A distributed approach to the optimal power flow problem and its privacy properties

Master's thesis

Sindri Magnússon

August 21, 2013

Abstract

In this thesis we address the optimal power flow (OPF) problem, where the goal is to find an optimal operating point of an electric network, which agrees to laws of physics and other physical limitations of the network. Traditionally the OPF problem has only been solved in the transmission network, which is responsible for transmitting the electricity from the power plants to cities. But with the introduction of the smart grid, the OPF problem has become relevant, not only in transmission networks, but also in distribution networks, which deliver electricity to end users.

Addressing the OPF problem in the distribution network is challenging in many aspects. Firstly, the general formulation is non-convex, and therefore designing algorithms, which are both efficient and optimal is usually challenging. However in practice, efficient algorithms are desirable, and therefore it is inevitable that one has to rely on heuristics to design solutions approaches for the general OPF problem. In this thesis we seek methods, based on convex optimization techniques to address the general OPF problem. During the solution approach, we capitalize on sequential approximations, in order to gracefully manipulate the non-convexity of the problem. Of course, the optimality of the algorithm is not guaranteed due to the non-convexity of the problem. Numerical results are provided to compare the proposed algorithm with other existing approaches.

Another challenge is the scalability of the related algorithms. The large size of the electrical network ruins the possibility of relying on centralized solution approaches, which are usually poor in scalability. Therefore, it is desirable to seek distributed methods, which have rich scalability properties. In this thesis we employ the state-of-the-art alternating direction method of multiplier (ADMM) method to enrich the proposed algorithms with scalability properties. The original non-convex OPF problem is broke into sub problems (one for every bus), which are coordinated via a thin protocol to find a good operating point. Numerical results are provided to show the applicability of our algorithm in large electrical network setups.

The proliferation of smart grid technologies, their processing information together with sophisticated statistical data mining and pattern recognition techniques have given rise to another threat, privacy breaches associated with the households life styles. Therefore, it is worth of investigating algorithms,

which can suppress such threats. Even though, we do not model mechanisms for privacy guarantees as an explicit design criterion of our solution approach, we discuss the potentials of our proposed algorithm, which are inherently accomplished, for preserving the privacy of household power demands.

Contents

Chapter 1

Introduction

1.1 The electrical grid and power flow

In todays connected society the consumer expects to have access to electricity when he wants and where he wants. The consumers range from small households to big international corporations. Unlike many other widely used energy resources electricity cannot be stored efficiently except in small doses. This means that to meet the demand of both the industry and other consumers the suppliers have to be able to generate the amount of electricity needed at each time simultaneously. The benefits of electricity over other energy resources are on the other hand that it can easily be transmitted over long distances. The technology that makes all this possible is the electrical network. The electrical network is an interconnected network that transfers the electricity from the power plants and other power generation stations to the consumers. So whenever a light is turned on, the electrical network is responsible for transferring the electricity from the power generation point to the light bowl. The components of the electrical network are buses and flow lines. The buses are simply the actors of the electric market so the households and industrial customers as well as the producers of the electricity. The flow lines are responsible for bringing the electricity between buses. Most electric networks are sparse. That means that each bus of the network is only connected to few others by flow line.

The electrical network is made up of two layers, namely the transmission network and the distribution network. The two layers are connected by electric substations. Most of the power needed is generated by big power plants.

These big power plants are often located far away from the demand centers and even in other countries. The purpose of the transmission network is to transfer the electricity from the big power plants to the electric substations that are located near demand centers. The electricity in the transmission network is transmitted at high voltage to reduce the energy loss in long distance transmission. The electric substations then reduce the magnitude of the voltages of the electricity and then the distribution network takes over. The distribution network consists of small distributed power generators as well as households and industrial consumers. The main responsibility of the distribution is to deliver the electricity to the end consumer.

The power companies have to be able to meet the maximum power demand at all times. At the same time they have to consider that generating power is costly, often has damaging effects on the environment and is not renewable in many cases. Therefore deciding on how much power to generate is not a trivial task. To make matters even worse information about individual consumption is very limited in the traditional electrical grid. The information gathered has been limited to sum of use over long periods for billing purposes. This means that the power companies have to rely solely on limited statistics when deciding the amount of power to inject into the network. This has been changing in recent years and is expected to change even more in the future. There has been a lot of growth of computational ability distributed around the network as well as high frequency information gathering about power consumption of the customers. This has happened with the arrival of the smart meter. The smart meter is a small computer device located at the customer that collects information about the power behavior of the customer. The smart meter gives the customer real time update on how he uses the electricity, how much it costs and tells him where he can save money. The smart meter also helps the power companies to use their power resources in a more efficient manner. For example it enables the power companies to use the price of electricity to reflect the status of the network. So they can lower the prices when few people are consuming power like during the night time, or if a lot of renewable resources are injecting power into the system for example when its windy or sunny. On the other hand prices can be raised if the power demand is so heavy that the power companies can hardly keep up the consumption. Customers can then adjust their consumption to the electrical network and are therefore persuaded to become more environmental friendly. This new generation of the distribution network goes by the name smart grid.

Power cannot be stored economically. Therefore the power needed at a specific time has to be created simultaneously. This means that the power companies have to generate the power and transfer it while the customer is demanding it. So to make sure that the need of every consumer in the network is met the power companies firstly have to make sure that the power plants are generating enough power and secondly have to adjust the flow through the network so the power reaches its destination. The control parameter that is used to control the flow through the network is called voltage. Each bus has a local voltage and therefore the voltages have to be controlled throughout the network. The flow of electricity through the network is not free because during the transmission some energy is lost. So the two parameters that have to be set so that the power demand of each bus in the network is met at each time are the voltage at each bus and power generated at each generator. There could be multiple feasible ways to choose the voltages and power generated to meet the consumers demand. Therefore it is interesting to find a feasible control in as efficient manner as possible. This is indeed a classic and well studied problem and it is called the optimal power flow (OPF) problem. The objectives differ but the most common ones are minimizing the cost of generating the power needed at each time and minimizing the losses in the network. The problem was first posed in the sixties and there have been multiple publications on the problem since then. The problem possesses two main difficulties. Firstly the set of feasible points is a non-convex set and therefore the problem is non-convex. Secondly most practical networks are of large scale and there rises the need for scalable algorithms. These two challenges make the problem an interesting research topic.

Traditionally the OPF problem has only been solved for the transmission network. This was mostly due to limited information about consumption in the distribution network. But with the introduction of the smart grid and high frequency data collection about consumers the OPF problem in the distribution network has become an interesting research topic. The OPF problem in the distribution network possesses two main difficulties. Firstly many solutions to the OPF problem in the transmission network have been based on some approximations that make the problem easier to solve. It turns out that these approximations perform poorly when applied to the distribution network, and therefore impose the need to solve the general non-convex OPF problem. Secondly solving the OPF problem in the distribution network raises some serious questions on privacy. Privacy issues of the smart grid are discussed in next section.

1.2 Privacy in the electrical grid

The discussion about the smart grid has been very focused on many possibilities of the new technology. Meanwhile, the discussion on how these possibilities can be abused has often been trailing behind. In this section we place a stronger emphasis on privacy issues that come up in the smart grid. By privacy issues we mean potential exposure of private data that was not initially intended to be unveiled without the consent or even the knowledge of the users. For further reading on privacy issues in the smart grid we refer the reader to [16, 11].

The power companies like to get individual data about power consumption of as many customers as they can, because this enables them to control the network in a more efficient way. Therefore the power companies try to persuade their customers to invest in the smart meter technology with appealing arguments. The smart meter gives the customer greater vision over his electrical use and billing information than ever before. The prices are variable and depend on the state of the system. This gives the conscious customers with good smart meters the option of saving money. Moreover, it is not just the power companies that benefit from a more efficient and renewable control of the power network. Everyone gains from less pollution and fewer blackouts. However there is a catch. The smart meter gathers high frequency information of the households about its power use. High frequency could be as often as every minute. The data is used both for pricing purposes and for helping the power companies with decision making. Even though the information is limited to the total use of power at each time, the power consumption of each appliance often leaves a unique signature. As a result, by using statistical data mining and other pattern recognition tools, it is not difficult to determine what appliances the consumer is using at a given time. This means that the data that the smart meter is collecting could be used to map out the behavior of the customer. For example when he is out of the house, sleeping, opening the fridge, watching tv, cooking, when the kids are home alone. There are few places as sacred to an individual as his home. Therefore jeopardizing the privacy of the home is a very sensitive issue. Different parties could use this information for their benefits in multiple ways. Government agencies could use it to look for criminal activity, insurance companies could use it to decide if it is profitable to sell you insurance, marketers could use it to personalize their campaigns, and the list goes on. This is a critical problem, usually known as non intrusive load

monitoring [15]. Nevertheless, the future of the smart grid demands that everyone invests in a smart meter. Certainly, customers will be suspicious about investing in a smart meter if it jeopardizes the privacy of their home. Therefore these privacy issues do not only threaten individual freedom but also the future development of the smart grid.

1.3 Literature review

The optimal power flow problem was first introduce by Carpentier in 1962 [7]. Since then multiple papers have been published and algorithm have been proposed for the problem. In this section we plan to review the most relevant papers on the OPF problem for the work of this thesis.

One of the most common simplification of the OPF problem is the DC approximation [6]. It imposes multiple assumptions on the problem, for example that the electrical network has no energy loss. The yielding optimization problem is convex with linear feasible set and convex objective function. The assumption of no energy losses is reasonable in the transmission network where the voltage magnitude is high to reduce losses in the long distance transmission. But in distribution network where voltage magnitude is low and the losses are much higher it becomes harder to defend the use of the DC approximation. There arises the need to look at the general non-convex problem when dealing with the OPF problem in the smart grid. Many approaches have been suggest for the non-convex OPF problem, see survey of methods in [14]. The drawback of most of the methods are that either they can only guarantee local optimum or are based on heuristic for which optimality cannot be proved.

In [12] the OPF problem is formulated as a semidefinite programing problem with additional rank constraint. But semidefinite programing problems are convex problems where a linear objective function is optimized over a conic set. As a result, if the rank constraint is relaxed the yielding optimization problem is convex. In [12] it is shown that the semidefinite relaxation is equivalent to the dual problem of the OPF problem. The results in [12] go even further and give sufficient conditions on the problem for the duality gap to be zero. Then it demonstrated how a solution to the OPF problem can be extracted from the solution to the relaxation in that case. Then [12] notes that the sufficient conditions are satisfied for the IEEE benchmark systems which are standard test network for the OPF problem. Therefore a global

solution for IEEE benchmark systems can obtained by solving a convex problem. But [12] only provides sufficient conditions and therefore nothing can be said about networks that do not meet the conditions. A simple and practical network which has a non zeros duality gap is presented in [3]. So the result from [12] cannot be generalized to all practical networks. In any case the semidefinite relaxation gives a lower bound on the optimal solution.

1.4 Outline of this thesis

The outline of the thesis is as follows. In chapter 2 we introduce the optimization theory that the contribution of this thesis is the based on. We start by discussing distributed optimization. But distributed optimization is when multiple parties collaborate towards optimizing a common objective function without any central control. Thereafter, we present privacy preserving optimization. But privacy preserving optimization is when multiple parties have a common interest in collaborating towards solving a global optimization problem without revealing private data that is necessary for the problem. In chapter 3 we formally present the OPF problem. We present both the standard centralized formulation and also present a distributed formulation of the problem. Based on the distributed formulation we build a distributed algorithm which is presented in chapter 4. It is noted in section 1.2 that one of the key challenges in the smart grid was privacy. Therefore we inspect the privacy properties of our proposed method with respect to the power demand of each household in chapter 5. In chapter 6 we test the algorithm on standard test networks for the problem. Then we conclude the thesis in chapter 7.

1.5 Notation

The following notation is used throughout this work:

- Boldface lower case and upper case letters represent vectors and matrices respectively.
- We denote the set of real and complex numbers, respectively, by $\mathbb{R}$ and $\mathbb{C}$ and the set of n times m real and complex matrices, respectively, by $\mathbb{R}^{n\times m}$ and $\mathbb{C}^{n\times m}$. We use calligraphy letters to represent all other sets.

- We use parentheses to construct column vectors from comma separated lists, e.g., $(\mathbf{a}, \mathbf{b}, \mathbf{c}) = [\mathbf{a^T}\ \mathbf{b^T}\ \mathbf{c^T}]\mathbf{^T}$.
- $j = \sqrt{-1}$
- $\mathbf{A} \bullet \mathbf{B}$, is the Hadamard product of the two matrices $\mathbf{A}$ and $\mathbf{B}$.
- z^* is the complex conjugate of z.
- $(\mathbf{A})_{n,k}$ is the n-th k-th component of the matrix $\mathbf{A}$.
- $(\mathbf{z})_n$ is the n-th component of the vector $\mathbf{z}$.

Chapter 2

Optimization theory

In this chapter we introduce the optimization theory and techniques that the contribution of this thesis is based on. We assume that the reader has a basic knowledge of convex optimization [5]. The outline of this chapter is as follows. In section 2.1 we introduce distributed optimization. Distributed optimization is where multiple parties collaborate to optimize a global objective function. One problem encountered when multiple parties want to find an optimal solution of a global objective funtion is that they have to share private information, and hence privacy can be breached. In section 2.2 we address the problem of privacy for the multi party optimization problem.

2.1 Distributed optimization

Many optimization applications involve number of parties that share a common interest and therefore want to collaborate towards that common goal. The classical way to solve such a problem is for each party involved to share their problem data to some central station that solves the problem. Distributed optimization is when the parties involved work together without any central control to solve the problem. This implies that each subsystem involved solves a sub problem and then the subsystems coordinate via message passing. Often this message passing is required only among the neighbors of the network. There are multiple reasons why distributed optimization methods might be preferable. For example if the data sets are huge and geographically distributed around the globe then it might be very costly to gather the data at one place. Another reason is scalability. The

problem size tends to grow as the distributed systems get larger but the size of the individual sub problems often stays the same. Yet another reason is privacy. The entities of a distributed system might not be willing to share some sensitive data that is needed to solve the optimization problem. We look more thoroughly into how distributed methods can be used to support privacy issues in section 2.2.

This section is organized as follows. In subsection 2.1.1 we introduce a class of optimization methods called decomposition methods. Moreover, we give some examples of simple decomposition algorithms. In subsection 2.1.2 we present a general frame work used for formulating distributed optimization problems. We will then generalize some of the decomposition algorithms from sub section 2.1.1 to our general framework. Finally, subsection 2.1.3 concisely presents the state of the art alternating direction method of multiplier (ADMM).

2.1.1 Introduction to decomposition methods

In this subsection we introduce and give examples of decomposition methods. More detailed information on the subject can be found in [21]. Decomposition methods consist of all optimization techniques that capitalize on the problem structure to solve an optimization problem by splitting it into smaller problems that can be handled independently. However, a light protocol is required to coordinate the sub problems. The advantage of decomposition methods becomes noticeable for example, in the case of very large problems, where the centralized algorithms can explode and in the case of problems, where the problem data are geographically distributed so that collecting them at a central location is impossible. But since complexity of most optimization algorithms grows more than linearly with problem size it might still be beneficial to use decomposition methods even if the sub problems are solved sequentially. Let us look at an example of a very simple decomposition method (borrowed from [21]):

Example 2.1.1. Consider the following optimization problem:

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}_1, \mathbf{x}_2) = f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) \\ &\text{subject to} && \mathbf{x}_1 \in \mathcal{C}_1, \mathbf{x}_2, \in \mathcal{C}_2. \end{aligned} \tag{2.1}$$

where the variables are $\mathbf{x}_1 \in \mathbb{R}^{n_1}$ and $\mathbf{x}_2 \in \mathbb{R}^{n_2}$ and $\mathcal{C}_1 \subseteq \mathbb{R}^{n_1}$ and $\mathcal{C}_2 \subseteq \mathbb{R}^{n_2}$. Here f_1 is a function of $\mathbf{x}_1$ and f_2 is a function of $\mathbf{x}_2$ and there is no coupling

between $\mathbf{x}_1$ and $\mathbf{x}_2$. So we can decompose this problem into the following two sub problems:

$$\begin{aligned} &\text{minimize} && f_1(\mathbf{x}_1) \\ &\text{subject to} && \mathbf{x}_1 \in \mathcal{C}_1 \end{aligned} \tag{2.2}$$

where the variables are $\mathbf{x}_1 \in \mathbb{R}^{n_1}$.

$$\begin{aligned} &\text{minimize} && f_2(\mathbf{x}_2) \\ &\text{subject to} && \mathbf{x}_2 \in \mathcal{C}_2 \end{aligned} \tag{2.3}$$

where the variables are $\mathbf{x}_2 \in \mathbb{R}^{n_2}$.

If we let $\mathbf{x}_1^\star$ and $\mathbf{x_2}^\star$ denote the optimal solutions of sub problems (2.2) and (2.3) respectively then $(\mathbf{x}_1^\star, \mathbf{x_2}^\star)$ is an optimal solution to problem (2.1).

In the example 2.1.1 above there is no coupling between the two sub problems. Therefore the problem is solved by just solving each of the sub problems once. We say that a problem is separable if it there is no coupling between the sub problems. Most optimization problems however do not behave so nicely and usually there is some coupling between the sub problems. Before we present more complicated examples of decomposition methods it is worthwhile to introduce some terminology that is commonly used when decomposition methods are discussed:

- Master problem: Is an optimization problem that is equivalent to the original problem but has a structure that can easily be decomposed into sub problems. The variables of the master problem are called master variables.

- Master algorithm: Is the algorithm that is used to solve the master problem, i.e., to find the optimal master variables. It is an iterative process where each iteration consists the following two steps. First, all the sub problems are solved. Then the solutions of all of the sub problems are coordinated, in order to update the master variables.

The master algorithm can either be carried out centrally, where one central unit gathers the solutions of the sub problems and coordinates, or distributively where the sub problems coordinate amongst themselves through message passing. To give the reader a better idea of decomposition methods and associated algorithms, we present two examples.

Primal decomposition

Let us consider the following optimization problem (borrowed from [21]):

$$\underset{\mathbf{x}_1,\mathbf{x}_2,\mathbf{y}}{\text{minimize}} \quad f(\mathbf{x}_1,\mathbf{x}_2,\mathbf{y}) = f_1(\mathbf{x}_1,\mathbf{y}) + f_2(\mathbf{x}_2,\mathbf{y}) \tag{2.4}$$

where $\mathbf{x}_1 \in \mathbb{R}^{n_1}, \mathbf{x}_2, \in \mathbb{R}^{n_2}, \mathbf{y} \in \mathbb{R}^n$ for $n_1, n_2, n \in \mathbb{N}$ and f_1, f_2, f are convex functions. For a fixed $\mathbf{y}$ problem (2.4) becomes equivalent to solving the following two separable sub problems:

Subproblem 1:

$$\text{minimize} \quad f_1(\mathbf{x}_1,\mathbf{y}) \tag{2.5}$$

where the variables are $\mathbf{x}_1 \in \mathbb{R}^{n_1}$.

Subproblem 2:

$$\text{minimize} \quad f_2(\mathbf{x}_2,\mathbf{y}) \tag{2.6}$$

where the variables are $\mathbf{x}_1 \in \mathbb{R}^{n_2}$.

For a fixed $\mathbf{y}$, we let $\phi_k(\mathbf{y})$ and $\mathbf{x}_k^\star(\mathbf{y})$ be the optimal value and optimal solution of sub problem k, respectively, where $k = 1, 2$. Note that $\phi_k(\mathbf{y})$ inherits the convexity from f_k since for $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{n_k}$ and $\alpha, \beta \in [0, 1]$ such that $\alpha + \beta = 1$, k we have that

$$\begin{aligned}
\phi_k(\alpha\mathbf{a} + \beta\mathbf{b}) &\leq f_k(\alpha\hat{\mathbf{x}}_k(\mathbf{a}) + \beta\hat{\mathbf{x}}_k(\mathbf{b}), \alpha\mathbf{a} + \beta\mathbf{b}) && (2.7)\\
&\leq \alpha f_k(\hat{\mathbf{x}}_k(\mathbf{a}), \mathbf{a}) + \beta f_k(\beta\hat{\mathbf{x}}_k(\mathbf{b}), \beta\mathbf{b}) && (2.8)\\
&= \alpha\phi(\mathbf{a}) + \beta\phi(\mathbf{b}). && (2.9)
\end{aligned}$$

Then we can can present the master problem that is equivalent to (2.4)

$$\text{minimize} \quad \phi(\mathbf{y}) = \phi_1(\mathbf{y}) + \phi_2(\mathbf{y}). \tag{2.10}$$

where the variables are $\mathbf{y} \in \mathbb{R}^n$. The functions ϕ_1 and ϕ_2 are convex and therefore problem (2.10) is also convex. Problem (2.10) is however not necessarily differentiable. A well studied and simple method to deal with non differentiable convex function is the subgradient method [20]. The subgradient method resembles the ordinary gradient method for differentiable functions with the following exception [20]. Instead of using gradients as in the ordinary gradient method the subgradient method uses subgradients, therefore the subgradient method can handle non-differentiable convex objective function. It is worth noting that the subgradient method is not a descent method.

We refer the reader to [22] and [20] for more details about subgradients and the subgradient method. For a convex function h and some point $\mathbf{w}$ in the domain of h we let $\partial h(\mathbf{w})$ be the set of all subgradients of the h at $\mathbf{w}$. To formally describe the master algorithm based on the subgradient method, we denote by $\mathbf{y}^{(n)}$ the $\mathbf{y}$ variable after n iterations of the algorithm. Then the $(n+1)$th iteration of the algorithm consists of the following 2 steps:

- Construct a subgradient $\mathbf{g}^{(n)}$ of ϕ such that $\mathbf{g}^{(n)} \in \partial\phi(\mathbf{y})$.
- Update $\mathbf{y}$: $\mathbf{y}^{(n+1)} = \mathbf{y}^{(n)} - \alpha^{(n)}\mathbf{g}^{(n)}$. Here $\alpha^{(n)}$ is called the step size which can be decided in multiple ways.

