewritten as

$$\begin{aligned}
&(\mathbf{v}_{\mathcal{V}'}, \mathbf{s}') \in \mathrm{PFE}(T', \mathbf{z}_{\mathcal{E}'}), && \mathbf{s}'_{\mathcal{V}''} = \mathbf{s}_{\mathcal{V}''},\\
&(|v_k|, s_k) \in C_k, && k \in \mathcal{V}'',\\
&(|v_j|, s'_j) \in C'_j,\\
&\arg(v_r) = 0,\\
&v_k = v_j + z_{kj} \left(\frac{\phi_k(|v_j|)}{v_j} \right)^*, && k =\in \mathrm{ch}_T(j).
\end{aligned}$$

The system above completes the proof, since the first four relations are the same as $\mathbf{v} \in \mathcal{F}_v(P')$ and the fourth relation is the same as item 2 in the premise of the theorem. □

4.1.4 CRN Reduction Theorem Example

We illustrate the reduction described in Theorem 1 on the CRN described in Figure 4.3. We choose the reducible node $j = 2$, for which we have $\mathrm{ch}_T(2) = \{3, 4\}$. We begin by computing the functions $\tilde{v}_3, \tilde{\sigma}_3, \tilde{v}_4, \tilde{\sigma}_4$, verify that $\tilde{v}_3$ and $\tilde{v}_4$ are invertible, and then proceed to compute the set U_2. The formulas in (4.1) and (4.2) with $j = 2$ and $k = 3$ yield:

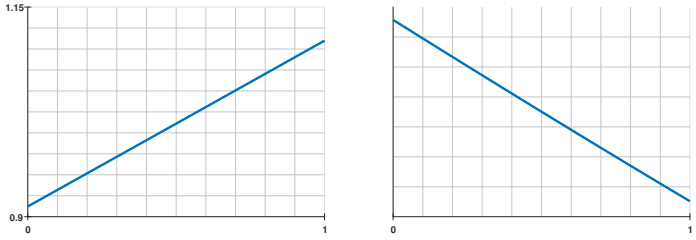
$$\begin{aligned}
\tilde{v}_3(t) &= \left| \nu_3(t) - \frac{(\sigma_3(t))^* z_{23}}{\nu_3(t)} \right| = \left| 0.9 + 0.2t - \frac{(-0.4 + 0.3\mathbf{i})(0.02 + 0.01\mathbf{i})}{0.9 + 0.2t} \right|\\
&= \sqrt{\frac{0.000125}{(0.2t + 0.9)^2} + (0.2t + 0.9)^2 + 0.022},\\
\tilde{\sigma}_3(t) &= \sigma_3(t) - z_{23} \frac{|\sigma_3(t)|^2}{(\nu_3(t))^2} = -0.4 - 0.3\mathbf{i} - (0.02 + 0.01\mathbf{i}) \frac{0.25}{(0.9 + 0.2t)^2}\\
&= \left[-\frac{0.005}{(0.2t + 0.9)^2} - 0.4 \right] + \mathbf{i} \cdot \left[-\frac{0.0025}{(0.2t + 0.9)^2} - 0.3 \right].
\end{aligned} \tag{4.13}$$

Similarly, computing $\tilde{v}_4$ and $\tilde{\sigma}_4$ yields:

$$\begin{aligned}
\tilde{v}_4(t) &= \sqrt{0.0208t^2 - 0.2608t + 1.105525},\\
\tilde{\sigma}_4(t) &= (-0.16t^2 + 0.16t + 0.2075) + \mathbf{i} \cdot (-0.24t^2 + 2.24t - 1.06375).
\end{aligned}$$

It is easy to show that $\tilde{v}_3$ and $\tilde{v}_4$ are monotone; this can also be illustrated graphically:

$\tilde{v}_3(t)$ $\qquad\qquad$ $\tilde{v}_4(t)$



Since $\tilde{v}_3$ is increasing and $\tilde{v}_4$ is decreasing, we have

$$\begin{aligned}\text{image}(\tilde{v}_3) &= [\tilde{v}_3(0), \tilde{v}_3(1)] = [0.9122, 1.1100],\\ \text{image}(\tilde{v}_4) &= [\tilde{v}_4(1), \tilde{v}_4(0)] = [0.9303, 1.0514].\end{aligned}$$

Using equation (4.3) we compute:

$$U_2 = [0.9, 1.1] \cap [0.9122, 1.1100] \cap [0.9303, 1.0514] = [0.9303, 1.0514]. \tag{4.14}$$

The assumptions of the tree reduction theorem indeed hold (invertibility of $\tilde{v}_3$ and $\tilde{v}_4$), and therefore $(v_1, v_2, v_3, v_4)^T \in \mathcal{F}_v(P)$ if and only if:

1. $(v_1, v_2)^T \in \mathcal{F}_v(P')$, where $P' = (T', \mathbf{z}', C_1, C_2')$, $T' = (\mathcal{V}', \mathcal{E}')$ is the reduction of T via node $j = 2$, $\mathbf{z}'$ is $\mathbf{z}_{\mathcal{E}'}$, and $C_2' = \text{image}(\nu_2, \sigma_2)$ with ν_2, σ_2 defined by

$$\begin{aligned}\nu_2(t) &= \underline{v}_2 \cdot (1-t) + \overline{v}_2 \cdot t = 0.9303 \cdot (1-t) + 1.0514 \cdot t\\ \sigma_2(t) &= \hat{s}_2 + \phi_3(\nu_2(t)) + \phi_4(\nu_2(t)),\end{aligned}$$

   where $\phi_3 = \tilde{\sigma}_3 \circ \tilde{v}_3^{-1}$ and $\phi_4 = \tilde{\sigma}_4 \circ \tilde{v}_4^{-1}$.

2. We have

$$\begin{aligned}v_3 &= v_2 + z_{23}\left(\frac{\phi_3(|v_2|)}{v_2}\right)^*,\\ v_4 &= v_2 + z_{24}\left(\frac{\phi_4(|v_2|)}{v_2}\right)^*.\end{aligned}$$

Next, we need to compute ϕ_3 and ϕ_4, which participate in the conclusion above. Let $v \in \mathbb{R}_+$; we will compute $\phi_3(v) = \left(\tilde{\sigma}_3 \circ \tilde{v}_3^{-1}\right)(v)$. Denoting $t = \tilde{v}_3^{-1}(v)$, we have that $\tilde{v}_3(t) = v$, meaning that:

$$\sqrt{\frac{0.000125}{(0.2t+0.9)^2} + (0.2t+0.9)^2 + 0.022} = v,$$

which readily implies that

$$(0.9+0.2t)^2 = \frac{v^2 - 0.022 + \sqrt{v^4 - 0.044v^2 - 0.000016}}{2}.$$

Therefore, taking into account the expression for $\tilde{\sigma}_3$ given in (4.13),

$$\begin{aligned}\phi_3(v) = \tilde{\sigma}_3(t) = &\left[-0.4 - \frac{0.1}{v^2 - 0.022 + \sqrt{v^4 - 0.044v^2 - 0.000016}}\right]\\ &+ \mathbf{i}\left[-0.3 - \frac{0.05}{v^2 - 0.022 + \sqrt{v^4 - 0.044v^2 - 0.000016}}\right]\end{aligned}$$

In a similar manner, ϕ_4 is computed from $\tilde{v}_4$ and $\tilde{\sigma}_4$:

$$\begin{aligned}\phi_4(v) = &\ 0.017751\sqrt{520000.0v^2 - 149777.0} - 7.6923v^2 - 2.8624\\ &+ \mathbf{i} \cdot (0.0073964\sqrt{520000.0v^2 - 149777.0} - 11.538v^2 + 6.8698).\end{aligned}$$

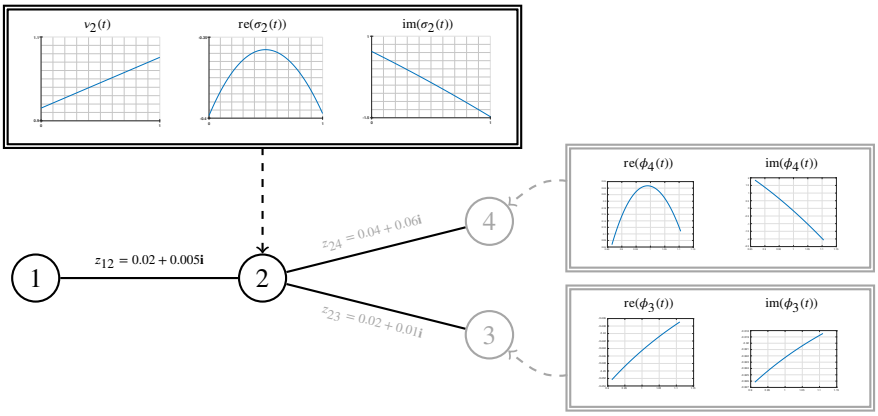


Figure 4.4: The network $P' = (T', \mathbf{z}', C_1, C_2')$, where T' are the black nodes and edges. The gray nodes and edges that were discarded by the reduction.

To summarize, having expressed the functions $\phi_3, \phi_4, v_2, \sigma_2$, and remembering that $P' = (T, \mathbf{z}', C_1, C_2')$ with $C_2' = \mathrm{image}(v_2, \sigma_2)$, we have the following mathematical relationship:

$$\begin{aligned} &(v_1, v_2, v_3, v_4)^T \in \mathcal{F}_v(P) \\ &\Longleftrightarrow \\ &(v_1, v_2)^T \in \mathcal{F}_v(P'), \\ &\qquad v_3 = v_2 + z_{23}\left(\frac{\phi_3(|v_2|)}{v_2}\right)^*, \\ &\qquad v_4 = v_2 + z_{24}\left(\frac{\phi_4(|v_2|)}{v_2}\right)^*. \end{aligned} \tag{4.15}$$

The formulas for v_3 and v_4 above follow from (4.7) for $j = 2$ and $k = 3, 4$. All the functions involved are graphically illustrated in Figure 4.4.

4.1.5 The Tree Reduction Method

The idea in the tree reduction method is to successively employ the reductions described in the tree reduction theorem (Theorem 1) in order to get a sequence of CRNs $P^{(0)} \equiv P, P^{(1)}, \ldots, P^{(m)}$ until we reach the CRN $P^{(m)}$ consisting only of the single node r, which is the root of P. During this process, we compute the voltage transfer functions $\{\phi_k\}_{k \in \mathcal{V},\, k \neq r}$ and the interval $[\underline{v}_r, \overline{v}_r]$ of possible values of the voltage at the root node r (recall that we assume that v_r is real). In addition, we test for the nonemptiness of $\mathcal{F}_v(P)$ by observing that $\mathcal{F}_v(P) \neq \emptyset$ if and only if all of the sets U_j computed during this phase are nonempty. The set of feasible vectors of the final CRN $P^{(m)}$ (consisting of the single node r) is extremely simple and is given by

$$\mathcal{F}_v(P^{(m)}) = [\underline{v}_r, \overline{v}_r].$$

The tree reduction method is formally summarized in Algorithm 1. We assume that we have a procedure named COMPUTE-REDUCTION-ORDER, which computes the sequence of nodes $\tau_1, \ldots, \tau_w$, with $\tau_w = r$, via which the reductions of the original tree are performed. Such a procedure can be easily implemented, for instance, by employing a Breadth-First Search algorithm.

Note that the tree reduction method is conceptual in nature, since it assumes that we have an efficient way to compute and represent the functions σ_k, ν_k and ϕ_k. Section 4.1.8 will describe an implementable version of the method.

We can now formally state the technical assumption which must hold for a CRN P, and is essential for the correctness of the tree reduction method applied on P.

Assumption (invertibility assumption)**.** *During the execution of the tree reduction method on the network P, the functions $\tilde{v}_k$ computed on line* 4 *are invertible.*

Algorithm 1 The Tree Reduction Method

Input: A CRN $P = (T, \mathbf{z}, \{C_j\}_{j\in\mathcal{V}})$ with $T = (\mathcal{V}, \mathcal{E})$ and $r = \text{root}(T)$
Output: the reduction order $(u_1, \ldots, u_w = r)$; the functions $\{\phi_k\}_{k=\mathcal{V},\ k\neq r}$; the interval $[\underline{v}_r, \overline{v}_r]$ of feasible values of v_r.
Steps:

$\tau_1, \ldots, \tau_w = r \leftarrow$ COMPUTE-REDUCTION-ORDER(T)
for all $j \in \{\tau_1, \ldots, \tau_w\}$ **do** ▷ *Invariant*: ν_k, σ_k exist for $k \in \text{ch}_T(j)$
  for all $k \in \text{ch}_T(j)$ **do**
    Compute $\tilde{\nu}_k$ as: $\tilde{\nu}_k(t) \equiv \left|\nu_k(t) - z_{kj}\frac{(\sigma_k(t))^*}{\nu_k(t)}\right|$ ▷ Eq (4.1)
    Compute $\tilde{\sigma}_k$ as: $\tilde{\sigma}_k(t) \equiv \sigma_k(t) - z_{kj}\frac{|\sigma_k(t)|^2}{(\nu_k(t))^2}$ ▷ Eq (4.2)
    Compute ϕ_k as: $\phi_k \equiv \tilde{\sigma}_k \circ \tilde{\nu}_k^{-1}$
  end for
  $U_j \leftarrow [\underline{u}_j, \overline{u}_j] \cap \left(\bigcap_{k\in\text{ch}_T(j)} \text{image}(\tilde{\nu}_k)\right)$ ▷ Eq (4.3)
  if $U_j = \emptyset$ **then**
    Terminate with an error: $\mathcal{F}_v(P) = \emptyset$
  else
    $[\underline{v}_j, \overline{v}_j] \leftarrow U_j$
  end if
  if $j \neq r$ **then**
    Compute ν_j as: $\nu_j(t) \equiv \nu_j^{min} \cdot (1-t) + \nu_j^{max} \cdot t$ ▷ Eq (4.5)
    Compute σ_j as: $\sigma_j(t) \equiv \hat{s}_j + \sum_{k\in\text{ch}_T(j)} \phi_k(\nu_j(t))$ ▷ Eq (4.6)
  end if
end for

It does not seem to be possible to verify the validity of the invertibility assumption on a given CRN a priori using only the data of the problem. However, this condition can be verified during the execution of the tree reduction method. Our numerical simulations suggest that when the impedances are small enough, the assumption holds. In any case, in Section 4.1.10 we describe a way to eliminate the above assumption.

Example Continued

The example from Section 4.1.4 described the first reduction used in the tree reduction method in which the CRN P with the four nodes $\{1, 2, 3, 4\}$ was reduced into the CRN P' with the two nodes $\{1, 2\}$. The tree reduction method will conduct at this point a second reduction – a reduction of the CRN $P' = (T', \mathbf{z}', C_1, C_2')$ via the node 1, which is reducible in T'. We have $\text{ch}_{T'}(1) = \{2\}$. Skipping the tedious computations, the process of computing the functions $\tilde{\nu}_2$ and $\tilde{\sigma}_2$ and the nonempty set U_1, verifying the invertibility of $\tilde{\nu}_2$ by observing that it is strictly increasing, computing the endpoints of the non-empty interval $U_1 = [\nu_1^{min}, \nu_1^{max}]$, and finally utilizing equations (4.5) and (4.6) yields:

$$[\nu_1^{min}, \nu_1^{max}] = [0.97, \infty] \cap [\tilde{\nu}_2(0), \tilde{\nu}_2(1)] = [0.97, 1.0663].$$

In addition, computing $\phi_2 = \tilde{\sigma}_2 \circ \tilde{\nu}_2^{-1}$ results in the function described in Figure 4.5. By defining $P'' =$

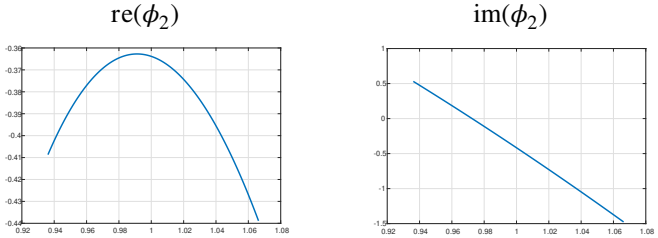


Figure 4.5: Real and imaginary parts of ϕ_2.

$(T'', \mathbf{z}'', \mathcal{C}_1')$ with $\mathcal{C}_1' = [v_1^{min}, v_1^{max}] \times \{0\}$, we can replace the statement $(v_1, v_2) \in \mathcal{F}_v(P')$ in Equation (4.15) with equivalent statements which follow from the CRN reduction theorem applied to P' with $j = 1$. The conclusion is that $(v_1, v_2, v_3, v_4)^T \in \mathcal{F}_v(P)$ if and only if

$$v_1 \in \mathcal{F}_v(P'') = [v_1^{min}, v_1^{max}], \tag{4.16}$$

$$v_2 = v_1 + z_{12} \left(\frac{\phi_2(|v_1|)}{v_1} \right)^*, \tag{4.17}$$

$$v_3 = v_2 + z_{23} \left(\frac{\phi_3(|v_2|)}{v_2} \right)^*, \tag{4.18}$$

$$v_4 = v_2 + z_{24} \left(\frac{\phi_4(|v_2|)}{v_2} \right)^*. \tag{4.19}$$

Algorithm 2 The Tree Expansion Method

Input: a CRN $P = (T, \mathbf{z}, \{\mathcal{C}_j\}_{j \in \mathcal{V}})$ with $T = (\mathcal{V}, \mathcal{E})$ and $r = \text{root}(T)$; two of the outputs of the tree reduction method – the reduction order $(\tau_1, \ldots, \tau_w = r)$, the voltage transfer functions $\{\phi_k\}_{k \in \mathcal{V}, k \neq r}$; a scalar $\alpha \in \mathbb{R}_+$

Output: a vector $(\mathbf{v}, \mathbf{s}) \in \mathcal{F}_{vs}(P)$ for which $v_r = \alpha$.

Steps:

1: $v_r \leftarrow \alpha$
2: **for all** $j \in u_w, \ldots, u_1$ **do** ▷ Invariant: v_j was already computed
3: **for all** $k \in \text{ch}_T(j)$ **do**
4: $v_k \leftarrow v_j + z_{kj} \left(\frac{\phi_k(|v_j|)}{v_j} \right)^*$ ▷ Eq. (4.7)
5: **end for**
6: **end for**
7: $\mathbf{s} \leftarrow \mathcal{Q}_P(\mathbf{v})$
8: Return: $(\mathbf{v}, \mathbf{s})$

4.1.6 The Tree Expansion Method

The system (4.16) - (4.19) concluding Example 4.1.5 illustrates how the outcomes of the tree reduction method, which are the reduction order, the voltage transfer functions $\{\phi_k\}_{k \neq r}$ and the interval of possible values of v_r, can be the basis of an algorithm for computing vectors in $\mathcal{F}_v(P)$. Indeed, to obtain a feasible vector we just need to choose $v_1 \in \mathcal{F}_v(P'') = [0.97, 1.0663]$ and then compute v_2, v_3, v_4 using the relations (4.17), (4.18) and (4.19). After finding the four voltages $\mathbf{v} = (v_1, v_2, v_3, v_4)^T$, the corresponding powers vector can be computed by the assumed relation $\mathbf{s} = \mathcal{Q}_P(\mathbf{v})$. The *tree expansion method* that computes a feasible pair $(\mathbf{v}, \mathbf{s}) \in \mathcal{F}_{vs}(P)$ given a value of $v_r \in [\underline{v}_r, \overline{v}_r]$ is described in Algorithm 2. Note that applying equation (4.7) is done in the order which is opposite to the reduction order.

4.1.7 The Tree Reduction/Expansion Method for Solving the OPF Problem

We can solve the OPF problem on a given restricted radial network by following the next steps: (i) represent the network as a CRN; (ii) employ the tree reduction method to obtain the reduction order, the voltage transfer functions and an interval of possible values of v_1; (iii) use the tree expansion method to compute a discretized version of $\mathcal{F}_{vs}(P)$ and (iv) pick the optimal solution out of the list of discretized pairs in $\mathcal{F}_{vs}(P)$. The algorithm executing these four steps is called the tree reduction/expansion method, and its details are given below. Next, we address the following remaining challenges: how is the conceptual algorithm implemented in practice; how do we cope with violations of the invertability assumption; which additional assumptions guarantee convergence to the optimum as the discretization density increases.

Algorithm 3 The Tree Reduction/Expansion Method

Input: the data of a restricted radial network P with structure $(T, \mathbf{z})$ where $T = (\mathcal{V}, \mathcal{E})$ and $r = \mathrm{root}(T)$, nodal constraints $\{C_j\}_{j\in\mathcal{V}}$, and objective f; the grid density parameter $m \in \mathbb{N},\ m \geq 1$;
Output: an optimal solution of (OPF) (up to discretization) on the network P.
Steps:

1. Represent P, ignoring f, as a CRN.
2. **Reduction:** Run the tree reduction method (Algorithm 1) on P and obtain the voltage transfer functions $\{\phi_k\}_{k\in\mathcal{V},\ k\neq r}$, the reduction order $\tau_1, \ldots, \tau_w = r$ and the interval of possible values of v_r, $[\underline{v}_r, \overline{v}_r]$.
3. **Expansion:** Construct a discretized version of $\mathcal{F}_{vs}(P)$: for any $\ell = 1, 2, \ldots, m$ employ the tree expansion method with input $(P, (\tau_1, \ldots, \tau_w), \{\phi_k\}, \underline{v}_r(1 - t_\ell) + \overline{v}_r t_\ell)$ where $t_\ell = \frac{2\ell-1}{2m}$ and obtain an output $(\mathbf{v}^\ell, \mathbf{s}^\ell) \in \mathcal{F}_{vs}(P)$.
4. Return an element of
$$\operatorname*{argmin}_{\mathbf{v},\mathbf{s}} \{f(\mathbf{v}, \mathbf{s}) : (\mathbf{v}, \mathbf{s}) \in \mathcal{S}\},$$
where $\mathcal{S} = \{(\mathbf{v}^\ell, \mathbf{s}^\ell) : \ell = 1, 2, \ldots, m\}$.

4.1.8 Implementation of the Tree Reduction Method

In contrast to the example detailed in Sections 4.1.4 and 4.1.5, an actual implementation of the tree reduction method will not rely on symbolic expressions to represent the functions ν_k, σ_k and ϕ_k. The reason for that is twofold: first, verifying that $\tilde{\nu}_k$ are invertible based on symbolic expressions is a challenging task on its own; second, inverting $\tilde{\nu}_k$, an operation which is required in order to compute the voltage transfer function ϕ_k, requires analytically solving high-order polynomial equations, which is known to be generally impossible for any polynomial of degree 5 and above. Therefore, an actual implementation of the algorithm will involve replacing all the one-dimensional functions by appropriate approximations. This is of course a modification of the algorithm, but fortunately, one-dimensional functions can be well approximated. Below we provide the exact details of how the one-dimensional functions were represented.

Representing ν_k, $\tilde{\nu_k}$, σ_k, and $\tilde{\sigma}_k$ The functions ν_k, $\tilde{\nu_k}$, σ_k, and $\tilde{\sigma}_k$ are represented by discretizing their argument $t \in [0, 1]$ at $d \geq 4$ ($d \in \mathbb{N}$) uniformly spaced points, where the discretization density d is a parameter of the algorithm. In other words, we define the vector

$$\mathbf{t} = \left(\tfrac{0}{d-1}, \tfrac{1}{d-1}, \tfrac{2}{d-1}, \ldots, \tfrac{d-1}{d-1}\right)^T.$$

We represent the functions ν_k, σ_k with the vectors $\boldsymbol{\nu}_k, \boldsymbol{\sigma}_k \in \mathbb{R}^d$ defined as

$$\boldsymbol{\nu}_k = \nu_k(\mathbf{t}), \boldsymbol{\sigma_k} = \sigma_k(\mathbf{t}), \tag{4.20}$$

where the functions are applied component-wise to the vector $\mathbf{t}$. Correspondingly, $\tilde{\nu}_k, \tilde{\sigma}_k$ with $j = \mathrm{parent}_T(k)$ are represented by the vectors $\tilde{\boldsymbol{\nu}}_k, \tilde{\boldsymbol{\sigma}}_k \in \mathbb{R}^d$, constructed by applying equation (4.2) and (4.1), component-wise, to the vectors defined in Equation (4.20):

$$\begin{aligned} \tilde{\boldsymbol{\nu}}_k &= \left|\boldsymbol{\nu_k} - z_{kj}\frac{\boldsymbol{\sigma_k}^*}{\boldsymbol{\nu}_k}\right| \\ \tilde{\boldsymbol{\sigma}}_k &= \boldsymbol{\sigma_k} - z_{kj}\frac{|\boldsymbol{\sigma}_k|^2}{\boldsymbol{\nu}_k^2} \end{aligned} \tag{4.21}$$

Invertibility and image of $\tilde{\nu}_k$ A continuous function that maps real numbers to real numbers is invertible if and only if it is strictly monotone. We perform approximate strict monotonicity checking of $\tilde{\nu}_k$ by verifying that the components of the vector $\tilde{\boldsymbol{\nu}}_k$ form a strictly monotone sequence. We assume that for a

large enough d, strict monotonicity of the sequence is a good indicator for the strict monotonicity of the function. Approximating image($\tilde{v}_k$) is also based on the discretization above:

$$\text{image}(\tilde{v}_k) \approx [\min(\tilde{\mathbf{v}}_k), \max(\tilde{\mathbf{v}}_k)]. \tag{4.22}$$

Representing ϕ_k Since we need to be able to compute $\phi_k(\cdot)$ at arbitrary points, we approximate these functions with cubic splines. We can verify, by induction on the structure of the tree T, that the functions ϕ_k are all piecewise smooth, and thus can be well-approximated by cubic splines [21, Ch. XII]. More specifically, since $\phi_k = \tilde{\sigma}_k \circ \tilde{v}_k^{-1}$, it satisfies

$$\phi_k((\tilde{\mathbf{v}}_k)_r) = (\tilde{\boldsymbol{\sigma}}_k)_r, \quad r = 1, \ldots, d. \tag{4.23}$$

Therefore, for any k the function ϕ_k is represented using an interpolating cubic spline, whose domain is the interval in equation (4.22) and approximates the equations in (4.23).

The spline's knots are the interpolation points, and end conditions are specified by the well-known 'not a knot' method [21, Ch. IV, pp. 43–48]. These parameters were chosen since they are widely used with spline interpolation, and have readily available implementations, such as the `csapi` MATLAB function [48].

4.1.9 Handling non-leaf PV-constrained nodes

To demonstrate the concept of our algorithm, we made a simplifying assumption which requires PV constrained nodes to be among the leaves of the rooted tree which defines the network. However, the tree-reduction theorem, and the resulting algorithms, can be easily extended for internal PV constrained nodes.

It can be easily shown that in Theorem 1, replacing the assumption that C_j is a PQ constraint an assumption that it is a PV constraint, defined by

$$\hat{v}_j(t) = \hat{u}_j, \quad \hat{\sigma}_j(t) = \hat{p}_j + \left((1-t)\underline{q}_j + t\overline{q}_j\right)\mathbf{i},$$

results in a theorem that differs from the original in the following manner:

- Equation (4.3) is replaced by

$$U_j = \{\hat{u}_j\} \cap \left(\bigcap k \in \text{ch}_T(j)\, \text{image}(\tilde{v}_k)\right);$$

- Equation (4.5) is replaced by

$$v_j(t) = \hat{u}_j;$$

- Equation (4.6) is replaced by

$$\sigma_j(t) = \hat{\sigma}_j(t) + \sum_{k \in \text{ch}_T(j)} \phi_k(\hat{u}_j).$$

In other words, after a reduction via a PV constrained node, we obtain a PV constrained leaf. The Tree Reduction/Expansion method is modified accordingly.

4.1.10 Eliminating the invertibility assumption

Eliminating the invertibility assumption requires us first to analyze the tree reduction/expansion method from a high level perspective. Consider a CRN P_{input} which is given as an input to the tree reduction method. Let Φ be a mapping from the nodes of P_{input} to their associated voltage transfer function, e.g. $\phi_k \equiv \Phi(k)$. The tree reduction method begins with an empty Φ, and each subsequent reduction results in a smaller CRN, while the voltage transfer functions associated with the discarded nodes are added to Φ.

Assume that at some stage during the execution of the tree reduction method on P_{input} we have the CRN $P = (T, \mathbf{z}, \{C_j\}_{j \in \mathcal{V}})$ and the voltage transfer functions Φ. The invariant maintained by the tree reduction method, which is ensured by Theorem 1, is that at each stage the pair (P, Φ) is a full characterization of $\mathcal{F}_v(P_{\text{input}})$. Formally, we write $\mathcal{F}_v(P, \Phi) = \mathcal{F}_v(P_{\text{input}})$, where $\mathcal{F}_v(P, \Phi)$ is the set of all voltage vectors obtainable by taking any $\mathbf{v}' \in \mathcal{F}_v(P)$ and applying the tree expansion method using the voltage transfer functions Φ.

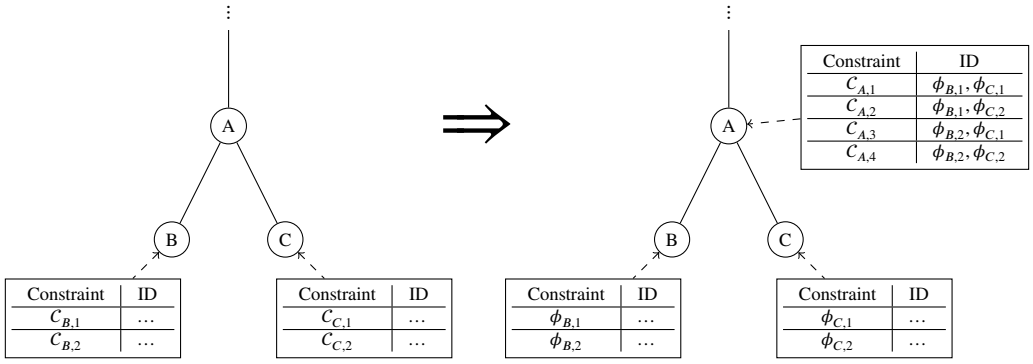


Figure 4.6: A reduction via node A on all network in parallel. *On the left*: two constraints associated with node B and two with node C, representing four networks. *On the right*: the result of the reduction. Four networks were reduced into four smaller networks. Node A is associated with four possible constraint curves, each one has an associated network ID in the form of the specific choice of child voltage transfer functions.