The master algorithm can be summarized as follows:

Algorithm 1: Primal decomposition.

Given an inital $\mathbf{y}^{(0)}$
Set $n = 0$
repeat
  Solve subproblem 1 to construct $\mathbf{g}_1^{(n)} \in \partial\phi_1(\mathbf{y}^{(n)})$.
  Solve subproblem 2 to construct $\mathbf{g}_2^{(n)} \in \partial\phi_2(\mathbf{y}^{(n)})$.
  Update $\mathbf{y}$: $\mathbf{y}^{(n+1)} = \mathbf{y}^{(n)} - \alpha^{(n)}\mathbf{g}^{(n)}$, where $\mathbf{g}^{(n)} = (\mathbf{g}_1^{(n)} + \mathbf{g}_2^{(n)})$.
  $n = n + 1$

Dual decomposition

Here we address problem (2.4) again. However, we formulate it a little differently as follows:

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}_1) = f_1(\mathbf{x}_1, \mathbf{y}_1) + f_2(\mathbf{x}_2, \mathbf{y}_2) \\ &\text{subject to} && \mathbf{y}_1 = \mathbf{y}_2 \end{aligned} \tag{2.11}$$

where the variables are $\mathbf{x} \in \mathbb{R}^{n_1}, \mathbf{x}_2 \in \mathbb{R}^{n_2}$ and $\mathbf{y}_1, \mathbf{y}_2 \in \mathbb{R}^n$. The only difference between (2.11) and (2.4) is that in (2.11) we have two copies of the variable $\mathbf{y}$ and force them to agree with a consistency constraint. The idea behind the dual decomposition is to solve the dual problem instead of the primal problem. Therefore we need some additional assumptions. Namely that the objective function is strictly convex and the duality gap is zero. The Lagrangian with respect to the consistency constraint is

$$L(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}_1, \mathbf{y}_2, \boldsymbol{\lambda}) = f_1(\mathbf{x}_1, \mathbf{y}_1) + f_2(\mathbf{x}_2, \mathbf{y}_2) + \boldsymbol{\lambda}^{\mathbf{T}}\mathbf{y}_1 - \boldsymbol{\lambda}^{\mathbf{T}}\mathbf{y}_2. \tag{2.12}$$

Notice that the Lagrangian is separable, and therefore we can minimize the Lagrangian separately. In particular, for fixed $\boldsymbol{\lambda}$, we solve

Subproblem 1:

$$\inf_{\mathbf{x}_1,\mathbf{y}_1} \quad f_1(\mathbf{x}_1,\mathbf{y}_1) + \boldsymbol{\lambda}^{\mathbf{T}}\mathbf{y}_1 \tag{2.13}$$

where the variables are $\mathbf{x}_1 \in \mathbb{R}^{n_1}$ and $\mathbf{y}_1 \in \mathbb{R}^n$.

Subproblem 2:

$$\inf_{\mathbf{x}_2,\mathbf{y}_2} \quad f_2(\mathbf{x}_2,\mathbf{y}_2) - \boldsymbol{\lambda}^{\mathbf{T}}\mathbf{y}_2 \tag{2.14}$$

where the variables are $\mathbf{x}_2 \in \mathbb{R}^{n_2}$ and $\mathbf{y}_2 \in \mathbb{R}^n$.

where the optimal value is given by let $g_1(\boldsymbol{\lambda})$ and $g_2(\boldsymbol{\lambda})$ respectively. We let $(\mathbf{x}_1^\star(\boldsymbol{\lambda}), \mathbf{y}_1^\star(\boldsymbol{\lambda}))$ and $(\mathbf{x}_2^\star(\boldsymbol{\lambda}), \mathbf{y}_2^\star(\boldsymbol{\lambda}))$ denote the solution of (2.13) and (2.14). Then the dual function of problem (2.11) can then be expressed as $g(\boldsymbol{\lambda}) = g_1(\boldsymbol{\lambda}) + g_2(\boldsymbol{\lambda})$ and the dual problem becomes:

$$\max g_1(\boldsymbol{\lambda}) + g_2(\boldsymbol{\lambda}) \tag{2.15}$$

where the variables are $\boldsymbol{\lambda} \in \mathbb{R}^n$. Or equivalently

$$\min -g_1(\boldsymbol{\lambda}) - g_2(\boldsymbol{\lambda}) \tag{2.16}$$

where the variables are $\boldsymbol{\lambda} \in \mathbb{R}^n$. It is easy to verify that $-\mathbf{y}_1^\star(\boldsymbol{\lambda})$ and $\mathbf{y}_2^\star(\boldsymbol{\lambda})$ are a subgradients of $-g_1$ and $-g_2$ at $\boldsymbol{\lambda}$. Therefore $\mathbf{y}_2^\star(\boldsymbol{\lambda}) - \mathbf{y}_1^\star(\boldsymbol{\lambda})$ is a subgradient of $-g$ at $\boldsymbol{\lambda}$. Now we can use the subgradient method as before. We denote the Lagrangian multiplier after n iterations by $\boldsymbol{\lambda}^{(n)}$ and let $\alpha^{(n)}$ be the step size at iteration n. The dual decomposition algorithm is as follows:

Algorithm 2: Dual decomposition.

Given an inital $\boldsymbol{\lambda}^{(0)}$
$n = 0$
repeat
  Solve subproblem 1 to construct $-\mathbf{y}_1^\star(\boldsymbol{\lambda}^{(n)}) \in \partial\left(-g_1(\boldsymbol{\lambda}^{(n)})\right)$.
  Solve subproblem 2 to construct $\mathbf{y}_2^\star(\boldsymbol{\lambda}^{(n)}) \in \partial\left(-g_2(\boldsymbol{\lambda}^{(n)})\right)$.
  Update $\mathbf{y}$: $\boldsymbol{\lambda}^{(n+1)} = \boldsymbol{\lambda}^{(n)} - \alpha^{(n)}(\mathbf{y}_2^\star(\boldsymbol{\lambda}^{(n)}) - \mathbf{y}_1^\star(\boldsymbol{\lambda}^{(n)}))$.
  $n = n + 1$

2.1.2 General decomposition structures for distributed problems.

In section 2.1.1 we introduced simple examples of decomposition methods. But for large distributed systems the decomposition structures can become quiet complex. For this reason it is of interest to have a general framework and notation to formulate a general distributed optimization problem. In this section we will present a general framework for that purpose. The framework we present here is based on [21].

Suppose we have K subsystems or entities. Let $\mathcal{K}$ denote the set of subsystems, so $\mathcal{K} = \{1, 2, \ldots, K\}$. We consider the case, where the K subsystems want to cooperate towards a common goal f that is distributed among the subsystems i.e.

$$f = \sum_{k \in \mathcal{K}} f_k, \tag{2.17}$$

where f_k is owned by subsystem k. Each subsystem $k \in \mathcal{K}$ has the following components:

- $\mathbf{x}_k \in \mathbb{R}^{n_k}$, a vector of private variables which are variables that only sub-system k uses and is aware of.
- $\mathbf{y}_k \in \mathbb{R}^{p_k}$, a vector of public variables which are variables shared or coupled with at least one other subsystem. In other words, each component of $\mathbf{y}_k$ is a local copy of a coupling variable that subsystem k keeps.
- $\mathcal{C}_k \subseteq \mathbb{R}^{n_k} \times \mathbb{R}^{p_k}$, local constraint set.
- $f_k : \mathcal{C}_k \to \mathbb{R}$, private objective function.

The global objective function can then be written as $f = \sum_{k=1}^{K} f_k(\mathbf{x}_k, \mathbf{y}_k)$.

Note that the subsystems cannot modify their public variables as they wish. This is because of coupling. That is each component of $\mathbf{y}_k$ is only a local copy pertaining to a coupling variable that subsystem k maintains. Therefore, components of $\mathbf{y}_k$ have to be consistent with the local copies of other subsystems pertaining to the same variable. We say that two subsystems are coupled if the subsystems maintain public variables, where at least one of their components is associated with the same coupling variable. Because all the local copies of the same coupling variable are equal they form an

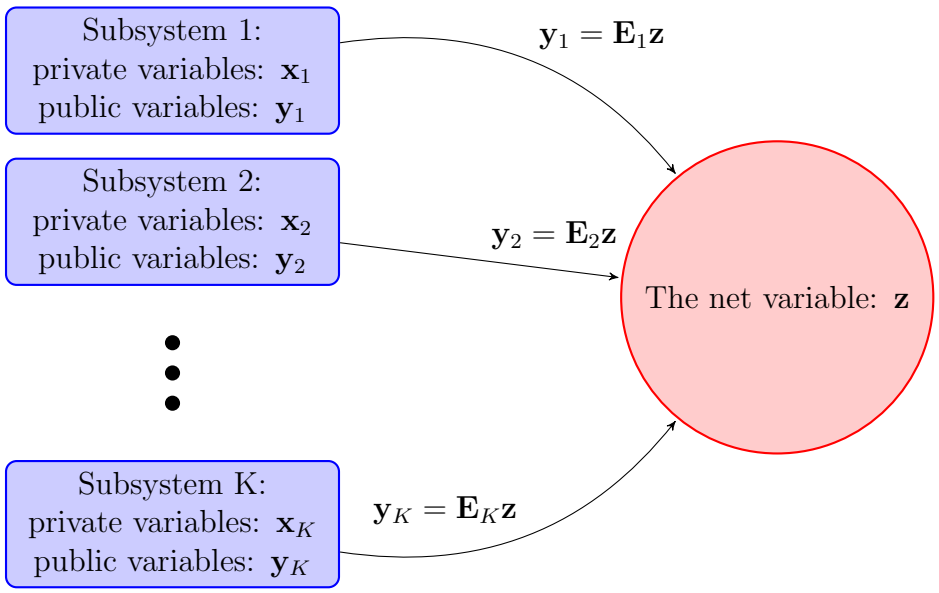


Figure 2.1: $\mathbf{x}_k$ is the local variable and $\mathbf{y}_k$ is the public variable of subsystem k, $k = 1, ..K$. Coupling is represented by the net variable $\mathbf{z}$, together with the consistency constraints $\mathbf{y}_k = \mathbf{E}_k\mathbf{z}$, $k = 1, \cdots, K$, where $\mathbf{E}_k$ projects net variables to the corresponding public variables.

equivalence class. Graphically these equivalence classes can be seen as an hyper edges or a net in a hypergraph[1] that connects between all the local copies of different subsystem. We will here after adapt a standard terminology and refer to these equivalence classes as nets and we assume that we have N nets. Let $\mathbf{z} \in \mathbb{R}^N$ be the coupling variable associated with the nets (equivalence classes), where component n of $\mathbf{z}$ is the coupling variable assigned to net n. We will refer to $\mathbf{z}$ as the net variables. To ensure that the local copies of the same net variable are equal, we need to define consistency constraints. In particular we have

$$\mathbf{y}_k = \mathbf{E}_k\mathbf{z}, \qquad k = 1, 2, \cdots, K, \tag{2.18}$$

where

$$(\mathbf{E}_k)_{n,m} = \begin{cases} 1 & \text{if } (\mathbf{y}_k)_n \text{ is in net } m; \\ 0 & \text{otherwise.} \end{cases} \tag{2.19}$$

[1]Hypergraph is a generalization of a graph where each edge can connect between an arbitrary number of nodes.

Note that the matrix $\mathbf{E}_k$ has some interesting properties:

- The operator $\mathbf{E}_k^{\mathbf{T}}$ embeds $\mathbf{w} \in \mathbb{R}^{p_k}$ into the space of net variables, $\mathbb{R}^N$, so that $(\mathbf{E}_k^{\mathbf{T}}\mathbf{w})_n$ is 0 if subsystem k has no public variable in net n but otherwise if subsystem k has a public variable $(\mathbf{y}_k)_n$ in net m then $(\mathbf{E}_k^{\mathbf{T}}\mathbf{w})_n = (\mathbf{w})_m$. More formally we have that:

$$(\mathbf{E}_k^{\mathrm{T}}\mathbf{w})_n = \begin{cases} (\mathbf{w})_m & \text{the public variable } (\mathbf{y}_k)_m \text{ is in net } m; \\ 0 & \text{otherwise.} \end{cases} \tag{2.20}$$

- $\mathbf{E}_k^{\mathbf{T}}\mathbf{E}_k \in \mathbb{R}^{N\times N}$ is a diagonal where diagonal element n is 1 if subsystem k is part of net n.

The global problem can then be formulated as

$$\begin{aligned} \text{minimize} \quad & \sum_{k=1}^{K} f_k(\mathbf{x}_k, \mathbf{y}_k) \\ \text{subject to} \quad & (\mathbf{x}_k, \mathbf{y}_k) \in \mathcal{C}_k, \qquad k = 1, \ldots, K \\ & \mathbf{y}_k = \mathbf{E}_k \mathbf{z}, \qquad k = 1, \ldots, K \\ & \mathbf{z} \in \mathcal{Z}, \end{aligned} \tag{2.21}$$

where the variables are $\mathbf{z}, \mathbf{x}_k$, and $\mathbf{y}_k$ for $k = 1, 2, \ldots, K$ and $\mathcal{Z}$ is a set known by each subsystem.

It is often convenient to have all the private and public variables, respectively, in one vector. For this purpose we stack all the private variables and all public variables in one vector denoted by $\mathbf{x} \in \mathbb{R}^{\sum_{k\in\mathcal{K}} n_k}$ and $\mathbf{y} \in \mathbb{R}^{\sum_{k\in\mathcal{K}} p_k}$. That is

$$\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_K) \tag{2.22}$$

$$\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_K). \tag{2.23}$$

We do the same for the $\mathbf{E}_k$ matrices i.e. define the matrix $\mathbf{E} \in \mathbb{R}^{\sum_{k\in\mathcal{K}} p_k \times N}$ so that

$$\mathbf{E} = \left[\mathbf{E}_1^{\mathrm{T}}, \mathbf{E}_2^{\mathrm{T}}, \cdots, \mathbf{E}_K^{\mathrm{T}}\right]^{\mathrm{T}}. \tag{2.24}$$

Then we can write the consistency constraints as

$$\mathbf{y} = \mathbf{E}\mathbf{z}. \tag{2.25}$$

Primal decomposition

We will now revisit the primal decomposition method from section 2.1.1 but this time for a problem on the form (2.21). We assume that the problem is convex. If we fix the net variables (and thereby also the public variables $\mathbf{y}_k$) in (2.21) we get a problem on the following form:

$$\begin{aligned} \text{minimize} \quad & \sum_{k=1}^{K} f_k(\mathbf{x}_k, \mathbf{y}_k) \\ \text{subject to} \quad & (\mathbf{x}_k, \mathbf{y}_k) \in \mathcal{C}_k, \quad k \in \mathcal{K}, \end{aligned} \tag{2.26}$$

where the variable is $\mathbf{x}_k \in \mathbb{R}^{n_k}$ for $k \in \mathcal{K}$. Note that the variables of this problem are split so that each variable only belongs to one subsystem. The problem is therefore separable, where the optimal solution is achieved by letting each subsystem k find , for fixed $\mathbf{y}_k = \mathbf{E}_k\mathbf{z}$, the solution for sub problem

$$\begin{aligned} \text{minimize} \quad & f_k(\mathbf{x}_k, \mathbf{y}_k) \\ \text{subject to} \quad & (\mathbf{x}_k, \mathbf{y}_k) \in \mathcal{C}_k, \end{aligned} \tag{2.27}$$

where the variable is $\mathbf{x}_k \in \mathbb{R}^{n_k}$. We denote the optimal value of problem (2.27) by $\phi_k(\mathbf{y}_k)$. The master problem can then be written as

$$\begin{aligned} \text{minimize} \quad & \phi(\mathbf{z}) = \sum_{k=1}^{K} \phi_k(\mathbf{E}_k\mathbf{z}) \\ \text{subject to} \quad & \mathbf{z} \in \mathcal{Z}, \end{aligned} \tag{2.28}$$

where the variable is $\mathbf{z} \in \mathbb{R}^N$. Note that $\phi_k(\mathbf{y}_k)$ is convex with respect to $\mathbf{y}_k$, which is a property inherited from the convexity of f_k. Moreover note that, $\phi_k(\mathbf{E}_k\mathbf{z})$ is a composition of ϕ_k with an affine mapping. Thus $\phi_k(\mathbf{E}_k\mathbf{z})$ becomes convex with respect to $\mathbf{z}$, and therefore $\phi(\mathbf{z})$ also becomes convex.

We apply projected subgradient method to find the solution of the master problem. Thus, we need to compute a subgradient of $\phi(\mathbf{z})$ at $\mathbf{z}$. To do this, we capitalize on the following two key results:

- Let g and h be a convex functions such $h(\mathbf{w}) = g(\mathbf{A}\mathbf{w})$ for some matrix $\mathbf{A}$ then $\partial h(\mathbf{w}) = \mathbf{A}^T \partial g(\mathbf{A}\mathbf{w})$.
- For a function $f = \sum_{n=1}^{r} f_n$ where $(f_n)_{n=1,\dots,r}$ are convex functions we have:

$$\partial f(\mathbf{w}) = \partial f_1(\mathbf{w}) + \cdots + \partial f_r(\mathbf{w}). \tag{2.29}$$

From these two properties and the fact that ϕ_k are all convex we get that:

$$\partial\phi(\mathbf{z}) = \sum \mathbf{E}_k^{\mathbf{T}} \partial\phi_k(\mathbf{E}_k\mathbf{z}). \tag{2.30}$$

So if $\mathbf{g}_k \in \partial\phi_k(\mathbf{E}_k\mathbf{z})$ for each $k \in \mathcal{K}$ then

$$\mathbf{g} = \sum \mathbf{E}_k^{\mathrm{T}} \mathbf{g}_k \in \partial\phi(\mathbf{z}). \tag{2.31}$$

Note that each component of $\mathbf{g}_k$ is associated with the same net as $\mathbf{y}_k$. We let $\mathbf{z}^{(n)}$ denote the value of the net variable (i.e. $\mathbf{z}$) after n iterations and we let $\alpha^{(n)}$ be the step size at iteration n. Then the steps of the algorithm based on the projected subgradient method are as follows:

Algorithm 3: Primal decomposition

Given an inital $\mathbf{z}^{(0)}$ such that $\mathbf{z}^{(0)} \in \mathcal{Z}$
$n = 0$
repeat
  Subsystem k solves (2.27) to construct $\mathbf{g}_k \in \partial\phi_k(\mathbf{E}_k\mathbf{z}^{(n)})$ for $k \in \mathcal{K}$
  Update the subgradiant $\mathbf{g} = \sum_{k=1}^{K} \mathbf{E}_k^{\mathbf{T}} \mathbf{g}_k$
  Update $\mathbf{z}$: $\mathbf{z}^{(n+1)} = \Pi_{\mathcal{Z}}(\mathbf{z}^{(n)} - \alpha^{(n)}\mathbf{g})$
  $n = n + 1$

If we thoroughly scrutinize $g = \sum \mathbf{E}_k^{\mathrm{T}} \mathbf{g}_k$, we note that nth component of $\mathbf{g}$ (i.e., $(\mathbf{g})_n$) is influenced by qth component of $\mathbf{g}_k$ (i.e., $(\mathbf{g}_k)_q$) if and only if $(\mathbf{g}_k)_q$ is associated with net n. Therefore the problem of finding the nth component of the subgradient $\mathbf{g}$ in each iteration can be solved in distributed fashion as follows. Each subsystem $k \in \mathcal{K}$ solves problem 2.27 to construct a subgradient $\mathbf{g}_k$, and note that for all $q = 1, \cdots, p_k$, $(\mathbf{g}_k)_q$ is associated to some net $n \in \{1, .., N\}$. Then each subsystem $k \in \mathcal{K}$ shares the component q of $\mathbf{g}_k$ (i.e., $(\mathbf{g}_k)_q$) with the subsystems in net n, where n is the net that $(\mathbf{g}_k)_q$ is associated with. So the only communication needed during the algorithm is, for all the subsystems that have a net in common to share their subgradient components associated with that net in each iteration.

Dual decomposition

We now like to present the dual decomposition method for the general framwork. Like before we assume that the the functions $f, f_1, \cdots, f_K$ are strictly

convex functions and that the duality gap for the problem is zero. The partial Lagrangian of problem associated with the constraints $\mathbf{y}_k = \mathbf{E}_k\mathbf{z}$ is

$$\begin{aligned} L(\mathbf{x},\mathbf{y},\mathbf{z},\mathbf{v}) &= \sum_{k=1}^{K} f_k(\mathbf{x}_k,\mathbf{y}_k) + \sum_{k=1}^{K} \mathbf{v}_k^{\mathrm{T}}(\mathbf{y}_k - \mathbf{E}_k\mathbf{z}) \\ &= \sum_{k=1}^{K} \big(f_k(\mathbf{x}_k,\mathbf{y}_k) + \mathbf{v}_k^{\mathbf{T}}\mathbf{y}_k\big) - \mathbf{v}^{\mathbf{T}}\mathbf{E}\mathbf{z}, \end{aligned}$$

where $\mathbf{v}_k$ is the dual variable associated with $\mathbf{y}_k = \mathbf{E}_k\mathbf{z}$ and

$$\mathbf{v} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_K). \tag{2.32}$$

The dual function is then

$$g(\mathbf{v}) = \inf_{\mathbf{x},\mathbf{y},\mathbf{z}} L(\mathbf{x},\mathbf{y},\mathbf{z},\mathbf{v}). \tag{2.33}$$

We see that for g to be bounded below we need the condition $\mathbf{E}^{\mathbf{T}}\mathbf{v} = 0$ which gives us that $\mathbf{v}^{\mathbf{T}}\mathbf{E}\mathbf{z} = 0$. So the Lagrangian can be decoupled and the dual function can be solved independently for each subsystem. Subsystem k then solves the problem

$$\begin{array}{ll} \text{minimize} & f_k(\mathbf{x}_k,\mathbf{y}_k) + \mathbf{v}_k^{\mathbf{T}}\mathbf{y}_k \\ \text{subject to} & (\mathbf{x}_k,\mathbf{y}_k) \in \mathcal{C}_k, \end{array} \tag{2.34}$$

where the variables are $\mathbf{x}_k \in \mathbb{R}^{n_1}$ and $\mathbf{y}_k \in \mathbb{R}^{p_k}$. We let $g_k(\mathbf{v}_k)$ be the optimal value and $(\mathbf{x}_k^\star(\mathbf{v}_k), \mathbf{y}_k^\star(\mathbf{v}_k))$ be the solution to problem (2.34) for a fixed $\mathbf{v}$. The dual problem then becomes

$$\begin{array}{ll} \text{maximize} & g(\mathbf{v}) = \displaystyle\sum_{k=1}^{K} g_k(\mathbf{v}_k) \\ \text{subject to} & \mathbf{E}^{\mathrm{T}}\mathbf{v} = 0 \end{array} \tag{2.35}$$

where the variable is $\mathbf{v}$. It is easy to verify that $-\mathbf{y}_k^\star(\mathbf{v}_k)$ is a subgradient of $-g_k$ at $\mathbf{v}_k$. So we can again use the projected subgradient method. We let $\mathbf{v}^{(n)}$ and $\mathbf{v}_k^{(n)}$ be the dual variables after n iterations and $\alpha^{(n)}$ be the step size

at iteration n. Then the steps of the dual decomposition are

Algorithm 4: Dual decomposition

Given an inital $\mathbf{v}$ such that $\mathbf{E^T v} = 0$
$n = 0$
repeat
  Subsystem k solves (2.34) to construct $-\mathbf{y}_k^\star(\mathbf{v}_k^{(n)}) \in \partial\left(-g_k(\mathbf{v}_k^{(n)})\right)$
  Update the subgradiant $-\mathbf{y} = -(\mathbf{y}_1^\star(\mathbf{v}_1^{(n)}), \cdots, \mathbf{y}_K^\star(\mathbf{v}_K^{(n)}))$
  Update $\mathbf{v}$: $\mathbf{v}^{(n+1)} = \Pi_{\{\mathbf{v}|\mathbf{E}^{\mathrm{T}}\mathbf{v}=0\}}(\mathbf{v}^{(n)} + \alpha^{(n)}\mathbf{y})$
  $n = n + 1$

2.1.3 Alternating Direction Method of Multipliers (ADMM)

In this section we will introduce the alternating direction method of multipliers (ADMM). The ADMM method is an attempt to improve the convergent properties and robustness of the dual decomposition method by the use of augmented Lagrangians and the method of multipliers. The ADMM method indeed has superior convergent properties than other decomposition methods.

This section is based on the material originally presented in [4] and we refer the reader to [4] for details. We start by presenting augmented Lagrangians and the method of multipliers. Then we present the basic theory behind the ADMM method. Next we present how the ADMM method can be used to address problems possessing the general decomposition structure from section 2.1.2, followed by a brief discussion on the use of ADMM methods in nonconvex problems.

Augmented lagrangians and the method of multipliers

Consider the following problem

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{Ax} = \mathbf{b}, \end{array} \tag{2.36}$$

where the variables are $\mathbf{x}$. In (2.36) we require that $\mathbf{Ax} - \mathbf{b} = 0$ therefore we can write (2.36) equivalently as

$$\begin{aligned} &\text{minimize} \quad f(\mathbf{x}) + (\rho/2)||\mathbf{Ax} - \mathbf{b}||_2^2 \\ &\text{subject to} \quad \mathbf{Ax} = \mathbf{b}, \end{aligned} \tag{2.37}$$

where the variables are $\mathbf{x}$. The Lagrangian of (2.37) is referred to as the augmented Lagrangian of (2.36) and it is given by

$$L_\rho(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}) + \mathbf{y}^{\mathrm{T}}(\mathbf{Ax} - \mathbf{b}) + (\rho/2)||\mathbf{Ax} - \mathbf{b}||_2^2. \tag{2.38}$$

Appling the dual ascent method on (2.36) using the augmented Lagrangian instead of the normal Lagrangian is called the method of multipliers. To formally present the corresponding algorithm, we denote by $\mathbf{x}^{(n)}$ and $\mathbf{y}^{(n)}$ the results at algorithm iteration n and by $\mathbf{x}^{(0)}$ and $\mathbf{y}^{(0)}$ the initial values. Then the dual ascent algorithm is

$$\mathbf{x}^{(n+1)} = \mathrm{argmin}_x L_\rho(\mathbf{x}, \mathbf{y}) \tag{2.39}$$

$$\mathbf{y}^{(n+1)} = \mathbf{y}^{(n)} + \rho(\mathbf{Ax}^{(n+1)} - b). \tag{2.40}$$

The step size in each iteration is taken as ρ. The method of multipliers can be shown to converge under much milder conditions than the dual decomposition method [4]. The method of multipliers cannot be decomposed because even though the objective function (therefore also the Lagrangian) is separable the augmented Lagrangian is not. The ADMM on the other hand preserves the separability properties as we will see in the next section.

ADMM

In this section we will examine the ADMM method. The method combines two valuable properties namely the decomposition properties of the Lagrangian of the dual decomposition and general convergent properties of the method of multipliers. To see this we the consider the following problem

$$\begin{aligned} &\text{minimize} \quad f(\mathbf{x}) + f(\mathbf{z}) \\ &\text{subject to} \quad \mathbf{Ax} + \mathbf{Bz} = \mathbf{c}, \end{aligned} \tag{2.41}$$

where the variables are $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{z} \in \mathbb{R}^m$ and $\mathbf{A} \in \mathbb{R}^{p \times n}, \mathbf{B} \in \mathbb{R}^{p \times m}$, and $\mathbf{c} \in \mathbb{R}^p$. We assume that the problem is convex. The augmented Lagrangian of problem (2.41) is

$$L_\rho(\mathbf{x}, \mathbf{z}, \lambda) = f(\mathbf{x}) + g(\mathbf{z}) + \boldsymbol{\lambda}^{\mathrm{T}}(\mathbf{Ax} + \mathbf{Bz} - \mathbf{c}) + (\rho/2)||\mathbf{Ax} + \mathbf{Bz} - \mathbf{c}||_2^2. \tag{2.42}$$

If we would apply the method of multipliers we would first update both $\mathbf{x}$ and $\mathbf{z}$ at the same time and then update λ. But here we can use the special structure of the problem and first update $\mathbf{x}$, then $\mathbf{z}$ and finaly $\boldsymbol{\lambda}$. To formally present the algorithm we let $\mathbf{x}^{(n)}, \mathbf{z}^{(n)}, \boldsymbol{\lambda}^{(n)}$ be the variables $\mathbf{x}, \mathbf{z}$ and $\boldsymbol{\lambda}$ respectively after n iterations. Then the algorithm can be summarized as follows:

Algorithm 5: ADMM

Given an inital $\mathbf{z}^{(0)}$ and $\boldsymbol{\lambda}^{(0)}$
$n = 0$
repeat
  $\mathbf{x}$ update: $\mathbf{x}^{(n+1)} = \text{argmin}_{\mathbf{x}} L_\rho(\mathbf{x}, \mathbf{z}^{(n)}, \boldsymbol{\lambda}^{(n)})$
  $\mathbf{y}$ update: $\mathbf{z}^{(n+1)} = \text{argmin}_{\mathbf{z}} L_\rho(\mathbf{x}^{(n+1)}, \mathbf{z}, \boldsymbol{\lambda}^{(n)})$
  $\mathbf{z}$ update: $\boldsymbol{\lambda}^{(n+1)} = \boldsymbol{\lambda}^k + \rho(\mathbf{A}\mathbf{x}^{(n+1)} + \mathbf{B}\mathbf{z}^{(n+1)} - \mathbf{c})$
  $n = n + 1$

In [4] the following general convergent properties of the ADMM method are proved. If f and g are closed proper and convex and the Lagrangian has a saddle point then the following holds:

- $\lim_{n\to\infty} \mathbf{A}\mathbf{x}^{(n)} + \mathbf{B}\mathbf{z}^{(n)} - \mathbf{c} = 0$.
- $\lim_{n\to\infty}(f(\mathbf{x}^{(n)}) + g(\mathbf{x}^{(n)})) = p^\star$, where $p^\star$ is the optimal value.
- $\lim_{n\to\infty} \mathbf{y}^{(n)} = \mathbf{y}^\star$, where is the optimal dual point.

The ADMM tends to have much faster convergent than the primal and dual decomposition methods we discussed in sections 2.1.1 and 2.1.2. However, many centralized algorithms have faster convergence properties than ADMM.

General consensus ADMM

We now like to apply the ADMM algorithm to our general decomposition structures. The considered problem is

$$\begin{array}{lll} \text{minimize} & \displaystyle\sum_{k=1}^{K} f_k(\mathbf{x}_k, \mathbf{y}_k) & \\ \text{subject to} & (\mathbf{x}_k, \mathbf{y}_k) \in \mathcal{C}_k, & k = 1, \ldots, K \\ & \mathbf{y}_k = \mathbf{E}_k \mathbf{z}, & k = 1, \ldots, K \\ & \mathbf{z} \in \mathcal{Z}, & \end{array} \tag{2.43}$$

where the variables are $\mathbf{x}_k, \mathbf{y}_k$ for $k \in \mathcal{K}$ and $\mathbf{z}_k$. We assume that the problem is convex. The partial augmented Lagrangian of (2.43) related to the consistency constraints is given by

$$L_\rho(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \sum_{k=1}^{K} \left(f_k(\mathbf{x}_k, \mathbf{y}_k) + \boldsymbol{\lambda}^{\mathrm{T}}(\mathbf{y}_k - \mathbf{E}_k \mathbf{z}) + (\rho/2)||\mathbf{y}_k - \mathbf{E}_k \mathbf{z}||_2^2 \right). \tag{2.44}$$

Now we can use the steps of the ADMM so that we first update $\mathbf{y}_k$, for all $k \in \mathcal{K}$, then $\mathbf{z}$ and finally $\boldsymbol{\lambda}_k$, for all $k \in \mathcal{K}$. For a formal presentation of the algorithm we denote by $\mathbf{x}_k^{(n)}$, $\mathbf{y}_k^{(n)}$, $\boldsymbol{\lambda}_k^{(n)}$ for $k \in \mathcal{K}$ and $\mathbf{z}^{(n)}$ the private, public, dual, and net variables, respectively, at the nth iterations. Furthermore, we denote by $(\mathbf{x}_k^\star(\mathbf{z}, \boldsymbol{\lambda}_k), \mathbf{y}_k^\star(\mathbf{z}, \boldsymbol{\lambda}_k))$ the solution to

$$\begin{array}{ll} \text{minimize} & f_k(\mathbf{x}_k, \mathbf{y}_k) + \boldsymbol{\lambda}_k^{\mathrm{T}}(\mathbf{y}_k - \mathbf{E}_k \mathbf{z}) + (\rho/2)||\mathbf{y}_k - \mathbf{E}_k \mathbf{z}||_2^2 \\ \text{subject to} & (\mathbf{x}_k, \mathbf{y}_k) \in \mathcal{C}_k, \end{array}$$

where the variables are $\mathbf{x}_k$ and $\mathbf{y}_k$ and similarly we denote by $\mathbf{z}^\star(\mathbf{y}, \boldsymbol{\lambda})$, where

$$\mathbf{y} = (\mathbf{y}_1, \cdots, \mathbf{y}_K) \tag{2.45}$$

$$\boldsymbol{\lambda} = (\boldsymbol{\lambda}_1, \cdots, \boldsymbol{\lambda}_K), \tag{2.46}$$

the solution to

$$\text{minimize} \quad \sum_{k \in \mathcal{K}} \left(\boldsymbol{\lambda}_k^{\mathrm{T}}(\mathbf{y}_k - \mathbf{E}_k \mathbf{z}) + (\rho/2)||\mathbf{y}_k - \mathbf{E}_k \mathbf{z}||_2^2 \right), \tag{2.47}$$

where the variable is $\mathbf{z}$. Then the ADMM algorithm can be summarized as follows:

Algorithm 6: ADMM

Given an inital $\mathbf{z}^{(0)}$ and $\boldsymbol{\lambda}^{(0)}$
$n = 0$
repeat
  $\mathbf{y}$ update: $\mathbf{y}_k^{(n+1)} = \mathbf{y}_k^\star(\mathbf{z}^{(n)}, \boldsymbol{\lambda}_k^{(n)})$, for $k \in \mathcal{K}$.
  $\mathbf{z}$ update: $\mathbf{z}^{(n+1)} = \mathbf{z}^\star(\mathbf{y}^{(n+1)}, \boldsymbol{\lambda}^{(n)})$.
  $\boldsymbol{\lambda}$ update: $\boldsymbol{\lambda}_k^{(n+1)} = \boldsymbol{\lambda}_k^{(n)} + \rho(\mathbf{y}_k^{(n+1)} - \mathbf{E}_k\mathbf{z}^{(n+1)})$, for $k \in \mathcal{K}$.
  $n = n + 1$

From the discussion in section 3.3 in [4] we have for all $n > 0$ that

$$\mathbf{E}^{\mathrm{T}}\boldsymbol{\lambda}^{(n)} = 0 \tag{2.48}$$

i.e.

$$\sum_{k\in\mathcal{K}} \mathbf{E}_k^{\mathrm{T}}\boldsymbol{\lambda}_k^{(n)} = 0. \tag{2.49}$$

Moreover, we can find an analytic solution to (2.47) and $\mathbf{z}^{(n+1)}$ can therefore be expressed as

$$\mathbf{z}^{(n+1)} = \left(\sum_{k\in\mathcal{K}} \mathbf{E}_k^{\mathrm{T}}\mathbf{E}_k\right)^{-1} \sum_{k\in\mathcal{K}} \mathbf{E}_k^{\mathrm{T}}\left(\mathbf{y}_k^{(n+1)} + (1/\rho)\boldsymbol{\lambda}_k^{(n)}\right) \tag{2.50}$$

$$= \mathrm{diag}(\omega_1^{-1}, \cdots, \omega_N^{-1})\left(\sum_{k\in\mathcal{K}} \mathbf{E}_k^{\mathrm{T}}\mathbf{y}_k^{(n+1)} + (1/\rho)\sum_{k\in\mathcal{K}} \mathbf{E}_k^{\mathrm{T}}\boldsymbol{\lambda}_k^{(n)}\right) \tag{2.51}$$

$$= \mathrm{diag}(\omega_1^{-1}, \cdots, \omega_N^{-1})\sum_{k\in\mathcal{K}} \mathbf{E}_k^{\mathrm{T}}\mathbf{y}_k^{(n+1)}, \tag{2.52}$$

The steps of the algorithm can be performed in distributed fashion as follows.heuristics First all the subsystems perform the public variable update. Then subsystem k shares the component q of $\mathbf{y}_k$ (i.e., $(\mathbf{y}_k)_q$) with the subsystems in net l, where l is the net that $(\mathbf{y}_k)_q$ is associated with. Then the subsystems can perform both the net variable update and the dual variable update locally.

ADMM for non-convex problems

Till now we have only focused on the ADMM method in the context of convex optimization. However, in [4, section 9] the potentials of using the ADMM as an heuristics for non-convex problems are discussed. Even though the resulting algorithms are only local methods, the ADMM might offer good scalability properties.

2.2 Privacy preserving optimization

In this section, we discuss privacy preserving multi party optimization problems. But multi party optimization problems are where multiple parties want to optimize a global objective function. Like in all optimization problems there is some data involved, objective function, constraints and variables. In the case of multi party optimization problems all the problem data is distributed between the parties. In many situations this data is very sensitive. So even though the parties would gain from cooperation they might not be willing to share their sensitive data. We can for example think of a medical classification problem, the problem of classifying and diagnosing diseases from massive data. For better results it would be of interest to have access to as much amount of data from as many hospitals as possible. But since patient records are very sensitive and personal information the hospitals might not be willing to share them. Another example is the optimal power flow problem introduced in section 1.1. In section 1.2, we noted that if the power use of a household is known then detailed information about the electric behavior of the household could be inferred. In this section we will look at how the optimization can be performed without the parties having to reveal their problem data. For a more thorough discussion on the subject we direct the interested reader to [24].

2.2.1 Approaches to privacy preserving multi party optimization

In the literature there are three existing approaches to cope with the problem of privacy preserving multi party optimization. Those are cryptography methods, transformation methods and decomposition methods. In sequence, we briefly present each of these methods.

Methods based on Cryptography

Cryptography approaches are based on using traditional cryptography methods to hide the sensitive data within the optimization method. For that purpose each step of the algorithm usually contains an encryption and decryption step. Most cryptography methods build an encryption and decryption on top of standard centralized optimization methods. The encryption and decryption is usually done on all the data in each step of the algorithm. Therefore both complexity and scalability can be put into question. Examples of methods based on cryptography can be found in [2], which presents privacy preserving interior point algorithm, and [23], which presents a privacy preserving version of the simplex method.

Methods based on transformation

Transformation approaches[2, 1] are based on using algebraic transformations that hide the problem data. Let us suppose that multiple parties are interested in collaborating towards solving an optimization problem and, furthermore, that the problem data is distributed between the parties and no party is willing to share their data. In some cases each party involved can transform there part of the data such that the original data is untraceable but the yielding optimization problem is equivalent to the original problem. If the transformation each party uses is one to one correspondence and each party only knows its own transformation then the optimization problem can be solved in following privacy preserving way. Each party shares there transformed data. Then we have a problem that is equivalent to the original problem except each parties private data cannot be traced. This equivalent problem can then be solved either by some third party or some of the participating parties. Then the solution is published and each party can re transform its own data back. This approach for solving the privacy preserving multi party optimization problem is called transformation method.

The problem with the transformation approach is that it is very dependent on the problem structure. It is therefore hard to generalize and each transformation has to be tailor made for each problem structure. Then it is not possible in nearly all cases to find transformation that gives an equivalent problem and hides the private data. One quality that transformation methods have over the cryptography methods is that the transformation usually just performed in the beginning and the end of the algorithm whereas the

encryption and decryption in the cryptography methods is performed in each iteration. Therefore the transformation does not affect the complexity of the solution algorithm less the complexity of the transformation is larger than the complexity of the solution method. An example of how transformation methods can be used outsource large-scale linear programing problem to a cloud in a privacy preserving manner is presented in [8].

Methods based on decompostion

In section 2.1.1 we introduced decomposition methods. In [24], it is observed that many well studied decomposition methods have inherited privacy properties. This is because the subsystems can often deal locally with the private data and there vanishes the need to share it. For further explanation we revisit the general decomposition structure from section 2.1.2. There each subsystem $k \in \mathcal{K}$ has a private objective function part f_k, private variables $\mathbf{x}_k$, private constraint set $\mathcal{C}_k$ and public variables $\mathbf{y}_k$. We assume that subsystem k is not willing to share its private data, f_k, $\mathbf{x}_k$ and $\mathcal{C}_k$. If the primal decomposition, dual decomposition, and the ADMM algorithms are considered (see section 2.1.2 and 2.1.3), we make the following observation. The only communication throughout both the ADMM and dual decomposition is that each subsystem shares each of its public variables with the subsystems that that have a public variable in the same net. The same applies for the primal decomposition except there the subsystems share there subgradient in the same way. Therefore, to preserve the privacy of private data, it is sufficient to require that the private data of each subsystem cannot be built from its the subgradients or its public variables. Decomposition methods do not require any cryptography protocol or transformation and therefore there is no overhead included in obtaining the privacy. In section 2.1.1 we noted that decomposition methods have many other good characteristics such as scalability. Therefore decomposition methods are good candidates for solving a large scale privacy preserving multi party optimization problems.

Chapter 3

Problem formulation

In this section we will state the optimal power flow problem in mathematical terms. We will focus on the mathematics of the problem and only present the necessary details about power systems for full understanding of the problem. For a rigorous understanding of power systems we refer the interested reader to books on the subject [19]. We will assume that our power system has no phase shifting transformers or charging capacitance. This assumption will not affect the mathematical properties of the formulation but simplify the presentation. For more information on phase shifting transformers and charging capacitance and how they can be integrated into the system, we direct the reader to [25]. The rest of the chapter is organized as follows. In section 3.1 we introduce the notation that is used throughout this thesis on the electrical network together with the related physics and physical constraints. Then we formulate the problem in section 3.2. Finally, in section 3.3 demonstrate how the formulation can be converted to the general decomposition structure for distributed problems 2.1.2.

3.1 Physics of the electrical networks

In this section we present concisely the physical properties of the electrical network and how they relate. Moreover, we discuss how the physical properties are related with each other.

Throughout this section and for the rest of this thesis we consider an electrical network with the following set of buses $\mathcal{N} = \{1, 2, \dots, N\}$ and set of flow lines $\mathcal{L} \subseteq \mathcal{N} \times \mathcal{N}$. Each bus in the network has a power load or

demand and we let $s_k^{\mathrm{L}} = p_k^{\mathrm{L}} + jq_k^{\mathrm{L}} \in \mathbb{C}$ be the power load for bus $k \in \mathcal{N}$. Some of the buses also have a power generator and we refer to these buses as generator buses. We let $\mathcal{G} \subseteq \mathcal{N}$ denote the set of generator buses and let $s_k^{\mathrm{G}} = p_k^{\mathrm{G}} + jq_k^{\mathrm{G}} \in \mathbb{C}$ be the power generated at generator bus $k \in \mathcal{G}$. It will also be convenient to talk about injected power at bus k. We let $s_k = p_k + jq_k$ denote the injected power. In particular we have

$$s_k = p_k + jq_k = \begin{cases} s_k^{\mathrm{G}} - s_k^{\mathrm{L}} & \text{if } k \in \mathcal{G} \ ; \\ -s_k^{\mathrm{L}} & \text{otherwise.} \end{cases} \tag{3.1}$$

We let $\mathbf{s} = \mathbf{p} + j\mathbf{q}$ be the vector where the kth component is s_k.

Besides power injection each bus has voltage and current injection. The voltages of all the buses are used to control the flow of electricity in the network. The current injection at bus k is the electric current that is injected into bus k. We use the following notation for voltage and current injection:

(a) $i_k = i_k^{\mathrm{Re}} + ji_k^{\mathrm{Im}}$: the current injection at node k.