Now assume that the next reduction is be performed via node j, and for some $k \in \mathrm{ch}_T(j)$ the function $\tilde{\nu}_k$ is not invertible. Clearly, the tree reduction theorem cannot be applied to P and the method will fail. However, assuming that $\tilde{\nu}_k$ has a finite number of extremum points, we can overcome this limitation. Let $0 = z_0 < z_1 < \cdots < z_\ell = 1$ be the extremum points of $\tilde{\nu}_k$. We can split the domain $\mathrm{dom}(\tilde{\nu}_k) = [0,1]$ into the sub-domains

$$[z_0, z_1],\ [z_1, z_2],\ \ldots,\ [z_{\ell-2}, z_{\ell-1}],\ [z_{\ell-1}, z_\ell].$$

On each of these domains $\tilde{\nu}_k$ is clearly strictly monotone and therefore invertible. This observation allows us to split the CRN P into the CRNs $P_1, \ldots, P_\ell$, in which the invertibility assumption, at least for node k, holds. For every $\rho \in \{1, \ldots, \ell\}$ we define the 'reparameterized' versions of ν_k, σ_k,

$$\begin{aligned} \nu_{k,\rho} &: [0,1] \to \mathbb{R},\ \nu_{k,\rho}(t) \equiv \nu_k((1-t)z_{\rho-1} + tz_\rho), \\ \sigma_{k,\rho} &: [0,1] \to \mathbb{C},\ \sigma_{k,\rho}(t) \equiv \sigma_k((1-t)z_{\rho-1} + tz_\rho), \\ & C_{k,\rho} \equiv \mathrm{image}(\nu_{k,\rho}, \sigma_{k,\rho}), \end{aligned}$$

and the CRN

$$P_\rho = (T, \mathbf{z}, C_{k,\rho}, \{C_j\}_{j \in \mathcal{V},\ j \neq k}).$$

By construction we have $C_k = \bigcup_{\rho=1}^{\ell} C_{k,\rho}$, and therefore $\mathcal{F}_v(P, \Phi) = \bigcup_{\rho=1}^{\ell} \mathcal{F}_v(P_\rho, \Phi)$. Thus, the set of pairs $\{(P_\rho, \Phi)\}_{\rho=1}^{\ell}$ remains a full characterization of $\mathcal{F}_v(P_{\mathrm{input}})$. The splitting process can continue with respect to each of the child nodes of j, until all CRNs satisfy the assumptions of the tree reduction theorem. Then, the tree reduction method can continue by performing the reduction via node j on all the pairs $\{(P_\rho, \Phi)\}$ in parallel.

Equipped with this idea, we can modify the tree reduction method to operate on a set of pairs instead of a single pair. Pairs are added to the set when a CRN is split as described above. A pair (P', Φ') is removed from the set when the algorithm concludes that $\mathcal{F}_v(P') = \emptyset$.

We call a network for which the above modification of the tree reduction method encounters only functions $\tilde{\nu}_k$ with a finite number of extremal points a **well behaved** network. In our experiments we did not encounter networks which are not well behaved up to the discretization accuracy, that is, networks in which the arrays representing $\tilde{\nu}_k$ had more than a few extremal points.

Implementation

The modification of the tree reduction method mentioned above raises several practical issues: detection of the extremal points, efficient representation of the set of CRNs at each stage, and the memory and time complexity it induces.

Extremal points detection In an actual implementation, the functions ν_k, σ_k are represented by their discrete approximations. For simplicity, our approach assumes that the extermal points of $\tilde{v}_k$ are the extremal elements in the array $\tilde{\boldsymbol{v}}_k$ which is defined in (4.20) and approximates $\tilde{v}_k$.

Set of pairs representation Note that when a pair (P, Φ) is split, the resulting pairs share a lot of data. In fact, they share all data, except for the constraint sets (which can also be represented by curves) that were split. Instead of duplicating entire networks, we associate each node with a *set* of constraint sets or voltage transfer functions, depending on weather or not the node was already discarded by a reduction. Each network is identified by a specific choice of a constraint set $\mathcal{C}_k$ for every leaf k out of the set of constraint sets associated with node k.

Initially, all sets are singletons. When a constraint curve C_k needs to be split, it is removed from the set associated with node k and replaced with the set of constraint sets which result from the above-mentioned splitting process.

Performing a reduction via node j on all networks in parallel is done by considering all possible choices of constraint sets associated with the children of node j, since each choice identifies a CRN. Each of the constraint sets associated with the leaf k is transformed into a voltage transfer function, and then all choices of voltage transfer functions of all the child nodes of j are used to compute the possible constraint sets associated with node j. To identify which network a constraint or a voltage transfer function belongs to, we store a 'pointer' to the specific child voltage transfer functions that were used to compute it. This process is described in Figure 4.6.

4.2 Solving fully constrained problems

A possible application of our algorithm is *reliably* solving a family of OPF problems on radial networks which have a finite number of feasible solutions, including the classical power flow problem (PF-C) on radial networks, presented in the literature review chapter (Section 2.1). When referring to the tree reduction method, we consider the version described above, which runs on several networks in parallel in order to remove the invertability assumption.

Recall, that the OPF problem on a restricted radial network is one-dimensional. The fact that $\mathcal{F}_{vs}(P)$ for a restricted radial network $P = (T, \mathbf{z}, \{C_j\}_{j\in\mathcal{V}})$ is a set of one-dimensional curves is, of course, not a coincidence. We consider a family of 'zero-dimensional' problems obtained when the objective f imposes an additional equality constraint. The feasible set for these problems is *finite*.

We consider three families of fully constrained problems. The first family, which includes the classical power-flow problem, is obtained by imposing a voltage constraint on the slack node r, that is $f(\mathbf{v}, \mathbf{s}) = \delta_{\hat{u}_r}(u_r) + \hat{f}(\mathbf{v}, \mathbf{s})$. Explicitly written, the problem is

$$\begin{aligned} \min_{\mathbf{v},\mathbf{s}} \quad & \hat{f}(\mathbf{v}, \mathbf{s}) \\ \text{s.t.} \quad & (\mathbf{v}, \mathbf{s}) \in \mathcal{F}_{vs}(P), \\ & u_r = \hat{u}_r \end{aligned} \tag{OPF-FC1}$$

This family is directly solved by the tree reduction/expansion method: we run the tree reduction method on P to characterize $\mathcal{F}_{vs}(P)$ by a set of terminal networks and their corresponding voltage transfer functions, and then run the tree expansion method with $\alpha = \hat{u}_r$ on each terminal pair to obtain $(\mathbf{v}, \mathbf{s}) \in \mathcal{F}_{vs}(P)$ that satisfies $u_r = \hat{u}_r$. Each terminal network and its corresponding voltage transfer function either produce a single pair $(\mathbf{v}, \mathbf{s})$, or report that the problem with the additional constraint is infeasible. Out of the set of feasible pairs, we pick a pair having the smallest value of $\hat{f}$.

The second family is obtained by imposing an additional constraint on a PQ constrained *leaf* node k. In addition to the existing constraint $s_k = \hat{s}_k$, we impose the constraint $u_k = \hat{u}_k$. The corresponding OPF problem can be written as

$$\begin{aligned} \min_{\mathbf{v},\mathbf{s}} \quad & \hat{f}(\mathbf{v}, \mathbf{s}) \\ \text{s.t.} \quad & (\mathbf{v}, \mathbf{s}) \in \mathcal{F}_{vs}(P), \\ & u_k = \hat{u}_k \end{aligned} \tag{OPF-FC2}$$

The problem is solved by a reduction to a problem of the first family (OPF-FC1), which is done in the following manner by utilizing the Leaf Reduction Lemma (Lemma 2) and the CRN Reduction Theorem:

1. locate the nodes $w_1, w_2, \ldots, w_N$ on the path from the leaf k to the root, namely, $w_1 = k$ and $w_N = \text{root}(T)$.
2. Run the tree reduction method. Obtain a set of terminal tuples $\{(P_r, \mathcal{U}_r, \Phi_r)\}_{r=1}^{\ell}$, where $\mathcal{U}$ is the set of the intervals U_j for each non-leaf node j computed by the tree reduction method.
3. For each terminal pair do:
   (a) set $u_k = \hat{u}_k$ and $s'_k = \hat{s}_k$
   (b) for $i = 1$ to $N - 1$:
      i. Apply equation (3.1): Compute $u_{w_{i+1}} = \left| u_{w_i} - \frac{z_{w_i,w_{i+1}} {s'_{w_i}}^*}{u_{w_i}} \right|$
      ii. If $u_{w_{i+1}} \notin U_{w_{i+1}}$, skip to the next terminal pair (infeasible)
      iii. otherwise, compute t such that $\nu_{w_{i+1}}(t) = u_{w+1}$, and set $s'_{w_{i+1}} = \sigma_{w_{i+1}}(t)$

At the end of the algorithm above, for each terminal pair we will obtain the corresponding root voltage u_r, which can be fed to the tree expansion method, or conclude that for that terminal pair the constraint $u_k = \hat{u}_k$ makes the problem infeasible.

4.3 Theoretical analysis

In the implementable version of the tree reduction/expansion method we use two discretization densities - m for discretizing the feasible set while solving the optimization problem given as input to Algorithm 3, and d for approximating the functions ϕ_k which characterize the feasible set as described in Section 4.1.8. In this section we assume that d is large enough so that the feasible set is characterized accurately, and analyze convergence w.r.t m. This assumption is backed by our numerical experiments in Section 4.4, which show that for a moderate value of m, the tree expansion method obtains points which are practically feasible.

Given a restricted radial network P with topology tree $T = (\mathcal{V}, \mathcal{E})$ and objective f, the tree reduction/expansion method characterizes $\mathcal{F}_{vs}(P)$ using a single one-dimensional curve, that is, a continuous function $\gamma : [0,1] \to \mathbb{C}^{\mathcal{V}} \times \mathbb{C}^{\mathcal{V}}$ such that $\mathcal{F}_{vs}(P) = \text{image}(\gamma)$. The tree expansion method is, in fact, an algorithm to evaluate $\gamma(\alpha)$ at any given point α. When extended to work without the invertability assumption, the method characterizes $\mathcal{F}_{vs}(P)$ using a finite number of one-dimensional curves, namely, the method finds a representation of continuous functions $\{\gamma_i\}_{u=1}^{n}$, such that

$$\mathcal{F}_{vs}(P) = \bigcup_{i=1}^{n} \text{image}(\gamma_i).$$

The number of curves n is exactly the number of terminal networks which remain after the extended tree reduction method runs on several CRNs in parallel. The optimal value of the OPF problem on the network P is, thus,

$$f^* = \inf_{\mathbf{v},\mathbf{s}} \{ f(\mathbf{v},\mathbf{s}) : (\mathbf{v},\mathbf{s}) \in \mathcal{F}_{vs}(P) \} = \inf_{t \in [0,1]} \left\{ \underbrace{\min_{i \in \{1,\ldots,n\}} f(\gamma_i(t))}_{\Gamma(t)} \right\}, \tag{4.24}$$

while the value produced by the tree reduction/expansion method is

$$f_m = \inf_{t \in [0,1]_m} \Gamma(t), \tag{4.25}$$

where $[a,b]_m$ denotes the discretization

$$[a,b]_m \equiv \left\{ a + \tfrac{2i-1}{2m}(b-a) \right\}_{i=1}^{m}.$$

Our objective is to study the convergence of f_m to f^* as $m \to \infty$.

Remark. Note, that $[a, b]_m$ does not include the endpoints a and b. We can, of course, sample the function at additional points, including the endpoints, since it clearly does not *worsen* the approximation. However, to prove our desired theoretical properties, we do not require the interval's endpoints.

We divide the study into two parts. For the first part, we define the following concept.

Definition. An extended real-valued function g is called solid piecewise-continuous, if there exists a *finite* partition $\mathrm{dom}(g) = \bigcup_{i \in I} D_i$, such that g restricted to D_i is continuous, and $\mathrm{cl}(D_i) = \mathrm{cl}(\mathrm{int}(D_i))$.

Intuitively, a solid piecewise-continous function is continuous on a finite number of sub-domains which are not 'flat' anywhere. Then, we prove the following result.

Theorem 2 (Grid search convergence)**.** *Suppose that g is a closed extended real-valued function which is solid piecewise-continuous, and that* $\mathrm{dom}(g)$ *is contained in the box* $[0, 1]^\ell$*. Let*

$$g^{(m)} = \min \left\{ g(\mathbf{x}) : \mathbf{x} \in ([0, 1]_m)^\ell \right\}.$$

Then,

$$\lim_{m \to \infty} g^{(m)} = \min g.$$

If we apply the theorem above with $g = \Gamma$ and $\ell = 1$, with the convention that $\Gamma \equiv +\infty$ outside the interval $[0, 1]$, we obtain our desired proof of convergence. However, Γ has to be solid piecewise continuous, and the second part of the analysis section is devoted to studying the assumptions on the power network's objective f implying that Γ almost satisfies the required assumption: we show that Γ is piecewise continuous, but the requirement that each region of continuity D_i is solid, i.e. $\mathrm{cl}(D_i) = \mathrm{cl}(\mathrm{int}(D_i))$, remains an assumption we need to make.

4.3.1 Grid search convergence

We prove Theorem 2 using tools from functional analysis regarding convergence of function sequences. A simpler proof can be derived directly, and the reason for using the more advanced tools is to derive results which will become useful later, when we analyze our algorithm for networks with several generators. Throughout this subsection, we denote by B the box $B \equiv [0, 1]^\ell$, and its grid discretization by $[B]_m = ([0, 1]_m)^\ell$.

Definition ([4], Section 4)**.** The extended-valued function g is the *epigraphical limit* of the sequence $\{g_m : m \in \mathbb{N}\}$ of extended-valued functions, denoted by $g = \text{epi-lim}_{m \to \infty} g_m$, if

(i) for any sequence $\{\mathbf{x}_m : d \in \mathbb{N}\}$ converging to $\mathbf{x}$, we have $\liminf_{m \to \infty} g_m(\mathbf{x}_m) \geq g(\mathbf{x})$, and,

(ii) there exists a sequence $\{\mathbf{x}_m : m \in \mathbb{N}\}$ converging to $\mathbf{x}$ such that $\limsup_{m \to \infty} g_m(\mathbf{x}_m) \leq g(\mathbf{x})$.

Definition (Eventual level boundedness)**.** The sequence $\{g_m : d \in \mathbb{N}\}$ of extended-valued functions is *eventually level-bounded* if for each $\alpha \in \mathbb{R}$ there exists $d_0 \in \mathbb{N}$ such that the sequence of sets $[g_m \leq \alpha]$ is bounded for all $d \geq d_0$.

The usefulness of the above concept comes from the following result.

Theorem 3 (Convergence of infima)**.** *Suppose that the function sequence $\{g_m : d \in \mathbb{N}\}$ is eventually level bounded, that $g = \text{epi-lim}_{m \to \infty} g_m$, and that g_m and g are closed and proper. Then,* $\inf g$ *is finite and* $\inf g = \lim_{m \to \infty} (\inf g_m)$.

Proof. [57], Theorem 7.33 □

Consequently, we will construct a sequence $\{g_m : m \in \mathbb{N}\}$ of approximations of g which satisfy the assumptions of the theorem above, and for which the grid search approximation $g^{(m)}$ is equal to $\inf g_m$. Then, applying the theorem above, we will prove Theorem 2. In fact, our g_m is a variant of the well-known nearest neighbor interpolant for g defined by the following rule given $\mathbf{x}$:

- If $\mathbf{x} \notin B$, then $g_m(\mathbf{x}) = +\infty$.
- Otherwise, $g_m(\mathbf{x}) = \min\{g(\mathbf{y}_1), \ldots, g(\mathbf{y}_k)\}$, where $\mathbf{y}_1, \ldots, \mathbf{y}_k$ are the nearest neighbors of $\mathbf{x}$ in the grid $[B]_m$ according to the $\|\cdot\|_\infty$ norm.

We call the resulting function g_m the minimum value nearest neighbor interpolant of g on the data sites $[B]_m$. An alternative way to look at g_m is by partitioning B into equally-sized *cells*, which are boxes of length $\frac{1}{d}$ along each dimension, each centered at a point of the grid $[B]_m$. Formally, the cells are the $\|\cdot\|_\infty$ norm balls

$$\mathcal{B}_\infty\left(\mathbf{y}, \tfrac{1}{2d}\right) = \left\{\mathbf{x} : \|\mathbf{x}-\mathbf{y}\|_\infty \leq \tfrac{1}{2d}\right\} \quad \mathbf{y} \in [B]_m.$$

If $\mathbf{x}$ lies in the interior of some cell $\mathcal{B}_\infty\left(\mathbf{y}_k, \frac{1}{2d}\right)$, its center $\mathbf{y}_k$ is the only nearest neighbor, and the value of $g_m(\mathbf{x})$ is the constant $g(\mathbf{y}_k)$. On the boundary between several cells we have several nearest neighboring grid points, and we take the minimum among the values associated with the neighboring box centers. Formally, with the convention that the $\min \emptyset = +\infty$ we have

$$\begin{aligned} g_m(\mathbf{x}) &= \min_{\mathbf{y}\in[B]_m} \left\{ f(\mathbf{y}) : \mathbf{x} \in \mathcal{B}_\infty\left(\mathbf{y}, \tfrac{1}{2d}\right)\right\} \\ &= \min_{\mathbf{y}\in[B]_m} \left\{ f(\mathbf{y}) + \delta_{\mathcal{B}_\infty\left(\mathbf{y}, \frac{1}{2d}\right)}(\mathbf{x})\right\} \end{aligned}$$

At the core of applying Theorem 3 lies the relation $g = \text{epi-lim}_{m\to\infty} g_m$, which is proved by the next theorem.

Lemma 5. *Let $\mathbb{E}$ be an Euclidean space, and suppose that $g : \mathbb{E} \to (-\infty, +\infty]$ is a solid piecewise continuous function whose domain is contained in B. Let $\{g_m : d \in \mathbb{N}\}$ be a sequence such that g_m is the minimum value nearest neighbor interpolant of g on the data sites $[B]_m$. Then, $g = \text{epi-lim}_{m\to\infty} g_m$.*

Proof. We show the result by showing that both parts of the definition of epi-lim hold.

1. Let $\{\mathbf{x}_m : m \in \mathbb{N}\}$ be a sequence converging to $\mathbf{x}$.

   Suppose that $\mathbf{x} \notin B$. Then, for some m_0 we have $\mathbf{x}_m \notin B$ for all $m \geq m_0$, meaning that $g_m(\mathbf{x}_m) = +\infty$ for all $m \geq m_0$. Thus, in this case,

   $$\liminf_{m\to\infty} g_m(\mathbf{x}_m) = +\infty = g(\mathbf{x}).$$

   Suppose that $\mathbf{x} \in B$, and let $\{m_1, m_2, \dots\}$ be the sequence of indices satisfying $\mathbf{x}_{m_k} \in B$. If the sequence above is finite, then

   $$\liminf_{\nu\to\infty} g_m(\mathbf{x}_m) = +\infty \geq g(\mathbf{x}).$$

   If the sequence above is infinite, let $\mathbf{y}_k$ be a nearest neighbor of $\mathbf{x}_{m_k}$ satisfying $g(\mathbf{y}_k) = g_m(\mathbf{x}_{m_k})$. Since $\|\mathbf{x}_{m_k} - \mathbf{y}_k\| \leq \frac{1}{2m}$ and $\mathbf{x}_{m_k} \to \mathbf{x}$, we have $\mathbf{y}_k \to \mathbf{x}$. Combining with the closedness of g we obtain,

   $$\liminf_{m\to\infty} g_m(\mathbf{x}_m) = \liminf_{k\to\infty} g_{m_k}(\mathbf{x}_{m_k}) = \liminf_{k\to\infty} g(\mathbf{y}_k) \geq g(\mathbf{x}).$$

   In all cases, we obtained

   $$\liminf_{m\to\infty} g_m(\mathbf{x}_m) \geq g(\mathbf{x}).$$

2. Let $\mathbf{x} \in \mathbb{E}$, and let $\{D_i\}_{i\in I}$ be the regions of continuity of g.

   Suppose that $\mathbf{x} \notin B$. Take the sequence $\mathbf{x}_m = \mathbf{x}$, which clearly converges to $\mathbf{x}$. By definition of g_m, we have $g_m(\mathbf{x}_m) = +\infty$, and hence

   $$\limsup_{m\to\infty} g_m(\mathbf{x}) = +\infty = g(\mathbf{x}).$$

   Suppose that $\mathbf{x} \in B$. We have two cases:

   - Suppose that $\mathbf{x} \notin \text{dom}(g)$. Take $\mathbf{x}_m$ to be the nearest neighbor of $\mathbf{x}$ selected when evaluating $g_m(\mathbf{x})$. Since $\text{dom}(g)$ is closed $\mathbb{E} \setminus \text{dom}(g)$ is open, and there is a neighborhood (ball) U of $\mathbf{x}$ such that $U \cap \text{dom}(g) = \emptyset$. Thus, there exists m_0 such that for all $m \geq m_0$ we have $\mathbf{x}_m \in U$, and therefore, for all $m \geq m_0$ we have $g_m(\mathbf{x}) = g(\mathbf{x}_m) = +\infty$. Hence,

     $$\limsup_{m\to\infty} g_m(\mathbf{x}_m) = +\infty = g(\mathbf{x}).$$

- Suppose that $\mathbf{x} \in \mathrm{dom}(g)$, and let $U(r) = \{\mathbf{z} : \|\mathbf{z} - \mathbf{x}\| < r\}$. Since the sets $\{D_i\}_{i\in I}$ form a partition of $\mathrm{dom}(g)$, there is a unique j for which $\mathbf{x} \in D_j$, and hence $\mathbf{x} \in \mathrm{cl}(D_j)$. Since $\mathrm{cl}(D_j) = \mathrm{cl}(\mathrm{int}(D_j))$, there exists m_0 such that for all $m \geq m_0$ we have $U_m \equiv U(1/m) \cap \mathrm{int}(D_j) \neq \emptyset$. Since U_m is open, there exists k_m such that there is a grid point $\mathbf{x}_m \in [B]_{k_m}$ which is also in U_m, meaning that $\mathbf{x}_m \in \mathrm{int}(D_j)$. For any $m < m_0$, let $\mathbf{x}_m = \mathbf{x}_{m_0}$. By construction, $\|\mathbf{x} - \mathbf{x}_m\| < \frac{1}{m}$ for all $m \geq m_0$, and hence the sequence $\{\mathbf{x}_m : m \in \mathbb{N}\}$ converges to $\mathbf{x}$. By continuity of g on D_j, we have

$$\limsup_{m\to\infty} g_m(\mathbf{x}_m) = \limsup_{m\to\infty} g(\mathbf{x}_m) = g(\mathbf{x}).$$

□

Proof of Theorem 2. By construction, we have $g^{(m)} = \inf g_m$, where g_m is the minimum value nearest neighbor interpolant of g on the data sites $[B]_m$. Moreover, each sub-level set $[g_m \leq \alpha]$ is a finite union of grid cells, which are norm balls. Thus, the sequence $\{g_m : m \in \mathbb{N}\}$ is level bounded. Since g is solid piecewise continuous, by Lemma 5 we have $g = \text{epi-lim}_{m\to\infty} g_m$. Finally, by Theorem 3, we have

$$\lim_{m\to\infty} g^{(m)} = \lim_{m\to\infty} (\inf g_m) = \inf g.$$

□

4.3.2 Analysis of Γ

We turn to the second part, in which we analyze the conditions which let us apply Theorem 2 to the function Γ defined at the beginning of this section. Recall, that

$$\Gamma(t) = \min_{i\in\{1,\ldots,n\}} f(\gamma_i(t)),$$

where γ_i are the curves describing the set $\mathcal{F}_{vs}(P)$.

The analysis relies on the concept of semi-algebraic sets and functions. We provide a brief review of the subject.

Definition.

- A set S is called semi-algebraic if it is of the form

$$S = \bigcup_{i=1}^{n} \bigcap_{j=1}^{m_i} \{\mathbf{x} : p_{ij}(\mathbf{x}) > 0, q_{ij}(\mathbf{x}) = 0\},$$

  where p_{ij}, q_{ij} are polynomials.

- A function $g : C \to \mathbb{R}^m$ with $C \subseteq \mathbb{R}^n$ called semi-algebraic, if its graph

$$\{(\mathbf{x}, \mathbf{t}) \in C \times \mathbb{R}^m : \mathbf{t} = g(\mathbf{x})\}$$

  is a semi-algebraic set.

Remark. Note that the definition assumes functions taking and returning real numbers. Therefore, in the context of semi-algebraic functions: when g is an extended real-valued function, it is treated as a regular function $g : \mathrm{dom}(f) \to \mathbb{R}$; when g accepts or returns a complex argument, it is treated as two separate real arguments - the real and imaginary parts.

Examples of semi-algebraic sets include polyhedra, ℓ_p norm-balls, and intersections, complements, and unions of semi-algebraic sets. According to the well-known Tarski-Seidenberg theorem, a projection of a semi-algebraic set onto a subset of the coordinates is also semi-algebraic. Examples of semi-algebraic functions include polynomials, absolute value, square root, and sums, products, integer powers, and compositions of semi-algebraic functions. More properties, theorems, and examples can be found in [18].

Recalling the CRN reduction theorem, it is easy to verify using the above composition rules that each of the voltage transfer functions $\{\phi_k\}$ produced by the tree reduction method is semi-algebraic. Recalling

the tree expansion method, using similar arguments we conclude that the curves $\{\gamma_i\}_{i=1}^n$ evaluated by the tree expansion method are also semi-algebraic. Thus, if the CRN's objective f is semi-algebraic, then for each $i \in \{1,\dots,n\}$ the function $f(\gamma_i(\cdot))$ is semi-algebraic as a composition of two semi-algebraic functions, and therefore a set defined by the relationship $y = f(\gamma_i(t))$ is semi-algebraic. To show that

$$\Gamma(t) = \min_{i\in\{1,\dots,m\}} f(\gamma_i(t))$$

is also semi-algebraic, we characterize its graph as the projection of the following set

$$\begin{aligned}&\{(t,y,\mathbf{w}) : 0 \le t \le 1,\ w_1 = f(\gamma_1(t)),\dots,w_m = f(\gamma_m(t)),\ y = \min\{w_1,\dots,w_m\}\}\\ =&\bigcap_{i=1}^m \{(t,y,\mathbf{w}) : 0 \le t \le 1,\ w_i = f(\gamma_i(t))\} \cap \bigcap_{i=1}^m \{(t,y,\mathbf{w}) : y \le w_i\} \cap \bigcup \{(t,y,\mathbf{w}) : y = w_i\}\end{aligned}$$

onto the (t, y) coordinates.

The semi-algebraicity of Γ is useful due to the following result.

Lemma 6 ([18], Exercise 2.22)**.** *Let $g : C \to \mathbb{R}$ be a semi-algebraic function. Then, there exists a finite partition $C = \bigcup_{i=1}^n C_i$, such that f restricted to C_i is continouus.*

Consequently, we conclude that *if the CRN's objective f is semi-algebraic, then Γ must be piecewise-continuous*. What we cannot conclude is that Γ is *solid* piecewise continuous, that is, we cannot conclude that for each region of continuity C_i we have $\mathrm{cl}(C_i) = \mathrm{cl}(\mathrm{int}(C_i))$. Moreover, if f is closed, then $f(\gamma_i(\cdot))$ is closed since γ_i is continuous for every i. Thus, in that case, Γ is closed as a finite minimum of closed functions. Summarizing the above, and using Theorem 2, we have reached the following conclusion.

Corollary 1. *Suppose that the CRN's objective f is closed and semi-algebraic. Then, Γ is closed, and there exists a finite partition $\mathrm{dom}(\Gamma) = \bigcup_{i=1}^n \mathrm{dom}(C_i)$ such that Γ restricted to each C_i is continuous.*

Moreover, if for each $i = 1,\dots,n$ we have $\mathrm{cl}(C_i) = \mathrm{cl}(\mathrm{int}(C_i))$, then the value $f^{(d)}$ produced by the tree reduction/expansion method converges to the optimal value f^* as $m \to \infty$.

4.4 Numerical experiments

We evaluated our implementation of the tree reduction/expansion method for solving the OPF problem on several cases to demonstrate the following features of our method:

- **Accuracy:** show that a moderate value of the approximation density d is enough for the pairs $(\mathbf{v}, \mathbf{s})$ produced by our method to be feasible.
- **Reliability:** we always find a global optimum for feasible problems, in contrast to MATPOWER [72], which to the best of our knowledge is considered as state of the art.
- **The number of parallel CRNs in practice:** verify that the number of CRNs on which the tree reduction method operates concurrently remains small.

We performed our numerical experiments on existing networks that were slightly modified to satisfy all the assumptions required from restricted radial networks. Specifically, we used the IEEE radial distribution networks with 13, 37, and 123 buses available at [56], a network with 47 buses used in [24], and a network with 69 buses used in [15]. We made our MATLAB implementation of the tree reduction/expansion method as an open source project on GitHub[1], together with the networks we used for our experiments.

4.4.1 Accuracy

In order to say that a pair $(\mathbf{v}, \mathbf{s})$ produced by our method is "practically feasible" for a restricted radial network P, we need to define some measures of deviation from feasibility. The definition of these measures

[1] `https://github.com/alexshtf/trem_opf_solver`

uses the following ingredients: the set of PQ-constrained nodes PQ, the set of PV-constrained nodes PV, the distance function

$$d(x;[\alpha,\beta]) = \begin{cases} x-\beta & x>\beta \\ \alpha - x & x<\alpha \\ 0 & otherwise, \end{cases}$$

which measures the distance of x from the interval $[\alpha,\beta]$. With the ingredients above, we define the following measures for deviation from the constraints of PQ constrained nodes:

$$E_{PQ,v}(\mathbf{v},\mathbf{s}) = \max_{j\in\mathcal{PQ}} d\left(|v_j|,[\underline{u}_j,\overline{u}_j]\right),$$
$$E_{PQ,s}(\mathbf{v},\mathbf{s}) = \max_{j\in\mathcal{PQ}} |s_j - \hat{s}_j|,$$

and the following measures for PV constrained nodes:

$$E_{PV,v}(\mathbf{v},\mathbf{s}) = \max_{j\in\mathcal{PV}} |\hat{u}_j - |v_j||$$
$$E_{PV,p}(\mathbf{v},\mathbf{s}) = \max_{j\in\mathcal{PV}} |\hat{p}_j - \mathrm{re}(s_j)|$$
$$E_{PV,q}(\mathbf{v},\mathbf{s}) = \max_{j\in\mathcal{PV}} d\left(\mathrm{im}(s_j),[\underline{q}_j,\overline{q}_j]\right)$$

Clearly, for a pair $(\mathbf{v},\mathbf{s}) \in \mathcal{F}_{vs}(P)$ all of the above measures are zero. Note that we do not measure the deviation from the constraints imposed by the power flow equations, since ,by definition, our method computes $\mathbf{s} = \mathcal{Q}_P(\mathbf{v})$.

For every network, we chose a set of approximation densities $d \in [2^3, 2^{12}]$ as inputs to the tree reduction method (Algorithm 1), and for each such density produced $m = 1000$ samples of the feasible set using the tree-expansion method (Algorithm 2), and took the maximum of each error measure over all the samples. In all our experiments, the measures $E_{PQ,v}$ and $E_{PV,q}$ were always zero. The results for the other measures, which appear in Figure 4.7, show that for $d \geq 2^{10}$ we obtain solutions which are practically feasible. In addition, very accurate results are obtained even for d as small as 2^5.

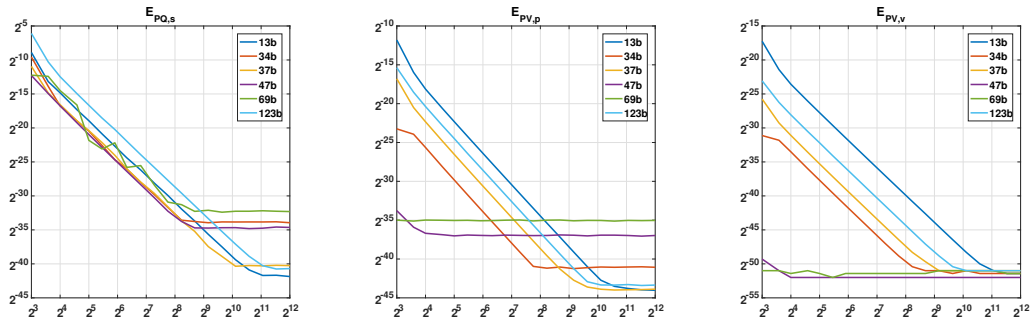


Figure 4.7: Constraint violation measures for different networks.