(b) $\mathbf{i} = \mathbf{i}^{\mathrm{Re}} + j\mathbf{i}^{\mathrm{Im}}$: the vector of current injections where component k is i_k.

(c) $v_k = v_k^{\mathrm{Re}} + jv_k^{\mathrm{Im}}$: the voltages on rectangular form.

(d) $\mathbf{v} = \mathbf{v}^{\mathrm{Re}} + j\mathbf{v}^{\mathrm{Im}}$: vector of voltages where component k is v_k.

Each flow line $(n, m) \in \mathcal{L}$ has an admittance value $y_{nm} = g_{nm} + jb_{nm} \in \mathbb{C}$, where $g_{nm} \in \mathbb{R}$ is the real part and $b_{nm} \in \mathbb{R}$ is the imaginary part of admittance of the flow line. The admittance y_{nm} is a measure of how easily the current flows through the flow line. To express the admittance over the whole network we introduce the admittance matrix of the network $\mathbf{Y} = \mathbf{G} + j\mathbf{B} \in \mathbb{C}^{N \times N}$, where $\mathbf{G} \in \mathbb{R}^{N \times N}$ is the real part and $\mathbf{B} \in \mathbb{R}^{N \times N}$ is the imaginary part of the admittance matrix. It is defined as follows:

$$\mathbf{Y} = \begin{cases} -y_{nm} & \text{if } (n, m) \in \mathcal{L} \ ; \\ y_{nn} + \sum_{(n,l) \in \mathcal{L}} y_{nl} & \text{if } n = m; \\ 0 & \text{if } (n, m) \notin \mathcal{L} \ , \end{cases} \tag{3.2}$$

where y_{nn} is the admittance to ground at bus n. Note that each bus is usually only connected to few other buses via flow line so the admittance matrix will be very sparse for large networks.

The relation between the current injection and the voltages can then be expressed as

$$\mathbf{i} = \mathbf{Y}\mathbf{v} = \left(\mathbf{G}\mathbf{v}^{\mathrm{Re}} - \mathbf{B}\mathbf{v}^{\mathrm{Im}}\right) + j\left(\mathbf{B}\mathbf{v}^{\mathrm{Re}} + \mathbf{G}\mathbf{v}^{\mathrm{Im}}\right). \tag{3.3}$$

We have the following relation between the injected power, voltages and current injection

$$\mathbf{s} = \mathbf{v} \bullet \mathbf{i}^* = \left(\mathbf{v}^{\mathrm{Re}} \bullet \mathbf{i}^{\mathrm{Re}} + \mathbf{v}^{\mathrm{Im}} \bullet \mathbf{i}^{\mathrm{Im}}\right) + j\left(\mathbf{v}^{\mathrm{Im}} \bullet \mathbf{i}^{\mathrm{Re}} - \mathbf{v}^{\mathrm{Re}} \bullet \mathbf{i}^{\mathrm{Im}}\right). \tag{3.4}$$

Note that from (3.3) and (3.4), we have

$$\mathbf{s} = \mathbf{p} + j\mathbf{q} = \mathbf{v} \bullet (\mathbf{Y}\mathbf{v})^*. \tag{3.5}$$

The system also has some operational limitations. For example each power generator has an upper and lower limit on how much real and reactive power it can generate. This we can represent in mathematical terms as

$$p_k^{\mathrm{G,min}} \leq p_k^{\mathrm{G}} \leq p_k^{\mathrm{G,max}} \qquad k \in \mathcal{G} \tag{3.6}$$

$$q_k^{\mathrm{G,min}} \leq q_k^{\mathrm{G}} \leq q_k^{\mathrm{G,max}} \qquad k \in \mathcal{G}. \tag{3.7}$$

We also have an upper and lower limit on the voltage magnitude on each bus

$$(v_k^{\min})^2 \leq ||v_k||_2^2 = (v_k^{\mathrm{Re}})^2 + (v_k^{\mathrm{Im}})^2 \leq (v_k^{\max})^2. \tag{3.8}$$

For each flow line $(n, m) \in \mathcal{L}$, we denote the current through the line from bus n to bus m by $i_{nm} = i_{nm}^{\mathrm{Re}} + j i_{nm}^{\mathrm{Im}}$. We have that $i_{nm} = y_{nm}(v_n - v_m)$. By doing simple calculations we get that [1]

$$i_{nm}^{\mathrm{Re}} = g_{nm}(v_n^{\mathrm{Re}} - v_m^{\mathrm{Re}}) + b_{nm}(v_n^{\mathrm{Im}} - v_m^{\mathrm{Im}}) \tag{3.9}$$

$$i_{nm}^{\mathrm{Im}} = b_{nm}(v_n^{\mathrm{Re}} - v_m^{\mathrm{Re}}) - g_{nm}(v_n^{\mathrm{Im}} - v_m^{\mathrm{Im}}). \tag{3.10}$$

Moreover, if we set

$$\mathbf{c}_{nm} = (g_{nm}, -g_{nm}, b_{nm}, -b_{nm}) \tag{3.11}$$

$$\mathbf{d}_{nm} = (b_{nm}, -b_{nm}, -g_{nm}, g_{nm}), \tag{3.12}$$

[1] This formula is a little bit different if we have phase shifting transformers or total charging capacitance but we still keep the linear relationship between i_{nm} and v_n, v_m.

then we get

$$i_{nm}^{\mathrm{Re}} = \mathbf{c}_{nm}^{\mathbf{T}}(v_n^{\mathrm{Re}}, v_m^{\mathrm{Re}}, v_n^{\mathrm{Im}}, v_m^{\mathrm{Im}}) \tag{3.13}$$

$$i_{nm}^{\mathrm{Im}} = \mathbf{d}_{nm}^{\mathbf{T}}(v_n^{\mathrm{Re}}, v_m^{\mathrm{Re}}, v_n^{\mathrm{Im}}, v_m^{\mathrm{Im}}).. \tag{3.14}$$

Note that $i_{nm} = -i_{mn}$.[2] For each flow line $(n, m) \in \mathcal{L}$ we also need to talk about the complex power that bus n injects into the flow line (n, m). We let $s_{nm} = p_{nm}^{\mathrm{Re}} + jq_{nm}^{\mathrm{Im}}$ denote the power injected by bus n into the flow line $(n, m) \in \mathcal{L}$. We have the following relationship between s_{nm}, the current and the voltages

$$s_{nm} = v_n i_{nm}^* = \left(v_n^{\mathrm{Re}} i_{nm}^{\mathrm{Re}} + v_n^{\mathrm{Im}} i_{nm}^{\mathrm{Im}}\right) + j\left(v_n^{\mathrm{Im}} i_{nm}^{\mathrm{Re}} - v_n^{\mathrm{Re}} i_{nm}^{\mathrm{Im}}\right). \tag{3.15}$$

The flow lines often have some operational limitation to the line to prevent overheating or for stability reasons. There are four types of limitations.[3]

- Limit on the current through the line, so for each $(n, m) \in \mathcal{L}$ we have $i_{nm}^{\max}$ such that

  $$||i_{nm}||_2^2 = (i_{nm}^{\mathrm{Re}})^2 + (i_{nm}^{\mathrm{Im}})^2 \leq (i_{nm}^{\max})^2. \tag{3.16}$$

  Note that since $i_{nm} = y_{nm}(v_n - v_m)$ this can also be written as

  $$|v_n - v_m| \leq \frac{i_{nm}^{\max}}{|y_{nm}|}, \tag{3.17}$$

  here $|\cdot|$ is the norm of a complex number.

- Limit on the apparent power through the line, so for each $(n, m) \in \mathcal{L}$ we have $s_{nm}^{\max}$ such that:

  $$||s_{nm}||_2^2 = p_{nm}^2 + q_{nm}^2 \leq (s_{nm}^{\max})^2. \tag{3.18}$$

- Limit on the real power through the line, so for each $(n, m) \in \mathcal{L}$ we have $p_{nm}^{\max}$ such that

  $$|p_{nm}| \leq p_{nm}^{\max}. \tag{3.19}$$

[2]Note that this does not apply when we have phase shifting transformers or total charging capacitance.

[3]The type of limitation that is chosen depends on the situation.

- Limit on the angle difference between the voltages on the buses connected to the line. That is for each line $(n, m) \in \mathcal{L}$ we have the following constraint:

$$|\theta_n - \theta_m| \leq \hat{\theta}_{n,m}, \tag{3.20}$$

where $v_n = |v_n|e^{j\theta}$ and $v_m = |v_m|e^{j\theta_m}$. In the original formulation of the optimal power flow problem, [7], polar form is used to represent the voltages. The use of the voltages on rectangular form is heavily adopted in the literature, because it makes the other constraints and the object function easier to deal with. When the rectangular form of the voltages is used then constraint (3.20) becomes

$$\left|\arctan\left(\frac{v_n^{\mathrm{Re}}}{v_n^{\mathrm{Re}}}\right) - \arctan\left(\frac{v_m^{\mathrm{Re}}}{v_m^{\mathrm{Re}}}\right)\right| \leq \hat{\theta}_{n,m}. \tag{3.21}$$

In practice, the angle difference tends to be small even if constraint (3.20) is relaxed and as a result the constraint above is usually not considered in the OPF problem [13].

3.2 Formulation of the optimal power flow problem

In this section we will present the OPF problem in mathematical terms. The formulation that is presented here is based on [13] and [25].

The OPF problem deals with finding a feasible control, with respect to physics and other limitations presented in section 3.1, that is considered optimal by some criteria. The control variables are the voltages which control how the electricity flows through the network and the power generation of each generating bus which controls how much power the generator generates into the system. The criteria that is used to measure the optimality of the control varies between applications. In this thesis we consider the most common criteria of minimize the cost of real power generation in the system. The cost of generating real power is usually modeled as a quadratic function. We denote the cost of generating power at bus $k \in \mathcal{G}$ by f_k^{G} and assume that f_k^{G} is quadratic function with positive coefficients:

$$f_k^{\mathrm{G}}(p_k^{\mathrm{G}}) = a_k^0(p_k^{\mathrm{G}})^2 + a_k^1 p_k^{\mathrm{G}} + a_k^2, \tag{3.22}$$

where $a_0, a_1, a_2 \geq 0$. We note that the second derivative of f_k^{G} is $2a_k^0 \geq 0$ so f_k^{G} is a convex function. The objective function of the problem can then expressed as

$$\sum_{k \in \mathcal{G}} f_k^{\mathrm{G}}(p_k^{\mathrm{G}}). \tag{3.23}$$

Together with the considered constraints above, we can present the optimal power flow probelm as

$$\begin{aligned}
\text{minimize} \quad & \sum_{k \in \mathcal{G}} f_k^{\mathrm{G}}(p_k^{\mathrm{G}}) && && \text{(3.24a)}\\
\text{subject to} \quad & p_k = p_k^{\mathrm{G}} - p_k^{\mathrm{L}} && k \in \mathcal{G} && \text{(3.24b)}\\
& q_k = q_k^{\mathrm{G}} - q_k^{\mathrm{L}} && k \in \mathcal{G} && \text{(3.24c)}\\
& p_k = -p_k^{\mathrm{L}} && k \in \mathcal{N} \setminus \mathcal{G} && \text{(3.24d)}\\
& q_k = -q_k^{\mathrm{L}} && k \in \mathcal{N} \setminus \mathcal{G} && \text{(3.24e)}\\
& p_k^{\mathrm{G,min}} \leq p_k^{\mathrm{G}} \leq p_k^{\mathrm{G,max}} && k \in \mathcal{G} && \text{(3.24f)}\\
& q_k^{\mathrm{G,min}} \leq q_k^{\mathrm{G}} \leq q_k^{\mathrm{G,max}} && k \in \mathcal{G} && \text{(3.24g)}\\
& \mathbf{i}^{\,\mathrm{Re}} = \mathbf{G}\mathbf{v}^{\mathrm{Re}} - \mathbf{B}\mathbf{v}^{\mathrm{Im}} && && \text{(3.24h)}\\
& \mathbf{i}^{\,\mathrm{Im}} = \mathbf{B}\mathbf{v}^{\mathrm{Re}} + \mathbf{G}\mathbf{v}^{\mathrm{Im}} && && \text{(3.24i)}\\
& \mathbf{p} = \mathbf{v}^{\mathrm{Re}} \bullet \mathbf{i}^{\mathrm{Re}} + \mathbf{v}^{\mathrm{Im}} \bullet \mathbf{i}^{\mathrm{Im}} && && \text{(3.24j)}\\
& \mathbf{q} = \mathbf{v}^{\mathrm{Im}} \bullet \mathbf{i}^{\mathrm{Re}} - \mathbf{v}^{\mathrm{Re}} \bullet \mathbf{i}^{\mathrm{Im}} && && \text{(3.24k)}\\
& (v_k^{\mathrm{min}})^2 \leq (v_k^{\mathrm{Re}})^2 + (v_k^{\mathrm{Im}})^2 && k \in \mathcal{N} && \text{(3.24l)}\\
& (v_k^{\mathrm{max}})^2 \geq (v_k^{\mathrm{Re}})^2 + (v_k^{\mathrm{Im}})^2 && k \in \mathcal{N} && \text{(3.24m)}\\
& i_{nm}^{\mathrm{Re}} = \mathbf{c}_{nm}^{\mathbf{T}}(v_n^{\mathrm{Re}}, v_n^{\mathrm{Re}}, v_k^{\mathrm{Im}}, v_k^{\mathrm{Im}}) && (n,m) \in \mathcal{L} && \text{(3.24n)}\\
& i_{nm}^{\mathrm{Im}} = \mathbf{d}_{nm}^{\mathbf{T}}(v_n^{\mathrm{Re}}, v_n^{\mathrm{Re}}, v_k^{\mathrm{Im}}, v_k^{\mathrm{Im}}) && (n,m) \in \mathcal{L} && \text{(3.24o)}\\
& p_{nm} = v_n^{\mathrm{Re}} i_{nm}^{\mathrm{Re}} + v_n^{\mathrm{Im}} i_{nm}^{\mathrm{Im}} && (n,m) \in \mathcal{L} && \text{(3.24p)}\\
& q_{nm} = v_n^{\mathrm{Im}} i_{nm}^{\mathrm{Re}} - v_n^{\mathrm{Re}} i_{nm}^{\mathrm{Im}} && (n,m) \in \mathcal{L} && \text{(3.24q)}\\
& (i_{nm}^{\mathrm{max}})^2 \geq (i_{nm}^{\mathrm{Re}})^2 + (i_{nm}^{\mathrm{Im}})^2 && (n,m) \in \mathcal{L} && \text{(3.24r)}\\
& (s_{nm}^{\mathrm{max}})^2 \geq p_{nm}^2 + q_{nm}^2 && (n,m) \in \mathcal{L} && \text{(3.24s)}\\
& p_{nm}^{\mathrm{max}} \geq |p_{nm}|, && (n,m) \in \mathcal{L} && \text{(3.24t)}
\end{aligned}$$

Where the variables are $\mathbf{v}^{\,\mathrm{Re}}, \mathbf{v}^{\,\mathrm{Im}}, \mathbf{i}^{\,\mathrm{Re}}, \mathbf{i}^{\,\mathrm{Im}}\mathbf{p}, \mathbf{q}$, $p_k^{\mathrm{G}}, q_k^{\mathrm{G}}$ for $k \in \mathcal{G}$, and i_{nm}^{Re}, i_{nm}^{Im}, p_{nm}, q_{nm} for $(n,m) \in \mathcal{L}$. We assume that $i_{nm}^{\mathrm{max}} = i_{mn}^{\mathrm{max}}$, $s_{nm}^{\mathrm{max}} = s_{mn}^{\mathrm{max}}$ and

$p_{nm}^{\max} = p_{mn}^{\max}$.[4] Here we have three types of constraints

- Linear constraints: (3.24b), (3.24c), (3.24d), (3.24e), (3.24f), (3.24g), (3.24h), (3.24i), (3.24n), and (3.24o).
- Non-linear convex constraints: (3.24m), (3.24r), (3.24s), and (3.24t).
- Non-convex constraints: (3.24l), (3.24j), (3.24k), (3.24p), and (3.24q).

The first two types of constraints, the linear constraint and the non-linear convex constraints, define a convex set. So if we only had these constraints we would have a convex problem since the objective function is convex. The non-convex constraints make the feasible set non-convex and thus result in a non-convex problem.

The non-convexity is not our only concern about the problem. Scalability is another issue that needs to be addressed. As we discussed in section 2.1.1, scalability issues are gracefully handled by decomposition techniques, provided there is a rich decomposition structure to be exploited. In the next section we equivalently reformulate the OPF problem by identifying its decomposition structure so that the distributed optimization techniques presented earlier can be employed to address the problem.

3.3 Distributed formulation

Note that problem (3.24) is a non-convex problem. In addition, most real world OPF applications are related with large scale networks, and therefore direct applications of centralized methods can be ineffective. For this reason we are inspired to go for a distributed algorithm. It is interesting to note that most real world examples of power networks are sparse. In other words, each bus in the network is only connected to few others by flow lines. This sparse connectivity allows us to identify the coupling of problem (3.24) and its decomposition structure, so that the decomposition techniques discussed in 2.1.2 can be readily employed. In particular, the decomposition can be done in such a way that each bus corresponds to one subsystem and its neighbors are the busses connected to it. This is mathematically modeled in

[4]Since it does not matter in which direction the the current or power is heading, it has the same effect on the flow line.

this section by introducing relevant net variables. The result is an equivalent reformulation of problem (3.24) in to the general form of 2.21.

We can split the variables of the problem into two categories, namely variables that are associated with a bus and variables that are associated with a flow line. The variables associated with bus $k \in \mathcal{N}$ are $p_k, q_k, v_k^{\mathrm{Re}}, v_k^{\mathrm{Im}}, i_k^{\mathrm{Re}}, i_k^{\mathrm{Im}}$ and $p_k^{\mathrm{G}}, q_k^{\mathrm{G}}$ if the bus is a generator. Similarly the variables associated with flow line $(n, m) \in \mathcal{L}$ are $i_{nm}^{\mathrm{Re}}, i_{nm}^{\mathrm{Im}}, p_{nm}, q_{nm}$. We can also split the constraints into ones that are associated with a bus or ones that are associated with a flow line. In particular, constraints (3.24b) - (3.24m) are associated with a bus while constraints (3.24n) - (3.24t) are associated with a flow line. Recall that we consider buses to be the subsystems. As a result, the problem is decomposed among the buses, where each has to optimize variables associated with buses as well as flow lines. Therefore the buses have to deal with both the variables and constraints that are associated to a flow line. We will start by introducing how we decouple the variables and constraints that are associated with buses. Then it becomes very easy to explain how we deal with the variables and constraints that are associated with flow lines.

As we already noted, each subsystem (i.e., bus) cannot be considered as an isolated unit in the network due to coupling. That is a variable associated with one bus affects a variable associated with another bus. For example changing the voltage of a bus affects the current injection on all of its neighbors due to constraints (3.24h) and (3.24i). So each bus needs to know the voltages of its neighbors to calculate its current injection. In other words, given a bus k, the voltages of all the neighbors of it, accounts for the public variables of bus k. Note that these public variables of bus k have to be consistent with its neighbors voltages. The required consistency is modeled by using consistency constraints.

To mathematically model the coupling variables and constraints, it is useful to introduce some notations. Let

$$\Omega_k = \{k\} \cup \{n | (k, n) \in \mathcal{L}\}, \qquad \text{and} \tag{3.25}$$

$$\omega_k = |\Omega_k|, \tag{3.26}$$

where Ω_k contains the neighbors of bus k and bus k itself and ω_k is the cardinality of Ω_k. Now we can formally introduce the public variables of bus k. In particular, we denote by $\mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}} \in \mathbb{R}^{\omega_k}$ the real and imaginary voltages of the buses in Ω_k. For clarity of the presentation, we organize $\mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}$ such

that the first component is associated with bus k, i.e., we have

$$(\mathbf{x}_k^{\mathrm{Re}})_1 = v_k^{\mathrm{Re}} \qquad (\mathbf{x}_k^{\mathrm{Im}})_1 = v_k^{\mathrm{Re}}. \tag{3.27}$$

Moreover, each component $n = 2, 3, \cdots, \omega_k$ of both $\mathbf{x}_k^{\mathrm{Re}}$ and $\mathbf{x}_k^{\mathrm{Im}}$ corresponds to the same bus in Ω_k. Now we can express constraints (3.24h)-(3.24t) in terms of public and private variables of bus k. Let us first consider constraints (3.24h) and (3.24i). Here, we construct the vector $\mathbf{g}_k + j\mathbf{b}_k \in \mathbb{C}^{\omega_k}$, where $\mathbf{g}_k$ ($\mathbf{b}_k$) is obtained by first extracting kth column of G (B) matrix and then extracting the rows corresponding to buses in Ω_k. The order of the components are preserved as in $\mathbf{x}_k^{\mathrm{Re}}$ and $\mathbf{x}_k^{\mathrm{Im}}$. Then constraints (3.24h) and (3.24i) are given by

$$i_k^{\mathrm{Re}} = \mathbf{g}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Re}} - \mathbf{b}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Im}} \qquad k \in \mathcal{N} \tag{3.28}$$

$$i_k^{\mathrm{Im}} = \mathbf{b}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Re}} + \mathbf{g}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Im}}. \qquad k \in \mathcal{N} \tag{3.29}$$

Then we can write (3.24j), (3.24k) as a relation between public and private variables, i.e.,

$$\mathbf{p}_k = (\mathbf{x}_k^{\mathrm{Re}})_1 i_k^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Im}})_1 i_k^{\mathrm{Im}} \qquad k \in \mathcal{N} \tag{3.30}$$

$$\mathbf{q}_k = (\mathbf{x}_k^{\mathrm{Im}})_1 i_k^{\mathrm{Re}} - (\mathbf{x}_k^{\mathrm{Re}})_1 i_k^{\mathrm{Im}} \qquad k \in \mathcal{N}. \tag{3.31}$$

In the case of (3.24l) and (3.24m) we define the constant vectors $\mathbf{v}_k^{\min}, \mathbf{v}_k^{\max} \in \mathbb{R}^{\omega_k}$ for each $k \in \mathcal{N}$. In particular the n-th components of $\mathbf{v}_k^{\min}$ and $\mathbf{v}_k^{\max}$ are $v_m^{\min}$ and $v_m^{\max}$, respectively, where m is the bus index that corresponds to the n-th component of $\mathbf{x}_k^{\mathrm{Re}}$ (or $\mathbf{x}_k^{\mathrm{Im}}$). Then we can write (3.24l) and (3.24m) as

$$[\mathbf{x}_k^{\mathrm{Re}}]^2 + [\mathbf{x}_k^{\mathrm{Im}}]^2 \geq [\mathbf{v}_k^{\min}]^2 \qquad k \in \mathcal{N} \tag{3.32}$$

$$[\mathbf{x}_k^{\mathrm{Re}}]^2 + [\mathbf{x}_k^{\mathrm{Im}}]^2 \leq [\mathbf{v}_k^{\max}]^2 \qquad k \in \mathcal{N}, \tag{3.33}$$

where $[\cdot]^2$ is the componentwise squared operation. For a more compact notation let

$$\mathcal{X}_k = \{(\mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}) \in \mathbb{C}^{\omega_k} | [\mathbf{v}_k^{\max}]^2 \geq [\mathbf{x}_k^{\mathrm{Re}}]^2 + [\mathbf{x}_k^{\mathrm{Im}}]^2 \geq [\mathbf{v}_k^{\min}]^2\}. \tag{3.34}$$

Then constraints (3.32) and (3.33) can be written for each $k \in \mathcal{N}$ as

$$(\mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}) \in \mathcal{X}_k. \tag{3.35}$$

Recall that the consistency among the public variables is ensured by consistency constraints. To do this, let us introduce the net variable $\mathbf{z} = (\mathbf{z}^{\mathrm{Re}}, \mathbf{z}^{\mathrm{Im}})$. Because every bus is connected to at least one other, we have

$$\mathbf{z}^{\mathrm{Re}} = \mathbf{v}^{\mathrm{Re}} \qquad \mathbf{z}^{\mathrm{Im}} = \mathbf{v}^{\mathrm{Im}}. \tag{3.36}$$

Particularized to our problem, ensuring consistency among the public variables is equivalent to making all the public variable (voltage) components that correspond to the same bus voltage are in agreement. This condition can mathematically be expressed as

$$\mathbf{x}_k^{\mathrm{Re}} = \mathbf{E}_k \mathbf{z}^{\mathrm{Re}} \qquad \mathbf{x}_k^{\mathrm{Im}} = \mathbf{E}_k \mathbf{z}^{\mathrm{Im}}, \tag{3.37}$$

where $\mathbf{E}_k \in \mathbb{R}^{\omega \times N}$ is given by

$$(\mathbf{E}_k)_{n,m} = \begin{cases} 1 & \text{if } (\mathbf{x}_k^{\mathrm{Re}})_n \text{ and } (\mathbf{x}_k^{\mathrm{Im}})_n \text{ are local copies of } v_m^{\mathrm{Re}} \text{ and } v_m^{\mathrm{Im}} \\ 0 & \text{otherwise.} \end{cases} \tag{3.38}$$

Constraints (3.24n) - (3.24t) can also be expressed in terms of private variables and the already defined public variables $\mathbf{x}_k^{\mathrm{R}}$ and $\mathbf{x}_k^{\mathrm{Im}}$, $k \in \mathcal{N}$. For compact presentation, we introduce $\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k \in \mathbb{R}^{\omega_k - 1}$ for each $k \in \mathcal{N}$. In particular,

- We denote by $\mathbf{i}_k^{\mathrm{Re}} \in \mathbb{R}^{\omega_k - 1}$ and $\mathbf{i}_k^{\mathrm{Im}} \in \mathbb{R}^{\omega_k - 1}$ the flow line variables i_{kn}^{Re} and i_{kn}^{Re}, respectively, for all $(k, n) \in \mathcal{L}$. We organize the vectors so that for all $j = 1, 2, ..., \omega_k$, the bus index corresponds to j-th component of $\mathbf{i}_k^{\mathrm{Re}}$ and $\mathbf{i}_k^{\mathrm{Im}}$ is same as the bus index corresponds to $(j+1)$-th component of $\mathbf{x}_k^{\mathrm{Re}}$ (or $\mathbf{x}^{\mathrm{Im}}$).

- We let $\hat{\mathbf{p}}_k \in \mathbb{R}^{\omega_k - 1}$ and $\hat{\mathbf{q}}_k \in \mathbb{R}^{\omega_k - 1}$ denote the flow line variables p_{kn} and q_{kn}, respectively, for all $(k, n) \in \mathcal{L}$. The order of the components is similar to that of $\mathbf{i}_k^{\mathrm{Re}}$ or $\mathbf{i}_k^{\mathrm{Im}}$.

To concisely present constraint (3.24n) and (3.24o), we require to define some more notations. Specifically, we define $\hat{\mathbf{C}}_k, \bar{\mathbf{C}}_k, \hat{\mathbf{D}}_k, \bar{\mathbf{D}}_k \in \mathbb{R}^{\omega_k - 1 \times \omega_k}$ that

emulate vectors $\mathbf{c}_{nm}$ and $\mathbf{d}_{nm}$ in (3.24n) and (3.24o), where

$$\bar{\mathbf{C}}_k = \begin{pmatrix} (\mathbf{g}_k)_2 & -(\mathbf{g}_k)_2 & 0 & \cdots & 0 \\ (\mathbf{g}_k)_3 & 0 & -(\mathbf{g}_k)_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (\mathbf{g}_k)_{\omega_k} & 0 & 0 & \cdots & -(\mathbf{g}_k)_{\omega_k} \end{pmatrix} \tag{3.39}$$

$$\hat{\mathbf{C}}_k = \begin{pmatrix} (\mathbf{b}_k)_2 & -(\mathbf{b}_k)_2 & 0 & \cdots & 0 \\ (\mathbf{b}_k)_3 & 0 & -(\mathbf{b}_k)_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (\mathbf{b}_k)_{\omega_k} & 0 & 0 & \cdots & -(\mathbf{b}_k)_{\omega_k} \end{pmatrix} \tag{3.40}$$

$$\bar{\mathbf{D}}_k = \begin{pmatrix} (\mathbf{b}_k)_2 & -(\mathbf{b}_k)_2 & 0 & \cdots & 0 \\ (\mathbf{b}_k)_3 & 0 & -(\mathbf{b}_k)_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (\mathbf{b}_k)_{\omega_k} & 0 & 0 & \cdots & -(\mathbf{b}_k)_{\omega_k} \end{pmatrix} \tag{3.41}$$

$$\hat{\mathbf{D}}_k = \begin{pmatrix} -(\mathbf{g}_k)_2 & (\mathbf{g}_k)_2 & 0 & \cdots & 0 \\ -(\mathbf{g}_k)_3 & 0 & (\mathbf{g}_k)_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -(\mathbf{g}_k)_{\omega_k} & 0 & 0 & \cdots & (\mathbf{g}_k)_{\omega_k} \end{pmatrix}. \tag{3.42}$$

Thus constraints (3.24n) and (3.24o) become

$$\mathbf{i}_k^{\mathrm{Re}} = \bar{\mathbf{C}}_k \mathbf{x}_k^{\mathrm{Re}} + \hat{\mathbf{C}}_k \mathbf{x}_k^{\mathrm{Im}}, k \in \mathcal{N} \tag{3.43}$$

$$\mathbf{i}_k^{\mathrm{Im}} = \bar{\mathbf{D}}_k \mathbf{x}_k^{\mathrm{Re}} + \hat{\mathbf{D}}_k \mathbf{x}_k^{\mathrm{Im}}, k \in \mathcal{N}. \tag{3.44}$$

Constraints (3.24p) and (3.24q) can be equivalently expressed as follows:

$$\hat{\mathbf{p}}_k = (\mathbf{x}_k^{\mathrm{Re}})_1 \mathbf{i}_k^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Im}})_1 \mathbf{i}_k^{\mathrm{Im}}, \qquad k \in \mathcal{N} \tag{3.45}$$

$$\hat{\mathbf{q}}_k = (\mathbf{x}_k^{\mathrm{Im}})_1 \mathbf{i}_k^{\mathrm{Re}} - (\mathbf{x}_k^{\mathrm{Re}})_1 \mathbf{i}_k^{\mathrm{Im}}, \qquad k \in \mathcal{N}. \tag{3.46}$$

For a more compact notation we define the functions

$$h_k^{\mathrm{CUR}} : \mathbb{R}^{4\omega_k - 2} \longrightarrow \mathbb{R}^{2(\omega_k - 1)} \tag{3.47}$$

$$h_k^{\mathrm{POW}} : \mathbb{R}^{4(\omega_k - 1)} \longrightarrow \mathbb{R}^{2(\omega_k - 1)} \tag{3.48}$$

such that:

$$h_k^{\mathrm{CUR}}(\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}) = \begin{pmatrix} \mathbf{i}_k^{\mathrm{Re}} - \left(\bar{\mathbf{C}}_k \mathbf{x}_k^{\mathrm{Re}} + \hat{\mathbf{C}}_k \mathbf{x}_k^{\mathrm{Im}} \right) \\ \mathbf{i}_k^{\mathrm{Im}} - \left(\bar{\mathbf{D}}_k \mathbf{x}_k^{\mathrm{Re}} + \hat{\mathbf{D}}_k \mathbf{x}_k^{\mathrm{Im}} \right) \end{pmatrix}. \tag{3.49}$$

and

$$h_k^{\mathrm{POW}}(\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k) = \begin{pmatrix} \hat{\mathbf{p}}_k - \left((\mathbf{x}_k^{\mathrm{Re}})_1 \mathbf{i}_k^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Im}})_1 \mathbf{i}_k^{\mathrm{Im}}\right) \\ \hat{\mathbf{q}}_k - \left((\mathbf{x}_k^{\mathrm{Im}})_1 \mathbf{i}_k^{\mathrm{Re}} - (\mathbf{x}_k^{\mathrm{Re}})_1 \mathbf{i}_k^{\mathrm{Im}}\right) \end{pmatrix}. \tag{3.50}$$

Then we can write constraints (3.24n) to (3.24q) equivalently as

$$h_k^{\mathrm{CUR}}(\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}) = 0, \qquad k \in \mathcal{N} \tag{3.51}$$

$$h_k^{\mathrm{POW}}(\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k, \bar{\mathbf{p}}_k, \bar{\mathbf{q}}_k) = 0, \qquad k \in \mathcal{N}. \tag{3.52}$$

For compact presentation and for the clarity of decomposition of constraints (3.24r)-(3.24t), we introduce $\mathbf{i}_k^{\max}, \mathbf{s}_k^{\max}, \mathbf{p}^{\max} \in \mathbb{R}^{\omega_k - 1}$. In particular,

- We denote by $\mathbf{i}_k^{\max} \in \mathbb{R}^{\omega_k - 1}$ the limit $i_{kn}^{\max}$ for all $(k, n) \in \mathcal{L}$. Remember that $i_{nm}^{\max} = i_{mn}^{\max}$. We organize the vector so that for all $j = 1, 2, ..., (\omega_k - 1)$, the bus index corresponds to j-th component of $\mathbf{i}_k^{\max}$ is same as the bus index corresponds to $(j+1)$-th component of $\mathbf{x}_k^{\mathrm{Re}}$ (or $\mathbf{x}_k^{\mathrm{Im}}$).

- We denote by $\mathbf{s}_k^{\max} \in \mathbb{R}^{\omega_k - 1}$ the limit $s_{kn}^{\max}$ for all $(k, n) \in \mathcal{L}$. Remember that $s_{nm}^{\max} = s_{mn}^{\max}$. The order of the components is similar to that of $\mathbf{i}_k^{\max}$.

- We denote by $\mathbf{p}_k^{\max} \in \mathbb{R}^{\omega_k - 1}$ the limit $p_{kn}^{\max}$ for all $(k, n) \in \mathcal{L}$. Remember that $p_{nm}^{\max} = p_{mn}^{\max}$. The order of the components is similar to that of $\mathbf{i}_k^{\max}$.

Then constraint (3.24r) can be written as

$$[\mathbf{i}_k^{\mathrm{Re}}]^2 + [\mathbf{i}_k^{\mathrm{Im}}]^2 \leq \mathbf{i}_k^{\max}, \qquad k \in \mathcal{N}, \tag{3.53}$$

where $[\cdot]^2$ is componentwise squaring. Similarly, constraint (3.24s) is given by

$$[\hat{\mathbf{p}}_k]^2 + [\hat{\mathbf{q}}_k]^2 \leq \mathbf{s}_k^{\max}, \qquad k \in \mathcal{N}. \tag{3.54}$$

Finally, constraint (3.24t) can be written as

$$|\hat{\mathbf{p}}_k| \leq \mathbf{p}_k^{\max}, \qquad k \in \mathcal{N}, \tag{3.55}$$

where $|\cdot|$ denotes absolute value. Note that each of these three constraints, (3.24r), (3.24s) and (3.24t), defines a convex set. Let us denote these set by

$\mathcal{I}_k^{\mathrm{FLOW}}, \mathcal{S}_k^{\mathrm{FLOW}}$ and $\mathcal{P}_k^{\mathrm{FLOW}}$ for each bus $k \in \mathcal{N}$. Then the constraints above are compactly given by

$$(\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}) \in \mathcal{I}_k^{\mathrm{FLOW}}, \qquad k \in \mathcal{N}, \tag{3.56}$$

$$(\hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k) \in \mathcal{S}_k^{\mathrm{FLOW}}, \qquad k \in \mathcal{N}, \tag{3.57}$$

$$\hat{\mathbf{p}}_k \in \mathcal{P}_k^{\mathrm{FLOW}}, \qquad k \in \mathcal{N}. \tag{3.58}$$

By replacing constraints (3.24h)-(3.24t) of problem (3.24) by (3.28), (3.29), (3.30), (3.31), (3.35), (3.37), (3.51), (3.52), (3.56), (3.57), and (3.58), we obtain the equivalent problem of the form (2.21) as

$$\begin{aligned}
\text{minimize} \quad & \sum_{k \in \mathcal{G}} f_k^{\mathrm{G}}(p_k^{\mathrm{G}}) \\
\text{subject to} \quad & p_k = p_k^{\mathrm{G}} - p_k^{\mathrm{L}}, && k \in \mathcal{G} \\
& q_k = q_k^{\mathrm{G}} - q_k^{\mathrm{L}}, && k \in \mathcal{G} \\
& p_k = -p_k^{\mathrm{L}}, && k \in \mathcal{N} \setminus \mathcal{G} \\
& q_k = -q_k^{\mathrm{L}}, && k \in \mathcal{N} \setminus \mathcal{G} \\
& p_k^{\mathrm{G,min}} \le p_k^{\mathrm{G}} \le p_k^{\mathrm{G,max}}, && k \in \mathcal{G} \\
& q_k^{\mathrm{G,min}} \le q_k^{\mathrm{G}} \le q_k^{\mathrm{G,max}}, && k \in \mathcal{G} \\
& i_k^{\mathrm{Re}} = \mathbf{g}_k^{\mathrm{T}} \mathbf{x}_k^{\mathrm{Re}} - \mathbf{b}_k^{\mathrm{T}} \mathbf{x}_k^{\mathrm{Im}}, && k \in \mathcal{N} \\
& i_k^{\mathrm{Im}} = \mathbf{b}_k^{\mathrm{T}} \mathbf{x}_k^{\mathrm{Re}} + \mathbf{g}_k^{\mathrm{T}} \mathbf{x}_k^{\mathrm{Im}}, && k \in \mathcal{N} \\
& p_k = (\mathbf{x}_k^{\mathrm{Re}})_1 i_k^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Im}})_1 i_k^{\mathrm{Im}}, && k \in \mathcal{N} \\
& q_k = (\mathbf{x}_k^{\mathrm{Im}})_1 i_k^{\mathrm{Re}} - (\mathbf{x}_k^{\mathrm{Re}})_1 i_k^{\mathrm{Im}}, && k \in \mathcal{N} \\
& h_k^{\mathrm{CUR}}(\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}) = 0, && k \in \mathcal{N} \\
& h_k^{\mathrm{POW}}(\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k) = 0, && k \in \mathcal{N} \\
& (\mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}) \in \mathcal{X}_k, && k \in \mathcal{N} \\
& (\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}) \in \mathcal{I}_k^{\mathrm{FLOW}}, && k \in \mathcal{N} \\
& (\hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k) \in \mathcal{S}_k^{\mathrm{FLOW}}, && k \in \mathcal{N} \\
& \hat{\mathbf{p}}_k \in \mathcal{P}_k^{\mathrm{FLOW}}, && k \in \mathcal{N} \\
& \mathbf{x}_k^{\mathrm{Re}} = \mathbf{E}_k \mathbf{z}^{\mathrm{Re}}, && k \in \mathcal{N} \\
& \mathbf{x}_k^{\mathrm{Im}} = \mathbf{E}_k \mathbf{z}^{\mathrm{Im}}, && k \in \mathcal{N},
\end{aligned} \tag{3.59}$$

where the variables are $\mathbf{x}_k^{\mathrm{Re}}$, $\mathbf{x}_k^{\mathrm{Im}}$,i_k^{Re}, i_k^{Im}, p_k, q_k, $\mathbf{z}^{\mathrm{Re}}$, $\mathbf{z}^{\mathrm{Im}}$, $\mathbf{i}_k^{\mathrm{Re}}$,$\mathbf{i}_k^{\mathrm{Im}}$, $\hat{\mathbf{p}}_k$, and $\hat{\mathbf{q}}_k$ for $k \in \mathcal{N}$ and ${p_k}^{\mathrm{G}}, q_k^{\mathrm{G}}$ for $k \in \mathcal{G}$.

Chapter 4

Solution methods

Because the problem is non-convex, it is not surprising that the optimal value of problem (3.59) is not achieved in general by employing local methods. However, compared with exponentially complex global methods, local methods are desirable in practice due to low complexity, even at the expense of global optimality. In this chapter we design a distributed solution approach to address problem (3.59). The method is not a global method, and therefore there is no guarantee that it achieves the optimal solution always. Nevertheless, the proposed algorithm can easily be scalable and, as a result, can be deployed in practice. Here, we capitalize on the distributed algorithms presented in section 2.1. In particular, we use ADMM as basis for our algorithm development, especially to achieve fast convergence properties, compared to the dual decomposition method [4]. In addition, the ADMM method is promising in the sense that it works on many non-convex problems as a good heuristic as discussed in [4, section 9]. In this chapter we present how we implemented the steps of the ADMM method for our problem (3.59). We start by presenting the outline of the algorithm in section 4.1, followed by a discussion on how the individual steps of the ADMM are implemented in sections 4.2 and 4.3. Thereafter, we discuss how the algorithm can be implemented in a distributed manner so that communication is limited to buses that are connected by a flow line. Finally, in section 4.5 we discuss feasibility issues of the proposed method.

4.1 Outline of the algorithm

Note that our problem (3.59) is in the form of general consensus problem (2.21). Therefore, the ADMM algorithm presented there can readily be applied. To do this, we start by defining some notations for simplicity. In particular, we stack the public variables, the matrices $\mathbf{E}_k$ for each bus k, and the net variables as follows:

$$\mathbf{x}_k = (\mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}) \qquad k \in \mathcal{N} \tag{4.1}$$

$$\bar{\mathbf{E}}_k = [\mathbf{E}_k^{\mathrm{T}} \quad \mathbf{E}_k^{\mathrm{T}}]^{\mathrm{T}} \qquad k \in \mathcal{N} \tag{4.2}$$

$$\mathbf{z} = (\mathbf{z}^{\mathrm{Re}}, \mathbf{z}^{\mathrm{Im}}). \tag{4.3}$$

We let $\mathbf{x}_k^{(n)}, \mathbf{y}_k^{(n)}$ for $k \in \mathcal{N}$, and $\mathbf{z}^{(n)}$ be the public, dual and the net variables respectively after n iterations. Moreover, $\mathbf{y}_k^{(0)}$ for $k \in \mathcal{N}$ and $\mathbf{z}^{(0)}$ are the initial values of the public variables and the net variables, respectively. Then the steps of the ADMM algorithm are the following:

- $\mathbf{x}_k$-Update: for $k \in \mathcal{G}$ we set $\mathbf{x}_k^{(n+1)}$ as the solution to the following problem:

$$\begin{aligned}
\text{minimize} \quad & f_k^{\mathrm{G}}(p_k^{\mathrm{G}}) + (\mathbf{y}_k^{(n)})^{\mathrm{T}}(\mathbf{x}_k - \bar{\mathbf{E}}_k\mathbf{z}^{(n)}) + (\rho/2)||\mathbf{x}_k - \bar{\mathbf{E}}_k\mathbf{z}^{(n)}||_2^2 \\
\text{subject to} \quad & p_k = p_k^{\mathrm{G}} - p_k^{\mathrm{L}} \\
& q_k = q_k^{\mathrm{G}} - q_k^{\mathrm{L}} \\
& p_k^{\mathrm{G,min}} \le p_k^{\mathrm{G}} \le p_k^{\mathrm{G,min}} \\
& q_k^{\mathrm{G,min}} \le q_k^{\mathrm{G}} \le q_k^{\mathrm{G,min}} \\
& i_k^{\mathrm{Re}} = \mathbf{g}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Re}} - \mathbf{b}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Im}} \\
& i_k^{\mathrm{Im}} = \mathbf{b}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Re}} + \mathbf{g}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Im}} \\
& p_k = (\mathbf{x}_k^{\mathrm{Re}})_1 i_k^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Im}})_1 i_k^{\mathrm{Im}} \\
& q_k = (\mathbf{x}_k^{\mathrm{Im}})_1 i_k^{\mathrm{Re}} - (\mathbf{x}_k^{\mathrm{Re}})_1 i_k^{\mathrm{Im}} \\
& h_k^{\mathrm{CUR}}(\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}) = 0 \\
& h_k^{\mathrm{POW}}(\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k) = 0 \\
& (\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}) \in \mathcal{I}_k^{\mathrm{FLOW}} \\
& (\hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k) \in \mathcal{S}_k^{\mathrm{FLOW}} \\
& \hat{\mathbf{p}}_k \in \mathcal{P}_k^{\mathrm{FLOW}} \\
& (\mathbf{x}^{\mathrm{Re}}, \mathbf{x}^{\mathrm{Im}}) \in \mathcal{X}_k,
\end{aligned} \tag{4.4}$$

where the variables are $p_k^{\mathrm{G}}, q_k^{\mathrm{G}}, p_k, q_k, i_k^{\mathrm{Re}}, i_k^{\mathrm{Im}}, \mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}, \mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k$.