4.4.2 Reliability

We compared our method to MATPOWER on networks which were randomly generated from our existing networks by pertubing every PQ constraint $C_j = [\underline{u}_j,\overline{u}_j] \times \{\hat{p}_j + \mathbf{i}\hat{q}_j\}$ into

$$[\underline{u}_j,\overline{u}_j] \times \{\hat{p}_j \cdot \alpha_j + (\hat{q}_j \cdot \beta_j)i\},$$

where α_j, β_j are uniformly distributed random variables in $[0,2]$. That is, we introduced uniform multiplicative noise to the prescribed power in the PQ constraints in our networks, while leaving the other constraints unchanged.

From each network, we generated 5000 random networks and solved the OPF problem on each random network using both MATPOWER and the tree reduction/expansion method, with the following *stability* objective function:

$$f(\mathbf{v}, \mathbf{s}) = \sum_{j \in \mathcal{PQ}} \left| |v_j| - \frac{1}{2}(\underline{u}_j + \overline{u}_j) \right|$$

Note that MATPOWER does not naturally support the function above, but its extension mechanism lets us specify this function quite easily. For each network we gathered statistics about the results of running both solvers on the randomly perturbed networks.

For each original network, we measured the percentage of randomly generated networks for which MATPOWER produced a feasible solution while our algorithm claimed that the network is infeasible. However, there were many networks for which MATPOWER could not find a solution since it did not converge, while our method did. Table 4.1 shows the percentage, out of the 5000 random networks we generated from each original networks, for which MATPOWER could not find a solution. As an example, a specific 13-bus network is shown in Figure 4.8. Finally, comparing the objective function value for

13-bus	34-bus	37-bus	47-bus	69-bus	123-bus
6.46%	5.22%	2.56%	2.78%	2.34%	39.64%

Table 4.1: The % of random networks, generated from each original network, for which our method found a solution while MATPOWER did not.

the randomly generated networks for which both algorithms reported a solution resulted in no significant difference. This leads us to the conclusion that for this set of experiments, if MATPOWER finds a solution, it is probably a global solution.

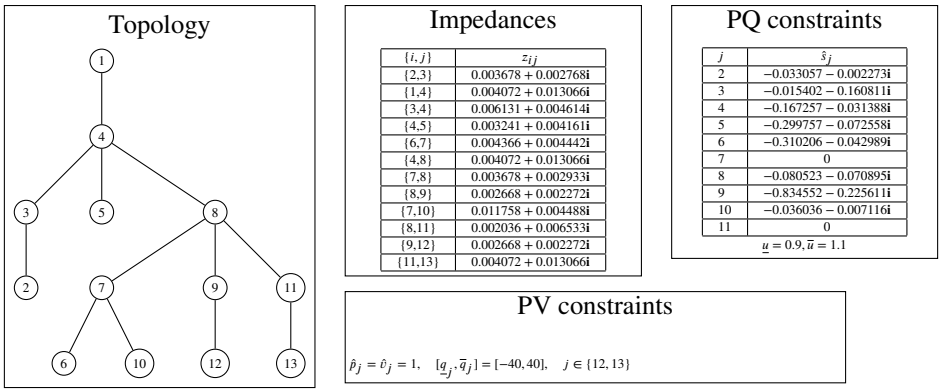


Figure 4.8: A 13-bus network for which MATPOWER does not converge given the OPF problem with the stability objective function.

The results reported above provide empirical evidence for our claim that in contrast to MATPOWER, our method finds the global solution, up to discretization, if one exists.

Chapter 5

Networks with Several Generators

In this chapter we describe an algorithm for solving OPF problems on a wider family of radial networks which may contain multiple generators. The algorithm, which we call the hierarchical decomposition method, bears some similarity to the tree reduction/expansion method described in Chapter 4 in the sense that both methods repeatedly reduce the size of the tree by 'cutting off' an entire sub-tree, and replacing it with a single node which aggregates the information from the sub-tree. If the sub-tree aggregation result could be computed exactly, we would obtain a globally optimal solution. Similarly to the tree reduction/expansion method, to the best of our knowledge the aggregation result cannot be computed exactly and has to be approximated. However, the approximation is not as simple, and a substantial portion of this chapter is devoted to its design. We finish the chapter with numerical experiments which evaluate the method's performance in practice using several approximation densities.

5.1 The model - compact radial networks

We concentrate on a family of radial networks which have compactly bounded nodal constraints for both consumers and generators, and a thermal constraint on each edge, which is defined using the edge power-loss function $\ell_{ij}(\cdot,\cdot)$ presented in Section 1.3.3. Consumers have the standard PQ constraints, while generator constraints are general compact sets, which in practice are usually boxes or simple polyhedral sets. The following definition precisely defines the family, and Figure 5.1 provides an example.

Definition. A power network $(T, \mathbf{z}, F)$ is called a *compact radial network*, if

- The topology graph $T = (\mathcal{V}, \mathcal{E})$ is a tree, whose nodes are partitioned into loads (consumers), and generators, which must be leaves. We denote by $\mathcal{G} \subseteq \mathcal{V}$ the set of generators, and require that every $i \in \mathcal{G}$ is a leaf. The loads are $\mathcal{V} \setminus \mathcal{G}$;
- The cost function F has the structure
$$F(\mathbf{v}, \mathbf{s}) = \sum_{i \in \mathcal{V}} f_i(|v_i|, s_i) + \sum_{(i,j) \in \mathcal{E}} \delta_{[|\ell_{ij}(\cdot,\cdot)| \le \overline{\ell}_{ij}]}(\mathbf{v}, \mathbf{s}),$$
where $\bar{\boldsymbol{\ell}} \in \mathbb{R}^{\mathcal{E}}_{+}$ are some given finite constants satisfying $\overline{\ell}_{ij} = \overline{\ell}_{ji}$;
- The function $f_i : \mathbb{R} \times \mathbb{C} \to (-\infty, +\infty]$ for any $i \in \mathcal{V}$ is closed and bounded;
- If $i \in \mathcal{G}$, then $\mathrm{dom}(f_i)$ is nonempty and compact with $(u, s) \in \mathrm{dom}(f_i) \implies u > 0$;
- If $i \notin \mathcal{G}$ then $\mathrm{dom}(f_i) = \mathrm{PQ}(\underline{u}_i, \overline{u}_i, \hat{s}_i)$, where $0 < \underline{u}_i < \overline{u}_i < +\infty$, and $\hat{s}_i \in \mathbb{C}$ are given constants.

A compact radial network is described using the tuple $(T, \mathbf{z}, \mathcal{G}, \mathbf{f}, \overline{\boldsymbol{\ell}})$, where $\mathbf{f}$ is the vector whose components are the functions f_i.

Compact nodal constraints and thermal constraints are desirable for many applications, and thus are not a restriction in most cases. In contrast, the requirement for the generators to be among the leaves of the topology of the tree may restrict the range of practical applications of our method. However, there are still many examples where the generators are among the leaves. In particular, such networks are common in small power grids, such as shipboard power grids (see [6, 23]).

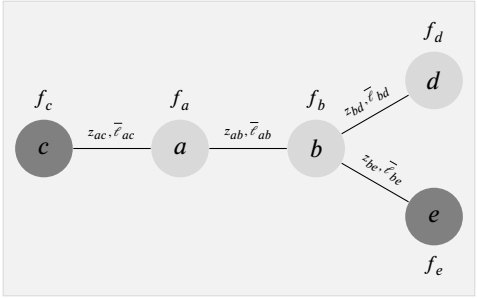


Figure 5.1: Compact radial network example. Each edge has associated impedance and thermal constraint bound. Each node has an associated nodal objective. Dark gray nodes are the generators.

5.2 Problem reformulation

Let $P = (T, \mathbf{z}, \mathcal{G}, \mathbf{f}, \overline{\boldsymbol{\ell}})$ be a compact radial network. We reformulate the OPF problem on P in a manner which is closely related to the hierarchy induced by the topology tree T. We begin our reformulation by dedicating a separate vector of decision variables to the voltage absolute values. The OPF problem on P can be written, equivalently, as

$$\begin{aligned} \min_{\mathbf{u},\mathbf{v},\mathbf{s}} \quad & \sum_{i\in\mathcal{V}} f_i(u_i, s_i) \\ \text{s.t.} \quad & |\mathbf{v}| = \mathbf{u}, \\ & (\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(T, \mathbf{z}), \\ & |\ell_{ij}(\mathbf{v}, \mathbf{s})| \le \overline{\ell}_{ij} \qquad (i,j) \in \mathcal{E} \end{aligned} \tag{OPF-CR}$$

Note that we may treat the complex voltages $\mathbf{v}$ as an auxiliary variable, and concentrate on the projection of the feasible set onto the $\mathbf{u}, \mathbf{s}$ coordinates. Indeed, $\mathbf{v}$ may be recovered from $\mathbf{u}, \mathbf{s}$, as shown in the proof of the following theorem, whose result is used to exploit the hierarchy induced by the network's topology tree (alternatively, Lemma 3 can be used to recover $\mathbf{v}$).

Theorem 4. *Suppose that $(T, \mathbf{z})$ is a network structure with a topology tree $T = (\mathcal{V}, \mathcal{E})$ rooted at $r = \mathrm{root}(T)$, and $\overline{\boldsymbol{\ell}} \in \mathbb{R}^{\mathcal{E}}_{+}$ is some constant vector with $\overline{\ell}_{ij} = \overline{\ell}_{ji}$. Then $(\mathbf{u}, \mathbf{s}) \in \mathbb{R}^{\mathcal{V}} \times \mathbb{C}^{\mathcal{V}}$ satisfies the following constraints with auxiliary variables $\mathbf{v} \in \mathbb{C}^{\mathcal{V}}$:*

$$\begin{aligned} & \mathbf{u} = |\mathbf{v}|, \\ & (\mathbf{v}, \mathbf{s}) \in \mathrm{PFE}(T, \mathbf{z}), \\ & |\ell_{ij}(\mathbf{v}, \mathbf{s})| \le \overline{\ell}_{ij}, \qquad (i,j) \in \mathcal{E} \end{aligned} \tag{ST-CR}$$

if and only if they satisfy the following system with auxiliary variables $\mathbf{s}' \in \mathbb{C}^{\mathcal{V}}$ comprising of

$$s'_r = 0, \tag{5.1}$$

and for all $i \in \mathcal{V}$ of

$$s'_i = s_i + \sum_{j\in \mathrm{ch}_T(i)} \tilde{s}_j, \tag{5.2}$$

$$|z_{ij}||\tilde{s}_j|^2 \le u_i^2 \overline{\ell}_{ij} \qquad \forall j \in \mathrm{ch}_T(i), \tag{5.3}$$

$$u_i = \left| u_j - \frac{z_{ij} s'^{*}_j}{u_j} \right| \qquad \forall j \in \mathrm{ch}_T(i), \tag{5.4}$$

with

$$\tilde{s}_j \equiv s'_j - z_{ij} \frac{|s'_j|^2}{u_j^2}, \quad i \in \mathcal{V}, j \in \mathrm{ch}_T(i).$$

The proof is technical in nature, and is based on the Full Tree Reduction Lemma and elementary manipulations of complex numbers.

Proof. Suppose that the assumptions of the theorem hold. Then from Lemma 3, the substitution $|v_i| = u_i$, and the fact that for $\{i, j\} \in \mathcal{E}$ either $j \in \mathrm{ch}_T(i)$ or $i \in \mathrm{ch}_T(j)$, we conclude that the pair $(\mathbf{u}, \mathbf{s})$ satisfies (ST-CR) for some $\mathbf{v}$ if and only if there exist $\mathbf{s}'$ such that:

$$|\mathbf{v}| = \mathbf{u}, \tag{5.5a}$$

$$s'_r = 0, \tag{5.5b}$$

$$s'_i = s_i + \sum_{j \in \mathrm{ch}_T(i)} \tilde{s}_j \qquad i \in \mathcal{V}, \tag{5.5c}$$

$$u_i = \left| u_j - \frac{z_{ij} s'^*_j}{u_j} \right| \qquad i \in \mathcal{V},\ j \in \mathrm{ch}_T(i), \tag{5.5d}$$

$$v_j = v_i + \frac{z_{ij} \tilde{s}^*_j}{v^*_i} \qquad i \in \mathcal{V},\ j \in \mathrm{ch}_T(i), \tag{5.5e}$$

$$|\ell_{ij}(\mathbf{v}, \mathbf{s})| \le \overline{\ell}_{ij} \qquad i \in \mathcal{V},\ j \in \mathrm{ch}_T(i), \tag{5.5f}$$

with $\tilde{s}_j$ defined by

$$\tilde{s}_j \equiv s'_j - z_{ij} \frac{|s'_j|^2}{u_j^2} \qquad i \in \mathcal{V}, j \in \mathrm{ch}_T(i)$$

By (5.5e) we have

$$v_i - v_j = -z_{ij} \frac{\tilde{s}^*_j}{v^*_i}.$$

Substituting the above and $u_i = |v_i|$, for nonzero z_{ij} into the constraint

$$|\ell_{ij}(\mathbf{v}, \mathbf{s})| = \frac{|v_i - v_j|^2}{|z_{ij}|} \le \overline{\ell}_{ij}$$

yields equivalence to

$$|\tilde{s}_j|^2 |z_{ij}| \le u_i^2 \overline{\ell}_{ij}.$$

Note, that the constraint above is correct for $z_{ij} = 0$, since in that case $\ell_{ij} \equiv 0$ by definition, and thus both the original and the transformed constraints always hold. With the above transformation, we obtain that the pair $(\mathbf{u}, \mathbf{s})$ satisfies (ST-CR) for some $\mathbf{v}$ if and only if there exists $\mathbf{s}'$ such that:

$$|\mathbf{v}| = \mathbf{u}, \tag{5.6a}$$

$$s'_r = 0, \tag{5.6b}$$

$$s'_i = s_i + \sum_{j \in \mathrm{ch}_T(i)} \tilde{s}_j \qquad i \in \mathcal{V}, \tag{5.6c}$$

$$u_i = \left| u_j - \frac{z_{ij} s'^*_j}{u_j} \right| \qquad i \in \mathcal{V},\ j \in \mathrm{ch}_T(i) \tag{5.6d}$$

$$v_j = v_i + \frac{z_{ij} \tilde{s}^*_j}{v^*_i} \qquad i \in \mathcal{V},\ j \in \mathrm{ch}_T(i), \tag{5.6e}$$

$$|\tilde{s}_j|^2 |z_{ij}| \le u_i^2 \overline{\ell}_{ij} \qquad i \in \mathcal{V},\ j \in \mathrm{ch}_T(i). \tag{5.6f}$$

Finally, we will see that (5.6e) and (5.6a) can be eliminated from the system above by showing that if all other constraints hold, we can always construct $\mathbf{v}$ such that the whole system (5.6) holds. Let $\theta_i = \arg(v_i)$, or equivalently suppose that v_i has the polar form $v_i = u_i e^{\mathbf{i}\theta_i}$. Substituting the polar form into (5.6e), multiplying both sides by $v^*_i = u_i e^{-\mathbf{i}\theta_i}$, and using the definition of $\tilde{s}_j$ we obtain the equivalent equation

$$u_i u_j e^{\mathbf{i}(\theta_j - \theta_i)} = u_i^2 + z_{ij} \tilde{s}^*_j = u_i^2 + z_{ij} s'^*_j - |z_{ij}|^2 \frac{|s'_j|^2}{u_j^2}.$$

By plugging the expression of u_i^2 obtained by squaring both sides of (5.6d) and canceling terms, we conclude that (5.6e) can be replaced with

$$\begin{aligned} u_i u_j e^{\mathbf{i}(\theta_j - \theta_i)} &= u_j^2 - z_{ij}^* s_j' && \leftarrow \text{Plugging } u_i^2 \\ &= u_j \left(u_j - \frac{z_{ij} s_j'^*}{u_j} \right)^* \end{aligned}$$

Denoting $\alpha_j = \arg\left(u_j - \frac{z_{ij} s_j'^*}{u_j} \right)$, and remembering (5.6d), we have

$$u_i u_j e^{\mathbf{i}(\theta_j - \theta_i)} = u_i u_j e^{-\mathbf{i}\alpha_j},$$

or equivalently

$$e^{\mathbf{i}(\theta_j - \theta_i)} = e^{-\mathbf{i}\alpha_j}.$$

We can *always* make the equation above hold, by choosing the root angle $\theta_r = 0$, and performing a BFS traversal from the root to propagate voltage angles from parent to child using the equation above. Hence, if the system (5.6) without (5.6a) and (5.6e) holds, then there exists $\mathbf{v}$ such that the whole system holds. Since the existance of $\mathbf{v}$ is ensured, we conclude that the pair $(\mathbf{u}, \mathbf{s})$ satisfies (ST-CR) if and only if there exists $\mathbf{s}' \in \mathbb{C}^{\mathcal{V}}$ such that

$$\begin{aligned} & s_r' = 0, \\ & s_i' = s_i + \sum_{j \in \mathrm{ch}_T(i)} \tilde{s}_j && i \in \mathcal{V}, \\ & u_i = \left| u_j - \frac{z_{ij} s_j'^*}{u_j} \right| && i \in \mathcal{V},\ j \in \mathrm{ch}_T(i), \\ & |\tilde{s}_j|^2 |z_{ij}| \le u_i^2 \overline{\ell}_{ij} && i \in \mathcal{V},\ j \in \mathrm{ch}_T(i), \end{aligned}$$

concluding the proof. □

We can use the theorem to reformulate (OPF-CR) on a compact radial network $(T, \mathbf{z}, \mathcal{G}, \mathbf{f}, \overline{\boldsymbol{\ell}})$ with $r = \mathrm{root}(T)$ into our desired form. At this stage, the root r may be chosen arbitrarily. First, we directly apply the theorem to the constraints, resulting in:

$$\begin{aligned} \min_{\mathbf{u},\mathbf{s},\mathbf{s}'} \quad & \sum_{i \in \mathcal{V}} f_i(u_i, s_i) \\ \text{s.t.} \quad & \text{Equation (5.1)} \\ & \text{Equations (5.2), (5.3), and (5.4)} && \forall i \in \mathcal{V} \end{aligned}$$

Note that for any $i \in \mathcal{V}$ the variables which participate in (5.2), (5.3), and (5.4) are the ones having the indices of $\mathrm{F}_i \equiv \{i\} \cup \mathrm{ch}_T(i)$ - the family of i. Next, recalling that voltages are constrained to be positive over $\mathrm{dom}(f_i)$ for every $i \in \mathcal{V}$, denoting

$$\alpha = \min_{i \in \mathcal{V}} \min_{(u,s)} \{ u \,:\, (u, s) \in \mathrm{dom}(f_i) \}, \tag{5.7}$$

and adding the redundant constraints $\mathbf{u}_{\mathrm{F}_i} \ge \alpha$, we obtain the following equivalent optimization problem

$$\begin{aligned} \min_{\mathbf{u},\mathbf{s},\mathbf{s}'} \quad & \sum_{i \in \mathcal{V}} f_i(u_i, s_i) \\ \text{s.t.} \quad & \text{Equation (5.1)}, \\ & \mathbf{u}_{\mathrm{F}_i} \ge \alpha, && \forall i \in \mathcal{V} \\ & \text{Equations (5.2), (5.3), and (5.4)} && \forall i \in \mathcal{V} \end{aligned}$$

Now, we formulate the above using indicator functions. To model the constraints, we introduce the following indicator function for any $i \in \mathcal{V}$:

$$\delta_i(\mathbf{u}_{\mathrm{F}_i}, \mathbf{s}'_{\mathrm{F}_i}, s_i) = \begin{cases} 0 & \mathbf{u}_{\mathrm{F}_i} \geq \alpha, \text{ and eqs. (5.2), (5.3), and (5.4) hold,} \\ +\infty & \text{otherwise,} \end{cases} \tag{5.8}$$

Now, by choosing a root node $r \in \mathcal{V}$, problem (OPF-CR) can be equivalently reformulated into the following form:

$$\min_{\mathbf{u},\mathbf{s},\mathbf{s}'} \; H(\mathbf{u}, \mathbf{s}, \mathbf{s}') = \delta_{\{0\}}(s'_r) + \sum_{i \in \mathcal{V}} \left[f_i(u_i, s_i) + \delta_i(\mathbf{u}_{\mathrm{F}_i}, \mathbf{s}'_{\mathrm{F}_i}, s_i) \right]. \tag{H-OPF}$$

The rest of this chapter is devoted to solving (H-OPF) problem instances.

5.3 The hierarchical decomposition method

Problem (H-OPF) enjoys a special structure which we exploit to devise the hierarchical decomposition method. To concentrate on the structure itself and avoid being distracted by other details, we treat problem (H-OPF) as a special case of the more general hierarchical optimization problem, which is defined below. After presenting the conceptual hierarchical decomposition method for this problem family and studying its basic properties, we return to our main course, and specialize the algorithm to (H-OPF) problems. Throughout this section, unless explicitly defined, $\mathbb{X}$ and $\mathbb{Y}$ are arbitrary Euclidean spaces, e.g. $\mathbb{R}^2$ or $\mathbb{R}^3$.

Definition (Hierarchical optimization problem)**.** Let $T = (\mathcal{V}, \mathcal{E})$ be a rooted tree. A hierarchical optimization problem over $\mathbb{X}$ and $\mathbb{Y}$ is a problem of the form

$$\min_{\substack{\mathbf{X} \in \mathbb{X}^{\mathcal{V}} \\ \mathbf{Y} \in \mathbb{Y}^{\mathcal{V}}}} \quad \sum_{j \in \mathcal{V}} g_j(\mathbf{X}_j, \mathbf{Y}_{\mathrm{F}_j}), \tag{HOP}$$

where g_j are extended real-valued functions. The functions g_j are called the *nodal objectives*. The problem is described by the tuple $(T, \mathbf{g})$.

Remark. $\mathbb{X}^{\mathcal{V}}$ (or $\mathbb{Y}^{\mathcal{V}}$) is a space of vectors indexed by $\mathcal{V}$, whose coordinates are themselves vectors in $\mathbb{X}$. We may think of $\mathbf{X} \in \mathbb{X}^{\mathcal{V}}$ as a matrix, whose columns $\mathbf{X}_j$ are indexed by $j \in \mathcal{V}$, and each column is an element in $\mathbb{X}$. Consequently, for $\mathcal{I} \subseteq \mathcal{V}$, we may think of $\mathbf{X}_{\mathcal{I}}$ as a sub-matrix composed of the columns of $\mathbf{X}$ whose indices are in $\mathcal{I}$. In particular, $\mathbf{Y}_{\mathrm{F}_j}$ is a sub-matrix composed of the columns of $\mathbf{Y}$ whose indices belong to the family of node j.

It is easy to verify that an (H-OPF) problem on the compact radial network $(T, \mathbf{z}, \mathcal{G}, \mathbf{f}, \overline{\boldsymbol{\mathscr{C}}})$ can be written as the hierarchical optimization problem $(T, \mathbf{g})$ with $\mathbb{X} = \mathbb{R}^2$ and $\mathbb{Y} = \mathbb{R}^3$, where:

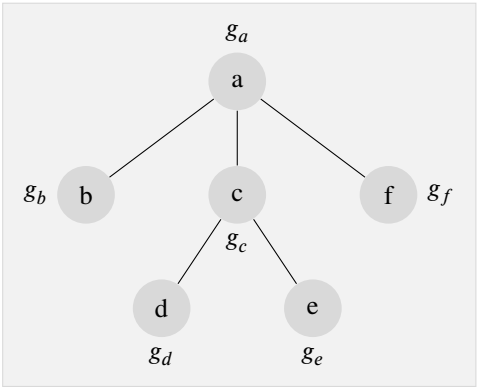
- $\mathbf{X}_j$ encodes $(\mathrm{re}(s_j), \mathrm{im}(s_j))^T$;
- $\mathbf{Y}_j$ encodes $(u_j, \mathrm{re}(s'_j), \mathrm{im}(s'_j))^T$;
- for $r = \mathrm{root}(T)$, the function g_r computes $\delta_{\{0\}}(s'_r) + f_r(u_r, s_r) + \delta_r(\mathbf{u}_{\mathrm{F}_r}, \mathbf{s}'_{\mathrm{F}_r}, s_r)$;
- for $j \neq r$, the function g_j computes $f_i(u_i, s_i) + \delta_i(\mathbf{u}_{\mathrm{F}_i}, \mathbf{s}'_{\mathrm{F}_i}, s_i)$.

Having described the generalization, we deviate away from the discussion of OPF problems, and devote the rest of this section to the description of a conceptual algorithm for solving hierarchical optimization problems.

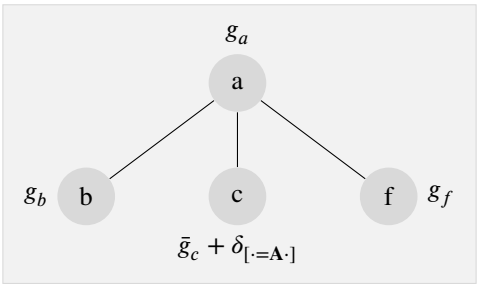
Definition (Sub-tree functions)**.** Let $(T, \mathbf{g})$ be a hierarchical optimization problem with $T = (\mathcal{V}, \mathcal{E})$, let $k \in \mathcal{V}$, and let $\mathcal{I} = \mathrm{sub}_T(k)$ - the nodes in the sub-tree rooted at k. The function $G_k : \mathbb{X}^{\mathcal{I}} \times \mathbb{Y}^{\mathcal{I}} \to (-\infty, +\infty]$ defined by

$$G_k(\mathbf{X}, \mathbf{Y}) = \sum_{j \in \mathcal{I}} g_j(\mathbf{X}_j, \mathbf{Y}_{\mathrm{F}_j})$$

is called the *sub-tree function* associated with node k. In other words, it is the sum of the nodal objectives associated with the sub-tree rooted at k.



(a) The data of an (HOP) problem



(b) The data a truncated hierarchical optimization problem, obtained by truncating the sub-tree rooted at c. The matrix $\mathbf{A}$ is unspecified.

Figure 5.2: Original and truncated (HOP) problems.

Example. For the (HOP) problem illustrated in Figure 5.2a we have $\mathcal{V} = \{a, b, c, d, e, f\}$, the problem is

$$\min_{\substack{\mathbf{X}\in\mathbb{X}^{\mathcal{V}} \\ \mathbf{Y}\in\mathbb{Y}^{\mathcal{V}}}} g_a(\mathbf{X}_a, \mathbf{Y}_a, \mathbf{Y}_b, \mathbf{Y}_c, \mathbf{Y}_f) + g_b(\mathbf{Y}_b, \mathbf{Y}_b) + g_c(\mathbf{X}_c, \mathbf{Y}_c, \mathbf{Y}_d, \mathbf{Y}_e) + g_f(\mathbf{X}_f, \mathbf{Y}_f) + g_d(\mathbf{X}_d, \mathbf{Y}_d) + g_e(\mathbf{X}_e, \mathbf{Y}_e),$$

and the sub-tree function at c is $G_c : \mathbb{X}^{\{c,d,e\}} \times \mathbb{Y}^{\{c,d,e\}} \to (-\infty, +\infty]$ defined by

$$G_c(\mathbf{X}, \mathbf{Y}) = g_c(\mathbf{X}_c, \mathbf{Y}_c, \mathbf{Y}_d, \mathbf{Y}_e) + g_d(\mathbf{X}_d, \mathbf{Y}_d) + g_e(\mathbf{X}_e, \mathbf{Y}_e)$$

Definition. Let $(T, \mathbf{g})$ be a hierarchical optimization problem with $T = (\mathcal{V}, \mathcal{E})$, and let $k \in \mathcal{V}$. The function $\bar{g}_k : \mathbb{Y} \to (-\infty, +\infty]$ defined by

$$\bar{g}_k(\mathbf{y}) = \min_{\mathbf{X},\mathbf{Y}} \{G_k(\mathbf{X}, \mathbf{Y}) : \mathbf{Y}_k = \mathbf{y}\}$$

is called the *absorbed sub-tree function*.

Alternatively, $\bar{g}_k$ is the partial minimum of G_k over all variables, except for $\mathbf{Y}_k$. We assume that the absorbed sub-tree function is well-defined. That is, when referring to $\bar{g}_k$ we implicitly assume that for any $\mathbf{y}$, either:

(i) the minimum in the definition of $\bar{g}_k(\mathbf{y})$ is attained, or

(ii) the minimization problem defining $\bar{g}_k(\mathbf{y})$ is infeasible, meaning that $\bar{g}_k(\mathbf{y}) = +\infty$.

Using the absorbed function, we can formulate problem (HOP) as a bi-level optimization problem using partial minimization: taking any node k, and denoting $\mathcal{J} = \mathcal{V} \setminus \mathrm{sub}_T(k)$, we can reformulate problem (HOP) as

$$\min_{\substack{\mathbf{X}\in\mathbb{X}^{\mathcal{V}} \\ \mathbf{Y}\in\mathbb{Y}^{\mathcal{V}}}} \left\{ \sum_{j\in\mathcal{V}} g_j(\mathbf{X}_j, \mathbf{Y}_{\mathrm{F}_j}) \right\} = \min_{\substack{\mathbf{X}\in\mathbb{Y}^{\mathcal{J}} \\ \mathbf{Y}\in\mathbb{Y}^{\mathcal{J}} \\ \mathbf{Y}_k\in\mathbb{Y}}} \left\{ \bar{g}_k(\mathbf{Y}_k) + \sum_{j\in\mathcal{J}} g_j(\mathbf{X}_j, \mathbf{Y}_{\mathrm{F}_j}) \right\}.$$

Clearly, given a solution of the optimization problem on the RHS, termed the 'outer' problem, we can subtitute the optimal $\mathbf{Y}_k$ into the problem defining $\bar{g}_k$, termed the 'inner' problem, and obtain an optimal solution of the problem on the LHS.

Note, that in the outer problem there is no variable $\mathbf{X}_k$, since it has been eliminated by the partial minimization resulting in $\bar{g}_k$. We reuse the name, and introduce a 'dummy' variable $\mathbf{X}_k = \mathbf{A}\mathbf{Y}_k$ to the outer problem, where $\mathbf{A}$ is some matrix of our choice. Its role is pedagogical in nature - to unify the presentation

of the algorithm, its specialization for (H-OPF) problems, and the associated theoretical analysis, under a single framework. The outer problem becomes

$$\min_{\substack{\mathbf{X}\in\mathbb{X}^{\mathcal{J}\cup\{k\}}\\ \mathbf{Y}\in\mathbb{Y}^{\mathcal{J}\cup\{k\}}}} \left\{ \bar{g}_k(\mathbf{Y}_k) + \delta_{[\cdot=\mathbf{A}\cdot]}(\mathbf{X}_k, \mathbf{Y}_k) + \sum_{j\in\mathcal{J}} g_j(\mathbf{X}_j, \mathbf{Y}_{\mathrm{F}_j}) \right\}$$

A careful observation of the above problem leads to the conclusion that it is a 'truncated' version of the original hierarchical optimization problem: its tree is obtained by replacing the entire sub-tree rooted at k with node k itself, and k's nodal objective is replaced with $\bar{g}_k + \delta_{[\cdot=\mathbf{A}\cdot]}$; see Figure 5.2 for an example. The operator which produces a truncated problem at node k given the matrix $\mathbf{A}$ is denoted by $\mathsf{truncate}(T, \mathbf{g}, k, \mathbf{A})$, and node k is referred to as the *anchor*. It will become apparent shortly why we prefer the truncated problem (which has the dummy variable $\mathbf{X}_k$) over its equivalent outer problem.