- $\mathbf{x}_k$-Update: for $k \in \mathcal{N} \setminus \mathcal{G}$ we set $\mathbf{x}_k^{(n+1)}$ as the solution to the following problem:

$$\begin{array}{ll} \text{minimize} & (\mathbf{y}_k^{(n)})^{\mathrm{T}}(\mathbf{x}_k - \bar{\mathbf{E}}_k \mathbf{z}^{(n)}) + (\rho/2)||\mathbf{x}_k - \bar{\mathbf{E}}_k \mathbf{z}^{(n)}||_2^2 \\ \text{subject to} & p_k = -p_k^{\mathrm{L}} \\ & q_k = -q_k^{\mathrm{L}} \\ & i_k^{\mathrm{Re}} = \mathbf{g}_k^{\mathrm{T}} \mathbf{x}_k^{\mathrm{Re}} - \mathbf{b}_k^{\mathrm{T}} \mathbf{x}_k^{\mathrm{Im}} \\ & i_k^{\mathrm{Im}} = \mathbf{b}_k^{\mathrm{T}} \mathbf{x}_k^{\mathrm{Re}} + \mathbf{g}_k^{\mathrm{T}} \mathbf{x}_k^{\mathrm{Im}} \\ & p_k = (\mathbf{x}_k^{\mathrm{Re}})_1 i_k^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Im}})_1 i_k^{\mathrm{Im}} \\ & q_k = (\mathbf{x}_k^{\mathrm{Im}})_1 i_k^{\mathrm{Re}} - (\mathbf{x}_k^{\mathrm{Re}})_1 i_k^{\mathrm{Im}} \\ & h_k^{\mathrm{CUR}}(\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}) = 0 \\ & h_k^{\mathrm{POW}}(\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k, \bar{\mathbf{p}}_k, \bar{\mathbf{q}}_k) = 0 \\ & (\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}) \in \mathcal{I}_k^{\mathrm{FLOW}} \\ & (\hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k), (\bar{\mathbf{p}}_k, \bar{\mathbf{q}}_k) \in \mathcal{S}_k^{\mathrm{FLOW}} \\ & \hat{\mathbf{p}}_k, \bar{\mathbf{p}}_k \in \mathcal{P}_k^{\mathrm{FLOW}} \\ & (\mathbf{x}^{\mathrm{Re}}, \mathbf{x}^{\mathrm{Im}}) \in \mathcal{X}_k, \end{array} \tag{4.5}$$

  where the variables are $p_k, q_k, i_k^{\mathrm{Re}}, i_k^{\mathrm{Im}}, \mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}, \mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k$.

- $\mathbf{z}$-Update: we set $\mathbf{z}^{(n+1)}$ as the solution to the following problem:

$$\text{minimize} \quad \sum_{k=1}^{N} (\mathbf{y}_k^{(n)})^{\mathrm{T}}(\mathbf{x}_k^{(n+1)} - \bar{\mathbf{E}}_k \mathbf{z}) + (\rho/2) \sum_{k=1}^{N} ||\mathbf{x}_k^{(n+1)} - \bar{\mathbf{E}}_k \mathbf{z}||_2^2, \tag{4.6}$$

  where the variable is $\mathbf{z}$.

- $\mathbf{y}_k$-Update: for $k \in \mathcal{N}$ we set:

$$\mathbf{y}_k^{(n+1)} = \mathbf{y}_k^{(n)} + \rho(\mathbf{x}_k^{(n+1)} - \bar{\mathbf{E}}_k \mathbf{z}^{(n+1)}) \ . \tag{4.7}$$

In the sequel, we discuss in detail variable updates (4.4), (4.5), (4.6), and (4.7).

4.2 The Subproblems: x-Update

In this section we present the solution method we use to address problem (4.4) and (4.5). In particular we will only consider problem (4.4) since problem (4.5) is a special case of problem (4.4). Note that problem (4.4) inherits the non-convexity properties from (3.59). Therefore, convex optimization techniques do not apply directly to find a solution. However, though the optimality is not guaranteed, our proposed method capitalizes on approximations together with convex optimization techniques to design a good heuristics which is efficient compared to global methods. In particular, we adopt the approximations used in [18], in the context of a centralized OPF solution approach, as basis for our method. Given $\mathbf{z}^{(n)}$ and $\mathbf{y}_k^{(n)}$ from the previous iteration, the optimization problem we have to solve is given by

$$
\begin{aligned}
\text{minimize} \quad & f_k^{\mathrm{G}}(p_k^{\mathrm{G}}) + (\mathbf{y}_k^{(n)})^{\mathrm{T}}(\mathbf{x}_k - \bar{\mathbf{E}}_k\mathbf{z}^{(n)}) + (\rho/2)||\mathbf{x}_k - \bar{\mathbf{E}}_k\mathbf{z}^{(n)}||_2^2 && \text{(4.8a)}\\
\text{subject to} \quad & p_k = p_k^{\mathrm{G}} - p_k^{\mathrm{L}} && \text{(4.8b)}\\
& q_k = q_k^{\mathrm{G}} - q_k^{\mathrm{L}} && \text{(4.8c)}\\
& p_k^{\mathrm{G,min}} \leq p_k^{\mathrm{G}} \leq p_k^{\mathrm{G,min}} && \text{(4.8d)}\\
& q_k^{\mathrm{G,min}} \leq q_k^{\mathrm{G}} \leq q_k^{\mathrm{G,min}} && \text{(4.8e)}\\
& i_k^{\mathrm{Re}} = \mathbf{g}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Re}} - \mathbf{b}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Im}} && \text{(4.8f)}\\
& i_k^{\mathrm{Im}} = \mathbf{b}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Re}} + \mathbf{g}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Im}} && \text{(4.8g)}\\
& p_k = (\mathbf{x}_k^{\mathrm{Re}})_1 i_k^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Im}})_1 i_k^{\mathrm{Im}} && \text{(4.8h)}\\
& q_k = (\mathbf{x}_k^{\mathrm{Im}})_1 i_k^{\mathrm{Re}} - (\mathbf{x}_k^{\mathrm{Re}})_1 i_k^{\mathrm{Im}} && \text{(4.8i)}\\
& \mathbf{i}_k^{\mathrm{Re}} = \bar{\mathbf{C}}_k\mathbf{x}_k^{\mathrm{Re}} + \hat{\mathbf{C}}_k\mathbf{x}_k^{\mathrm{Im}} && \text{(4.8j)}\\
& \mathbf{i}_k^{\mathrm{Im}} = \bar{\mathbf{D}}_k\mathbf{x}_k^{\mathrm{Re}} + \hat{\mathbf{D}}_k\mathbf{x}_k^{\mathrm{Im}} && \text{(4.8k)}\\
& \hat{\mathbf{p}}_k = \left((\mathbf{x}_k^{\mathrm{Re}})_1\mathbf{i}_k^{\mathrm{Re}} - (\mathbf{x}_k^{\mathrm{Im}})_1\mathbf{i}_k^{\mathrm{Im}}\right) && \text{(4.8l)}\\
& \hat{\mathbf{q}}_k = \left((\mathbf{x}_k^{\mathrm{Im}})_1\mathbf{i}_k^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Re}})_1\mathbf{i}_k^{\mathrm{Im}}\right) && \text{(4.8m)}\\
& (\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}) \in \mathcal{I}_k^{\mathrm{FLOW}} && \text{(4.8n)}\\
& (\hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k) \in \mathcal{S}_k^{\mathrm{FLOW}} && \text{(4.8o)}\\
& \hat{\mathbf{p}}_k \in \mathcal{P}_k^{\mathrm{FLOW}} && \text{(4.8p)}\\
& (\mathbf{x}_k^{\mathrm{Re}})_l^2 + (\mathbf{x}_k^{\mathrm{Im}})_l^2 \leq (\mathbf{v}_k^{\max})_l^2, \qquad l \in \Omega_k && \text{(4.8q)}\\
& (\mathbf{x}_k^{\mathrm{Re}})_l^2 + (\mathbf{x}_k^{\mathrm{Im}})_l^2 \geq (\mathbf{v}_k^{\min})_l^2, \qquad l \in \Omega_k && \text{(4.8r)}
\end{aligned}
$$

where the variables are $p_k^{\mathrm{G}}, q_k^{\mathrm{G}}, p_k, q_k, i_k^{\mathrm{Re}}, i_k^{\mathrm{Im}}, \mathbf{x}_k^{\mathrm{Re}}, \mathbf{x}_k^{\mathrm{Im}}, \mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}, \hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k$. As noted earlier, the problem is non-convex, and therefore we trade optimality in the solution method for an improvement in the efficiency of the method so that it is desirable in practice. To do this we rely on convex approximations of the non-convex constraints.

Most of the constraints of the problem are convex. However, constraints (4.8h), (4.8i), (4.8l), (4.8m), and (4.8r) are non-convex. In the case of non-convex constraints (4.8h) and (4.8i) we use the linear Taylor approximation in the point

$$\big((\mathbf{x}_k^{\mathrm{Re}})_1, (\mathbf{x}_k^{\mathrm{Im}})_1, i_k^{\mathrm{Re}}, i_k^{\mathrm{Im}})\big) = (\alpha^{\mathrm{Re}}, \alpha^{\mathrm{Im}}, w^{\mathrm{Re}}, w^{\mathrm{Im}}). \tag{4.9}$$

Then the approximated versions of constraints (4.8h) and (4.8i) are given by

$$p_k = \alpha^{\mathrm{Re}} i_k^{\mathrm{Re}} + \alpha^{\mathrm{Im}} i_k^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Re}})_1 w^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Im}})_1 w^{\mathrm{Im}} - (\alpha^{\mathrm{Re}} w^{\mathrm{Re}} + \alpha^{\mathrm{Im}} w^{\mathrm{Im}}) \tag{4.10}$$

$$q_k = \alpha^{\mathrm{Im}} i_k^{\mathrm{Re}} - \alpha^{\mathrm{Re}} i_k^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Im}})_1 w^{\mathrm{Re}} - (\mathbf{x}_k^{\mathrm{Re}})_1 w^{\mathrm{Im}} - (\alpha^{\mathrm{Im}} w^{\mathrm{Re}} - \alpha^{\mathrm{Re}} w^{\mathrm{Im}}). \tag{4.11}$$

We treat the non-convex constraints (4.8l) and (4.8m) in similar fashion and use the linear Taylor approximation in the point

$$\big((\mathbf{x}_k^{\mathrm{Re}})_1, (\mathbf{x}_k^{\mathrm{Im}})_1, i_k^{\mathrm{Re}}, i_k^{\mathrm{Im}})\big) = (\alpha^{\mathrm{Re}}, \alpha^{\mathrm{Im}}, \mathbf{w}^{\mathrm{Re}}, \mathbf{w}^{\mathrm{Im}}). \tag{4.12}$$

Then the approximated versions of constraints (4.8l) and (4.8m) are given by

$$\hat{\mathbf{p}}_k = \alpha^{\mathrm{Re}} i_k^{\mathrm{Re}} + \alpha^{\mathrm{Im}} i_k^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Re}})_1 \mathbf{w}^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Im}})_1 \mathbf{w}^{\mathrm{Im}} - (\alpha^{\mathrm{Re}} \mathbf{w}^{\mathrm{Re}} + \alpha^{\mathrm{Im}} \mathbf{w}^{\mathrm{Im}}) \tag{4.13}$$

$$\hat{\mathbf{q}}_k = \alpha^{\mathrm{Im}} i_k^{\mathrm{Re}} - \alpha^{\mathrm{Re}} i_k^{\mathrm{Re}} + (\mathbf{x}_k^{\mathrm{Im}})_1 \mathbf{w}^{\mathrm{Re}} - (\mathbf{x}_k^{\mathrm{Re}})_1 \mathbf{w}^{\mathrm{Im}} - (\alpha^{\mathrm{Im}} \mathbf{w}^{\mathrm{Re}} - \alpha^{\mathrm{Re}} \mathbf{w}^{\mathrm{Im}}). \tag{4.14}$$

In the case of non-convex constraint (4.8r), we note that for any $l \in \{1, 2, ..., \omega_k\}$, the values of $(\mathbf{x}_k^{\mathrm{Re}})_l$ and $(\mathbf{x}^{\mathrm{Im}})_l$ represents a donut, see Figure 4.1a. In other words, the 2-dimensional set

$$\mathcal{X}_k^l = \left\{ ((\mathbf{x}_k^{\mathrm{Re}})_l, (\mathbf{x}^{\mathrm{Im}})_l) \,\middle|\, (\mathbf{v}_k^{\min})_l^2 \leq (\mathbf{x}_k^{\mathrm{Re}})_l^2 + (\mathbf{x}^{\mathrm{Im}})_l^2 \leq (\mathbf{v}_k^{\max})_l^2 \right\} \tag{4.15}$$

is a donut. Therefore, we consider a convex subset of $\mathcal{X}_k^l$ instead, which we denote by $\mathcal{XX}_k^l$, see Figure 4.1b. Specifically, $\mathcal{XX}_k^l$ is the intersection of $\mathcal{X}_k^n$ and the halfspace

$$\{\big((\mathbf{x}_k^{\mathrm{Re}})_l, (\mathbf{x}_k^{\mathrm{Im}})_l\big) \,|\, a_k^l (\mathbf{x}_k^{\mathrm{Re}})_l + b_k^n (\mathbf{x}_k^{\mathrm{Im}})_n \geq c_k^n\} \tag{4.16}$$

where

$$a_k^l = \text{sign}\left((\mathbf{E}_k\mathbf{z}^{\text{Re}})_l\right)\left(\frac{(\mathbf{v}_k^{\min})_l^2}{1+\left((\mathbf{E}_k\mathbf{z}^{\text{Im}})_l/(\mathbf{E}_k\mathbf{z}^{\text{Re}})_l\right)^2}\right)^{1/2} \tag{4.17}$$

$$b_k^l = \left(\frac{(\mathbf{E}_k\mathbf{z}^{\text{Im}})_l}{(\mathbf{E}_k\mathbf{z}^{\text{Re}})_l}\right)a_k^l \tag{4.18}$$

$$c_k^l = (\mathbf{v}_k^{\min})_l^2 \tag{4.19}$$

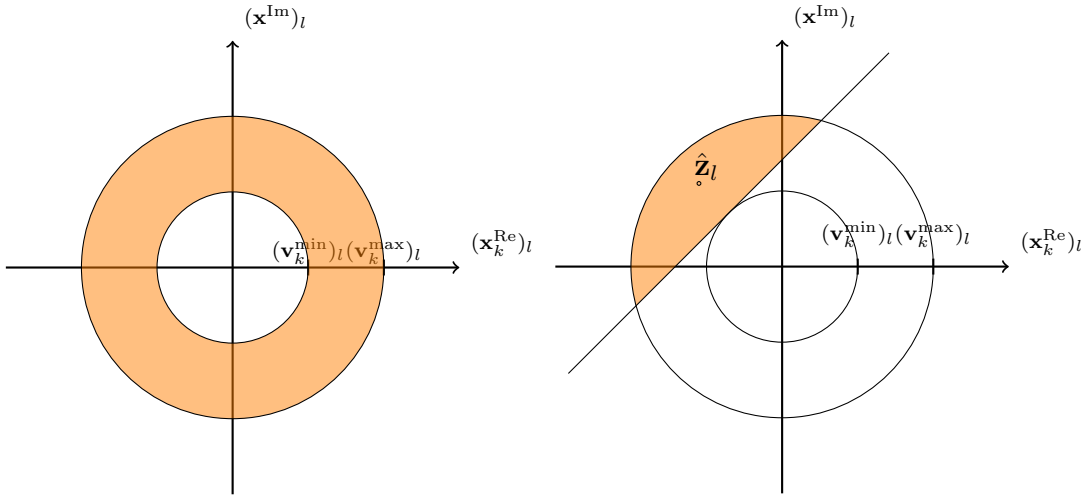
if $(\mathbf{E}_k\mathbf{z}^{\text{Re}})_l \neq 0$, where $\text{sign}(r)$ is the sign of the real number r, and

$$a_k^l = 0 \tag{4.20}$$

$$b_k^l = \text{sign}\left((\mathbf{E}_k\mathbf{z}^{\text{Im}})_l\right) \tag{4.21}$$

$$c_k^l = (\mathbf{v}_k^{\min})_l \tag{4.22}$$

if $(\mathbf{E}_k\mathbf{z}^{\text{Re}})_l = 0$. Note that the set $\mathcal{XX}_k^l$ is uniquely determined by the net variables $(\mathbf{E}_k\mathbf{z}^{\text{Re}})_l$ and $(\mathbf{E}_k\mathbf{z}^{\text{Im}})_l$. Therefore the set $\mathcal{XX}_k^l$ is identical for all the local copies of $(\mathbf{E}_k\mathbf{z}^{\text{Re}})_l$ and $(\mathbf{E}_k\mathbf{z}^{\text{Im}})_l$.



(a) The feasibility set for the variables $(\mathbf{x}_k^{\text{Re}})_l$ and $(\mathbf{x}^{\text{Im}})_l$.

(b) Here we see the $\mathcal{XX}_k^l$. Here $\hat{\mathbf{z}}_l = (\mathbf{E}_k\mathbf{z}^{\text{Re}})_l + j(\mathbf{E}_k\mathbf{z}^{\text{Im}})_l$.

Figure 4.1

By replacing constraints (4.8h), (4.8i), (4.8l), and (4.8m) by (4.10), (4.11), (4.13), and (4.14) and by replacing (4.8r) and (4.8q) by

$$((\mathbf{x}_k^{\text{Re}})_l, (\mathbf{x}_k)_l^{\text{Im}})) \in \mathcal{XX}_k^l \qquad l \in \Omega_k \tag{4.23}$$

we obtain the approximated sub problem

$$
\begin{aligned}
\text{minimize} \quad & f_k^{\mathrm{G}}(p_k^{\mathrm{G}}) + (\mathbf{y}_k^{(n)})^{\mathrm{T}}(\mathbf{x}_k - \bar{\mathbf{E}}_k\mathbf{z}^{(n)}) + (\rho/2)||\mathbf{x}_k - \bar{\mathbf{E}}_k\mathbf{z}^{(n)}||_2^2 && \text{(4.24a)}\\
\text{subject to} \quad & p_k = p_k^{\mathrm{G}} - p_k^{\mathrm{L}} && \text{(4.24b)}\\
& q_k = q_k^{\mathrm{G}} - q_k^{\mathrm{L}} && \text{(4.24c)}\\
& p_k^{\mathrm{G,min}} \le p_k^{\mathrm{G}} \le p_k^{\mathrm{G,min}} && \text{(4.24d)}\\
& q_k^{\mathrm{G,min}} \le q_k^{\mathrm{G}} \le q_k^{\mathrm{G,min}} && \text{(4.24e)}\\
& i_k^{\mathrm{Re}} = \mathbf{g}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Re}} - \mathbf{b}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Im}} && \text{(4.24f)}\\
& i_k^{\mathrm{Im}} = \mathbf{b}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Re}} + \mathbf{g}_k^{\mathrm{T}}\mathbf{x}_k^{\mathrm{Im}} && \text{(4.24g)}\\
& \text{Constraint 4.10} && \text{(4.24h)}\\
& \text{Constraint 4.11} && \text{(4.24i)}\\
& \mathbf{i}_k^{\mathrm{Re}} = \bar{\mathbf{C}}_k\mathbf{x}_k^{\mathrm{Re}} + \hat{\mathbf{C}}_k\mathbf{x}_k^{\mathrm{Im}} && \text{(4.24j)}\\
& \mathbf{i}_k^{\mathrm{Im}} = \bar{\mathbf{D}}_k\mathbf{x}_k^{\mathrm{Re}} + \hat{\mathbf{D}}_k\mathbf{x}_k^{\mathrm{Im}} && \text{(4.24k)}\\
& \text{Constraint 4.13} && \text{(4.24l)}\\
& \text{Constraint 4.14} && \text{(4.24m)}\\
& (\mathbf{i}_k^{\mathrm{Re}}, \mathbf{i}_k^{\mathrm{Im}}) \in \mathcal{I}_k^{\mathrm{FLOW}} && \text{(4.24n)}\\
& (\hat{\mathbf{p}}_k, \hat{\mathbf{q}}_k) \in \mathcal{S}_k^{\mathrm{FLOW}} && \text{(4.24o)}\\
& \hat{\mathbf{p}}_k \in \mathcal{P}_k^{\mathrm{FLOW}} && \text{(4.24p)}\\
& ((\mathbf{x}_k^{\mathrm{Re}})_l, (\mathbf{x}_k)_l^{\mathrm{Im}})) \in \mathcal{XX}_k^l, \qquad l \in \Omega_k, && \text{(4.24q)}
\end{aligned}
$$

where the variables are p_k^{G}, q_k^{G}, p_k, q_k, i_k^{Re}, i_k^{Im}, $\mathbf{x}_k^{\mathrm{Re}}$, $\mathbf{x}_k^{\mathrm{Im}}$, $\mathbf{i}_k^{\mathrm{Re}}$, $\mathbf{i}_k^{\mathrm{Im}}$, $\hat{\mathbf{p}}_k$, and $\hat{\mathbf{q}}_k$. This problem is convex and therefore efficiently solved by using convex optimization techniques. Now we can use the following iterative process to cope with problem (4.8). Let $(\mathbf{x}_k^{\mathrm{Re}})^{(r)}$ and $(\mathbf{x}_k^{\mathrm{Im}})^{(r)}$ be the variables $\mathbf{x}_k^{\mathrm{Re}}$ and $\mathbf{x}_k^{\mathrm{Im}}$, respectively, at iteration r and $(\mathbf{x}_k^{\mathrm{Re}})^{(0)}$ and $(\mathbf{x}_k^{\mathrm{Re}})^{(0)}$ be initial value. Then we adjust constraints (4.10), (4.11), (4.13), (4.14), by setting

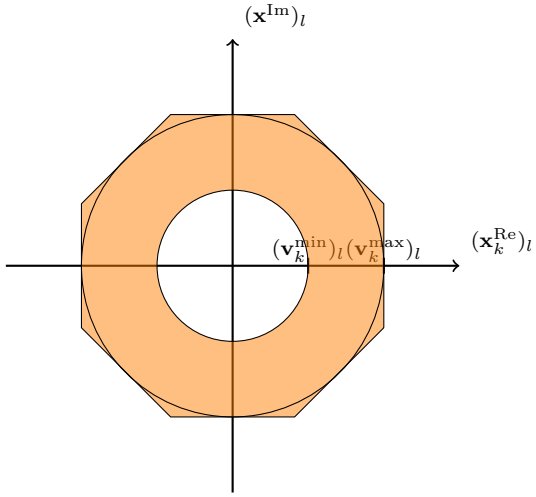
$$
\begin{aligned}
\alpha^{\mathrm{Re}} &= (\mathbf{x}_k^{\mathrm{Re}})_1^{(r)} && \text{(4.25)}\\
\alpha^{\mathrm{Im}} &= (\mathbf{x}_k^{\mathrm{Re}})_1^{(r)} && \text{(4.26)}\\
w^{\mathrm{Re}} &= \mathbf{g}_k^{\mathrm{T}}(\mathbf{x}_k^{\mathrm{Re}})^{(r)} - \mathbf{b}_k^{\mathrm{T}}(\mathbf{x}_k^{\mathrm{Im}})^{(r)} && \text{(4.27)}\\
w^{\mathrm{Im}} &= \mathbf{b}_k^{\mathrm{T}}(\mathbf{x}_k^{\mathrm{Re}})^{(r)} + \mathbf{g}_k^{\mathrm{T}}(\mathbf{x}_k^{\mathrm{Im}})^{(r)} && \text{(4.28)}\\
\mathbf{w}^{\mathrm{Re}} &= \bar{\mathbf{C}}_k(\mathbf{x}_k^{\mathrm{Re}})^{(r)} + \hat{\mathbf{C}}_k(\mathbf{x}_k^{\mathrm{Im}})^{(r)} && \text{(4.29)}\\
\mathbf{w}^{\mathrm{Im}} &= \bar{\mathbf{D}}_k(\mathbf{x}_k^{\mathrm{Re}})^{(r)} + \hat{\mathbf{D}}_k(\mathbf{x}_k^{\mathrm{Im}})^{(r)}. && \text{(4.30)}
\end{aligned}
$$

Then we set $(\mathbf{x}_k^{\mathrm{Re}})^{(r+1)}$ and $(\mathbf{x}_k^{\mathrm{Re}})^{(r+1)}$ as the solution to problem (4.24).

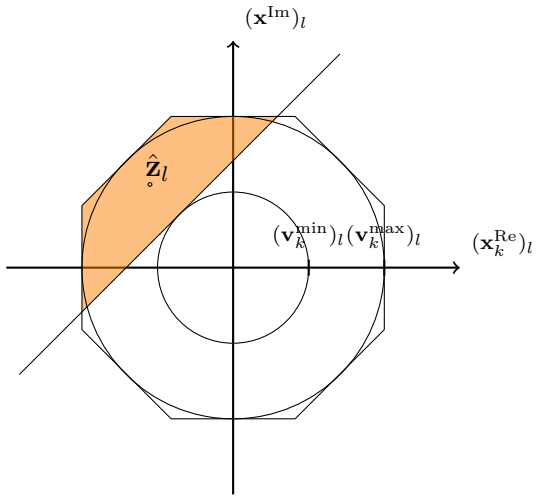
Note that the nonlinear boundary of the set $\mathcal{XX}_n^l$ can be approximated by affine functions yielding in a quadratic optimization problem. In which case, sophisticated QP solving tools can be applied to obtain a solution to 4.24. To mathematically express the associated problem, we denote by $\mathcal{XXX}_k^l$ the convex set constructed by affine approximations of the nonlinear boundary of $\mathcal{XX}_k^l$. To do this, we simply approximate the exterior boundary of the donut $\mathcal{X}_k^n$ by an equilateral octagon, see figures 4.2a and 4.2b. The linearized version of constraint (4.24q) is then

$$((\mathbf{x}_k^{\mathrm{Re}})_l, (\mathbf{x}_k)_l^{\mathrm{Im}})) \in \mathcal{XXX}_k^l, \qquad l \in \Omega_k. \tag{4.31}$$

Now we simply solve problem 4.24 but use constraint (4.31) instead of (4.24q). If the yielding optimization problem returns a point $((\mathbf{x}_k^{\mathrm{Re}})_l, (\mathbf{x}_k)_l^{\mathrm{Im}}))$ such that $((\mathbf{x}_k^{\mathrm{Re}})_l, (\mathbf{x}_k)_l^{\mathrm{Im}})) \in \mathcal{XXX}_k^l$ but $((\mathbf{x}_k^{\mathrm{Re}})_l, (\mathbf{x}_k)_l^{\mathrm{Im}})) \notin \mathcal{XX}_k^l$ then we simply update the set $\mathcal{XXX}_k^l$ by adding a new linear constraint on exterior boundary of the donut so that $((\mathbf{x}_k^{\mathrm{Re}})_l, (\mathbf{x}_k)_l^{\mathrm{Im}})) \notin \mathcal{XXX}_k^l$ and solve again. Therefore, the point that is eventually returned will be in the set $\mathcal{XX}_k^l$. The same approach can be applied to the convex constraints (4.24n) and (4.24p).



(a) The exterior boundary of the donut approximated by an octagon.



(b) Here we see the the inital set $\mathcal{XXX}_k^l$. Here $\hat{\mathbf{z}}_l = (\mathbf{E}_k \mathbf{z}^{\mathrm{Re}})_l + j(\mathbf{E}_k \mathbf{z}^{\mathrm{Im}})_l$.

Figure 4.2

4.3 The z-Update and y-Update

In this section we present how the **z**- and **y**-Updates are performed. Let us start by discussing the **z**-Update. From section 4.1 we have that $\mathbf{z}^{(n+1)}$ is the solution to

$$\text{minimize} \quad \sum_{k=1}^{N} (\mathbf{y}_k^{(n)})^{\mathrm{T}} (\mathbf{x}_k^{(n+1)} - \bar{\mathbf{E}}_k \mathbf{z}) + (\rho/2) \sum_{k=1}^{N} ||\mathbf{x}_k^{(n+1)} - \bar{\mathbf{E}}_k \mathbf{z}||_2^2, \tag{4.32}$$

where the variable is **z**. As noted in section 2.1.3 an analytic solution can be obtained resulting in

$$\begin{aligned} \mathbf{z}^{(n+1)} &= \left(\sum_{k=1}^{N} \bar{\mathbf{E}}_k^{\mathrm{T}} \bar{\mathbf{E}}_k \right)^{-1} \sum_{k=1}^{N} \bar{\mathbf{E}}_k^{\mathrm{T}} \mathbf{x}_k^{(n+1)} \\ &= \operatorname{diag}\left(\omega_1^{-1}, \cdots, \omega_N^{-1}, \omega_1^{-1}, \cdots, \omega_N^{-1}\right) \sum_{k=1}^{N} \bar{\mathbf{E}}_k^{\mathrm{T}} \mathbf{x}_k^{(n+1)}. \end{aligned} \tag{4.33}$$

This is simply the average of the public variables of each net. Therefore, the **z**-Update of each net variable boils down to collecting the public variables associated to the net a and taking averages.

The **y**-Update can now readily be obtained from

$$\mathbf{y}_k^{(n+1)} = \mathbf{y}_k^{(n)} + \rho(\mathbf{x}_k^{(n+1)} - \bar{\mathbf{E}}_k \mathbf{z}^{(n+1)}) \ . \tag{4.34}$$

After the net variables have been computed the **y**-Update can then be performed in parallel on each bus.

4.4 Distributed implementation

The algorithm can essentially be broken down to three steps, **x**-, **z**-, and **y**-Updates. Let us examine how these steps can be accomplished in a distributed fashion where message passing is limited to buses that are connected by a flow line.

First each bus updates its public variables by solving a a non-convex optimization problem, this is the **x**-Update. In the next step of the algorithm, the **z**-Update, the net variables are updated by taking the average of all the public variables that account for a common net. Recall that the net variables

$(\mathbf{z}^{\mathrm{Re}})_l$ and $(\mathbf{z}^{\mathrm{Im}})_l$ represent the voltage of bus l. Therefore the only bus that is connected by a flow line to all the buses that maintain a local copy of $(\mathbf{z}^{\mathrm{Re}})_l$ and $(\mathbf{z}^{\mathrm{Im}})_l$ is bus l. Therefore, to avoid communication between buses that are not connected by a flow line only bus l performs the $\mathbf{z}$-Update of $(\mathbf{z}^{\mathrm{Re}})_l$ and $(\mathbf{z}^{\mathrm{Im}})_l$. To do the averaging needed for the $\mathbf{z}$-Update each bus simply collects the public variables that are a copies of its voltage from its neighbors. After the bus has performed its $\mathbf{z}$-update it can broadcasts it to all of its neighbors. Then each bus has enough information to perform the $\mathbf{y}$-update which can be performed in parallel by the subsystems.

4.5 Feasibility Issues

The OPF problem is non-convex and therefore we can anticipate that the proposed ADMM algorithm returns an infeasible point. However, numerical experiences suggests that the proposed algorithm always returns a feasible solution with a good accuracy, see chapter 6. We have no analytical explanations of these promising experiences and leave further theoretical analysis of the method for a future research. In this section we roughly examine some aspects of those issues.

The relation between the power injections at bus and the voltages is quadratic, in particular for bus k we have

$$p_k + jq_k = ((\mathbf{x}_k^{\mathrm{Re}})_1 + j(\mathbf{x}_k^{\mathrm{Im}})_1)\left((\mathbf{g}^{\mathrm{T}} + j\mathbf{b}^{\mathrm{T}})(\mathbf{x}_k^{\mathrm{Re}} + j\mathbf{x}_k^{\mathrm{Im}})^*\right). \tag{4.35}$$

Moreover, the feasible power injections at bus k are $p_k \in \mathcal{P}_k$ and $q_k \in \mathcal{Q}_k$ where

$$\mathcal{P}_k = \left\{ p \in \mathbb{R} \;\middle|\; \begin{array}{ll} p_k^{\mathrm{G,min}} - p_k^{\mathrm{L}} \le p \le p_k^{\mathrm{G,max}} - p_k & \text{if } k \in \mathcal{G} \\ p = -p_k^{\mathrm{L}} & \text{if } k \in \mathcal{N} \setminus \mathcal{G} \end{array} \right\} \tag{4.36}$$

$$\mathcal{Q}_k = \left\{ q \in \mathbb{R} \;\middle|\; \begin{array}{ll} q_k^{\mathrm{G,min}} - q_k^{\mathrm{L}} \le q \le q_k^{\mathrm{G,max}} - q_k & \text{if } k \in \mathcal{G} \\ q = -q_k^{\mathrm{L}} & \text{ifl } k \in \mathcal{N} \setminus \mathcal{G} \end{array} \right\}. \tag{4.37}$$

Therefore, even though the $\mathbf{x}$-Update returns feasible public variables that give feasible power injections the voltages that correspond to the net variables might result in an unfeasible power injections. That is, $\mathbf{p_z} \in \prod_{k\in\mathcal{N}} \mathcal{P}_k$ and $\mathbf{q_z} \in \prod_{k\in\mathcal{N}} \mathcal{Q}_k$ where

$$\mathbf{p_z} + j\mathbf{q_z} = (\mathbf{z}^{\mathrm{Re}} + \mathbf{z}^{\mathrm{Im}})(\mathbf{Y}(\mathbf{z}^{\mathrm{Re}} + \mathbf{z}^{\mathrm{Im}}))^* \tag{4.38}$$

Therefore we cannot guaranty feasible power injections till the algorithm has converges to a point where the consistency constraints

$$\mathbf{x}_k = \mathbf{E}_k \mathbf{z} \qquad k \in \mathcal{N}. \tag{4.39}$$

are valid.

The object function objective of the ADMM method is

$$f_k^{\mathrm{G}}(p_k^{\mathrm{G}}) + (\mathbf{y}_k)^{\mathrm{T}}(\mathbf{x}_k - \mathbf{E}_k \mathbf{z}) + (\rho/2)||\mathbf{x}_k - \mathbf{E}_k \mathbf{z}||_2^2. \tag{4.40}$$

Here ρ is a penalty parameter that casts penalty to the objective function if the consistency constraint is violated. Therefore, if ρ is large relative to the objective function then more effort is put in finding a feasible point than finding a good objective value. On the other hand if ρ is relatively small compared to the objective function then each subsystem focuses more on optimizing its part of the objective on the expense of consistency. Therefore, we have a tradeoff between consistency and the objective value.

Chapter 5

Privacy properties

Section 1.2 discusses key privacy issues in the smart grid. In particular, it was highlighted that high frequency information about the power demand of a household can be used to extract detailed information of its electrical use. This power demand information is however necessary when solving the OPF problem, see (3.24b)-(3.24e). Nevertheless, as pointed out in section 2.2, approaches based on decomposition can be used to address privacy in some cases, particularly if the problem is decomposed so that each party deals locally with its private data. Note that methods based on decomposition are not always privacy preserving because the private data might be reconstructed using other information. In this chapter, we explore the privacy properties of our proposed method proposed in chapter 4.

We can divide the buses into two groups, load bus which represent consumers in the network, and generator buses that represent power plants. The set of load buses includes all k such that $k \in \mathcal{N} \setminus \mathcal{G}$ and the set of generator buses includes all k such that $k \in \mathcal{G}$. Moreover, the private data is the power demands of the load buses, i.e., s_k^{L}, where $k \in \mathcal{N} - \mathcal{G}$. For this discussion we suppose that the adversaries that want to reconstruct the power demand of load bus k are simple all the buses in the network beside bus k. The adversaries can use their local knowledge such as their own voltage and the admittance in the flow lines they are connected to and global knowledge such as the network topology. In addition, they can use the information obtained by message passing during the algorithm such as the voltages of their neighbors. Furthermore, we suppose that no two buses in the network can share information.

We start by investigating the information that is required by bus $l \neq k$ for

reconstructing the power demand of bus $k \in \mathcal{N} \setminus \mathcal{G}$. From equation (3.1) we have that the power injection of the bus, s_k, is equal to the negative power demand i.e. $-s_k^{\mathrm{L}}$. Moreover, from equation (3.4) we have that

$$s_k = v_k i_k^*.$$

In addition, from (3.2) and (3.3),

$$i_k = \left(y_{k,k} + \sum_{l \in \Omega_k \setminus \{k\}} y_{k,l} \right) v_k + \sum_{l \in \Omega_k \setminus \{k\}} y_{k,l} v_l. \tag{5.1}$$

Therefore, the information needed to reconstruct the power demand of bus k is the voltage of k, the voltage of every neighbor of k, the admittance of every flow line connected to k, and the admittance to ground at bus k.

Let us now explore which bus has which part of the information needed to reconstruct the power demand of bus k.

- The voltage of k, v_k, is known by all neighbors of bus k.
- The voltage of bus l, v_l, where l is a neighboring bus of k is known by all neighbors of bus l.
- The admittance to ground at bus k, y_k is only known by bus k.
- The admittance of the flow line (k, l) is only known by bus k and bus l.

Since the admittance to ground at bus k is known only by bus k, even if all the buses of the network beside bus k collaborate they will not be able to build the full power demand of bus k. However, the admittance to ground at bus k is zero in many situations and therefore we also need to look at the case where $y_k = 0$. Only buses k and l know the admittance in the flow line (k, l). Therefore, all the neighbors of bus k need to take part in the reconstruction of the power demand at bus k. Moreover, if we assume that $y_k = 0$ then if all the neighbors of bus k combine their knowledge they have all the data needed to reconstruct the power demand of bus k by using equation (5.1). Therefore, load buses that have only one neighbor have no privacy given that $y_k = 0$. However, if the bus has more than one neighbor, then the solution method yields privacy for the power demand of the bus.

Chapter 6

Numerical results

In this chapter we will look at how the method we have provided behaves on test networks. All simulations where performed on a PC with Windows 7 Enterprise 64-bit operating system with Intel core i7-2600 CPU @ 3.40Hz and 16GB RAM. The simulations where created with matlab.

This chapter will be organized as follows. In section 6.1 we introduce the test networks we use. In section 6.2 we present the metric we used to measure feasibility after each iteration of the algorithm. Implementation details are then presented in section 6.3. Finally, the results are presented in sections 6.4 and 6.5.

6.1 Test networks

The matlab simulation package for power system Matpower[17] provides range of test networks for the OPF problem. Among those are the IEEE benchmark networks, which are standard test networks for the optimal power flow problem. We test our algorithm on three of the IEEE benchmark networks, namely the IEEE 14, 30, and 118 bus networks and we refer to them as IEEE14, IEEE30, and IEEE118, respectively. We also test our algorithm on Case9, a 9 bus network also provided by Matpower. The lower bound to the OPF problem given in appendix A is exact for all the networks above. We are therefore interested in testing the algorithm on a network where this lower bound is not tight. An example of a practical network with 3 buses, where the lower bound becomes non tight lower when an apparent power flow limit is introduced on one of the flow lines, is presented in [3]. For

comparison, we test our algorithm on the network both with and without the apparent power on the flow line. To distinguish between the two we will refer to the network without apparent power flow limit as case3A and with apparent power flow limit as case3B. In the following table we can see some statistics about the six test networks.

Network	Number of Buses	Number Flow lines	Average number of flow lines	Maximum number of flow lines	Mininum number of flow lines
Case3A	3	3	2	2	2
Case3B	3	3	2	2	2
Case9	9	9	2	3	1
IEEE14	14	20	2.8571	5	1
IEEE30	30	41	2.7333	7	1
IEEE118	118	186	3.0339	9	1

6.2 Measuring feasibility

The OPF problem is non-convex and hence obtaining a feasible solution can become an issue. Therefore, we need to examine if our proposed method converges to a feasible point. For that purpose we need a metric to measure the feasibility after each iteration. In particular we examine the voltages that correspond to the net variable $\mathbf{z}$ after the $\mathbf{z}$-Update. Recall that all the variables of the problems are uniquely determined by the voltages and therefore we can easily determine if the net variables result in a feasible solution of the OPF problem. If we inspect the steps of the algorithm thoroughly we observe that after the $\mathbf{z}$-Update has been performed the only constraints that might be violated are that the resulting power injections of the buses, constraints (3.24b)-(3.24g), and power injections in the flow lines, constraint (3.24s), might be infeasible. In our simulations we did run into troubles with respecting the constraint that limits the power injections of a flow line. Therefore we limit our analysis on feasible power injections of a bus.

Let us start by defining the metric

$$\operatorname{dist}(w, \mathcal{A}) = \inf\{|w - r| \big| r \in \mathcal{A}\} \tag{6.1}$$

between a subset $\mathcal{A} \subseteq \mathbb{R}$ and a scalar $w \in \mathbb{R}$ and let $\mathbf{z}^{\mathrm{Re}} + j\mathbf{z}^{\mathrm{Im}}$ be the net variable after the $\mathbf{z}$-Update. The power injections resulting from $\mathbf{z}^{\mathrm{Re}} + j\mathbf{z}^{\mathrm{Im}}$

are then

$$\mathbf{s} = \mathbf{p} + j\mathbf{q} = \left(\mathbf{z}^{\mathrm{Re}} + j\mathbf{z}^{\mathrm{Im}}\right) \bullet \left(\mathbf{Y}\left(\mathbf{z}^{\mathrm{Re}} + j\mathbf{z}^{\mathrm{Im}}\right)\right)^{*}. \tag{6.2}$$

Then we define the distance of the power injections $\mathbf{p} + j\mathbf{q}$ from the feasible set by

$$\mathrm{error}(\hat{\mathbf{p}} + j\hat{\mathbf{q}}) = \frac{1}{N}\sum_{k\in\mathcal{N}} \left(\mathrm{dist}(\hat{\mathbf{p}}_k, \mathcal{P}_k) + \mathrm{dist}(\hat{\mathbf{q}}_k, \mathcal{Q}_k)\right). \tag{6.3}$$

where N is the number of buses in the network and $\mathcal{P}_k$ and $\mathcal{Q}_k$ are the sets of feasible real and reactive power injections of bus k, i.e.