The partial minimization procedure can be performed multiple times - having truncated the problem at some anchor k, we may repeat the process, until a 'simple enough' terminal problem is obtained. When this problem is solved, we solve the obtained sequence of inner problems to 'expand' the optimal solution of the terminal problem to obtain an optimal solution of the original input problem. That is exactly the hierarchical decomposition method, described below.

Hierarchical decomposition

Input: An (HOP) problem $(T, \mathbf{g})$; A matrix $\mathbf{A}$.
Output: An optimal solution of $(T, \mathbf{g})$.
Truncation stage:

1. $i \leftarrow 1$
2. Set $(T^{(0)}, \mathbf{g}^{(0)}) \leftarrow (T, \mathbf{g})$
3. While $(T^{(i-1)}, \mathbf{g}^{(i-1)})$ is not 'simple enough'
   (a) Select an anchor k_i.
   (b) $(T^{(i)}, \mathbf{g}^{(i)}) \leftarrow \mathsf{truncate}(T^{(i-1)}, \mathbf{g}^{(i-1)}, k_i, \mathbf{A})$
   Reminder: $g^{(i)}_{k_i}$ is the function $\bar{g}^{(i-1)}_{k_i} + \delta_{[\cdot=\mathbf{A}\cdot]}$, where $\bar{g}^{(i-1)}_{k_i}$ was obtained by partial minimization.
   (c) $i \leftarrow i + 1$
4. $m \leftarrow i - 1$

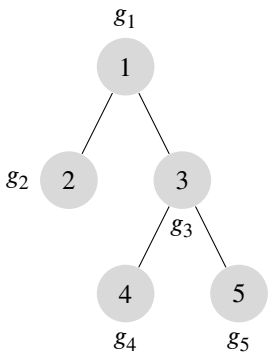
Recovery stage:

1. Solve problem $(T^{(m)}, \mathbf{g}^{(m)})$.
2. for $i = m$ to 1
   (a) Discard $\mathbf{X}_{k_i}$, and solve the inner problem defining $\bar{g}^{(i-1)}_{k_i}$ by using the optimal $\mathbf{Y}^*_{k_i}$.

The similarity to the tree reduction/expansion method is immediately apparent - the problem is first reduced, by removing nodes, until a terminal problem is obtained, and then the terminal problem's solution is expanded back. Clearly, the algorithm is conceptual in nature. For the algorithm to become implementable, we need describe how the absorbed functions are computed, and how the terminal and the inner problems are solved for a particular specialization of the method. From a practical perspective, of course, anchors should be chosen such that the inner problem is 'tractable' in some sense. Otherwise, we will neither have the computational means to truncate, nor to perform the recovery stage.

5.3.1 Hierarchical decomposition example

To reveal the key practical aspects of the conceptual method, we demonstrate it on a small example. Assume that $\mathbb{X} = \mathbb{Y} = \mathbb{R}$. In this case $\mathbb{X}^{\mathcal{V}} = \mathbb{Y}^{\mathcal{V}}$ are spaces of regular vectors whose components are real numbers, and thus we will use the lower-case vector notation for $\mathbf{x} \in \mathbb{X}^{\mathcal{V}}$ instead of the matrix notation $\mathbf{X}$. We are given the (HOP) problem



with

$$
\begin{aligned}
&g_1(x_1, y_1, y_2, y_3) = x_1^2 + y_1^2 + y_2^2 + y_3^2 && \mathrm{dom}(g_1) = [0,1]^4,\\
&g_2(x_2, y_2) = 3x_2^2 - x_2 + y_2, && \mathrm{dom}(g_2) = [0,1]^2,\\
&g_3(x_3, y_3, y_4, y_5) = y_3(x_3 + y_4 + y_5), && \mathrm{dom}(g_3) = \{(x_3, y_3, y_4, y_5) : x_3, y_3 \in [0,1],\ x_3 + y_4 + y_5 = y_3^2\},\\
&g_4(x_4, y_4) = x_4 + y_4, && \mathrm{dom}(g_4) = [0,1]^2,\\
&g_5(x_5, y_5) = x_5 + y_5, && \mathrm{dom}(g_5) = [0,1]^2.
\end{aligned}
$$

Since g_3 is non-convex, the problem seems hard to solve at first glance.

Truncation stage

We set $(T^{(0)}, \mathbf{g}^{(0)})$ to be the given problem. Let us choose node $k_1 = 3$ as the anchor. The sub-tree function at node 3 is

$$G_3(x_3, x_4, x_5, y_3, y_4, y_5) = y_3 x_3 + x_4 + x_5 + (1 + y_3)y_4 + (1 + y_3)y_5$$

with

$$\mathrm{dom}(G_3) = \{(x_3, x_4, x_5, y_3, y_4, y_5) : x_i, y_i \in [0,1],\ x_3 + y_4 + y_5 = y_3^2\}$$

The absorbed function $\bar{g}_3(y)$ is the partial minimum of G_3 over all variables, except for y_3, namely,

$$\bar{g}_3(y) = \min_{\substack{x_3, x_4 \\ x_5, y_4, y_4}} \left\{ yx_3 + x_4 + x_5 + (1 + y)y_4 + (1 + y)y_5 : x_i, y_i \in [0,1],\ x_3 + y_4 + y_5 = y^2 \right\} + \delta_{[0,1]}(y)$$

Using separability, we obtain

$$
\begin{aligned}
\bar{g}_3(y) = \min_{x_3, y_4, y_5} \{ yx_3 + (1 + y)y_4 + (1 + y)y_5 : x_3, y_4, y_4 \in [0,1],\ x_3 + y_4 + y_5 = y^2 \}& \\
+ \min_{x_4 \in [0,1]} \{x_4\} + \min_{x_5 \in [0,1]} \{x_5\} + \delta_{[0,1]}(y)&
\end{aligned}
$$

Clearly, the last two minima are 0, and are attained at $x_4 = x_5 = 0$. The first minimum is a linear program, and since $y \in [0,1]$ its feasible set is a simplex whose vertices are $(y^2, 0, 0)$, $(0, y^2, 0)$, and $(0, 0, y^2)$. Using the fact that $y \in [0,1]$, the first minimum is

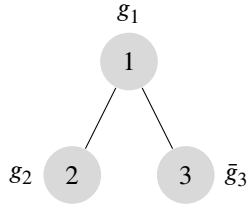
$$\min\{y^3, (1 + y)y^2, (1 + y)y^2\} = y^3,$$

and it is attained at $(x_3, y_4, y_5) = (y^2, 0, 0)$. To summarize, the absorbed function is

$$\bar{g}_3(y) = y^3 + \delta_{[0,1]}(y),$$

and the inner problem's solution is $(x_3, x_4, x_5, y_4, y_5) = (y^2, 0, 0, 0, 0, 0)$.

We obtained the truncated problem $(T^{(1)}, \mathbf{g}^{(1)})$ illustrated below:



Explicitly written, the problem $(T^{(1)}, \mathbf{g}^{(1)})$ (without the pedagogical variable x_3) is

$$\min_{x_1,x_2,y_1,y_2,y_3} \quad g_1(x_1, y_1, y_2, y_3) + g_2(x_2, y_2) + \bar{g}_3(y_3).$$

Using separability, the terminal problem $(T^{(1)}, \mathbf{g}^{(1)})$ can be written as

$$\min_{x_1\in[0,1]}\{x_1^2\} + \min_{x_2\in[0,1]}\{3x_2^2 - x_2\} + \min_{y_1\in[0,1]}\{y_1^2\} + \min_{y_2\in[0,1]}\{y_2^2 + y_3\} + \min_{y_3\in[0,1]}\{y_3^2 + y_3^3\}$$

Due to its simplicity, we deem $(T^{(1)}, \mathbf{g}^{(1)})$ as our terminal problem, and the truncation stage ends here.

Recovery stage

The optimal value of the terminal problem is clearly $-\frac{1}{12}$, which is attained when all variables are $(x_1, y_1, y_2, y_3) = \mathbf{0}$ and $x_2 = \frac{1}{6}$. Substituting $y_3 = 0$ into the problem defining $\bar{g}_3(y_3)$, we obtain an optimal solution of the inner problem, which is attained at $(x_3, x_4, x_5, y_4, y_4) = \mathbf{0}$.

Summary

We chose node 3 as our anchor, since its inner problem is an easy to solve linear program, and it led to a simple separable outer problem. This example demonstrates that the choice of the anchor node is critical - it should be chosen to serve two purposes: tractability of the inner problem, and simplification of the outer problem.

In contrast to the example above, the actual inner problems we will encounter after specializing the method to (H-OPF) problems do not possess a known closed form solution. Therefore, we resort to approximation, meaning that when we specialize the method to (H-OPF) problems we will select anchor nodes such that we have the computational means to approximate the absorbed function and inner problem's optimal solution.

5.3.2 Algorithm well-definedness

Recall, that we assumed that when referring to absorbed functions, we implicitly assume their well definedness. However, for the algorithm to be well-defined, we need this assumption to hold at every iteration of the truncation stage. It turns out that the following simple property ensures the well-definedness of the algorithm, and we will see that it holds for the specialization of the algorithm for (H-OPF) problems.

Definition (compactness property)**.** We say that the *compactness property* holds for the hierarchical optimization problem $(T, \mathbf{g})$, if for any node k

(i) the function g_k is closed, and

(ii) $\mathrm{dom}(G_k)$ for the sub-tree function G_k is compact.

Since the infimum of a closed function over a compact set is always finite and attained ([9], Theorem 2.12), $\bar{g}_k$ is well defined for any node k, including the chosen anchor node. Thus, to ensure that the hierarchical decomposition method is well-defined, it is sufficient to show that the compactness property is preserved under truncation, which is ensured by the following Lemma.

Lemma 7. *If the compactness property holds for the hierarchical optimization problem* $(T, \mathbf{g})$*, then for any node* k *and any matrix* $\mathbf{A}$ *it also holds for the hierarchical optimization problem* $\mathsf{truncate}(T, \mathbf{g}, k, \mathbf{A})$.

The proof of Lemma 7 requires the following result.

Definition. An extended real-valued function f is called *inf-compact* if for any $\alpha \in \mathbb{R}$ the sub-level set $[f \le \alpha]$ is compact.

Lemma 8 (Inf-compactness preservation (Corollary 3.2 in [26]))**.** *Let* $f : (\mathbb{E}_1 \times \mathbb{E}_2) \to (-\infty, +\infty]$ *be an inf-compact function. Then,*

$$g(\mathbf{y}) = \inf_{\mathbf{x}\in\mathbb{E}_1} f(\mathbf{x}, \mathbf{y})$$

is also inf-compact.

Proof of Lemma 7. Let $(T, \mathbf{g})$ be a hierarchical optimization problem with $T = (\mathcal{V}, \mathcal{E})$, let $k \in \mathcal{V}$, let $(T', \mathbf{g}') = \mathsf{truncate}(T, \mathbf{g}, k, \mathbf{A})$ with $T' = (\mathcal{V}', \mathcal{E}')$, and let $\bar{g}_k$ be the absorbed function in $(T, \mathbf{g})$ at node k. Assume that the compactness property holds for $(T, \mathbf{g})$.

Closedness of $\mathbf{g}'$ By construction, we have $g'_j = g_j$ for all $j \in \mathcal{V}' \setminus \{k\}$ which are closed according to the compactness property, and $g'_k(\mathbf{x}, \mathbf{y}) = \bar{g}_k(\mathbf{x}) + \delta_{[\cdot=\mathbf{A}\cdot]}(\mathbf{x}, \mathbf{y})$. Thus, to show that the functions in $\mathbf{g}'$ are closed, we only need show the closedness of $\bar{g}_k$. The sub-tree function G_k is a sum of nodal objectives, and thus closed, meaning that its sub-level sets are closed. Moreover, $\mathrm{dom}(G_k)$ is compact by the compactness property, and thus the sub-level sets are bounded. We conclude all sub-level sets G_k are compact, and by Lemma 8 the function $\bar{g}_k$ is inf-compact, implying that all its level sets are closed, meaning thus $\bar{g}_k$ is closed.

Compactness of sub-tree function domains in $(T', \mathbf{g}')$ Let $j \in \mathcal{V}'$, and let G'_j be the sub-tree function at node j with respect to the problem $(T', \mathbf{g}')$. We look at three cases.

$j \neq k$ **and** $k \notin \mathrm{des}_{T'}(j)$ By construction, $G'_j = G_j$, where G_j is the sub-tree function with respect to $(T, \mathbf{g})$. Thus, by the compactness property, $\mathrm{dom}(G'_j) = \mathrm{dom}(G_j)$ is compact.

$j = k$ **or** $k \in \mathrm{des}_{T'}(j)$ By construction, we have

$$G'_j(\mathbf{X}, \mathbf{Y}) = \underbrace{\sum_{\substack{i \in \mathcal{V}' \\ i \neq k}} g_i(\mathbf{X}_i, \mathbf{Y}_{\mathrm{F}_i}) + \bar{g}_k(\mathbf{Y}_k)}_{(*)} + \delta_{[\cdot=\mathbf{A}\cdot]}(\mathbf{X}_k, \mathbf{Y}_k)$$

$$G_j(\mathbf{X}, \mathbf{Y}) = \sum_{\substack{i \in \mathcal{V}' \\ i \neq k}} g_i(\mathbf{X}_i, \mathbf{Y}_{\mathrm{F}_i}) + g_k(\mathbf{X}_k, \mathbf{Y}_k) + \sum_{i \in \mathrm{des}_T(k)} g_i(\mathbf{X}_i, \mathbf{Y}_{\mathrm{F}_i})$$

The effective domain of $(*)$ is the projection of $\mathrm{dom}(G_j)$, which is assumed to be compact, onto a subset of the coordinates. Together with the constraint imposed by $\delta_{[\cdot=\mathbf{A}\cdot]}$, we conclude that $\mathrm{dom}(G'_j)$ must be compact as well.

□

5.4 Hierarchical decomposition applied to (H-OPF)

Recall, that an (H-OPF) problem on the compact radial network $(T, \mathbf{z}, \mathcal{G}, \mathbf{f}, \overline{\boldsymbol{\ell}})$ with a rooted tree T can be written as the hierarchical optimization problem $(T, \mathbf{g})$ with $\mathbb{X} = \mathbb{R}^2$ and $\mathbb{Y} = \mathbb{R}^3$, and:

- $\mathbf{X}_j$ encodes $(\mathrm{re}(s_j), \mathrm{im}(s_j))^T$;
- $\mathbf{Y}_j$ encodes $(u_j, \mathrm{re}(s'_j), \mathrm{im}(s'_j))^T$;
- for $r = \mathrm{root}(T)$, the nodal function g_r computes $\delta_{\{0\}}(s'_r) + f_r(u_r, s_r) + \delta_r(\mathbf{u}_{\mathrm{F}_r}, \mathbf{s}'_{\mathrm{F}_r}, s_r)$;
- for $j \neq r$, the nodal function g_j computes $f_i(u_i, s_i) + \delta_i(\mathbf{u}_{\mathrm{F}_i}, \mathbf{s}'_{\mathrm{F}_i}, s_i)$.

We devote this chapter to the specialization of the hierarchical decomposition method to the problem described above. Consequently, throughout the chapter, we write the nodal, sub-tree and absorbed functions in terms of the problem's original variables $\mathbf{u}, \mathbf{s}, \mathbf{s}'$ instead of $\mathbf{X}$ and $\mathbf{Y}$. However, when discussing properties such as continuity and closedness of functions, we are referring to these functions in terms of the real variables $\mathbf{X}$ and $\mathbf{Y}$.

To apply the hierarchical decomposition method to the problem, we need to specify the following: how the root of the tree is chosen, how anchors are selected, how absorbed functions are computed, how the terminal problem is solved, and which matrix $\mathbf{A} \in \mathbb{R}^{2\times 3}$ is used for the truncation process. As demonstrated by the simple example in Section 5.3.1, these decisions are interconnected, and the most significant decision is the anchor selection strategy.

Recall, that we do not aim at an exact solution of the inner problems, and approximate them instead. The theoretical analysis of the conditions which ensure the convergence of the approximation we describe in this chapter to the exact solution are analyzed separately in Chapter 6, since the analysis is quite exhaustive, and may impede the readability of this chapter. Without going into the details, we point out that

convergence is ensured under quite a restrictive set of assumptions, and applies only to a subset of (H-OPF) problems. The weakness of the theoretical result is, of course, not a surprise. For most algorithms for solving non-convex problems the theoretical result is even weaker - they are proved to converge to a stationary point, which might not even be a local optimum. Since the theory for our algorithm is insufficient, this chapter also contains the results of numerical experiments which demonstrate the performance of our algorithm.

The specialization of the hierarchical decomposition method to (H-OPF) problems relies on the careful analysis of the sub-tree function associated with node $k \neq \text{root}(T)$:

$$H_k(\mathbf{u}, \mathbf{s}, \mathbf{s}') \equiv \sum_{i \in \text{sub}_T(k)} [f_i(u_i, s_i) + \delta_i(\mathbf{u}_{\text{F}_i}, \mathbf{s}'_{\text{F}_i}, s_i)], \tag{5.9}$$

When $k = r = \text{root}(T)$ the sub-tree function H_r coincides with the (H-OPF) objective H.

5.4.1 Well definedness - the compactness property

Recall that for the hierarchical decomposition method to be well-defined, the compactness property (Definition 5.3.2) is sufficient. To show that the compactness property holds in our specialization, we first require the following result.

Lemma 9. *Let P be a compact radial network with a rooted topology tree $T = (\mathcal{V}, \mathcal{E})$, and let $i \in \mathcal{V}$. Then:*

(a) $\delta_i(\mathbf{u}_{\text{F}_i}, \mathbf{s}'_{\text{F}_i}, s_i)$ is a closed extended real-valued function.

(b) Let $C = \text{ch}_T(i)$, let $S \subseteq \mathbb{C} \times \mathbb{R}^C \times \mathbb{C}^C \times \mathbb{C}^C$ be a bounded set, and suppose that $i \notin \text{L}(T)$. Then, the set

$$\{(u_i, s'_i) : \delta_i(\mathbf{u}_{\text{F}_i}, \mathbf{s}'_{\text{F}_i}, s_i) = 0,\ (s_i, \mathbf{u}_C, \mathbf{s}_C, \mathbf{s}'_C) \in S\}$$

is bounded.
In other words, the constraints imposed by δ_i enforce boundedness of (u_i, s'_i) if the rest of the coordinates belong to a bounded set.

Proof. We prove each item separately. Recall the definition of $\delta_i(\mathbf{u}_{\text{F}_i}, \mathbf{s}'_{\text{F}_i}, s_i)$.

(a) The equations and inequalities imposed by $\delta_i(\mathbf{u}_{\text{F}_i}, \mathbf{s}'_{\text{F}_i}, s_i)$ are sub-level and level sets of continuous functions, meaning that $\delta_i(\mathbf{u}_{\text{F}_i}, \mathbf{s}'_{\text{F}_i}, s_i)$ is an indicator function of a closed set. Thus, $\delta_i(\mathbf{u}_{\text{F}_i}, \mathbf{s}'_{\text{F}_i}, s_i)$ is closed.

(b) Fix $j \in \text{ch}_T(j)$. If $z_{ij} = 0$ then by definition of $\tilde{s}_j$ we have $\tilde{s}_j = s'_j$. Otherwise, $\tilde{s}_j$ is a continuous function of s'_j and u_j. Since by assumption s'_j, u_j are bounded, then $\tilde{s}_j$ must also be bounded.

Since s_i is also bounded, then by (5.2) s'_i is bounded as a sum of bounded quantities by the triangle inequality. The right-hand side in (5.4) is a continuous function of u_j, s'_j, which as we recall are bounded. Thus u_i must be bounded too.

□

Now we prove our desired result.

Lemma 10. *Let $(T, \mathbf{z}, \mathcal{G}, \mathbf{f}, \overline{\mathscr{E}})$ be a compact radial network, let $r = \text{root}(T)$, and let $(T, \mathbf{g})$ be the corresponding hierarchical optimization problem. Then the compactness property holds in $(T, \mathbf{g})$.*

Proof. We prove each item in the compactness property separately. Throughout the proof, functions accepting complex arguments are treated as equivalent functions accepting the corresponding real and imaginary components. Let $T = (\mathcal{V}, \mathcal{E})$ and let $q \in \mathcal{V}$.

Closedness of nodal functions Suppose that $q \neq r$. In that case, the nodal function is $f_q(u_q, s_q) + \delta_q(\mathbf{u}_{\text{F}_q}, \mathbf{s}'_{\text{F}_q}, s_q)$. Since P is a compact radial network, f_q is closed. By Lemma 9, δ_q is closed. Therefore, the nodal function is closed as a sum of closed functions. The argument for the case of $q = r$ follows immediately by using closedness arguments for $\delta_{\{0\}}(s'_r)$.

Domain compactness of sub-tree functions Recall that H_q defined in (5.9) is the sub-tree function.

Closedness of $\mathrm{dom}(H_q)$: The compactness of every $\mathrm{dom}(f_i)$ for $i \in \mathrm{sub}_T(q)$, which follows from the definition of a compact radial network, implies that $\mathrm{dom}(f_i)$ is closed. $\mathrm{dom}(\delta_i)$ is closed by Lemma 9. Therefore $\mathrm{dom}(H_q)$ is closed as an intersection of closed sets.

Boundedness of $\mathrm{dom}(H_q)$: The compactness of every $\mathrm{dom}(f_i)$ for $i \in \mathrm{sub}_T(q)$, implied by the definition of a compact radial network, also implies the boundedness of $\mathrm{dom}(f_i)$. Therefore, $\mathrm{dom}\left(\sum_{i\in\mathrm{sub}_T(q)} f_i\right)$ is bounded as a Cartesian product of bounded sets, which implies that all coordinates u_i and s_i must be bounded. Proving that s'_i are bounded for all $i \in \mathrm{sub}_T(q)$, which is done by induction on the distance of i from the leaves, concludes this item.
When $i \in \mathrm{L}(T)$, meaning the distance from the leaves is 0, by definition of $\delta_i(\mathbf{u}_{\mathrm{F}_i}, \mathbf{s}'_{\mathrm{F}_i}, s_i)$ we have $s'_i = s_i$, and in thus the boundedness of s_i implies the boundedness of s'_i.
Assume that $i \notin \mathrm{L}(T)$. By the induction hypothesis s'_j is bounded for all $j \in \mathrm{ch}_T(i)$. Together with the boundedness of s_i and of u_j for $j \in \mathrm{ch}_T(i)$, Lemma 9 implies that s'_i is bounded.

□

5.4.2 Absorbed function and truncated problem

Let $P = (T, \mathbf{z}, \mathcal{G}, \mathbf{f}, \overline{\mathscr{E}})$ and $q \neq \mathrm{root}(T)$ be the chosen anchor. Recalling the sub-tree function defined in (5.9), the absorbed function at node q is

$$\begin{aligned}\bar{f}_q(u,s) &= \min_{\mathbf{u},\mathbf{s},\mathbf{s}'}\{H_q(\mathbf{u},\mathbf{s},\mathbf{s}') : u_q = u, s'_q = s\} \\ &= \min_{\mathbf{u},\mathbf{s},\mathbf{s}'}\left\{\sum_{i\in\mathrm{sub}_T(q)}[f_i(u_i,s_i)+\delta_i(\mathbf{u}_{\mathrm{F}_i},\mathbf{s}'_{\mathrm{F}_i},s_i)] : u_q = u, s'_q = s\right\}\end{aligned} \tag{5.10}$$

while the truncated problem, denoting $T = (\mathcal{V}, \mathcal{E})$ and $r = \mathrm{root}(T)$, is

$$\min_{\mathbf{u},\mathbf{s},\mathbf{s}'} \quad \delta_{\{0\}}(s'_r) + \sum_{i\in\mathcal{V}\setminus\mathrm{sub}_T(q)}[f_i(u_i,s_i)+\delta_i(\mathbf{u}_{\mathrm{F}_i},\mathbf{s}'_{\mathrm{F}_i},s_i)] + \bar{f}_q(u_q,s'_q) + \delta_{[\cdot=\mathbf{A}\cdot]}(s_q,(u_q,s'_q)^T).$$

Note, that $(u_q, s_q) \in \mathrm{dom}(\bar{f}_i)$, by construction, implies $u_q \geq \alpha$, where α is defined in (5.7). Therefore, together with the choice of

$$\mathbf{A} = \begin{pmatrix}0 & 1\end{pmatrix}, \tag{5.11}$$

which imposes the equality $s_q = s'_q$, we obtain the following equivalent truncated problem

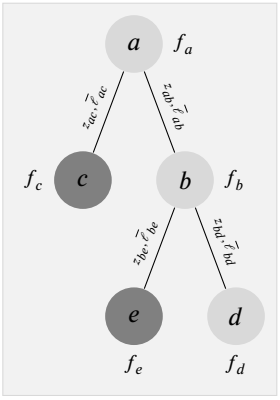
$$\min_{\mathbf{u},\mathbf{s},\mathbf{s}'} \quad \delta_{\{0\}}(s'_r) + \sum_{i\in\mathcal{V}\setminus\mathrm{sub}_T(q)}[f_i(u_i,s_i)+\delta_i(\mathbf{u}_{\mathrm{F}_i},\mathbf{s}'_{\mathrm{F}_i},s_i)] + \underbrace{\delta_{[\cdot\geq\alpha]}(u_q) + \delta_{\{(x,x)\}}(s_q,s'_q)}_{\delta_q(u_q,s_q,s'_q)} + \bar{f}_q(u_q,s_q). \tag{5.12}$$

By the compactness property, $\mathrm{dom}(\bar{f}_q)$ must be compact, and can be regarded as the cost associated with a generator. Thus, by carefully inspecting the above problem, we conclude that it is exactly the (H-OPF) objective associated with the compact radial network obtained by removing the entire sub-tree rooted at q, and replacing it with the node q, which now has the role of a generator whose associated cost function is $\bar{f}_q$. We denote the resulting network by $P' = \mathsf{truncate}(P, q)$, and illustrate it in Figure 5.3.

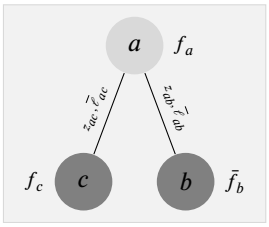
To summarize, we conclude that if $(T, \mathbf{g})$ is the hierarchical optimization problem which corresponds to the compact radial network P, then hierarchical optimization problem $\mathsf{truncate}(T, \mathbf{g}, q, \mathbf{A})$ with A defined in (5.11) is an (H-OPF) problem on the compact radial network $\mathsf{truncate}(P, q)$.

5.4.3 Anchor choice and network augmentation

Note that if q contains at least two generators in its sub-tree, the result of $\mathsf{truncate}(P)$ produces a network which has less generators than P. Thus, such an anchor choice will, eventually, lead to a terminal problem with one generator, which can be solved by the tree reduction/expansion method. For example, the anchor chosen in Figure 5.3 is, *not* a beneficial anchor node.



(a) The compact radial network in Figure 5.1, rooted at node a



(b) The network on the left, truncated at node b. The nodal objective f_b was replaced with the absorbed function f'_b, and node b became a generator.

Figure 5.3: Truncated network example

Recall, that the anchor q should also be chosen such that we have the computational means to approximate the solution of the inner problem defining $\bar{f}_q$ as well. To satisfy this requirement, for reasons which will become apparent by the end of this chapter, we choose the anchor q such that the following conditions hold:

$$|\operatorname{sub}_T(q) \cap \mathcal{G}| \geq 2 \tag{5.13a}$$

$$|\operatorname{sub}_T(j) \cap \mathcal{G}| \leq 1 \qquad j \in \operatorname{ch}_T(q) \tag{5.13b}$$

That is, we choose an anchor which contains at least two generators in its sub-tree, and each child contains at most one generator in its sub-tree.

When multiple generators are present in the network, and the $\operatorname{root}(T)$ is chosen to be a non-generator node, such a choice is always possible. Recalling that the generators are a subset of the leaves in a compact radial network, an anchor can be found by 'climbing' from the leaves to the root, until a common ancestor of several generators has been found. This idea is formally described as Algorithm 4.

Another step we need to make in order to have the computational means to approximate the inner problem is to augment the network with additional nodes which we call *surrogates*. The process is best explained visually - see Figure 5.4. Formally, between the anchor q and each $j \in \operatorname{ch}_T(q)$ we insert a *surrogate* node r: node r becomes the parent of j and replaces j as a child of q, the edge connecting q and r is assigned zero impedance and zero power-loss bound, the edge connecting r and j inherits the original properties of the edge which connected q to j, and node r itself is assigned the cost function $f_r(u, s) = \delta_{\{0\}}(s) + \delta_{[\underline{u}_q, \bar{u}_q]}(u)$ which is an indicator of a PQ constraint. It is easy to verify that the constraints imposed by δ_q and δ_r in the network obtained after surrogate insertion collapse back to the constraints imposed by δ_q in the original network.

A similar idea can be used to ensure that inequality (5.13b) holds as an equality, that is, each child of an anchor has exactly one generator in its sub-tree. If it is not the case, we can 'isolate' our desired children under a common node q': insert an auxiliary node q' between the anchor q and the children that which do have a generator in their sub-tree, such that q' is connected to its parent q by an edge with zero impedance and zero power-loss bound. For similar reasons as above, both the original and the augmented networks have equivalent associated (H-OPF) problems. Then, we can use q' as the anchor, instead of q.

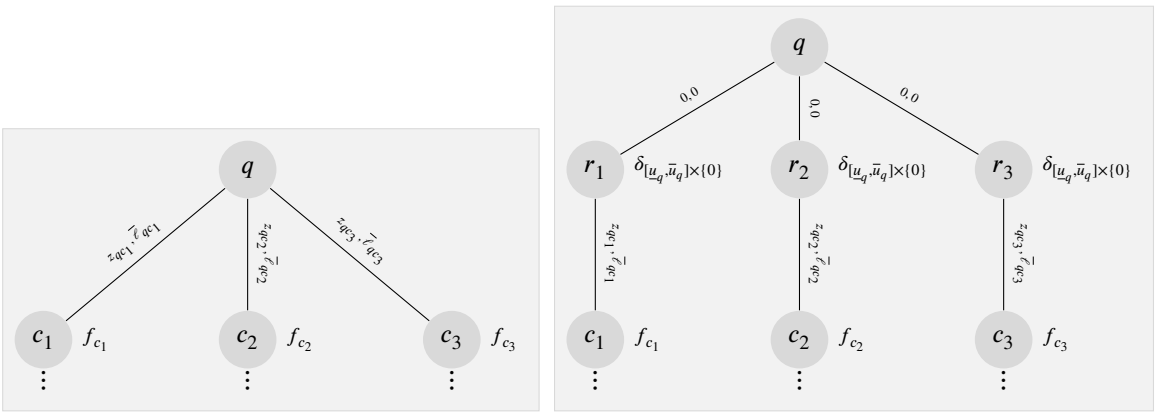
Clearly, given a network $P = (T, \mathbf{z}, \mathcal{G}, \mathbf{f}, \overline{\boldsymbol{\ell}})$, we can, in advance, find all the nodes which will become anchors when the hierarchical decomposition method runs, by simulating the truncation process without actually computing any absorbed function. Hence, we denote by $P' = \mathsf{augment}(P)$ the operator which locates of all the nodes which will become anchors during the hierarchical decomposition method, and augments the network P as described above with surrogate nodes and nodes which ensure that (5.13b) holds as an equality.

Algorithm 4 Find Anchor

Input: a rooted topology tree $T = (\mathcal{V}, \mathcal{E})$; a set of generator nodes $\mathcal{G}$.
Output: an anchor node.
Steps:

1: $(N, q) \leftarrow$ FIND-ANCHOR-REC$(T, \text{root}(T))$
2: **return** q

Returns either $(\emptyset, q)$ denoting the anchor q, or $(N, \emptyset)$ denoting N generators in the sub-tree
3: **function** FIND-ANCHOR-REC(T, j)
4: $N \leftarrow |\{j\} \cap \mathcal{G}|$
5: **for all** $i \in \text{ch}_T(j)$ **do**
6: $(N_i, P_i) \leftarrow$ FIND-ANCHOR-REC(T, i)
7: **if** $N_i = \emptyset$ **then**
8: **return** $(\emptyset, P_i)$
9: **else**
10: $N \leftarrow N + N_i$
11: **end if**
12: **end for**
13: **if** $N \geq 2$ **then**
14: **return** $(\emptyset, j)$
15: **else**
16: **return** $(N, \emptyset)$
17: **end if**
18: **end function**



(a) Before surrogate insertion

(b) After surrogate insertion at q

Figure 5.4: Surrogate insertion example

5.4.4 Examination of the inner problem

Let $P = (T, \mathbf{z}, \mathcal{G}, \mathbf{f}, \overline{\boldsymbol{\ell}})$ be a compact radial network, which has been augmented by the $\mathsf{augment}(\cdot)$ operator, and let q be an anchor node chosen to satisfy conditions (5.13). Our aim is to reformulate the optimization problem defining the absorbed function $\bar{f}_q$ in order to discover the essential properties which allow us to approximate its optimal value and solution.

Suppose that $r_1, \dots, r_\ell$ are the surrogates inserted between q and its original children $j_1, \dots, j_\ell$, and denote $\mathcal{I}_k = \mathrm{sub}_T(r_k)$. The absorbed function (5.10), by separating the optimization objective into distinct sums corresponding to each child of q, can be written as

$$\begin{aligned}\bar{f}_q(u,s) &= \min_{\mathbf{u},\mathbf{s},\mathbf{s}'} \left\{ f_q(u_q, s_q) + \delta_q(\mathbf{u}_{\mathrm{F}_q}, \mathbf{s}'_{\mathrm{F}_q}, s_q) + \sum_{k=1}^{\ell} H_{r_k}(\mathbf{u}_{\mathcal{I}_k}, \mathbf{s}_{\mathcal{I}_k}, \mathbf{s}'_{\mathcal{I}_k}) \,:\, u_q = u, s'_q = s \right\} \\ &= \min_{\mathbf{u},\mathbf{s},\mathbf{s}'} \left\{ \delta_q(\mathbf{u}_{\mathrm{F}_q}, \mathbf{s}'_{\mathrm{F}_q}, s_q) + \sum_{k=1}^{\ell} \left[\frac{1}{\ell} f_q(u_q, s_q) + H_{r_k}(\mathbf{u}_{\mathcal{I}_k}, \mathbf{s}_{\mathcal{I}_k}, \mathbf{s}'_{\mathcal{I}_k}) \right] \,:\, u_q = u, s'_q = s \right\}\end{aligned}$$

Since $r_1, \dots, r_\ell$ are all surrogates, the constraints imposed by δ_q are

$$u_q = u_{r_1}, \;\; \dots, u_q = u_{r_\ell}, \;\; s'_q = s_q + \sum_{k=1}^{\ell} s'_{r_k}.$$

Moreover, since q is not a leaf, according to the definition of a compact radial network it must be a PQ constrained node, that is, s_q is constrained to be the constant $\hat{s}_q$ and f_q is a function of u_q only. Since r is a surrogate, s_r is constrained to be zero. Combining the information above, the absorbed function can be written as

$$\bar{f}_q(u,s) = \min_{\mathbf{u},\mathbf{s},\mathbf{s}'} \left\{ \sum_{k=1}^{\ell} \left[\frac{1}{\ell} f_q(u_{r_k}) + \delta_{\{u\}}(u_{r_k}) + \delta_{\{0\}}(s_{r_k}) + H_{r_k}(\mathbf{u}_{\mathcal{I}_k}, \mathbf{s}_{\mathcal{I}_k}, \mathbf{s}'_{\mathcal{I}_k}) \right] \,:\, s = \hat{s}_q + \sum_{k=1}^{\ell} s'_{r_k} \right\}.$$

Reformulating the minimization problem above as a bi-level such that the variables $s'_{r_1}, \dots, s'_{r_\ell}$ remain in the outer problem, we obtain $\bar{f}_q(u,s) = g(u, s - \hat{s}_q)$ where g is defined by

$$g(u,s) = \min_{s'_{r_1}, \dots, s'_{r_\ell}} \left\{ \sum_{k=1}^{\ell} g_k(u, s'_{r_k}) \,:\, s = \sum_{k=1}^{\ell} s'_{r_k} \right\},$$

with

$$g_k(u, s'_{r_k}) = \min_{\substack{\mathbf{u} \in \mathbb{R}^{\mathcal{I}_k} \\ \mathbf{s} \in \mathbb{C}^{\mathcal{I}_k} \\ \mathbf{s}' \in \mathbb{C}^{\mathcal{I}_k \setminus \{r_k\}}}} \left\{ \frac{1}{\ell} f_q(u_{r_k}) + \delta_{\{u\}}(u_{r_k}) + \delta_{\{0\}}(s_{r_k}) + H_{r_k}(\mathbf{u}_{\mathcal{I}_k}, \mathbf{s}_{\mathcal{I}_k}, \mathbf{s}'_{\mathcal{I}_k}) \right\}$$

Now, let us look deeper into g_k. Recall, that since each r_k is a surrogate, it has a single child, denoted by j_k. Let $\mathcal{J}_k = \mathrm{sub}_T(j_k)$. Thus,

$$H_{r_k}(\mathbf{u}, \mathbf{s}, \mathbf{s}') = f_{r_k}(u_{r_k}, s_{r_k}) + \delta_{r_k}((u_{r_k}, u_{j_k}), s_{r_k}, (s'_{r_k}, s'_{j_k})) + H_{j_k}(\mathbf{u}_{\mathcal{J}_k}, \mathbf{s}_{\mathcal{J}_k}, \mathbf{s}'_{\mathcal{J}_k})$$

Since r_k is a surrogate, we have $f_{r_k} = \delta_{[\underline{u}_q, \bar{u}_q] \times \{0\}}$, and thus the objective in the minimization problem defining g_k is

$$\frac{1}{\ell} f_q(u_{r_k}) + \delta_{\{u\}}(u_{r_k}) + \delta_{[\underline{u}_q, \bar{u}_q]}(u_{r_k}) + \delta_{\{0\}}(s_{r_k}) + \delta_{r_k}((u_{r_k}, u_{j_k}), (s'_{r_k}, s'_{j_k}), s_{r_k}) + H_{j_k}(\mathbf{u}_{\mathcal{J}_k}, \mathbf{s}_{\mathcal{J}_k}, \mathbf{s}'_{\mathcal{J}_k})$$

The constraint (5.2) as imposed by δ_{r_k} is

$$s'_{r_k} = s_{r_k} + \tilde{s}_{j_k},$$

which can be equivalently written as

$$-s_{r_k} = -s'_{r_k} + \tilde{s}_{j_k}.$$

Thus, we can 'exchange the roles' of $-s_{r_k}, -s'_{r_k}$ with s'_{r_k}, s_{r_k}, respectively, and obtain

$$g_k(u,s) = \min_{\substack{\mathbf{u}\in\mathbb{R}^{\mathcal{I}_k} \\ \mathbf{s}\in\mathbb{C}^{\mathcal{I}_k} \\ \mathbf{s}'\in\mathbb{C}^{\mathcal{I}_k}}} \left\{ \delta_{\{0\}}(s'_{r_k}) + \delta_{\{u\}}(u_{r_k}) + \frac{1}{\ell} f_q(u_{r_k}) + \delta_{\{-s\}}(s_{r_k}) \right. \\ \left. + \delta_{r_k}(\mathbf{u}_{\mathrm{F}_{r_k}}, \mathbf{s}'_{\mathrm{F}_{r_k}}, s_{r_k}) + H_{j_k}\left(\mathbf{u}_{\mathcal{J}_k}, \mathbf{s}_{\mathcal{J}_k}, \mathbf{s}'_{\mathcal{J}_k}\right) \right\} + \delta_{[\underline{u}_q, \overline{u}_q]}(u) \tag{5.14}$$

An important observation to make is that for any given (u, s), the minimization problem above is exactly the (H-OPF) objective of the compact radial network obtained by 'detaching' the sub-tree rooted at r_k from the network, and replacing the cost associated with r_k with

$$\tilde{f}_{r_k}(u_{r_k}, s_{r_k}) = \delta_{\{u\}}(u_{r_k}) + \frac{1}{\ell} f_q(u_{r_k}) + \delta_{\{-s\}}(s_{r_k}). \tag{5.15}$$

By construction, this network has only one generator, and the consumer r_k has its associated voltage absolute value and apparent power constrained to be the constants u and s. Thus, its associated OPF problem is a fully constrained problem on a restricted radial network, meaning that the algorithm described in Section 4.2 can be used to evaluate $g_k(u, s)$ for any (u, s).

We denote by $\mathsf{detach}(P, q, r_k)$ a function which produces such networks for any given (u, s). To be precise, $\mathsf{detach}(P, q, r_k)(u, s)$ results in the network obtained by detaching the sub-tree rooted at r_k, and replacing its associated cost function by $\tilde{f}_{r_k}$ above. The reason for defining such an operator is that an algorithm which evaluates g_k uses the function $P' = \mathsf{detach}(P, q, r_k)$, to obtain for any given (u, s) the network $P'(u, s)$ whose associated (H-OPF) problem has $g_k(u, s)$ as its optimal value.

The approximation algorithm for g which we will describe below requires access to a bounding box of $\mathrm{dom}(g_k)$ for all $k = 1, \ldots, \ell$. Fortunately, we can obtain such a box by constructing a convex relaxation of the constraints imposed by the objective defining g_k. Explicitly writing the constraints imposed by the f_i functions and the δ_i functions participating in the objective defining g_k, and recalling that all nodes except generators have PQ constraints, we obtain the following system in $\mathbf{u}, \mathbf{s}, \mathbf{s}', \tilde{\mathbf{s}}$:

$$\begin{aligned}
s_i &= \hat{s}_i && i \in \mathcal{J}_k \setminus \mathcal{G} \\
\underline{u}_i \le u_i &\le \overline{u}_i && i \in \mathcal{I}_k \\
(u_i, s_i) &\in \mathrm{dom}(f_i) && i \in \mathcal{G} \cap \mathcal{J}_k \\
(*)\ u_i &= \left| u_j - z_{ij} \frac{s'^{*}_j}{u_j} \right| && i \in \mathcal{I}_k,\ j \in \mathrm{ch}_T(i), \\
s'_i &= s_i + \sum_{j \in \mathrm{ch}_T(i)} \tilde{s}_j && i \in \mathcal{I}_k \\
\tilde{s}_j &= s'_j - z_{ij} \frac{|s'_j|^2}{u_j^2} && i \in \mathcal{I}_k,\ j \in \mathrm{ch}_T(i) \\
|\tilde{s}_j^2||z_{ij}| &\le u_j^2 \overline{\ell}_{ij} && i \in \mathcal{I}_k,\ j \in \mathrm{ch}_T(i)
\end{aligned}$$

We relax the constraint $(u_i, s_i) \in \mathrm{dom}(f_i)$ for $i \in \mathcal{G}$ by replacing $\mathrm{dom}(f_i)$ with its bounding box $[\underline{u}_i, \overline{u}_i] \times$

$[\underline{p}_i, \overline{p}_i] + \mathbf{i}[\underline{q}_i, \overline{q}_i]$. By substituting $w_i = u_j^2$ and $t_j = \frac{|s'_j|^2}{u_j^2}$, and squaring both sides of $(*)$, we obtain:

$$
\begin{aligned}
&s_i = \hat{s}_i, && i \in \mathcal{J}_k \setminus \mathcal{G},\\
&\underline{u}_i^2 \le w_i \le \overline{u}_i^2, && i \in \mathcal{I}_k,\\
&s_i \in [\underline{p}_i, \overline{p}_i] + \mathbf{i}[\underline{q}_i, \overline{q}_i], && i \in \mathcal{G} \cap \mathcal{J}_k,\\
&w_i = w_i - 2\operatorname{re}(z_{ij} s'^{*}_j) + t_j, && i \in \mathcal{I}_k,\ j \in \operatorname{ch}_T(i),\\
&s'_i = s_i + \sum_{j \in \operatorname{ch}_T(i)} \tilde{s}_j, && i \in \mathcal{I}_k,\\
&\tilde{s}_j = s'_j - z_{ij} t_j, && i \in \mathcal{I}_k,\ j \in \operatorname{ch}_T(i),\\
&|\tilde{s}_j^2||z_{ij}| \le w_j \overline{\ell}_{ij}, && i \in \mathcal{I}_k,\ j \in \operatorname{ch}_T(i),\\
&t_j = \frac{|s'_j|^2}{w_j}, && j \in \mathcal{I}_k.
\end{aligned}
$$

Finally, relaxing the last equality by replacing it with an inequality, we obtain the following system in $\mathbf{w}, \mathbf{s}, \mathbf{s}', \tilde{\mathbf{s}}', \mathbf{t}$, which is convex assuming complex numbers are represented by their real and imaginary parts:

$$
\begin{aligned}
&s_i = \hat{s}_i, && i \in \mathcal{J}_k \setminus \mathcal{G}, && \text{(5.16a)}\\
&\underline{u}_i^2 \le w_i \le \overline{u}_i^2, && i \in \mathcal{I}_k, && \text{(5.16b)}\\
&s_i \in [\underline{p}_i, \overline{p}_i] + \mathbf{i}[\underline{q}_i, \overline{q}_i], && i \in \mathcal{G} \cap \mathcal{J}_k, && \text{(5.16c)}\\
&w_i = w_i - 2\operatorname{re}(z_{ij} s'^{*}_j) + t_j, && i \in \mathcal{I}_k,\ j \in \operatorname{ch}_T(i), && \text{(5.16d)}\\
&s'_i = s_i + \sum_{j \in \operatorname{ch}_T(i)} \tilde{s}_j, && i \in \mathcal{I}_k, && \text{(5.16e)}\\
&\tilde{s}_j = s'_j - z_{ij} t_j, && i \in \mathcal{I}_k,\ j \in \operatorname{ch}_T(i), && \text{(5.16f)}\\
&|\tilde{s}_j^2||z_{ij}| \le w_j \overline{\ell}_{ij}, && i \in \mathcal{I}_k,\ j \in \operatorname{ch}_T(i), && \text{(5.16g)}\\
&t_j \ge \frac{|s'_j|^2}{w_j}, && j \in \mathcal{I}_k. && \text{(5.16h)}
\end{aligned}
$$

Consequently, we may obtain a bounding box on $\operatorname{dom}(g_k)$ by minimizing and maximizing u_{r_k}, $\operatorname{re}(s_{r_k})$, and $\operatorname{im}(s_{r_k})$ subject to (5.16) using a convex programming solver of our choice. The relaxation can be further 'strenghtened' by adding additional constraints which follow from the Power Preservation Lemma (Lemma 1), where the additional variables $\mathbf{q}$ model possible edge power losses:

$$
\begin{aligned}
&\sum_{i \in \mathcal{I}_k} s_i = \sum_{\substack{i \in \mathcal{I}_k \\ j \in \operatorname{ch}_T(i)}} q_{ij}, && && \text{(5.17a)}\\
&|q_{ij}| \le \overline{\ell}_{ij} && i \in \mathcal{I}_k,\ j \in \operatorname{ch}_T(i) && \text{(5.17b)}
\end{aligned}
$$

To summarize, we have reached the following conclusions:

- $\bar{f}_q(u, s)$ can be written as $\bar{f}_q(u, s) = g(u, \hat{s}_q - s)$, where g has the following structure:

$$
g(u, s) = \min_{s_1, \ldots, s_\ell \in \mathbb{C}} \left\{ \sum_{k=1}^{\ell} g_k(u, s_k) \,:\, s = \sum_{k=1}^{\ell} s_k \right\}. \tag{5.18}
$$

  We denote the corresponding optimal set by:

$$
\bar{F}_q(u, s) = \operatorname*{argmin}_{s_1, \ldots, s_\ell \in \mathbb{C}} \left\{ \sum_{k=1}^{\ell} g_k(u, s_k) \,:\, s = \sum_{k=1}^{\ell} s_k \right\} \tag{5.19}
$$

- We have the computational means to evaluate each $g_k(u, s)$ at any point, and obtain a bounding box on each $\mathrm{dom}(g_k)$:
  - $g_k(u, s)$ can be evaluated by solving an OPF problem on the fully constrained network with one generator, obtained by detaching the sub-tree rooted at r_k, and replacing f_{r_k} with $\tilde{f}_{r_k}$ defined in (5.15);
  - we defined the operator $D = \mathsf{detach}(P, q, r_k)$ which returns the function D such that $D(u, s)$ produces the above-mentioned detached network;
  - a bounding box on $\mathrm{dom}(g_k)$ can be computed by solving nine convex problems subject to (5.16) and (5.17) which minimize and maximize u, $\mathrm{re}(s_{r_k})$, and $\mathrm{im}(s_{r_k})$;
  - an optimal solution of the problem defining $\bar{f}_q$ can be obtained by solving the problem defining g above, and then solving the inner problem defining each g_k.

5.4.5 The full algorithm

At this stage, we combine the information we learned up until now to explicitly write the hierarchical decomposition specialized to (H-OPF) problems. The basic building block are networks with one generator. The 'simple enough' terminal network has one generator. The truncation operator reduces the number of generators by choosing an appropriate anchor, and each inner problem is an infimal convolution of (H-OPF) optimal values problems on networks with one generator, parameterized by the voltage u. Thus, the hierarchical decomposition method we devised reduces a problem on a network with several generators to a set of problems on networks with one generator.

The algorithm is described below in Algorithm 5. We assume that the computations marked with $\triangleright$ are carried out exactly. The next subsection discusses our approximation scheme for these steps.

Algorithm 5 Hierarchical Decomposition for (H-OPF) problems

Input: a compact radial network $P = (T, \mathbf{z}, \mathcal{G}, \mathbf{f}, \overline{\boldsymbol{\mathscr{C}}})$ with $r = \mathrm{root}(T) \notin \mathcal{G}$;
Output: either $(\mathbf{u}, \mathbf{s})$ which is an optimal solution of the (H-OPF) problem on P, or $\emptyset$ if the problem is infeasible;
Preparation stage:

1. $P \leftarrow \mathsf{augment}(P)$
   Ensure each child of an anchor is a surrogate with 1 generator in its sub-tree

Reduction stage:

2. $P^{(0)} = (T^{(0)}, \mathbf{z}^{(0)}, \mathcal{G}^{(0)}, \mathbf{f}^{(0)}, \overline{\boldsymbol{\mathscr{C}}}^{(0)}) \leftarrow P$
3. $i \leftarrow 1$
4. while $|\mathcal{G}| > 1$:
   (a) $q \leftarrow \mathsf{Find-Anchor}(T^{(i-1)}, \mathcal{G}^{(i-1)})$ using Algorithm 4
   Invariant: every $r \in \mathrm{ch}_{T^{(i-1)}}(q)$ is a surrogate
   (b) $\mathcal{D}^{(i)} \leftarrow \{(r, D) : r \in \mathrm{ch}_{T^{(i-1)}}(i),\ D = \mathsf{detach}(P^{(i-1)}, q, r)\}$
   (c) $\triangleright\ P^{(i)} = (T^{(i)}, \mathbf{z}^{(i)}, \mathcal{G}^{(i)}, \mathbf{f}^{(i)}, \overline{\boldsymbol{\mathscr{C}}}^{(i)}) \leftarrow \mathsf{truncate}(P^{(i-1)}, q)$
   q becomes a leaf generator in $P^{(i)}$; $\bar{f}_q$ is its associated cost function
   (d) $i \leftarrow i + 1$

Recovery stage:

4. $m \leftarrow i - 1$
5. $\mathcal{I} \leftarrow$ the nodes of $T^{(m)}$.
6. $\mathbf{u}_{\mathcal{I}}, \mathbf{s}_{\mathcal{I}}, \mathbf{s}'_{\mathcal{I}} \leftarrow$ an optimal solution of the (H-OPF) associated with $P^{(m)}$
7. for $i = m - 1$ to 1:
   (a) $q \leftarrow$ the only generator in $\mathcal{G}^{(i)}$
   (b) $\triangleright$ pick $\bar{s} \in \bar{F}_q(u, \hat{s}_q - s_q)$, where $\bar{F}_q$ is defined for $(T^{(i)}, \mathbf{z}^{(i)}, \mathcal{G}^{(i)}, \mathbf{f}^{(i)}, \overline{\boldsymbol{\mathscr{C}}}^{(i)})$
   (c) foreach $(r, D) \in \mathcal{D}^{(i)}$:
      i. $(T', \mathbf{z}', \mathcal{G}', \mathbf{f}', \overline{\boldsymbol{\mathscr{C}}}') \leftarrow D(u_q, \bar{s}_r)$
      ii. $\mathcal{J} \leftarrow$ the nodes of T'
      iii. $\mathbf{u}_{\mathcal{J}}, \mathbf{s}_{\mathcal{J}}, \mathbf{s}'_{\mathcal{J}} \leftarrow$ solve the fully constrained (H-OPF) problem associated with $(T', \mathbf{z}', \mathcal{G}', \mathbf{f}', \overline{\boldsymbol{\mathscr{C}}}')$

5.4.6 Approximation scheme

In the light of the conclusions above, we study an approximation scheme for the class of functions

$$g(\mathbf{x}, t) = \min_{\mathbf{x}_1, \ldots, \mathbf{x}_\ell} \left\{ \sum_{k=1}^{\ell} g_k(\mathbf{x}_k, t) : \mathbf{x} = \sum_{k=1}^{\ell} \mathbf{x}_k \right\},$$

where for every k:

- $g_k : \mathbb{E} \times \mathbb{R} \to (-\infty, +\infty]$ is a closed extended real-valued functions;
- $\mathrm{dom}(g_k) \subseteq \mathcal{B} \equiv [0, 1]^n$;
- for g_k we are given an evaluation oracle.

The box $\mathcal{B} \equiv [0,1]^n$ which bounds each $\mathrm{dom}(g_k)$ chosen w.l.o.g, in order to simplify the analysis and notation. In our case, t encodes the voltage absolute value u, and $\mathbf{x}$ encodes $(\mathrm{re}(s), \mathrm{im}(s))$, both after a corresponding transformation which ensures that $(\mathbf{x}, t)$ fit into the box $\mathcal{B}$. The function g can thought of an optimal value of a resource allocation problem, where t is a price of some commodity which is related to the resources we allocated, and x_k is the amount of the k-th resource[1].

Our approximation uses a building block we already introduced in Section 4.3.1 - the minimum value nearest neighbor interpolant of ϕ with $\mathrm{dom}(\phi) \subseteq \mathcal{B}$ on the grid $[\mathcal{B}]_d$:

$$\phi^{(d)}(\mathbf{x}) = \min_{\mathbf{y}} \left\{ \phi(\mathbf{y}) \,:\, \mathbf{y} \in [\mathcal{B}]_d, \|\mathbf{y} - \mathbf{x}\|_\infty \leq \tfrac{1}{2d} \right\}$$

Using the above, we approximate g in the following manner:

- choose a discretization density d;
- construct the minimum value nearest-neighbor interpolant $g_k^{(d)}$ of g_k for every k on the grid $[\mathcal{B}]_d$;
- convolve the interpolants, namely, define the approximation $g^{(d)}$ as

$$g^{(d)}(\mathbf{x}, t) = \min_{\mathbf{x}_1, \ldots, \mathbf{x}_\ell} \left\{ \sum_{k=1}^{\ell} g_k^{(d)}(\mathbf{x}_k, t) \,:\, \mathbf{x} = \sum_{k=1}^{\ell} \mathbf{x}_k \right\}.$$

We also define the corresponding minimizer set

$$G^{(d)}(\mathbf{x}, t) = \operatorname*{argmin}_{\mathbf{x}_1, \ldots, \mathbf{x}_\ell} \left\{ \sum_{k=1}^{\ell} g_k^{(d)}(\mathbf{x}_k, t) \,:\, \mathbf{x} = \sum_{k=1}^{\ell} \mathbf{x}_k \right\}$$

We treat $G^{(d)}$ as an approximation of the minimizers, in the sense that elements in this set achieve an approximate optimal value. The following technical, but simple derivations lead to an algorithm which can both compute $g^{(d)}(\mathbf{x}, t)$ and an element in $G^{(d)}(\mathbf{x}, t)$ for any given $(\mathbf{x}, t)$.

Computing $g^{(d)}(\mathbf{x}, t)$ and obtaining an element in $G^{(d)}(\mathbf{x}, t)$

By definition of $g_k^{(d)}$ as a minimum value nearest neighbor interpolant, we have

$$g_k^{(d)}(\mathbf{x}, t) = \min_{\mathbf{y}, w} \left\{ g_k(\mathbf{y}, w) \,:\, (\mathbf{y}, w) \in [\mathcal{B}]_d, \|\mathbf{y} - \mathbf{x}\|_\infty \leq \frac{1}{2d}, |t - w| \leq \frac{1}{2d} \right\}$$

By plugging the above into the definition of $g^{(d)}$, and changing the order of minimization we obtain:

$$\begin{aligned} g^{(d)}(\mathbf{x}, t) &= \min_{\mathbf{x}_1, \ldots, \mathbf{x}_\ell} \left\{ \sum_{k=1}^{\ell} \min_{\mathbf{y}_k, w_k} \left\{ g_k(\mathbf{y}_k, w_k) \,:\, (\mathbf{y}_k, w_k) \in [\mathcal{B}]_d, \|\mathbf{y}_k - \mathbf{x}_k\|_\infty \leq \frac{1}{2d}, |t - w_k| \leq \frac{1}{2d} \right\} \,:\, \mathbf{x} = \sum_{k=1}^{\ell} \mathbf{x}_k \right\} \\ &= \min_{\substack{\mathbf{x}_1, \ldots, \mathbf{x}_\ell \\ \mathbf{y}_1, \ldots, \mathbf{y}_\ell \\ w_1, \ldots, w_\ell}} \left\{ \sum_{k=1}^{\ell} g_k(\mathbf{y}_k, w_k) \,:\, (\mathbf{y}_k, w_k) \in [\mathcal{B}]_d, \|\mathbf{y}_k - \mathbf{x}_k\|_\infty \leq \frac{1}{2d}, \mathbf{x} = \sum_{k=1}^{\ell} \mathbf{x}_k, |t - w_k| \leq \frac{1}{2d} \right\} \\ &\underset{(*)}{=} \min_{\substack{\mathbf{y}_1, \ldots, \mathbf{y}_\ell \\ w_1, \ldots, w_\ell}} \left\{ \sum_{i=1}^{\ell} g_i(\mathbf{y}_i, w_i) \,:\, (\mathbf{y}_i, w_i) \in [\mathcal{B}]_d, \left\| \mathbf{x} - \sum_{i=1}^{\ell} \mathbf{y}_i \right\|_\infty \leq \frac{\ell}{2d}, |t - w_i| \leq \frac{1}{2d} \right\}, \end{aligned} \tag{Q}$$

where $(*)$ follows from our knowledge about the Minkowski sum of ℓ_∞ balls, namely,

$$\mathbf{x}_i \in \mathcal{B}_\infty\left(\mathbf{y}_i, \frac{1}{2d}\right), \mathbf{x} = \sum_{i=1}^{\ell} \mathbf{x}_i \quad \iff \quad \mathbf{x} \in \sum_{i=1}^{\ell} \mathcal{B}_\infty\left(\mathbf{y}_i, \frac{1}{2d}\right) = \mathcal{B}_\infty\left(\sum_{i=1}^{\ell} \mathbf{y}_i, \frac{\ell}{2d}\right) \tag{5.20}$$

[1] In fact, voltage is a kind of a price - the price, in terms of energy units, of moving a specific particle a distance of 1 meter. Thus, we allocate the apparent power $\mathbf{s}$ which corresponds to the given voltage.

By construction, having solved (Q), its optimal value is $g^{(d)}(\mathbf{x},t)$, and a corresponding minimizer in $G^{(d)}(\mathbf{x},t)$ may be recovered by taking the minimizer of (Q) and solving the *convex* feasibility problem to the left of the $\Longleftrightarrow$ sign in Equation 5.20.

An algorithm for solving (Q) is obtained by another small technical step. By introducing the auxiliary variable $\mathbf{y} = \sum_{i=1}^{\ell} \mathbf{y}_i$ and changing the minimization order, we obtain:

$$\begin{aligned}
g^{(d)}(\mathbf{x},t) &= \min_{\substack{\mathbf{y}_1,\ldots,\mathbf{y}_\ell,\mathbf{y}\\ w_1,\ldots,w_\ell}} \left\{ \sum_{i=1}^{\ell} g_i(\mathbf{y}_i,w_i) \,:\, (\mathbf{y}_i,w_i)\in[\mathcal{B}]_d, \|\mathbf{x}-\mathbf{y}\|_\infty \le \frac{\ell}{2d}, |t-w_i| \le \frac{1}{2d},\ \mathbf{y} = \sum_{i=1}^{\ell} \mathbf{y}_i \right\} \\
&= \min_{w_1,\ldots,w_\ell,\mathbf{y}} \left\{ \underbrace{\min_{\mathbf{y}_1,\ldots,\mathbf{y}_\ell} \left\{ \sum_{i=1}^{\ell} g_i(\mathbf{y}_i,w_i) \,:\, (\mathbf{y}_i,w_i)\in[\mathcal{B}]_d, \mathbf{y} = \sum_{i=1}^{\ell} \mathbf{y}_i \right\}}_{\gamma(\mathbf{y},w_i)} \,:\, |t-w_i| \le \frac{1}{2d},\ \|\mathbf{x}-\mathbf{y}\|_\infty \le \frac{\ell}{2d} \right\} \\
&= \min_{\mathbf{y},w} \left\{ \gamma(\mathbf{y},w) \,:\, |t-w| \le \frac{1}{2d},\ \|\mathbf{x}-\mathbf{y}\|_\infty \le \frac{\ell}{2d} \right\} \\
&= \min_{\mathbf{y},w,\xi} \left\{ \xi \,:\, (\mathbf{y},w,\xi)\in \mathrm{grf}(\gamma), |t-w| \le \frac{1}{2d},\ \|\mathbf{x}-\mathbf{y}\|_\infty \le \frac{\ell}{2d} \right\} \qquad \text{(HQ)}
\end{aligned}$$

Where $\mathrm{grf}(\gamma)$ is the *graph* of γ:

$$\mathrm{grf}(\gamma) \equiv \{(\mathbf{y},w,\gamma(\mathbf{y},w)) \,:\, (\mathbf{y},w)\in \mathrm{dom}(\gamma)\}.$$

Recalling that $[\mathcal{B}]_d$ is a finite set, we conclude that so is $\mathrm{grf}(\gamma)$. The graph, augmented with a corresponding minimizer $(\mathbf{y}_1,\ldots,\mathbf{y}_\ell)$ of the problem defining γ can be pre-computed into a table of the following form:

$\mathbf{y}$	w	γ	$\mathbf{y}_1$	$\cdots$	$\mathbf{y}_\ell$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Given $(\mathbf{x},t)$, solving (HQ) is done by locating the rows in the table satisfying

$$\|\mathbf{x}-\mathbf{y}\|_\infty \le \frac{\ell}{2d}, \quad |t-w| \le \frac{1}{2d},$$

and among resulting rows selecting one having the minimum value of γ. A corresponding optimal solution of (Q) is readily available in the $\mathbf{y}_1,\ldots,\mathbf{y}_\ell$ columns. Locating such rows is the solution of a well-known problem called *proximity query*, and is efficiently solved by existing spatial search data structures[2], i.e. [64].

The graph of γ can be computed by a naïve algorithm which forms all possible tuples of the form $(\mathbf{y} = \sum_{i=1}^{\ell} \mathbf{y}_i, w, \sum_{i=1}^{\ell} g_i(y_i,w), \mathbf{y}_1,\ldots,\mathbf{y}_\ell)$, and from each group of tuples having the same value $(\mathbf{y},w)$ it selects one which has the smallest third coordinate. This algorithm is formally written in Algorithm 6. The resulting method to evaluate $g^{(d)}$ and obtain the corresponding element of $G^{(d)}$ is written in Algorithm 7.

A major drawback of Algorithm 6 is the fact that it has to compute and sort the set $\tilde{\Gamma}$, whose size may be as large as $O(|[\mathcal{B}]_d|^\ell)$. The algorithm can be substantially improved by observing that for each w, the function $\gamma(\cdot,w)$ is an *infimal convolution*. To be precise, denoting $\gamma_i = g_i + \delta_{[\mathcal{B}]_d}$ we have

$$\gamma(\mathbf{y},w) = \min_{\mathbf{y}_1,\ldots,\mathbf{y}_\ell} \left\{ \sum_{i=1}^{\ell} \gamma_i(\mathbf{y}_i,w) \,:\, \mathbf{y} = \sum_{i=1}^{\ell} \mathbf{y}_i \right\}$$

Defining the pairwise $\oplus$ operator as

$$(\phi\oplus\psi)(\mathbf{y},w) = \min_{\mathbf{y}_1,\mathbf{y}_2} \{\phi(\mathbf{y}_1,w) + \psi(\mathbf{y}_1,w) \,:\, \mathbf{y} = \mathbf{y}_1 + \mathbf{y}_2\},$$

we obtain that $\gamma = \gamma_1 \oplus \cdots \oplus \gamma_\ell$. That is, the functions can be convolved incrementally: we first convolve γ_1 and γ_2, and then repeatedly convolve the result with γ_k for $k = 3,\ldots,\ell$. Consequently, the (augmented)

[2] In MATLAB such a data structure can be constructed by the `createns` function, and a proximity query can be performed using the `rangesearch` function.

Algorithm 6 Approximation - pre-computation (naïve)

Input: the functions $g_1, \dots, g_\ell$; the grid density d.
Output: a set $\Gamma \subseteq \mathbb{E} \times \mathbb{R} \times \mathbb{R} \times \mathbb{E}^\ell$.
Steps:

1. compute $\Gamma_i = \{g_i(\mathbf{y}, w) : (\mathbf{y}, w) \in [\mathcal{B}]_d \cap \mathrm{dom}(g_i)\}$ for $i = 1, \dots, \ell$
2. compute

$$\tilde{\Gamma} = \left\{ \left(\mathbf{y}, w, \gamma, (\mathbf{y}_1, \dots, \mathbf{y}_m)\right) : \mathbf{y} = \sum_{i=1}^{\ell} \mathbf{y}_i,\ \gamma = \sum_{i=1}^{\ell} \gamma_i,\ (\mathbf{y}_i, w, \gamma_i) \in \Gamma_i \right\}$$

3. sort the items of $\tilde{\Gamma}$ in lexicographic ascending order of the $(\mathbf{y}, w, \gamma)$ coordinates.

   Invariant: elements with the same $(\mathbf{y}, w)$ are grouped into contiguous subsequences, and in each such subsequence the items with the minimum value of γ appears first.

4. construct Γ by taking the first element from each subsequence.

Algorithm 7 Approximation - evaluation

Input: the pre-computation step output Γ; the discretization denisty d; the query point $(\mathbf{x}, t)$.
Output: $+\infty$ if $(\mathbf{x}, t) \notin \mathrm{dom}(g^{(d)})$
$(a, \mathbf{x}_1, \dots, \mathbf{x}_m)$ such that $a = g^{(d)}(\mathbf{x}, t) = \sum_{k=1}^{\ell} g_k^{(d)}(\mathbf{x}_k)$ and $\mathbf{x} = \sum_{k=1}^{m} \mathbf{x}_k$ otherwise
Steps:

1. perform a proximity query - compute the set

$$A = \{(\gamma, \mathbf{y}_1, \dots, \mathbf{y}_m) : (\mathbf{y}, w, \gamma, (\mathbf{y}_1, \dots, \mathbf{y}_\ell)) \in \Gamma,\ \|\mathbf{y} - \mathbf{x}\| \leq \tfrac{\ell}{2d},\ |t - w| \leq \tfrac{1}{2d}\}$$

2. If a is empty, exit and return $+\infty$.
3. set $(\gamma, \mathbf{y}_1, \dots, \mathbf{y}_\ell)$ as any item in

$$\operatorname*{argmin}_{(\gamma', \mathbf{y}'_1, \dots, \mathbf{y}'_\ell) \in A} \{\gamma'\}.$$

4. set $a = \gamma$.
5. set $\mathbf{x}_1, \dots, \mathbf{x}_\ell$ as any solution of the feasibility problem

$$\mathbf{x}_i \in \mathcal{B}_\infty\left(\mathbf{y}_i, \frac{1}{2d}\right), \mathbf{x} = \sum_{i=1}^{\ell} \mathbf{x}_i$$

graph of γ can be computed incrementally in a similar manner. The resulting algorithm formally written in Algorithm 8.

We point out that the construction of the set Γ in Algorithm 8 is highly parallelizable. Computing the graphs Γ_i is easily parallelizable. Combining graphs requires sorting, which can be done in parallel using modern divide and conquer techniques, such as the algorithm suggested in [58]. The algorithm fits well with the highly parallel nature of modern computers, which are rapidly gaining more parallel processing power as time passes.

Algorithm 8 Approximation - pre-computation

Input: the functions $g_1, \ldots, g_\ell$; the grid density d.
Output: a set $\Gamma \subseteq \mathbb{E} \times \mathbb{R} \times \mathbb{R} \times \mathbb{E}^\ell$.
Steps:

1: **for all** $i = 1, \ldots, \ell$ **do**
2: $\quad \Gamma_i = \mathrm{grf}\left(\gamma_i\right) \equiv \{g_i(\mathbf{y}, w) : (\mathbf{y}, w) \in [\mathcal{B}]_d \cap \mathrm{dom}(g_i)\}$
3: **end for**
4: set $\Gamma \leftarrow \{(\mathbf{y}, w, \gamma, (\mathbf{y})) : (\mathbf{y}, w, \gamma) \in \Gamma_1\}$
5: **for all** $i = 2, \ldots, \ell$ **do**
6: $\quad \Gamma \leftarrow$COMBINE-GRAPHS(Γ, Γ_i)
7: **end for**
8: Return Γ

9: **function** COMBINE-GRAPHS(Γ, Γ')
10: $\quad$ compute

$$\tilde{\Gamma} = \{(\mathbf{z} + \mathbf{y}', w, \alpha + \beta, (\mathbf{y}_1, \ldots, \mathbf{y}_k, \mathbf{y}')) : (\mathbf{z}, w, \gamma, (\mathbf{y}_1, \ldots, \mathbf{y}_k)) \in \Gamma,\ (\mathbf{y}', w, \beta) \in \Gamma'\}$$

11: $\quad$ sort the items of $\tilde{\Gamma}$ in lexicographic ascending order
Invariant: elements with the same $(\mathbf{y}, w)$ are grouped into contiguous subsequences, and in each such subsequence the items with the minimum value of γ appears first.
12: $\quad$ return the set formed by taking the first element from each subsequence.
13: **end function**

5.4.7 Numerical results

Since the conceptual method is exact, the purpose of the experiments is to assess the effect of the approximation - how far is the output of the algorithm from being feasible, and how far it is from being optimal. The root node was chosen among the consumer nodes to be a node with maximum *closeness centrality* $C(i) = (\sum_{j \in \mathcal{V}} d(i, j))^{-1}$, where $d(i, j)$ is the number of edges on the (unique) path between i and j.

We performed our experiments on five networks: one network comprising of 12, 28, and 85 nodes, and two networks comprising of 15 nodes. The full data of each network, including the topology graph, edge impedances and power-loss bounds, and nodal constraints is described in Appendix A. The objective function we used was $\sum_{i \in \mathcal{G}} \mathrm{re}(s_i)$, that is, we aimed to minimize the amount of physical power generated by the generators. We ran our algorithm with approximation densities $d \in \{20, 25, 30, 35, 40, 45, 50, 55, 60\}$ for the absorbed functions. The tree reduction / expansion method we used to solve the inner and terminal problems used approximation density 128.

Let $(\mathbf{u}^d, \mathbf{s}^d)$ be the output produced by our algorithm on some network, and let f^* be the optimal value of the OPF problem on that network. We use

$$\mathrm{E}_f^d = 100 \times \frac{\sum_{i \in \mathcal{G}} \mathrm{re}(s_i^d) - f^*}{f^*}$$

to measure sub-optimality. Computing f^* was done using MATPOWER[72] as a local solver started from a large amount of random points. Since the output of our algorithm are voltage absolute values $\mathbf{u}$ and apparent powers $\mathbf{s}$, it is natural to use Lemma 3 to measure infeasibility. First, the vector $\mathbf{s}'$ is computed from $(\mathbf{u}^d, \mathbf{s}^d)$ by applying equation (3.18b) from the leaves toward the root. Then, we define the 'power

error' as the violation of Equation (3.18a), namely,

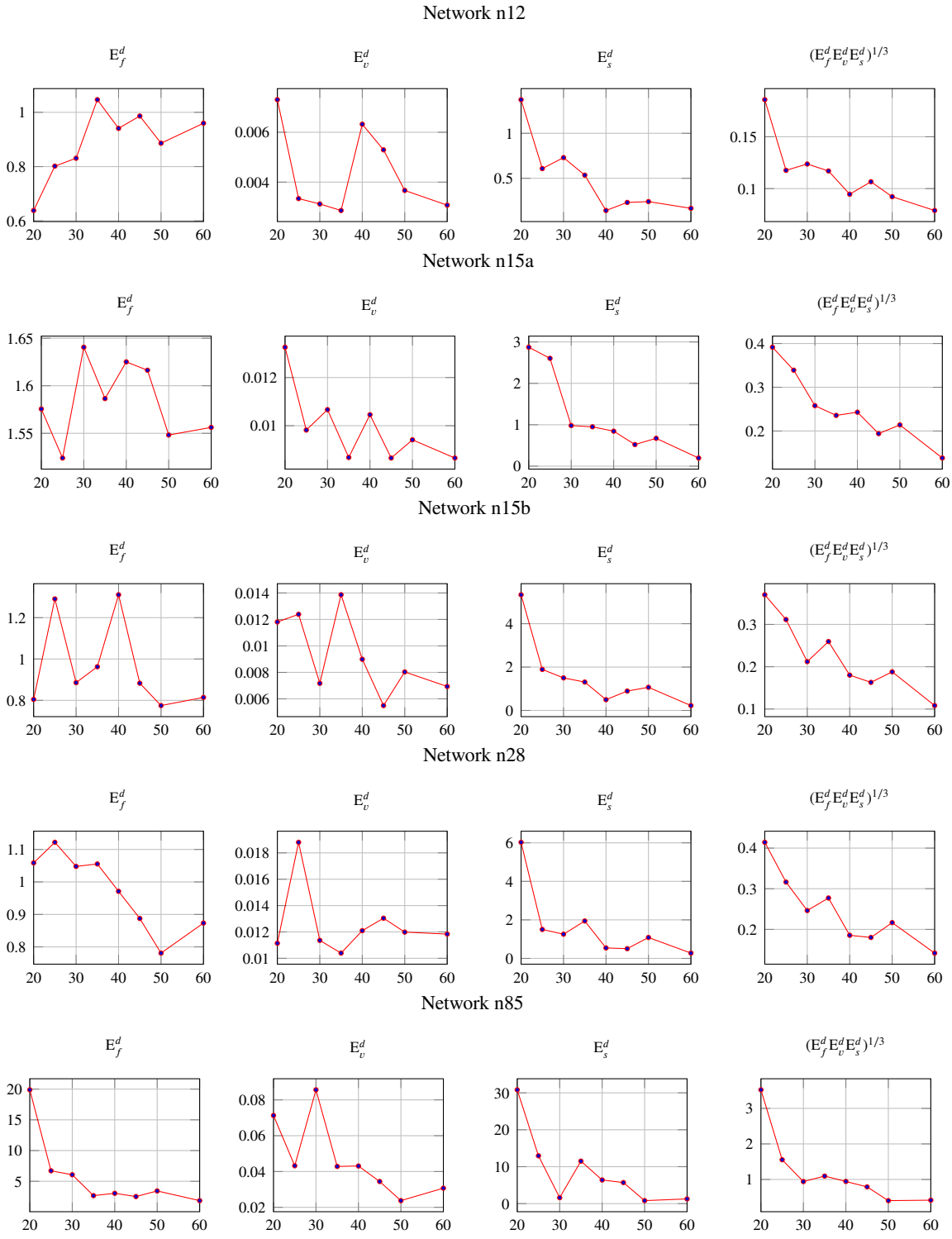
$$E_s^d = |s'_r|,$$

and the 'voltage error' as the worst violation of Equation (3.18c), namely,

$$E_v^d = \max_{\substack{i \in \mathcal{V} \\ j \in \mathrm{ch}_T(i)}} \left| u_i^d - \left| u_j^d - z_{ij} \frac{s_j'^*}{u_j^d} \right| \right|.$$

An evident property of optimization algorithms which do not ensure feasibility, is the fact that the output of the algorithm approaches feasibility with the computational effort, and thus the function values might *increase*. Thus, we use the geometric mean of the error measures $\sqrt[3]{\mathrm{E}_f^d E_s^d E_v^d}$ to evaluate the convergence of our algorithm. The geometric mean was chosen because each error measure has its own units and scale. Figure 5.5 contains these measures for our test networks. We indeed observe a steady dicrease of the geometric mean as the grid density increases. However, we also observe that the accuracy obtained is quite moderate and a higher approximation density d is required. Unfortunately, the memory requirements of our unoptimized MATLAB implementation limited our ability to further increase d.

Figure 5.5: Hierarchical Decomposition Method errors due to approximation. The x-axis is the discretization density d.



Chapter 6

Approximation Convergence Analysis for Networks with Several Generators

The goal of this chapter is to provide the theoretical justification for the approximation scheme we described in Section 5.4.6 for solving OPF problems on several generators to indeed makes sense. The reason for this material to reside in a separate chapter is twofold: *(i)* the practical impact of the result is weak, but its derivation is quite exhaustive, which might impede the readability of Chapter 5; *(ii)* the results about a convergent approximation for parameterized infimal convolutions stand in their own right, and might be useful in other applications.

We concentrate on a single truncation of the hierarchical decomposition method described in Section 5.3 when it is applied to the OPF problem on a compact radial network at a given anchor node q. The corresponding truncated problem defined in (5.12), where the function is $\bar{f}_q(u, s)$ is defined by $\bar{f}_q(u, s) = g(u, \hat{s}_q - s)$ with g having the form

$$g(u, s) = \min_{\mathbf{s}_1, \ldots, \mathbf{s}_\ell} \left\{ \sum_{i=1}^{\ell} g_i(u, s_i) \,:\, s = \sum_{i=1}^{\ell} s_i \right\}, \tag{6.1}$$

and each g_i is the optimal value of an OPF problem on a smaller compact radial network obtained by detaching the sub-tree rooted at the i-th child of q. The function g cannot be computed exactly, and is approximated by a function $g^{(d)}$ given density $d \in \mathbb{N}$ using the scheme described in Section 5.4.6. By adding the auxiliary variable $w = \hat{s}_q - s_q$ to the truncated problem in Equation (5.12) amounts to minimizing

$$\underbrace{\delta_{\{0\}}(s'_r) + \sum_{i \in \mathcal{V} \setminus \mathrm{sub}_T(q)} [f_i(u_i, s_i) + \delta_i(\mathbf{u}_{\mathrm{F}_i}, \mathbf{s}'_{\mathrm{F}_i}, s_i)] + \delta_{[\cdot \geq \alpha]}(u_q) + \delta_q(u_q, s_q, s'_q) + \delta_{[\cdot + \cdot = \hat{s}_q]}(s_q, w)}_{h} + g(u_q, w),$$

From a high-level perspective, the truncated problem is of the form

$$\min_{\mathbf{z}, \mathbf{x}, t} \quad h(\mathbf{z}, \mathbf{x}, t) + g(\mathbf{x}, t), \tag{6.2}$$

where g has the structure in (6.1), and we approximate g with a family of functions $g^{(d)}$. In this chapter, the first section adopts the the high-level perspective, and is dedicated to to studying the conditions on h and g_i which ensure convergence of the optimal value, namely, that

$$\lim_{d \to \infty} \min(h + g^{(d)}) = \min(h + g)$$

In the second section, we inspect the concrete h and g_i functions for the problem at hand, and study the conditions on the power network which ensure, up to a technical assumption, that the conditions above hold.

It is important to note that even if all assumptions do hold, this result has quite limited impact. The hierarchical decomposition method may truncate a problem which was already truncated, resulting in an

'approximation of an approximation'. Thus, the results shown here are applicable to a small family of problems in which only one truncation is sufficient. For the general case, we do not possess any convergence results and rely on the numerical experiments to demonstrate our method's performance. Moreover, the approximation scheme employs the tree reduction/expansion method described in Chapter 4, which itself produces an approximate output. We assume that the tree reduction/expansion method is accurate enough to be considered 'exact', but this assumption has not been verified by years of experience with the method (in contrast to, for example, algorithms for solving eigenvalue problems).

6.1 Approximation convergence analysis

The approximation scheme described in Section 5.4.6 approximates functions of the form

$$g(\mathbf{x},t)=\min_{\mathbf{x}_1,\ldots,\mathbf{x}_\ell}\left\{\sum_{i=1}^{\ell}g_k(\mathbf{x}_i,t)\,:\,\mathbf{x}=\sum_{i=1}^{\ell}\mathbf{x}_i\right\},$$

where for every i:

- $g_i:\mathbb{E}\times\mathbb{R}\to(-\infty,+\infty]$ is a closed extended real-valued functions;
- $\mathrm{dom}(g_i)\subseteq\mathcal{B}\equiv[0,1]^n$;
- for g_i we are given an evaluation oracle.

This setup is assumed throughout the section. The following theorem is our approximation convergence result.

Theorem 5. *Assume that:*

(i) *g_i is bounded and solid piecewise continuous*[1]*, and* $\mathrm{dom}(g_i)$ *is compact, for each* $i=1,\ldots,\ell$*;*

(ii) *h is closed, and there exists a finite partition* $\mathrm{dom}(h)=\bigcup_{j=1}^{m}D_j$ *such that h restricted to each D_j is uniformly continuous.*

Then,

$$\lim_{d\to\infty}(\min h+g^{(d)})=\min h+g.$$

Remark. that the assumption on h of being 'piecewise uniformly continuous' is not an actual restriction. Most objective functions associated with OPF problems are polynomials, or other elementary functions which are uniformly continuous on any bounded set.

Our analysis heavily relies on the results of Section 4.3.1 and on the epigraphical convergence concept. We advice the reader to briefly review these results. The following result is our first step toward a proof of the theorem.

Lemma 11. *Suppose that* $\mathrm{dom}(g_i)$ *is compact and* $g_i=\text{epi-lim}_{d\to\infty}\,g_i^{(d)}$ *for each* $i=1,\ldots,\ell$*. Then,*

$$g=\operatorname*{epi\text{-}lim}_{d\to\infty}g^{(d)}.$$

Proof. The proof is very similar to the proof of Proposition 4.2 in [4]. We prove the lemma for the case $m=2$, that is,

$$g(\mathbf{x},t)=\inf_{\mathbf{z},\mathbf{w}}\{g_1(\mathbf{z},t)+g_2(\mathbf{w},t)\,:\,\mathbf{z}+\mathbf{w}=\mathbf{x}\},$$

and

$$g^{(d)}(\mathbf{x},t)=\inf_{\mathbf{z},\mathbf{w}}\{g_1^{(d)}(\mathbf{z},t)+g_2^{(d)}(\mathbf{w},t)\,:\,\mathbf{z}+\mathbf{w}=\mathbf{x}\}.$$

Since the infimal convolution is associative, the general result follows.

[1] Recall Definition 4.3

Let $\mathbf{x} \in \mathbb{E}$ and $t \in \mathbb{R}$ be arbitrary. By Proposition 7.4 in [57], g_1, g_2 are closed since they are epigraphical limits. Thus, since $\mathrm{dom}(g_1), \mathrm{dom}(g_2)$ are compact, there exist minimizers $\mathbf{z}$ and $\mathbf{w}$ in the definition of $g(\mathbf{x}, t)$ satisfying $\mathbf{x} = \mathbf{z} + \mathbf{w}$ such that

$$g(\mathbf{x}, t) = g_1(\mathbf{z}, t) + g_2(\mathbf{w}, t).$$

Since $\text{epi-lim}_{d\to\infty} g_i^{(d)} = g_i$, there exist sequences $\{(\mathbf{z}^{(d)}, t^{(d)}) : d \in \mathbb{N}\}$ and $\{(\mathbf{w}^{(d)}, t^{(d)}) : d \in \mathbb{N}\}$ converging to $(\mathbf{z}, t)$ and $(\mathbf{w}, t)$, respectively, such that

$$\limsup_{d\to\infty} g_1^{(d)}(\mathbf{z}^{(d)}, t^{(d)}) \le g_1(\mathbf{z}, t), \quad \limsup_{d\to\infty} g_2^{(d)}(\mathbf{w}^{(d)}, t^{(d)}) \le g_2(\mathbf{w}, t).$$

Letting $\mathbf{x}^{(d)} = \mathbf{z}^{(d)} + \mathbf{w}^{(d)}$, we have that $\{(\mathbf{x}^{(d)}, t^{(d)}) : d \in \mathbb{N}\}$ converges to $(\mathbf{x}, t)$, and by definition of $g^{(d)}$ we have

$$\begin{aligned}\limsup_{d\to\infty} g^{(d)}(\mathbf{x}^{(d)}, t^{(d)}) &\le \limsup_{d\to\infty} \{g_1(\mathbf{z}^{(d)}, t^{(d)}) + g_2(\mathbf{w}^{(d)}, t^{(d)})\} \\ &\le \limsup_{d\to\infty} g_1(\mathbf{z}^{(d)}, t^{(d)}) + \limsup_{d\to\infty} g_2(\mathbf{w}^{(d)}, t^{(d)}) \\ &\le g_1(\mathbf{z}, t) + g_2(\mathbf{w}, t) = g(\mathbf{x}, t)\end{aligned}$$

Let $\{(\mathbf{x}^{(d)}, t^{(d)}) : d \in \mathbb{N}\}$ be an arbitrary sequence converging to $(\mathbf{x}, t)$. In particular, $\{\mathbf{x}^{(d)} : d \in \mathbb{N}\}$ converges to $\mathbf{x}$ and $\{t^{(d)} : d \in \mathbb{N}\}$ converges to t. Since the sets $\mathrm{dom}(g_i^{(d)})$ are compact and $g_i^{(d)}$ are closed, for each $d \in \mathbb{N}$ there exist minimizers $(\mathbf{z}^{(d)}, \mathbf{w}^{(d)})$ in the definition of $g^{(d)}$ satisfying $\mathbf{x}^{(d)} = \mathbf{z}^{(d)} + \mathbf{w}^{(d)}$ such that

$$g^{(d)}(\mathbf{x}^{(d)}, t^{(d)}) = g_1^{(d)}(\mathbf{z}^{(d)}, t^{(d)}) + g_2^{(d)}(\mathbf{w}^{(d)}, t^{(d)}).$$

Since $\mathrm{dom}(g_i^{(d)}) \subseteq \mathcal{B}$, for all $d \in \mathbb{N}$ the vectors $(\mathbf{z}^{(d)}, \mathbf{w}^{(d)})$ are contained in the bounded set $\mathcal{B}$, and hence the sequence $\{(\mathbf{z}^{(d)}, \mathbf{w}^{(d)}) : d \in \mathbb{N}\}$ contains subsequences which converge to the cluster points of the sequence. For each such convergent subsequence $\{(\mathbf{z}^{(d_j)}, \mathbf{w}^{(d_j)}) : j \in \mathbb{N}\}$ converging to some cluster point $(\hat{\mathbf{z}}, \hat{\mathbf{w}})$ we must have $\mathbf{x} = \hat{\mathbf{z}} + \hat{\mathbf{w}}$, and by definition of epigraphical limits we have

$$\begin{aligned}\liminf_{j\to\infty} [g_1^{(d_j)}(\mathbf{z}^{(d_j)}, t^{(d_j)}) + g_2^{(d_j)}(\mathbf{w}^{(d_j)}, t^{(d_j)})] &\ge \liminf_{j\to\infty} g_1^{(d_j)}(\mathbf{z}^{(d_j)}, t^{(d_j)}) + \liminf_{j\to\infty} g_2^{(d_j)}(\mathbf{z}^{(d_j)}, t^{(d_j)}) \\ &\ge g_1(\hat{\mathbf{z}}, t) + g_2(\hat{\mathbf{w}}, t) \ge g(\mathbf{x}, t).\end{aligned}$$

Therefore,

$$\liminf_{d\to\infty} g^{(d)}(\mathbf{x}^{(d)}, t^{(d)}) = \liminf_{d\to\infty} [g_1^{(d)}(\mathbf{z}^{(d)}, t^{(d)}) + g_2^{(d)}(\mathbf{w}^{(d)}, t^{(d)})] \ge g(\mathbf{x}, t).$$

□

Recalling Lemma 5, we immediately obtain the following result.

Corollary 2. *Suppose that for any $i \in \{1, \dots, \ell\}$ the function g_i is solid piecewise continuous, and* $\mathrm{dom}(g_i)$ *is compact. Then,*

$$g = \operatorname*{epi-lim}_{d\to\infty} g^{(d)}$$

To continue our analysis, we require more advanced results on epigraphical convergence.

Definition (Set limits)**.** Let $\{C_d : d \in \mathbb{N}\}$ be a sequence of subsets of $\mathbb{E}$. Let $\mathcal{N}_\infty$ be the set of all infinite sequences of the form $\{d_0, d_0 + 1, \dots\}$ and let $\mathcal{N}_\infty^\#$ be the set of all infinite subsequences of $\mathbb{N}$. We define,

$$\liminf_{d\to\infty} C_d = \bigcap_{N\in\mathcal{N}_\infty^\#} \mathrm{cl}\left(\bigcup_{d\in N} C_d\right), \tag{6.3}$$

$$\limsup_{d\to\infty} C_d = \bigcap_{N\in\mathcal{N}_\infty} \mathrm{cl}\left(\bigcup_{d\in N} C_d\right), \tag{6.4}$$

and if

$$C = \limsup_{d\to\infty} C_d = \liminf_{d\to\infty} C_d,$$

then C is called the Painleve-Kuratowski limit (or just 'the limit') of the sequence, which is denoed by $\lim_{d\to\infty} C_d$.

In fact, the reason for the term 'epigraphical conergence' is because it is equivalent to the convergence of the epigraphs of the functions in the Painleve-Kuratowski sense, that is,

$$g = \operatorname*{epi\text{-}lim}_{d\to\infty} g^{(d)} \iff \operatorname{epi}(g) = \lim_{d\to\infty} \operatorname{epi}(g^{(d)})$$

A proof can be found in [57], Definition 7.1 and Proposition 7.2. From the above we can derive the following simple result.

Lemma 12. *Let $\{\phi_d : d \in \mathbb{N}\}$ be a sequence of extended real-valued functions defined on the Euclidean space $\mathbb{E}$ with $\phi = \operatorname{epi\text{-}lim}_{d\to\infty} \phi_d$. Let $C \subseteq \mathbb{E}$. Then,*

$$\operatorname*{epi\text{-}lim}_{d\to\infty}[\phi_d + \delta_C] = \phi + \delta_{\operatorname{cl}(C)}$$

Proof. First, since $\phi = \operatorname{epi\text{-}lim}_{d\to\infty} \phi_d$, we have

$$\operatorname{epi}(\phi) = \limsup_{d\to\infty} \operatorname{epi}(\phi_d) = \liminf_{d\to\infty} \operatorname{epi}(\phi_d).$$

Note, that

$$\operatorname{epi}(\phi_d + \delta_C) = \operatorname{epi}(\phi_d) \cap (C \times \mathbb{R}).$$

By the distributive law for any index set $N \subseteq \mathbb{N}$ we also have

$$\begin{aligned}
&\operatorname{cl}\left(\bigcup_{d\in N} [\operatorname{epi}(\phi_d) \cap (C \times \mathbb{R})]\right) \\
&= \operatorname{cl}\left[(C \times \mathbb{R}) \cap \bigcup_{d\in N} \operatorname{epi}(\phi_d)\right] \\
&= (\operatorname{cl}(C) \times \mathbb{R}) \cap \operatorname{cl}\left(\bigcup_{d\in N} \operatorname{epi}(\phi_d)\right).
\end{aligned}$$

Hence,

$$\begin{aligned}
\liminf_{d\to\infty} \operatorname{epi}(\phi_d + \delta_C) &= \liminf_{d\to\infty} [\operatorname{epi}(\phi_d) \cap (C \times \mathbb{R})] \\
&= \bigcap_{N\in\mathcal{N}_\infty^\#} \operatorname{cl}\left(\bigcup_{d\in N} [\operatorname{epi}(\phi_d) \cap (C \times \mathbb{R})]\right) \\
&= \bigcap_{N\in\mathcal{N}_\infty^\#} (\operatorname{cl}(C) \times \mathbb{R}) \cap \operatorname{cl}\left(\bigcup_{d\in N} \operatorname{epi}(\phi_d)\right) \\
&= (\operatorname{cl}(C) \times \mathbb{R}) \cap \bigcap_{N\in\mathcal{N}_\infty^\#} \operatorname{cl}\left(\bigcup_{d\in N} \operatorname{epi}(\phi_d)\right) \\
&= (\operatorname{cl}(C) \times \mathbb{R}) \cap \liminf_{d\to\infty} \operatorname{epi}(\phi_d) \\
&= (\operatorname{cl}(C) \times \mathbb{R}) \cap \operatorname{epi}(\phi) \\
&= \operatorname{epi}(\phi + \delta_{\operatorname{cl}(C)})
\end{aligned}$$

Similarly,

$$\limsup_{d\to\infty} \operatorname{epi}(\phi_d + \delta_C) = \operatorname{epi}(\phi + \delta_{\operatorname{cl}(C)})$$

Since both limits are equal, we conclude that

$$\lim_{d\to\infty} \operatorname{epi}(\phi_d + \delta_C) = \operatorname{epi}(\phi + \delta_{\mathrm{cl}(C)}),$$

and therefore

$$\operatorname*{epi-lim}_{d\to\infty}[\phi_d + \delta_C] = \phi + \delta_{\mathrm{cl}(C)}.$$

□

Recall that if a function $\phi : C \to \mathbb{R}$ is uniformly continuous on $C \subseteq \mathbb{E}$, then it can be extended to a continuous function $\Phi : \mathrm{cl}(C) \to \mathbb{R}$ which agrees with ϕ on C.

Lemma 13. *Let $\phi : \mathbb{E} \to (-\infty, +\infty]$ be an extended real-valued function which is uniformly continuous on $\mathrm{dom}(\phi)$, assume that $\mathrm{dom}(\phi)$ is bounded, and let Φ be the continuous extension of ϕ to $\mathrm{cl}(\mathrm{dom}(\phi))$. Let $\{\omega_d : d \in \mathbb{N}\}$ be a sequence of extended real-valued functions on $\mathbb{E}$ with the epigraphical limit $\operatorname{epi-lim}_{d\to\infty} \omega_d = \omega$. Then,*

$$\operatorname*{epi-lim}_{d\to\infty}[\phi + \omega_d] = \Phi + \omega.$$

Proof. By the Tietze Extension Theorem ([54], Theorem 35.1), the continuous function Φ can be further extended to a continuous function $\hat{\Phi}$ on the entire space $\mathbb{E}$. By construction, we have $\phi = \Phi + \delta_{\mathrm{dom}(\phi)} = \hat{\Phi} + \delta_{\mathrm{dom}(\phi)}$. Thus,

$$\phi + \omega_d = \hat{\Phi} + \delta_{\mathrm{dom}(\phi)} + \omega_d.$$

By lemma 12, we have

$$\operatorname*{epi-lim}_{d\to\infty}[\delta_{\mathrm{dom}(\phi)} + \omega_d] = \delta_{\mathrm{cl}(\mathrm{dom}(\phi))} + \operatorname*{epi-lim}_{d\to\infty} \omega_d = \delta_{\mathrm{cl}(\mathrm{dom}(\phi))} + \omega$$

By Exercise 7.8 in [57], since $\hat{\Phi}$ is continuous, we have

$$\operatorname*{epi-lim}_{d\to\infty}[\phi + \omega_d] = \operatorname*{epi-lim}_{d\to\infty}[\hat{\Phi} + \delta_{\mathrm{dom}(\phi)} + \omega_d] = \hat{\Phi} + \operatorname*{epi-lim}_{d\to\infty}[\delta_{\mathrm{dom}(\phi)} + \omega_d]$$

Combining both, we have

$$\operatorname*{epi-lim}_{d\to\infty}[\phi + \omega_d] = \hat{\Phi} + \delta_{\mathrm{cl}(\mathrm{dom}(\phi))} + \omega = \Phi + \omega$$

□

We are now ready to prove our desired result.

Proof of Theorem 5. Assume the premises of the theorem: each g_i is solid piecewise continuous with $\mathrm{dom}(g_i)$ being compact, and there exists a finite partition $\mathrm{dom}(h) = \bigcup_{j=1}^{m} D_j$ such that h restricted to D_j is uniformly continuous.

Since $D_1, \ldots, D_\ell$ form a partition of $\mathrm{dom}(h)$, we have

$$\min(g^{(d)} + h) = \min_{j=1,\ldots,m} \left\{ \min(g^{(d)} + \delta_{D_j} + h) \right\}. \tag{6.5}$$

Fix $j \in \{1, \ldots, m\}$, and let $h_j = h + \delta_{D_j}$. By assumption, h_j is uniformly continuous on $\mathrm{dom}(h_j) \equiv D_j$. By Corollary 2 we have $g = \operatorname{epi-lim}_{d\to\infty} g^{(d)}$. Denoting by $\bar{h}_j$ the continuous extension of h_j to $\mathrm{cl}(D_j)$, by Lemma 13 we obtain

$$\operatorname*{epi-lim}_{d\to\infty} \left(h_j + g^{(d)}\right) = \left(\operatorname*{epi-lim}_{d\to\infty} g^{(d)}\right) + \bar{h}_j = g + \bar{h}_j$$

Recall, that each $g^{(d)}$ is closed, and thus $g^{(d)} + h_j$ is closed too. Moreover, since $\mathrm{dom}(h + g^{(d)})$ is bounded, so is $\mathrm{dom}(g^{(d)} + h_j)$, meaning that $g^{(d)} + h_j$ is level bounded. Invoking Theorem 3 we obtain

$$\lim_{d\to\infty} [\min(g^{(d)} + h_j)] = \min(g + \bar{h}_j).$$

Substituting the above into (6.5), we obtain

$$\lim_{d\to\infty}(h+g^{(d)}) = \min_{j=1,\ldots,m}\left[\min(g+\bar{h}_j)\right].$$

Since h is closed, for each $(\mathbf{x},\mathbf{y},t)\in \mathrm{cl}(D_j)$ we have $\bar{h}_j(\mathbf{x},\mathbf{y},t)\geq h(\mathbf{x},\mathbf{y},t)$. Thus,

$$\lim_{d\to\infty}(\min h+g^{(d)}) = \min_{i=1,\ldots,m}\left[\min(g+h_j)\right] = \min(g+h)$$

□

6.2 Power network conditions

In this section we leave the high-level perspective, and specialize the functions h, g as described at the beginning of this chapter to the application of the hierarchical decomposition method to a compact radial network $P = (T, \mathbf{z}, \mathcal{G}, \mathbf{f}, \overline{\boldsymbol{\ell}})$ which was described in Chapter 5. This section is tightly coupled to the concepts defined in Chapter 5, including the definition of a compact radial network, the (H-OPF) problem, the detach operator, and the $\delta_i(\mathbf{u}_{\mathrm{F}_i}, \mathbf{s}'_{\mathrm{F}_i}, s_i)$ functions. Throughout this section we assume that h, g are the functions described at the beginning of this chapter.

The following is our convergence result.

Theorem 6. *Suppose that the nodal objective f_k is semi-algebraic and uniformly continuous on* $\mathrm{dom}(f_k)$ *for every node k. Then:*

1. *for every i, there is a finite partition of* $\mathrm{dom}(g_i) = \bigcup_{j\in\mathcal{I}_i} D_{ij}$, *such that g_i restricted to D_{ij} is continuous;*
2. *if, in addition, we have* $\mathrm{cl}(D_{ij}) = \mathrm{cl}(\mathrm{int}(D_{ij}))$ *for all $i = 1,\ldots,\ell$ and $j\in\mathcal{I}_i$, then we have optimal value convergence, namely,*

$$\lim_{d\to\infty}\min(h+g^{(d)}) = \min(h+g)$$

We first need to introduce additional results in semi-algebraic analysis. Recalling the definition of a semi-algebraic set in Section 4.3.2, it is easy to verify that such sets can be defined by a predicate composed of a combination of polynomial sign conditions using 'and', 'or', and 'not' logical operators (a boolean formula). For example,

$$\{(x,y,z)\in\mathbb{R}^3 : x^2+xy\geq 0 \wedge (xz+yz<0 \vee xy+zy=0)\}.$$

It turns out we can extend the range of predicates which define semi-algebraic sets.

Definition (First order polynomial formula)**.** A predicate $\Phi : \mathbb{E}\to\{\texttt{true},\texttt{false}\}$ with $\mathbb{E}$ being a Euclidean space, is called a first order polynomial formula (FOPF) if either

- $\Phi(\mathbf{x})$ is a polynomial sign condition. That is, $\Phi(\mathbf{x})\equiv p(\mathbf{x})\rhd 0$, where p is a polynomial and $\rhd$ is $>,<,=,\geq$, or $\leq$; or,
- $\Phi(\mathbf{x})$ is either of the form $\exists\mathbf{y} : \Psi(\mathbf{x},\mathbf{y})$ or $\forall\mathbf{y} : \Psi(\mathbf{x},\mathbf{y})$, where Ψ is a first order polynomial formula; or,
- $\Phi(\mathbf{x})$ is either of the form $\Psi_1(\mathbf{x})\wedge\Psi_2(\mathbf{x})$, $\Psi_1(\mathbf{x})\vee\Psi_2(\mathbf{x})$, or $\neg\Psi_1(\mathbf{x})$, where Ψ_1,Ψ_2 are first order polynomial formulas, and $\wedge$, $\vee$, and $\neg$ are the logical 'and', 'or', and 'not' operators.

Example.

- The following is an FOPF, which defines a predicate on $(x,y)\in\mathbb{R}^2$:

$$(\exists z : zx^2-zx+1>0)\wedge(\forall(a,b) : (ax^2+bxy>0)\vee(ax-bxy<0))$$

- Let $g(\mathbf{x}) = \min_{\mathbf{y}} f(\mathbf{x},\mathbf{y})$, and assume that the minimum is well-defined, e.g., the infimum is attained for any $\mathbf{x}\in\mathrm{dom}(g)$. The graph of g is the set of all pairs $(\mathbf{x},z)$ such that $z=g(\mathbf{x})$, or equivalently

$$(\exists\mathbf{y} : z=f(\mathbf{x},\mathbf{y}))\wedge(\forall\mathbf{y} : z\leq f(\mathbf{x},\mathbf{y})).$$

  If f is a polynomial, then the above is an FOPF which defines a predicate on z and $\mathbf{x}$.

Theorem 7 (Tarski-Seidenberg ([18], Thereom 2.6))**.** *Let Φ be an FOPF, and let*

$$S = \{\mathbf{x} \in \mathbb{E} : \Phi(\mathbf{x})\}.$$

Then S is semi-algebraic. That is, there exists a predicate Ψ, which is composed of polynomial sign conditions and logical operators, but without the $\forall$ and $\exists$ operators, such that

$$S = \{\mathbf{x} \in \mathbb{E} : \Psi(\mathbf{x})\}.$$

The Tarski-Seidenberg theorem immediately implies many interesting results, including the semi-algebraicity of the boundary, closure, and interior of semi-algebraic sets, and the domain of semi-algebraic function. We use it to obtain the following result.

Lemma 14. *Let $f : \mathbb{E}_1 \times \mathbb{E}_2 \to (-\infty, +\infty]$ be a semi-algebraic extended real-valued function, assume that $g(\mathbf{x}, \cdot)$ is bounded below, let*

$$g(\mathbf{x}) = \min_{\mathbf{y}} f(\mathbf{x}, \mathbf{y}),$$

and assume that the minimum is well-defined for any $\mathbf{x}$, e.g., either it is attained or the minimization problem is infeasible. Then, g is semi-algebraic.

Proof. Since f is semi-algebraic, by definition, there exists a boolean formula $\Phi(\mathbf{x}, \mathbf{y}, z)$ such that for any $(\mathbf{x}, \mathbf{y}, z) \in \mathbb{E}_1 \times \mathbb{E}_2 \times \mathbb{R}$ we have

$$z = f(\mathbf{x}, \mathbf{y}) \iff \Phi(\mathbf{x}, \mathbf{y}, z)$$

The predicate $f(\mathbf{x}, \mathbf{y}) \geq z$ can be formally written as

$$\begin{gathered} \exists w : w = f(\mathbf{x}, \mathbf{y}) \wedge w \geq z \\ \iff \\ \exists w : \Phi(\mathbf{x}, \mathbf{y}, w) \wedge w \geq z. \end{gathered}$$

Since minimum is well defined, the graph of g can be written as

$$\begin{aligned} &\{(\mathbf{x}, z) \in \mathbb{E}_1 \times \mathbb{R} : z = g(\mathbf{x})\} \\ &\quad = \{(\mathbf{x}, z) \in \mathbb{E}_1 \times \mathbb{R} : (\exists \mathbf{y} : z = f(\mathbf{x}, \mathbf{y})) \wedge (\forall \mathbf{y} : f(\mathbf{x}, \mathbf{y}) \geq z)\}. \\ &\quad = \{(\mathbf{x}, z) \in \mathbb{E}_1 \times \mathbb{R} : (\exists \mathbf{y} : \Phi(\mathbf{x}, \mathbf{y}, z)) \wedge (\forall \mathbf{y}\ \exists w : \Phi(\mathbf{x}, \mathbf{y}, w) \wedge w \geq z)\} \end{aligned}$$

We conclude that the graph of g can be defined by an FOPF, and by the Tarski-Seidenberg theorem it is semi-algebraic. Hence, by definition, g is semi-algebraic. □

Lemma 14 implies the following result.

Lemma 15. *Suppose that the nodal objectives f_i of the compact radial network P are semi-algebraic. Then, for every i there exists a finite partition of $\mathrm{dom}(g_i) = \bigcup_{j \in \mathcal{I}_i} D_{ij}$ such that f_i restricted to each D_{ij} is continuous.*

Proof. We will use the fact that the indicator function of a semi-algebraic set C is semi-algebraic, since

$$\mathrm{grf}(\delta_C) = \{(\mathbf{x}, y) : \mathbf{x} \in C, y = 0\}.$$

Let r_i be the i-th child of q and let j_i be the (only) child of r_i. Recall, that $g_i(u, s)$, defined in (5.14), is the optimal value of an (H-OPF) objective on the compact radial network $\mathsf{detach}(P, q, r_i)(u, s)$. Denoting the nodes in the network family obtained by the $\mathsf{detach}(P, q, r_i)$ operator by $\mathcal{J}_i$, we can equivalently write define g_i by

$$\begin{aligned} g_i(u, s) = \min_{\substack{\mathbf{u} \in \mathbb{R}^{\mathcal{J}_i} \\ \mathbf{s} \in \mathbb{C}^{\mathcal{J}_i} \\ \mathbf{s}' \in \mathbb{C}^{\mathcal{J}_i}}} \Bigg\{ & \delta_{\{0\}}(s'_{r_i}) + \delta_{[\cdot = \cdot]}(u_{r_i}, u) + \frac{1}{\ell} f_q(u_{r_i}) + \delta_{[\cdot + \cdot = 0]}(s, s_{r_i}) \\ & + \delta_{r_i}(\mathbf{u}_{\mathrm{F}_{r_i}}, \mathbf{s}'_{\mathrm{F}_{r_i}}, s_{r_i}) + H_{j_i}\left(\mathbf{u}_{\mathcal{J}_i}, \mathbf{s}_{\mathcal{J}_i}, \mathbf{s}'_{\mathcal{J}_i}\right) \Bigg\} + \delta_{[\underline{u}_q, \overline{u}_q]}(u) \end{aligned}$$

Note, that the minimization objective above is composed of the functions f_i associated with a subset of the nodes, and indicator functions. By assumption, all functions f_i are semi-algebraic. The indicator functions $\delta_{\{0\}}$, $\delta_{[\cdot=\cdot]}$, and $\delta_{[\cdot+\cdot=0]}$ are semi-algebraic since the corresponding sets are semi-algebraic by definition. The constraints imposed by the indicator functions δ_i can be equivalently written as the following system with auxiliary variables $\tilde{\mathbf{s}} \in \mathbb{C}^{\mathrm{ch}_T(i)}$

$$
\begin{aligned}
&s'_i = s_i + \sum_{k \in \mathrm{ch}_T(i)} \tilde{s}_k, && k \in \mathrm{ch}_T(i),\\
&u_i^2 u_k^2 = u_k^4 - z_{ik} {s'_k}^* u_k^2 - z_{ik}^* s'_k u_k^2 + z_{ik}^2 |s'_k|^2 && k \in \mathrm{ch}_T(i),\\
&|\tilde{s}_k|^2 |z_{ik}|^2 \le u_i^2 \overline{\ell}_{ik} && k \in \mathrm{ch}_T(i),\\
&u_k \ge \alpha && k \in \mathrm{ch}_T(i),\\
&\tilde{s}_k u_k^2 = s'_k - z_{ik} |s'_k|^2 && k \in \mathrm{ch}_T(i).
\end{aligned}
$$

The conditions above define a system of polynomial equations and inequalities in the real variables obtained assuming rectangular complex representation, and thus describe a semi-algebraic set of the tuples $(s_i, \mathbf{u}, \mathbf{s}, \mathbf{s}', \tilde{\mathbf{s}})$ which satisfy them. Its projection onto the $(s_i, \mathbf{u}, \mathbf{s}, \mathbf{s}')$ coordinates by the Tarski-Seidenberg theorem is also semi-algebraic. Hence, δi is semi-algebraic as an indicator of the above projection.

The minimization objective defining g_i is thus semi-algebraic, and therefore g_i is semi-algebraic by Lemma 14. The desired result follows directly from Lemma 6. □

Now we are ready to prove our desired result.

Proof of Theorem 6. Conclusion (i) follows immediately by Lemma 15. We thus proceed to proving (ii).

Suppose that for each partition $\mathrm{dom}(g_i) = \bigcup_{j \in \mathcal{I}_i} D_{ij}$ satisfies $\mathrm{cl}(D_{ij}) = \mathrm{cl}(\mathrm{int}(D_{ij}))$. Since the nodal objectives f_i of a compact radial network are bounded, g_i must be bounded as well. By Lemma 10, since the compactness property holds, $\mathrm{dom}(g_i)$ must be compact. An indicator function is constant, and thus by definition uniformly continuous on its domain. The nodal objectives f_i are assumed to be uniformly continuous. Hence, the function h, which is a sum of nodal objectives and indicator functions is uniformly continuous on its domain. Invoking Theorem 5 we conclude that

$$\lim_{d \to \infty} \min(h + g^{(d)}) = \min(h + g)$$

□

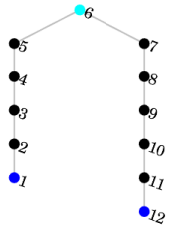
Appendix A

Problem Data for Networks with Several Generators

Figures A.1, A.2, A.3, A.4, A.5 contain the topology, edge impedances and power-loss bounds, and the constraints for the consumer and generator nodes. Each topology tree is visualized along with its chosen root, the generators, and the anchor nodes which the hierarchical decomposition method (see Section 5.3) chooses. The power-loss bounds were computed

Figure A.1: Network n12. Cyan - anchors. Blue - generators.

Topology.



Consumers

j	$\hat{s}_j$	$\underline{u}_j$	$\overline{u}_j$	j	$\hat{s}_j$	$\underline{u}_j$	$\overline{u}_j$
2	$-60-60\mathbf{i}$	0.9	1.1	3	$-40-30\mathbf{i}$	0.9	1.1
4	$-55-55\mathbf{i}$	0.9	1.1	5	$-30-30\mathbf{i}$	0.9	1.1
6	$-20-15\mathbf{i}$	0.9	1.1	7	$-55-55\mathbf{i}$	0.9	1.1
8	$-45-45\mathbf{i}$	0.9	1.1	9	$-40-40\mathbf{i}$	0.9	1.1
10	$-35-30\mathbf{i}$	0.9	1.1	11	$-40-30\mathbf{i}$	0.9	1.1

Generators

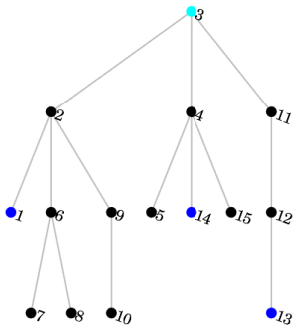
j	$\underline{p}_j$	$\overline{p}_j$	$\underline{q}_j$	$\overline{q}_j$	$\underline{u}_j$	$\overline{u}_j$
1	0	888.668	-572.619	-572.619	0.9	1.1
12	0	888.668	-572.619	-572.619	0.9	1.1

Edges

i	j	z_{ij}	$\overline{\ell}_{ij}$	i	j	z_{ij}	$\overline{\ell}_{ij}$
2	1	$9.03306\times10^{-6}+3.76033\times10^{-6}\mathbf{i}$	65.4097	3	2	$9.78512\times10^{-6}+4.08264\times10^{-6}\mathbf{i}$	60.3621
4	3	$1.7314\times10^{-5}+7.21488\times10^{-6}\mathbf{i}$	34.1203	5	4	$2.63471\times10^{-5}+1.09835\times10^{-5}\mathbf{i}$	22.4209
6	5	$9.03306\times10^{-6}+3.76033\times10^{-6}\mathbf{i}$	65.4097	7	6	$8.28099\times10^{-6}+3.44628\times10^{-6}\mathbf{i}$	71.353
8	7	$3.63884\times10^{-5}+1.00413\times10^{-5}\mathbf{i}$	16.9543	9	8	$4.66281\times10^{-5}+1.31983\times10^{-5}\mathbf{i}$	13.2068
10	9	$2.38843\times10^{-5}+6.76033\times10^{-6}\mathbf{i}$	25.783	11	10	$1.25124\times10^{-5}+3.53719\times10^{-6}\mathbf{i}$	49.2203
12	11	$1.02314\times10^{-5}+2.90083\times10^{-6}\mathbf{i}$	60.1805				

Figure A.2: Network n15a. Cyan - anchors. Blue - generators.

Topology



Consumers

j	$\hat{s}_j$	$\underline{u}_j$	$\overline{u}_j$	j	$\hat{s}_j$	$\underline{u}_j$	$\overline{u}_j$
2	$-44.1 - 44.9883\mathbf{i}$	0.9	1.1	3	$-70 - 71.41\mathbf{i}$	0.9	1.1
4	$-140 - 142.82\mathbf{i}$	0.9	1.1	5	$-44.1 - 44.9883\mathbf{i}$	0.9	1.1
6	$-140 - 142.82\mathbf{i}$	0.9	1.1	7	$-140 - 142.82\mathbf{i}$	0.9	1.1
8	$-70 - 71.41\mathbf{i}$	0.9	1.1	9	$-70 - 71.41\mathbf{i}$	0.9	1.1
10	$-44.1 - 44.9883\mathbf{i}$	0.9	1.1	11	$-140 - 142.82\mathbf{i}$	0.9	1.1
12	$-70 - 71.41\mathbf{i}$	0.9	1.1	15	$-140 - 142.82\mathbf{i}$	0.9	1.1

Generators

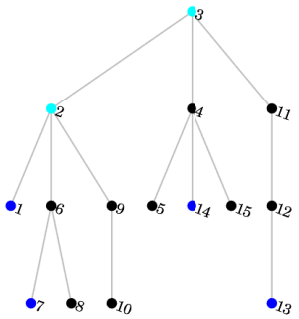
j	$\underline{p}_j$	$\overline{p}_j$	$\underline{q}_j$	$\overline{q}_j$	$\underline{u}_j$	$\overline{u}_j$
1	0	1687.85	-1595.79	-1595.79	0.9	1.1
13	0	1687.85	-1595.79	-1595.79	0.9	1.1
14	0	1687.85	-1595.79	-1595.79	0.9	1.1

Edges

i	j	z_{ij}	$\overline{\ell}_{ij}$	i	j	z_{ij}	$\overline{\ell}_{ij}$
2	1	$1.11826 \times 10^{-5} + 1.09379 \times 10^{-5}\mathbf{i}$	40.9142	3	2	$9.6714 \times 10^{-6} + 9.45983 \times 10^{-6}\mathbf{i}$	47.307
6	2	$2.11345 \times 10^{-5} + 1.42554 \times 10^{-5}\mathbf{i}$	25.1052	9	2	$1.66378 \times 10^{-5} + 1.12223 \times 10^{-5}\mathbf{i}$	31.8903
4	3	$6.95132 \times 10^{-6} + 6.79926 \times 10^{-6}\mathbf{i}$	65.8184	11	3	$1.48391 \times 10^{-5} + 1.00091 \times 10^{-5}\mathbf{i}$	35.7558
5	4	$1.25907 \times 10^{-5} + 8.49256 \times 10^{-6}\mathbf{i}$	42.1408	14	4	$1.84364 \times 10^{-5} + 1.24355 \times 10^{-5}\mathbf{i}$	28.7791
15	4	$9.89273 \times 10^{-6} + 6.67273 \times 10^{-6}\mathbf{i}$	53.6338	7	6	$8.99339 \times 10^{-6} + 6.06612 \times 10^{-6}\mathbf{i}$	58.9971
8	6	$1.03424 \times 10^{-5} + 6.97603 \times 10^{-6}\mathbf{i}$	51.3019	10	9	$1.39398 \times 10^{-5} + 9.40248 \times 10^{-6}\mathbf{i}$	38.0627
12	11	$2.02351 \times 10^{-5} + 1.36488 \times 10^{-5}\mathbf{i}$	26.2209	13	12	$1.66378 \times 10^{-5} + 1.12223 \times 10^{-5}\mathbf{i}$	31.8903

Figure A.3: Network n15b. Cyan - anchors. Blue - generators.

Topology.



Consumers

j	$\hat{s}_j$	$\underline{u}_j$	$\overline{u}_j$	j	$\hat{s}_j$	$\underline{u}_j$	$\overline{u}_j$
2	$-44.1 - 44.9883\mathbf{i}$	0.9	1.1	3	$-70 - 71.41\mathbf{i}$	0.9	1.1
4	$-140 - 142.82\mathbf{i}$	0.9	1.1	5	$-44.1 - 44.9883\mathbf{i}$	0.9	1.1
6	$-140 - 142.82\mathbf{i}$	0.9	1.1	8	$-70 - 71.41\mathbf{i}$	0.9	1.1
9	$-70 - 71.41\mathbf{i}$	0.9	1.1	10	$-44.1 - 44.9883\mathbf{i}$	0.9	1.1
11	$-140 - 142.82\mathbf{i}$	0.9	1.1	12	$-70 - 71.41\mathbf{i}$	0.9	1.1

Generators

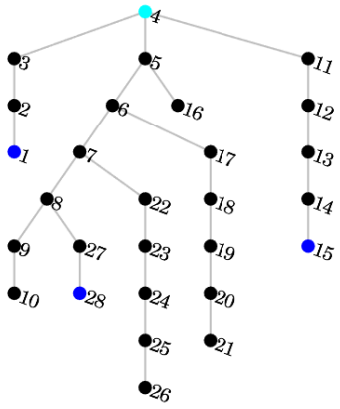
j	$\underline{p}_j$	$\overline{p}_j$	$\underline{q}_j$	$\overline{q}_j$	$\underline{u}_j$	$\overline{u}_j$
1	0	1687.85	−1595.79	−1595.79	0.9	1.1
7	0	1687.85	−1595.79	−1595.79	0.9	1.1
13	0	1687.85	−1595.79	−1595.79	0.9	1.1
14	0	1687.85	−1595.79	−1595.79	0.9	1.1

Edges

i	j	z_{ij}	$\overline{\ell}_{ij}$	i	j	z_{ij}	$\overline{\ell}_{ij}$
2	1	$1.11826 \times 10^{-5} + 1.09379 \times 10^{-5}\mathbf{i}$	40.9142	3	2	$9.6714 \times 10^{-6} + 9.45983 \times 10^{-6}\mathbf{i}$	47.307
6	2	$2.11345 \times 10^{-5} + 1.42554 \times 10^{-5}\mathbf{i}$	25.1052	9	2	$1.66378 \times 10^{-5} + 1.12223 \times 10^{-5}\mathbf{i}$	31.8903
4	3	$6.95132 \times 10^{-6} + 6.79926 \times 10^{-6}\mathbf{i}$	65.8184	11	3	$1.48391 \times 10^{-5} + 1.00091 \times 10^{-5}\mathbf{i}$	35.7558
5	4	$1.25907 \times 10^{-5} + 8.49256 \times 10^{-6}\mathbf{i}$	42.1408	14	4	$1.84364 \times 10^{-5} + 1.24355 \times 10^{-5}\mathbf{i}$	28.7791
15	4	$9.89273 \times 10^{-6} + 6.67273 \times 10^{-6}\mathbf{i}$	53.6338	7	6	$8.99339 \times 10^{-6} + 6.06612 \times 10^{-6}\mathbf{i}$	58.9971
8	6	$1.03424 \times 10^{-5} + 6.97603 \times 10^{-6}\mathbf{i}$	51.3019	10	9	$1.39398 \times 10^{-5} + 9.40248 \times 10^{-6}\mathbf{i}$	38.0627
12	11	$2.02351 \times 10^{-5} + 1.36488 \times 10^{-5}\mathbf{i}$	26.2209	13	12	$1.66378 \times 10^{-5} + 1.12223 \times 10^{-5}\mathbf{i}$	31.8903

Figure A.4: Network n28. Cyan - anchors. Blue - generators.

Topology. Cyan - anchors. Blue - generators.



Consumers

j	$\hat{s}_j$	$\underline{u}_j$	$\overline{u}_j$	j	$\hat{s}_j$	$\underline{u}_j$	$\overline{u}_j$
2	$-35.28 - 35.9927\mathbf{i}$	0.9	1.1	3	$-14 - 14.2828\mathbf{i}$	0.9	1.1
4	$-35.28 - 35.9927\mathbf{i}$	0.9	1.1	5	$-14 - 14.2828\mathbf{i}$	0.9	1.1
6	$-35.28 - 35.9927\mathbf{i}$	0.9	1.1	7	$-35.28 - 35.9927\mathbf{i}$	0.9	1.1
8	$-35.28 - 35.9927\mathbf{i}$	0.9	1.1	9	$-14 - 14.2828\mathbf{i}$	0.9	1.1
10	$-14 - 14.2828\mathbf{i}$	0.9	1.1	11	$-56 - 57.1312\mathbf{i}$	0.9	1.1
12	$-35.28 - 35.9927\mathbf{i}$	0.9	1.1	13	$-35.28 - 35.9927\mathbf{i}$	0.9	1.1
14	$-14 - 14.2828\mathbf{i}$	0.9	1.1	16	$-35.28 - 35.9927\mathbf{i}$	0.9	1.1
17	$-8.96 - 9.14099\mathbf{i}$	0.9	1.1	18	$-8.96 - 9.14099\mathbf{i}$	0.9	1.1
19	$-35.28 - 35.9927\mathbf{i}$	0.9	1.1	20	$-35.28 - 35.9927\mathbf{i}$	0.9	1.1
21	$-14 - 14.2828\mathbf{i}$	0.9	1.1	22	$-35.28 - 35.9927\mathbf{i}$	0.9	1.1
23	$-8.96 - 9.14099\mathbf{i}$	0.9	1.1	24	$-56 - 57.1312\mathbf{i}$	0.9	1.1
25	$-8.96 - 9.14099\mathbf{i}$	0.9	1.1	26	$-35.28 - 35.9927\mathbf{i}$	0.9	1.1

Generators

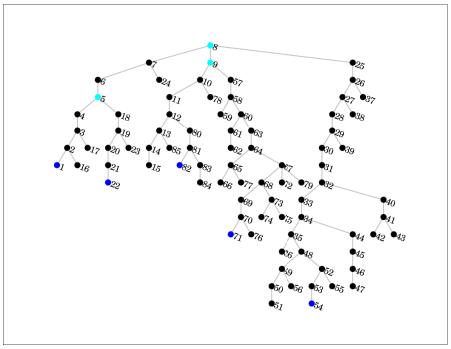
j	$\underline{p}_j$	$\overline{p}_j$	$\underline{q}_j$	$\overline{q}_j$	$\underline{u}_j$	$\overline{u}_j$
1	0	5073.83	−2793.76	−2793.76	0.9	1.1
15	0	5073.83	−2793.76	−2793.76	0.9	1.1
28	0	5073.83	−2793.76	−2793.76	0.9	1.1

Edges

i	j	z_{ij}	$\overline{\ell}_{ij}$	i	j	z_{ij}	$\overline{\ell}_{ij}$
2	1	$9.89256 \times 10^{-6} + 6.77686 \times 10^{-6}\mathbf{i}$	120.088	3	2	$1.4843 \times 10^{-5} + 1.01736 \times 10^{-5}\mathbf{i}$	80.0228
4	3	$1.07934 \times 10^{-5} + 7.39669 \times 10^{-6}\mathbf{i}$	110.053	5	4	$1.52975 \times 10^{-5} + 1.04793 \times 10^{-5}\mathbf{i}$	77.6587
11	4	$2.33306 \times 10^{-5} + 9.68595 \times 10^{-6}\mathbf{i}$	57.0042	6	5	$1.2595 \times 10^{-5} + 8.6281 \times 10^{-6}\mathbf{i}$	94.3215
16	5	$2.10744 \times 10^{-5} + 8.7438 \times 10^{-6}\mathbf{i}$	63.1128	7	6	$1.57438 \times 10^{-5} + 1.07851 \times 10^{-5}\mathbf{i}$	75.4572
17	6	$1.12893 \times 10^{-5} + 4.68595 \times 10^{-6}\mathbf{i}$	117.809	8	7	$9.89256 \times 10^{-6} + 6.77686 \times 10^{-6}\mathbf{i}$	120.088
22	7	$1.27934 \times 10^{-5} + 5.30579 \times 10^{-6}\mathbf{i}$	103.971	9	8	$5.39669 \times 10^{-6} + 3.69421 \times 10^{-6}\mathbf{i}$	220.184
27	8	$4.5124 \times 10^{-6} + 1.86777 \times 10^{-6}\mathbf{i}$	294.86	10	9	$9.44628 \times 10^{-6} + 6.47107 \times 10^{-6}\mathbf{i}$	125.762
12	11	$9.78512 \times 10^{-6} + 4.05785 \times 10^{-6}\mathbf{i}$	135.937	13	12	$8.28099 \times 10^{-6} + 3.43802 \times 10^{-6}\mathbf{i}$	160.601
14	13	$3.76033 \times 10^{-6} + 1.56198 \times 10^{-6}\mathbf{i}$	353.649	15	14	$4.5124 \times 10^{-6} + 1.87603 \times 10^{-6}\mathbf{i}$	294.669
18	17	$6.7686 \times 10^{-6} + 2.80992 \times 10^{-6}\mathbf{i}$	196.488	19	18	$1.27934 \times 10^{-5} + 5.30579 \times 10^{-6}\mathbf{i}$	103.971
20	19	$1.12893 \times 10^{-5} + 4.68595 \times 10^{-6}\mathbf{i}$	117.809	21	20	$2.93554 \times 10^{-5} + 1.21818 \times 10^{-5}\mathbf{i}$	45.3078
23	22	$9.02479 \times 10^{-6} + 3.7438 \times 10^{-6}\mathbf{i}$	147.382	24	23	$7.52066 \times 10^{-6} + 3.12397 \times 10^{-6}\mathbf{i}$	176.824
25	24	$3.76033 \times 10^{-6} + 1.56198 \times 10^{-6}\mathbf{i}$	353.649	26	25	$3.00826 \times 10^{-6} + 1.24793 \times 10^{-6}\mathbf{i}$	442.147
28	27	$2.2562 \times 10^{-6} + 9.33884 \times 10^{-7}\mathbf{i}$	589.72				

Figure A.5: Network n85. Cyan - anchors. Blue - generators.

Topology



Consumers

j	$\hat{s}_j$	$\underline{u}_j$	$\overline{u}_j$	j	$\hat{s}_j$	$\underline{u}_j$	$\overline{u}_j$	j	$\hat{s}_j$	$\underline{u}_j$	$\overline{u}_j$
2	0	0.9	1.1	3	0	0.9	1.1	4	$-56-57.13\mathbf{i}$	0.9	1.1
5	0	0.9	1.1	6	$-35.28-35.99\mathbf{i}$	0.9	1.1	7	0	0.9	1.1
8	$-35.28-35.99\mathbf{i}$	0.9	1.1	9	0	0.9	1.1	10	0	0.9	1.1
11	$-56-57.13\mathbf{i}$	0.9	1.1	12	0	0.9	1.1	13	0	0.9	1.1
14	$-35.28-35.99\mathbf{i}$	0.9	1.1	15	$-35.28-35.99\mathbf{i}$	0.9	1.1	16	$-35.28-35.99\mathbf{i}$	0.9	1.1
17	$-112-114.3\mathbf{i}$	0.9	1.1	18	$-56-57.13\mathbf{i}$	0.9	1.1	19	$-56-57.13\mathbf{i}$	0.9	1.1
20	$-35.28-35.99\mathbf{i}$	0.9	1.1	21	$-35.28-35.99\mathbf{i}$	0.9	1.1	23	$-56-57.13\mathbf{i}$	0.9	1.1
24	$-35.28-35.99\mathbf{i}$	0.9	1.1	25	$-35.28-35.99\mathbf{i}$	0.9	1.1	26	$-56-57.13\mathbf{i}$	0.9	1.1
27	0	0.9	1.1	28	$-56-57.13\mathbf{i}$	0.9	1.1	29	0	0.9	1.1
30	$-35.28-35.99\mathbf{i}$	0.9	1.1	31	$-35.28-35.99\mathbf{i}$	0.9	1.1	32	0	0.9	1.1
33	$-14-14.28\mathbf{i}$	0.9	1.1	34	0	0.9	1.1	35	0	0.9	1.1
36	$-35.28-35.99\mathbf{i}$	0.9	1.1	37	$-56-57.13\mathbf{i}$	0.9	1.1	38	$-56-57.13\mathbf{i}$	0.9	1.1
39	$-56-57.13\mathbf{i}$	0.9	1.1	40	$-35.28-35.99\mathbf{i}$	0.9	1.1	41	0	0.9	1.1
42	$-35.28-35.99\mathbf{i}$	0.9	1.1	43	$-35.28-35.99\mathbf{i}$	0.9	1.1	44	$-35.28-35.99\mathbf{i}$	0.9	1.1
45	$-35.28-35.99\mathbf{i}$	0.9	1.1	46	$-35.28-35.99\mathbf{i}$	0.9	1.1	47	$-14-14.28\mathbf{i}$	0.9	1.1
48	0	0.9	1.1	49	0	0.9	1.1	50	$-36.28-37.01\mathbf{i}$	0.9	1.1
51	$-56-57.13\mathbf{i}$	0.9	1.1	52	0	0.9	1.1	53	$-35.28-35.99\mathbf{i}$	0.9	1.1
55	$-56-57.13\mathbf{i}$	0.9	1.1	56	$-14-14.28\mathbf{i}$	0.9	1.1	57	$-56-57.13\mathbf{i}$	0.9	1.1
58	0	0.9	1.1	59	$-56-57.13\mathbf{i}$	0.9	1.1	60	$-56-57.13\mathbf{i}$	0.9	1.1
61	$-56-57.13\mathbf{i}$	0.9	1.1	62	$-56-57.13\mathbf{i}$	0.9	1.1	63	$-14-14.28\mathbf{i}$	0.9	1.1
64	0	0.9	1.1	65	0	0.9	1.1	66	$-56-57.13\mathbf{i}$	0.9	1.1
67	0	0.9	1.1	68	0	0.9	1.1	69	$-56-57.13\mathbf{i}$	0.9	1.1
70	0	0.9	1.1	72	$-56-57.13\mathbf{i}$	0.9	1.1	73	0	0.9	1.1
74	$-56-57.13\mathbf{i}$	0.9	1.1	75	$-35.28-35.99\mathbf{i}$	0.9	1.1	76	$-56-57.13\mathbf{i}$	0.9	1.1
77	$-14-14.28\mathbf{i}$	0.9	1.1	78	$-56-57.13\mathbf{i}$	0.9	1.1	79	$-35.28-35.99\mathbf{i}$	0.9	1.1
80	$-56-57.13\mathbf{i}$	0.9	1.1	81	0	0.9	1.1	83	$-35.28-35.99\mathbf{i}$	0.9	1.1

Generators

j	$\underline{p}_j$	$\overline{p}_j$	$\underline{q}_j$	$\overline{q}_j$	$\underline{u}_j$	$\overline{u}_j$
1	0	$1.761 \cdot 10^4$	-9683	-9683	0.9	1.1
22	0	$1.761 \cdot 10^4$	-9683	-9683	0.9	1.1
54	0	$1.761 \cdot 10^4$	-9683	-9683	0.9	1.1
82	0	$1.761 \cdot 10^4$	-9683	-9683	0.9	1.1
71	0	$1.761 \cdot 10^4$	-9683	-9683	0.9	1.1

Edges

i	j	z_{ij}	$\overline{\ell}_{ij}$	i	j	z_{ij}	$\overline{\ell}_{ij}$	i	j	z_{ij}	$\overline{\ell}_{ij}$
2	1	$8.926 \cdot 10^{-7} + 6.198 \cdot 10^{-7}\mathbf{i}$	1178	3	2	$1.347 \cdot 10^{-6} + 9.256 \cdot 10^{-7}\mathbf{i}$	783.1	16	2	$6.017 \cdot 10^{-6} + 2.496 \cdot 10^{-6}\mathbf{i}$	196.5
4	3	$1.793 \cdot 10^{-6} + 1.231 \cdot 10^{-6}\mathbf{i}$	588.4	17	3	$3.76 \cdot 10^{-6} + 1.562 \cdot 10^{-6}\mathbf{i}$	314.4	5	4	$8.926 \cdot 10^{-7} + 6.116 \cdot 10^{-7}\mathbf{i}$	1183
6	5	$3.595 \cdot 10^{-6} + 2.463 \cdot 10^{-6}\mathbf{i}$	293.7	18	5	$6.777 \cdot 10^{-6} + 2.81 \cdot 10^{-6}\mathbf{i}$	174.5	7	6	$2.248 \cdot 10^{-6} + 1.537 \cdot 10^{-6}\mathbf{i}$	470
8	7	$9.893 \cdot 10^{-6} + 6.777 \cdot 10^{-6}\mathbf{i}$	106.7	24	7	$7.521 \cdot 10^{-6} + 3.124 \cdot 10^{-6}\mathbf{i}$	157.2	9	8	$8.926 \cdot 10^{-7} + 6.116 \cdot 10^{-7}\mathbf{i}$	1183
25	8	$3.76 \cdot 10^{-6} + 1.562 \cdot 10^{-6}\mathbf{i}$	314.4	10	9	$4.942 \cdot 10^{-6} + 3.388 \cdot 10^{-6}\mathbf{i}$	213.6	57	9	$2.256 \cdot 10^{-6} + 9.339 \cdot 10^{-7}\mathbf{i}$	524.2
11	10	$4.496 \cdot 10^{-6} + 3.083 \cdot 10^{-6}\mathbf{i}$	234.8	78	10	$5.264 \cdot 10^{-6} + 2.182 \cdot 10^{-6}\mathbf{i}$	224.6	12	11	$4.496 \cdot 10^{-6} + 3.083 \cdot 10^{-6}\mathbf{i}$	234.8
13	12	$4.942 \cdot 10^{-6} + 3.388 \cdot 10^{-6}\mathbf{i}$	213.6	80	12	$6.017 \cdot 10^{-6} + 2.496 \cdot 10^{-6}\mathbf{i}$	196.5	14	13	$2.248 \cdot 10^{-6} + 1.537 \cdot 10^{-6}\mathbf{i}$	470
85	13	$6.769 \cdot 10^{-6} + 2.81 \cdot 10^{-6}\mathbf{i}$	174.7	15	14	$2.694 \cdot 10^{-6} + 1.843 \cdot 10^{-6}\mathbf{i}$	392.1	19	18	$5.264 \cdot 10^{-6} + 2.182 \cdot 10^{-6}\mathbf{i}$	224.6
20	19	$3.76 \cdot 10^{-6} + 1.562 \cdot 10^{-6}\mathbf{i}$	314.4	23	19	$1.504 \cdot 10^{-6} + 6.198 \cdot 10^{-7}\mathbf{i}$	786.8	21	20	$6.769 \cdot 10^{-6} + 2.81 \cdot 10^{-6}\mathbf{i}$	174.7
22	21	$1.279 \cdot 10^{-5} + 5.306 \cdot 10^{-6}\mathbf{i}$	92.42	26	25	$3.008 \cdot 10^{-6} + 1.248 \cdot 10^{-6}\mathbf{i}$	393	27	26	$4.512 \cdot 10^{-6} + 1.868 \cdot 10^{-6}\mathbf{i}$	262.1
37	26	$3.008 \cdot 10^{-6} + 1.248 \cdot 10^{-6}\mathbf{i}$	393	28	27	$2.256 \cdot 10^{-6} + 9.339 \cdot 10^{-7}\mathbf{i}$	524.2	38	27	$8.281 \cdot 10^{-6} + 3.438 \cdot 10^{-6}\mathbf{i}$	142.8
29	28	$4.512 \cdot 10^{-6} + 1.868 \cdot 10^{-6}\mathbf{i}$	262.1	30	29	$4.512 \cdot 10^{-6} + 1.868 \cdot 10^{-6}\mathbf{i}$	262.1	39	29	$4.512 \cdot 10^{-6} + 1.868 \cdot 10^{-6}\mathbf{i}$	262.1
31	30	$2.256 \cdot 10^{-6} + 9.339 \cdot 10^{-7}\mathbf{i}$	524.2	32	31	$1.504 \cdot 10^{-6} + 6.198 \cdot 10^{-7}\mathbf{i}$	786.8	33	32	$1.504 \cdot 10^{-6} + 6.198 \cdot 10^{-7}\mathbf{i}$	786.8
40	32	$3.76 \cdot 10^{-6} + 1.562 \cdot 10^{-6}\mathbf{i}$	314.4	34	33	$6.769 \cdot 10^{-6} + 2.81 \cdot 10^{-6}\mathbf{i}$	174.7	35	34	$5.264 \cdot 10^{-6} + 2.182 \cdot 10^{-6}\mathbf{i}$	224.6
44	34	$8.281 \cdot 10^{-6} + 3.438 \cdot 10^{-6}\mathbf{i}$	142.8	36	35	$1.504 \cdot 10^{-6} + 6.198 \cdot 10^{-7}\mathbf{i}$	786.8	48	35	$5.264 \cdot 10^{-6} + 2.182 \cdot 10^{-6}\mathbf{i}$	224.6
41	40	$8.281 \cdot 10^{-6} + 3.438 \cdot 10^{-6}\mathbf{i}$	142.8	42	41	$2.256 \cdot 10^{-6} + 9.339 \cdot 10^{-7}\mathbf{i}$	524.2	43	41	$3.76 \cdot 10^{-6} + 1.562 \cdot 10^{-6}\mathbf{i}$	314.4
45	44	$7.529 \cdot 10^{-6} + 3.124 \cdot 10^{-6}\mathbf{i}$	157	46	45	$7.529 \cdot 10^{-6} + 3.124 \cdot 10^{-6}\mathbf{i}$	157	47	46	$4.512 \cdot 10^{-6} + 1.868 \cdot 10^{-6}\mathbf{i}$	262.1
49	48	$1.504 \cdot 10^{-6} + 6.198 \cdot 10^{-7}\mathbf{i}$	786.8	52	48	$1.129 \cdot 10^{-5} + 4.686 \cdot 10^{-6}\mathbf{i}$	104.7	50	49	$3.008 \cdot 10^{-6} + 1.248 \cdot 10^{-6}\mathbf{i}$	393
56	49	$4.512 \cdot 10^{-6} + 1.868 \cdot 10^{-6}\mathbf{i}$	262.1	51	50	$3.76 \cdot 10^{-6} + 1.562 \cdot 10^{-6}\mathbf{i}$	314.4	53	52	$3.76 \cdot 10^{-6} + 1.562 \cdot 10^{-6}\mathbf{i}$	314.4
55	52	$4.512 \cdot 10^{-6} + 1.868 \cdot 10^{-6}\mathbf{i}$	262.1	54	53	$4.512 \cdot 10^{-6} + 1.868 \cdot 10^{-6}\mathbf{i}$	262.1	58	57	$6.769 \cdot 10^{-6} + 2.81 \cdot 10^{-6}\mathbf{i}$	174.7
59	58	$1.504 \cdot 10^{-6} + 6.198 \cdot 10^{-7}\mathbf{i}$	786.8	60	58	$4.512 \cdot 10^{-6} + 1.868 \cdot 10^{-6}\mathbf{i}$	262.1	61	60	$6.017 \cdot 10^{-6} + 2.496 \cdot 10^{-6}\mathbf{i}$	196.5
63	60	$1.504 \cdot 10^{-6} + 6.198 \cdot 10^{-7}\mathbf{i}$	786.8	62	61	$8.281 \cdot 10^{-6} + 3.43 \cdot 10^{-6}\mathbf{i}$	142.8	64	63	$6.017 \cdot 10^{-6} + 2.496 \cdot 10^{-6}\mathbf{i}$	196.5
65	64	$1.504 \cdot 10^{-6} + 6.198 \cdot 10^{-7}\mathbf{i}$	786.8	67	64	$3.76 \cdot 10^{-6} + 1.562 \cdot 10^{-6}\mathbf{i}$	314.4	66	65	$1.504 \cdot 10^{-6} + 6.198 \cdot 10^{-7}\mathbf{i}$	786.8
77	65	$7.521 \cdot 10^{-7} + 3.058 \cdot 10^{-7}\mathbf{i}$	1577	68	67	$7.521 \cdot 10^{-6} + 3.124 \cdot 10^{-6}\mathbf{i}$	157.2	72	67	$1.504 \cdot 10^{-6} + 6.198 \cdot 10^{-7}\mathbf{i}$	786.8
79	67	$4.512 \cdot 10^{-6} + 1.868 \cdot 10^{-6}\mathbf{i}$	262.1	69	68	$9.025 \cdot 10^{-6} + 3.744 \cdot 10^{-6}\mathbf{i}$	131	73	68	$9.785 \cdot 10^{-6} + 4.058 \cdot 10^{-6}\mathbf{i}$	120.8
70	69	$3.76 \cdot 10^{-6} + 1.562 \cdot 10^{-6}\mathbf{i}$	314.4	71	70	$4.512 \cdot 10^{-6} + 1.868 \cdot 10^{-6}\mathbf{i}$	262.1	76	70	$4.512 \cdot 10^{-6} + 1.868 \cdot 10^{-6}\mathbf{i}$	262.1
74	73	$2.256 \cdot 10^{-6} + 9.339 \cdot 10^{-7}\mathbf{i}$	524.2	75	73	$8.281 \cdot 10^{-6} + 3.438 \cdot 10^{-6}\mathbf{i}$	142.8	81	80	$3.008 \cdot 10^{-6} + 1.248 \cdot 10^{-6}\mathbf{i}$	393
82	81	$7.521 \cdot 10^{-7} + 3.058 \cdot 10^{-7}\mathbf{i}$	1577	83	81	$9.025 \cdot 10^{-6} + 3.744 \cdot 10^{-6}\mathbf{i}$	131	84	83	$8.281 \cdot 10^{-6} + 3.438 \cdot 10^{-6}\mathbf{i}$	142.8

Bibliography

[1] M. A. Abido. Optimal power flow using tabu search algorithm. *Electric Power Components and Systems*, 30(5):469–483, 2002.

[2] M.A. Abido. Optimal power flow using particle swarm optimization. *International Journal of Electrical Power and Energy Systems*, 24(7):563 – 571, 2002.

[3] V. Ajjarapu and C. Christy. The continuation power flow: a tool for steady state voltage stability analysis. In *[Proceedings] Conference Papers 1991 Power Industry Computer Application Conference*, pages 304–311, May 1991.

[4] H. Attouch and R.J.-B. Wets. Epigraphical analysis. *Annales de l'Institut Henri Poincare (C) Non Linear Analysis*, 6:73 – 100, 1989.

[5] Xiaoqing Bai, Hua Wei, Katsuki Fujisawa, and Yong Wang. Semidefinite programming for optimal power flow problems. *International Journal of Electrical Power and Energy Systems*, 30(6):383 – 392, 2008.

[6] T. L. Baldwin and S. A. Lewis. Distribution load flow methods for shipboard power systems. *IEEE Transactions on Industry Applications*, 40(5):1183–1190, Sept 2004.

[7] L. Bam and W. Jewell. Review: power system analysis software tools. In *IEEE Power Engineering Society General Meeting, 2005*, pages 139–144 Vol. 1, June 2005.

[8] M. Baran and F. F. Wu. Optimal sizing of capacitors placed on a radial distribution system. *IEEE Transactions on Power Delivery*, 4(1):735–743, Jan 1989.

[9] Amir Beck. *First-Order Methods in Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2017.

[10] D. Bienstock. Progress on solving power flow problems. *Optima*, 93:1–7, nov 2003.

[11] S. Bose, D. F. Gayme, K. M. Chandy, and S. H. Low. Quadratically constrained quadratic programs on acyclic graphs with application to power flow. *IEEE Transactions on Control of Network Systems*, 2(3):278–287, Sep 2015.

[12] S. Bose, D. F. Gayme, S. Low, and K. M. Chandy. Optimal power flow over tree networks. In *2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1342–1348, sep 2011.

[13] Richard H. Byrd, Jorge Nocedal, and Richard A. Waltz. *Knitro: An Integrated Package for Nonlinear Optimization*, pages 35–59. Springer US, Boston, MA, 2006.

[14] Florin Capitanescu and Louis Wehenkel. Experiments with the interior-point method for solving large scale optimal power flow problems. *Electric Power Systems Research*, 95:276 – 283, 2013.

[15] M. Chakravorty and D. Das. Voltage stability analysis of radial distribution networks. *International Journal of Electrical Power and Energy Systems*, 23(2):129 – 135, 2001.

[16] G. W. Chang, S. Y. Chu, and H. L. Wang. An improved backward/forward sweep load flow algorithm for radial distribution systems. *IEEE Transactions on Power Systems*, 22(2):882–884, May 2007.

[17] Changbo Chen and Marc Moreno Maza. Algorithms for computing triangular decomposition of polynomial systems. *Journal of Symbolic Computation*, 47(6):610 – 642, 2012. Advances in Mathematics Mechanization.

[18] Michel Coste. An introduction to semialgebraic geometry. 2002.

[19] D. Das, D.P. Kothari, and A. Kalam. Simple and efficient method for load flow solution of radial distribution networks. *International Journal of Electrical Power and Energy Systems*, 17(5):335 – 346, 1995.

[20] J. C. Das. *Load Flow Optimization and Optimal Power Flow*. CRC Press, oct 2017.

[21] Carl de Boor. *A Practical Guide to Splines*, volume 27 of *Applied Mathematical Sciences*. Springer-Verlag New York, 2001.

[22] U. Eminoglu and M. H. Hocaoglu. Distribution systems forward/backward sweep-based power flow algorithms: A review and comparison study. *Electric Power Components and Systems*, 37(1):91–110, 2008.

[23] H. Fang, G. Yang, and C. Pan. A parallel power flow algorithm for ladder-shaped shipboard power system. In *2014 Seventh International Symposium on Computational Intelligence and Design*, volume 1, pages 548–551, Dec 2014.

[24] M. Farivar, C. R. Clarke, S. H. Low, and K. M. Chandy. Inverter var control for distribution systems with renewables. In *2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 457–462, Oct 2011.

[25] M. Farivar and S. H. Low. Branch flow model: Relaxations and convexification - part i. *IEEE Transactions on Power Systems*, 28(3):2554–2564, Aug 2013.

[26] Eugene A. Feinberg, Pavlo O. Kasyanov, and Mark Voorneveld. Berges maximum theorem for noncompact image sets. *Journal of Mathematical Analysis and Applications*, 413(2):1040 – 1046, 2014.

[27] Stephen Frank and Steffen Rebennack. An introduction to optimal power flow: Theory, formulation, and examples. *IIE Transactions*, 48(12):1172–1197, 2016.

[28] Stephen Frank, Ingrida Steponavice, and Steffen Rebennack. Optimal power flow: a bibliographic survey i. *Energy Systems*, 3(3):221–258, 2012.

[29] L. Gan, N. Li, U. Topcu, and S. H. Low. Exact convex relaxation of optimal power flow in radial networks. *IEEE Transactions on Automatic Control*, 60(1):72–87, Jan 2015.

[30] L. Gan, N. Li, U. Topcu, and S. H. Low. Exact convex relaxation of optimal power flow in radial networks. *IEEE Transactions on Automatic Control*, 60(1):72–87, Jan 2015.

[31] B. Ghaddar, J. Marecek, and M. Mevissen. Optimal power flow as a polynomial optimization problem. *IEEE Transactions on Power Systems*, 31(1):539–546, Jan 2016.

[32] M. Giuntoli, P. Pelacchi, and D. Poli. On the use of simplified reactive power flow equations for purposes of fast reliability assessment. In *Eurocon 2013*, pages 992–997, July 2013.

[33] A. F. Glimn and G. W. Stagg. Automatic calculation of load flows. *Transactions of the American Institute of Electrical Engineers. Part III: Power Apparatus and Systems*, 76(3):817–825, April 1957.

[34] J. Duncan Glover, Mulukutla S. Sarma, and Thomas Overbye. *Power System Analysis and Design*. Cengage Learning, 5 edition, jan 2011.

[35] A. Gomez and L. G. Franquelo. An efficient ordering algorithm to improve sparse vector methods. *IEEE Transactions on Power Systems*, 3(4):1538–1544, Nov 1988.

[36] A. Gopalakrishnan, A. U. Raghunathan, D. Nikovski, and L. T. Biegler. Global optimization of optimal power flow using a branch amp; bound algorithm. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 609–616, Oct 2012.

[37] M. H. Haque. Efficient load flow method for distribution systems with radial or mesh configuration. *IEE Proceedings - Generation, Transmission and Distribution*, 143(1):33–38, Jan 1996.

[38] K. Iba, H. Suzuki, M. Egawa, and T. Watanabe. Calculation of critical loading condition with nose curve using homotopy continuation method. *IEEE Transactions on Power Systems*, 6(2):584–593, May 1991.

[39] J. Kardos, D. Kourounis, O. Schenk, and R. Zimmerman. Complete results for a numerical evaluation of interior point solvers for large-scale optimal power flow problems. *ArXiv e-prints*, July 2018.

[40] B. Kocuk, S. S. Dey, and X. A. Sun. Inexactness of sdp relaxation and valid inequalities for optimal power flow. *IEEE Transactions on Power Systems*, 31(1):642–651, jan 2016.

[41] L.L. Lai, J.T. Ma, R. Yokoyama, and M. Zhao. Improved genetic algorithms for optimal power flow under both normal and contingent operation states. *International Journal of Electrical Power and Energy Systems*, 19(5):287 – 292, 1997.

[42] J. Lavaei and S. H. Low. Zero duality gap in optimal power flow problem. *IEEE Transactions on Power Systems*, 27(1):92–107, feb 2012.

[43] K. Lehmann, A. Grastien, and P. Van Hentenryck. Ac-feasibility on tree networks is np-hard. *IEEE Transactions on Power Systems*, 31(1):798–801, Jan 2016.

[44] W. Li, X. Han, and B. Zhang. A comparison of power flow by different ordering schemes. In *2011 4th International Conference on Electric Utility Deregulation and Restructuring and Power Technologies (DRPT)*, pages 742–745, July 2011.

[45] S. H. Low. Convex relaxation of optimal power flow - part i: Formulations and equivalence. *IEEE Transactions on Control of Network Systems*, 1(1):15–27, mar 2014.

[46] S. H. Low. Convex relaxation of optimal power flow - part ii: Exactness. *IEEE Transactions on Control of Network Systems*, 1(2):177–189, jun 2014.

[47] S. Magnússon, P. C. Weeraddana, and C. Fischione. A distributed approach for the optimal power-flow problem based on admm and sequential convex approximations. *IEEE Transactions on Control of Network Systems*, 2(3):238–253, sept 2015.

[48] MathWorks. Cubic spline interpolation - matlab csapi. `http://www.mathworks.com/help/curvefit/csapi.html`. Accessed: 2017-04-10.

[49] D. Mehta, H. D. Nguyen, and K. Turitsyn. Numerical polynomial homotopy continuation method to locate all the power flow solutions. *IET Generation, Transmission Distribution*, 10(12):2972–2980, 2016.

[50] F. Milano. An open source power system analysis toolbox. *IEEE Transactions on Power Systems*, 20(3):1199–1206, Aug 2005.

[51] J. A. Momoh, R. Adapa, and M. E. El-Hawary. A review of selected optimal power flow literature to 1993. i. nonlinear and quadratic programming approaches. *IEEE Transactions on Power Systems*, 14(1):96–104, feb 1999.

[52] J. A. Momoh, M. E. El-Hawary, and R. Adapa. A review of selected optimal power flow literature to 1993. ii. newton, linear programming and interior point methods. *IEEE Transactions on Power Systems*, 14(1):105–111, feb 1999.

[53] A. J. Monticelli, A. Garcia, and O. R. Saavedra. Fast decoupled load flow: hypothesis, derivations, and testing. *IEEE Transactions on Power Systems*, 5(4):1425–1431, Nov 1990.

[54] James Raymond Munkres. *Topology*. Prentice Hall, Inc, 2 edition, 2015.

[55] Lynn Powell. *Power System Load Flow Analysis*. McGraw-Hill Education, 1 edition, dec 2004.

[56] IEEE Power and Energy Society. Distribution test feeders. `https://ewh.ieee.org/soc/pes/dsacom/testfeeders/`, 2000. Accessed: 2017-04-10.

[57] R. Tyrrell Rockafellar and Roger J. B. Wets. *Variational Analysis*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[58] Erik Sintorn and Ulf Assarsson. Fast parallel gpu-sorting using a hybrid algorithm. *Journal of Parallel and Distributed Computing*, 68(10):1381 – 1388, 2008. General-Purpose Processing using Graphics Processing Units.

[59] B. Stott and O. Alsac. Fast decoupled load flow. *IEEE Transactions on Power Apparatus and Systems*, PAS-93(3):859–869, May 1974.

[60] B. Stott, J. Jardim, and O. Alsac. Dc power flow revisited. *IEEE Transactions on Power Systems*, 24(3):1290–1300, Aug 2009.

[61] W. F. Tinney and C. E. Hart. Power flow solution by newton's method. *IEEE Transactions on Power Apparatus and Systems*, PAS-86(11):1449–1460, Nov 1967.

[62] W. F. Tinney and J. W. Walker. Direct solutions of sparse network equations by optimally ordered triangular factorization. *Proceedings of the IEEE*, 55(11):1801–1809, Nov 1967.

[63] A. Trias. The holomorphic embedding load flow method. In *2012 IEEE Power and Energy Society General Meeting*, pages 1–8, July 2012.

[64] Jeffrey K. Uhlmann. Satisfying general proximity / similarity queries with metric trees. *Information Processing Letters*, 40(4):175 – 179, 1991.

[65] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, Mar 2006.

[66] H. Wang, C. E. Murillo-Sanchez, R. D. Zimmerman, and R. J. Thomas. On computational issues of market-based optimal power flow. *IEEE Transactions on Power Systems*, 22(3):1185–1193, Aug 2007.

[67] W. Wei, J. Wang, N. Li, and S. Mei. Optimal power flow of radial networks and its variations: A sequential convex optimization approach. *IEEE Transactions on Smart Grid*, 8(6):2974–2987, Nov 2017.

[68] Z. Yang, H. Zhong, Q. Xia, and C. Kang. Solving opf using linear approximations: fundamental analysis and numerical demonstration. *IET Generation, Transmission Distribution*, 11(17):4115–4125, 2017.

[69] D. C. Yu and H. Wang. A new parallel lu decomposition method. *IEEE Transactions on Power Systems*, 5(1):303–310, Feb 1990.

[70] Xiao-Ping Zhang, Ping Ju, and E. Handschin. Continuation three-phase power flow: a tool for voltage stability analysis of unbalanced three-phase power systems. *IEEE Transactions on Power Systems*, 20(3):1320–1329, Aug 2005.

[71] Wu Zhongxi and Zhou Xiaoxin. Power system analysis software package (psasp)-an integrated power system analysis tool. In *POWERCON '98. 1998 International Conference on Power System Technology. Proceedings (Cat. No.98EX151)*, volume 1, pages 7–11 vol.1, Aug 1998.

[72] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas. Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Transactions on Power Systems*, 26(1):12–19, feb 2011.