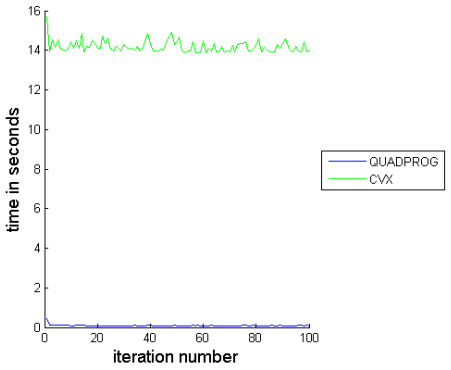
$$\mathcal{P}_k = \left\{ p \in \mathbb{R} \;\middle|\; \begin{array}{ll} p_k^{\mathrm{G,min}} - p_k^{\mathrm{L}} \le p \le p_k^{\mathrm{G,max}} - p_k & \text{if } k \in \mathcal{G} \\ p = -p_k^{\mathrm{L}} & \text{if } k \in \mathcal{N} \setminus \mathcal{G} \end{array} \right\} \tag{6.4}$$

$$\mathcal{Q}_k = \left\{ q \in \mathbb{R} \;\middle|\; \begin{array}{ll} q_k^{\mathrm{G,min}} - q_k^{\mathrm{L}} \le q \le q_k^{\mathrm{G,max}} - q_k & \text{if } k \in \mathcal{G} \\ q = -q_k^{\mathrm{L}} & \text{ifl } k \in \mathcal{N} \setminus \mathcal{G} \end{array} \right\}. \tag{6.5}$$

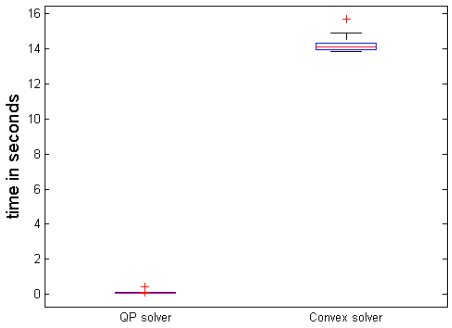
6.3 Implementation details

In section 4.1 it is discussed how the steps of the ADMM method, **x**-update, **z**-update and **y**-update, can be implemented for our problem. We did not have the opportunity to implement a distributed version of the algorithm. Therefore, we execute each step in sequence for all the buses for going to the next step. The **z**-update and **y**-update are simply performed by taking linear combinations of vectors, while the **x**-Update requires solving a non-convex problem. The approach to the solving the **x**-Update is based on sequentially solving convex problem. Furthermore, based on the discussion in section 4.2 this can be achived by using either convex or quadratic programing (QP) solvers. We implemented both versions and, for convex problems we used CVX [10, 9], a modeling language for convex problems that has an built in convex solver, and for the QP problems we used is quadprog, built in matlab QP solver, together with our own routines to transform the problem into a form suitable for quadprog. The advantage of using a modeling language, such as CVX, is the code gets cleaner and simpler. The downside side is that it results in a lot of overhead when many smaller problems are solved. In the case of our algorithm, an iteration of the master algorithm consists solving the non-convex sub problem for each bus and, which is done by sequentially solving convex problems. We tested all two implementations on a simple example, the three bus network net3A. We ran 100 iterations of the master algorithm with $\rho = 10^6$. In figure 6.1 we see clearly that the implementation

based on solving QP problems runs much faster. Then to verify that the two implementations are indeed equivalent we look at distance to the feasible set for each implementation in figure 6.2. From these results we see that our implementation based on using a general convex solver solver highly inefficient for larger networks, while the implementation based on using a QP solver is promising. Therefore, we will use the implementation based on using a QP solver to run the rest of the simulations.

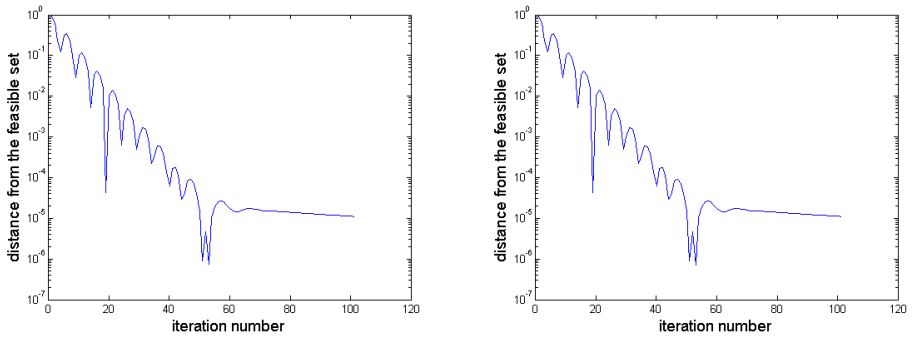


(a) Here we see how much time each iteration took in the ADMM master algorithm for the different solvers.



(b) Here we see a boxplot where each box represents a different solver. The center is the median and the edges of the box are the 25th and 75th percentiles.

Figure 6.1: Here we see how much time each iteration in the ADMM master algorithm takes for the convex and QP solvers.



(a) QUADPROG (b) Convex solver

Figure 6.2: Here we see the distance from the feasible set for both the convex and QP solvers.

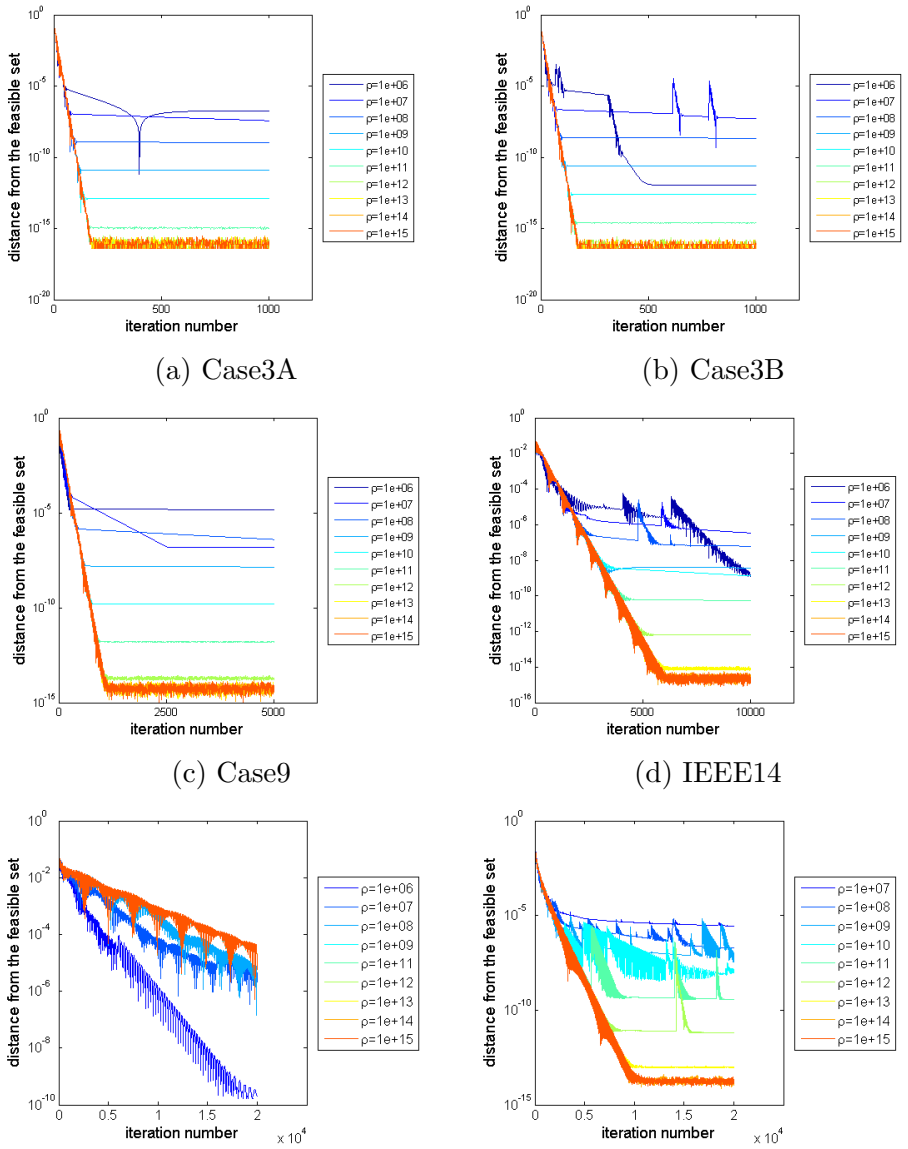
6.4 Results

In this section we demonstrate the results obtained by testing the algorithm on the six networks introduced in section 6.1. Furthermore we compare the results to the lower bound given in appendix A and to the solution obtained by using centralized OPF solver provided by Matpower. For these simulations the values of ρ range from 10^6 to 10^{15}. For each network we run as many iterations as we deem necessary but still not more than $2 \cdot 10^4$.

In figure 6.3 we look the metric given by equation (6.3) after each iteration of the algorithm. In the case of the IEEE30 network the algorithm does not converge in $2 \cdot 10^4$ iterations. However, the feasibility measure has a decreasing trend for all values of ρ and would probably eventually converge if we would run enough iterations. Moreover, for $\rho = 10^6$ the feasibility measure goes down to $2 \cdot 10^{-10}$. The algorithm converges in all other cases for the other 5 networks, except for IEEE14 with $\rho = 10^6$. We observe the following trend for the other 5 networks. For large values of ρ we manage to drive the algorithm to a point very close to the feasible set. Then when we look at declining values of ρ we observe that distance to the feasible set grows.

We are not only interested in obtaining a feasible point but we also want the algorithm to converge to a point that gives a good objective value. To examine the performance we look at both the objective function value and the distance to the feasible set after the last iteration of the algorithm, see

figure 6.4 and table 6.1. We noted above that the algorithm did not converge for the IEEE30. Therefore IEEE30 stands out in figure 6.4. But remarkably for $\rho = 10^6$ the algorithm returned a point very close to the feasible set with an objective value equall to the optimal value, see table 6.1. For the other 5 networks we observe the following trend. For a small value of ρ we get very good value of the objective function in all cases very close to the lower bound. But as we increase ρ the algorithm converges to a point that is closer to the feasible set but gives worse value of the objective function.

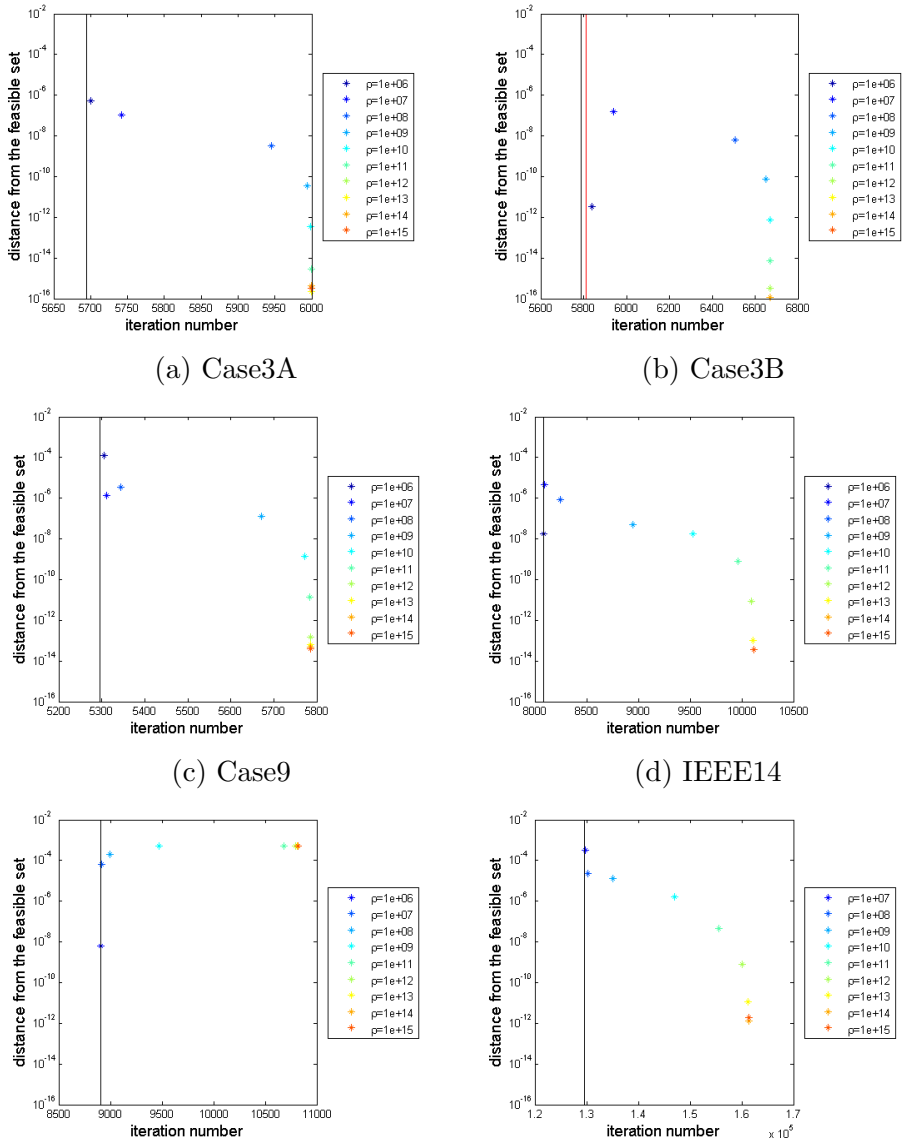


(a) Case3A (b) Case3B

(c) Case9 (d) IEEE14

(e) IEEE30 (f) IEEE118

Figure 6.3: Here we see the distance from the feasability set (equation 6.3) after each iteration in the ADMM. On the x-axis we have the iteration number and on the y-axis we have the distance form the feasible set.



(a) Case3A (b) Case3B

(c) Case9 (d) IEEE14

(e) IEEE30 (f) IEEE118

Figure 6.4: Here we see the value of the objective function on the x-axis and the distance from the feasible set, equation 6.3, on the y-axis after the algorithm has finished. For all the networks the black line indicates the lower bound given by the rank relaxation. For all the networks Matpower obtained solution equal to the lower bound except for case3B and there the solution obtained by Matpower is given by the red line.

ρ	Case3A		Case3B		Case9	
	obj	error	obj	error	obj	error
10^{6}	5699.9	$1.8 \cdot 10^{-07}$	5840.2	$1.1 \cdot 10^{-12}$	5305.0	$1.4 \cdot 10^{-05}$
10^{7}	5742.5	$3.7 \cdot 10^{-08}$	5939.5	$5.4 \cdot 10^{-08}$	5312.1	$1.5 \cdot 10^{-07}$
10^{8}	5945.8	$1.1 \cdot 10^{-09}$	6505.2	$2.1 \cdot 10^{-09}$	5344.3	$3.9 \cdot 10^{-07}$
10^{9}	5993.4	$1.2 \cdot 10^{-11}$	6652.2	$2.4 \cdot 10^{-11}$	5671.9	$1.4 \cdot 10^{-08}$
10^{10}	5998.6	$1.2 \cdot 10^{-13}$	6668.4	$2.5 \cdot 10^{-13}$	5772.0	$1.6 \cdot 10^{-10}$
10^{11}	5999.2	$9.6 \cdot 10^{-16}$	6670.0	$2.5 \cdot 10^{-15}$	5783.4	$1.6 \cdot 10^{-12}$
10^{12}	5999.2	$0.0 \cdot 10^{+00}$	6670.2	$3.7 \cdot 10^{-17}$	5784.6	$1.6 \cdot 10^{-14}$
10^{13}	5999.2	$3.7 \cdot 10^{-17}$	6670.2	$0.0 \cdot 10^{+00}$	5784.7	$8.2 \cdot 10^{-15}$
10^{14}	5999.2	$7.4 \cdot 10^{-17}$	6670.2	$3.7 \cdot 10^{-17}$	5784.7	$5.5 \cdot 10^{-15}$
10^{15}	5999.2	$1.1 \cdot 10^{-16}$	6670.2	$7.4 \cdot 10^{-17}$	5784.7	$4.1 \cdot 10^{-15}$
MP	5694.5		5812.6		5296.7	
LB	5694.5		5789.9		5296.7	

ρ	IEEE14		IEEE30		IEEE118	
	obj	error	obj	error	obj	error
10^{6}	8081.5	$1.2 \cdot 10^{-09}$	8906.1	$2.0 \cdot 10^{-10}$	-	-
10^{7}	8091.7	$3.4 \cdot 10^{-07}$	8911.6	$2.1 \cdot 10^{-06}$	129744.2	$2.8 \cdot 10^{-06}$
10^{8}	8242.2	$6.1 \cdot 10^{-08}$	8995.4	$6.9 \cdot 10^{-06}$	130212.5	$1.9 \cdot 10^{-07}$
10^{9}	8949.4	$3.7 \cdot 10^{-09}$	9471.5	$1.7 \cdot 10^{-05}$	135022.9	$1.1 \cdot 10^{-07}$
10^{10}	9527.0	$1.3 \cdot 10^{-09}$	-	-	146994.4	$1.4 \cdot 10^{-08}$
10^{11}	9965.1	$5.5 \cdot 10^{-11}$	10685.2	$1.8 \cdot 10^{-05}$	155777.0	$4.1 \cdot 10^{-10}$
10^{12}	10096.5	$6.3 \cdot 10^{-13}$	10798.3	$1.8 \cdot 10^{-05}$	160619.6	$8.2 \cdot 10^{-12}$
10^{13}	10111.6	$7.1 \cdot 10^{-15}$	10813.0	$1.8 \cdot 10^{-05}$	161204.0	$9.6 \cdot 10^{-14}$
10^{14}	10113.1	$2.6 \cdot 10^{-15}$	10814.6	$1.8 \cdot 10^{-05}$	161348.1	$1.1 \cdot 10^{-14}$
10^{15}	10113.3	$2.7 \cdot 10^{-15}$	10818.5	$1.8 \cdot 10^{-05}$	161362.8	$1.6 \cdot 10^{-14}$
MP	8081.5		8906.1		129660	
LB	8081.5		8906.1		129660	

Table 6.1: Here we see the results presented in figure 6.4. The line marked MP contains the results provided by Matpower and the line marked LB gives the lower bound given by relaxing the rank constraint.

6.5 Varying ρ

In the last section we observed the following trend. For large value of ρ the algorithm converged to a point very close to being feasible but not very optimal. For smaller values of ρ the algorithm converged to a point farther away from the feasible set but very close to being optimal. This inspires the following idea. We choose two ρs, ρ_1 and ρ_2, where ρ_1 is chosen to give a good objective function value and ρ_2 to force the algorithm closer to the feasible set. Then we run the algorithm n iterations with ρ_1 for n iterations get a point with a good object function value and after that we continue with ρ_2. We performed three experiments and the results can be seen in table 6.2. In all three cases we manage to get a solution with both good objective value and very close to the feasible set.

network	ρ_1	ρ_2	n	error	obj
Case3A	10^6	10^{15}	300	0	5701.9
Case3B	10^6	10^{15}	300	$7.4 \cdot 10^{-17}$	5843.2
Case9	10^6	10^{15}	1000	$1.1796 \cdot 10^{-14}$	5311.1

Table 6.2: Here we see the results of three experiments where we vary ρ.

Chapter 7

Conclusion

In this thesis we studied the optimal power flow (OPF) problem in the setting of the smart grid. The general OPF problem is non-convex and therefore designing algorithms that are both efficient and guarantee global optimality can be difficult. Therefore, obtaining practical algorithms often boils down to constructing good heuristics that find good suboptimal solutions efficiently. In addition, most real life electrical networks are of large scale and, therefore centralized algorithms can easily explode when employed in practice. As a result, practical algorithms must possess good scalability properties. Furthermore, for solving the OPF problem in the distribution network one has to relay on information that, if breached, compromises the privacy of the consumers. Therefore, privacy concerns have emerged in the discussion on the OPF problem in the smart grid.

In this thesis we tried to address some of the key problems mentioned above. In particular, we designed a suboptimal distributed algorithm for the general non-convex OPF problem, which is presented in chapter 4. The method was based on convex optimization together with sequential approximation techniques to avoid non-convexity issues of the original problem. Furthermore, we used the state of the art alternating direction method of multipliers to obtain a distributed algorithm where each bus of the network participates in the solution. The algorithm was implemented so that the sub problem associated with each bus only uses local information and information from neighboring buses in the network. Because of the sparsity of most real life electrical networks this ensures that the local sub problems will be of small scale. Therefore, the proposed algorithm can gracefully be scalable and, consequently, can be applied on real life networks.

We discussed the privacy properties of the proposed algorithm in chapter 5. In particular, we considered the privacy of the power demand of the load buses, where the adversaries where the other buses in the network. The analysis was limited to the case where the adversaries could only use local information and information obtained from message passing during the algorithm. Furthermore, no two buses where allowed to share information. We conclueded that within this frame privacy was obtained for load buses that had at least two neighbors in the network.

Finally we presented numerical results in chapter 6. There we inspected both the feasibility of solutions produced by the algorithm as well as the objective function values. We observed that both where sensitive to the algorithm parameter ρ, which imposes a penalty to violating the consistency constraints in the objective function. In particular, for increasing ρ the algorithm converged to points that where feasible with higher accuracy on the expense of a good objective value. Furthermore, for smaller values of ρ we obtained solutions with objective values close to the optimal solution in all cases.

Besed on this work, we suggest the following directions for future reasearch. One of our suggestions is to investigate the theoretical properties of the algorithm we have proposed. For example by examining the optimality conditions and explore if the algorithm converges to the necessary conditions for local optimality. Furhermore, an interesting direction would be to find an optimal value of ρ and inspect good stopping criterias. Another continuations of this work could be to embed total privacy of the power demand into the algorithm, for example by using transformations. Finally, it would be interesting to run the algorithm on large scale networks to really test the scalability of the algorithm.

Appendix A

Semidefinite programming with a rank constraint

In this section we will briefly present a semidefinite programming (SDP) formulation of the OPF problem with additional rank constraint. Therefore, if the rank constraint is relaxed then the problem becomes semidefinite programming. But semidefinite programming problem are convex problems that deal with minimizing a linear function over a cone:

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}} \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \mathbf{x}_1\mathbf{F}_1^{(k)} + \cdots + \mathbf{x}_n\mathbf{F}_n^{(k)} + \mathbf{G}^{(k)} \preceq 0 \qquad k = 1, 2, \cdots m \end{aligned} \tag{A.1}$$

where $\mathbf{c} \in \mathbb{R}^n$ and $\mathbf{G}^{(k)}, \mathbf{F}_1^{(k)}, \cdots, \mathbf{F}_m^{(k)} \in \mathbb{R}^{m\times m}$ are symmetric square matrices. Semidefinite programs are convex and can be solved in polynomial time for a given accuracy.

The formulation we present here is based on [12] and is also discussed in section 1.3 of this thesis. Let us denote the standard basis for $\mathbb{R}^N$ by $\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_N$. We have from [12] that if we set

$$\mathbf{Y}_k^{\mathrm{BUS}} := \mathbf{e}_k \mathbf{e}_k^T \mathbf{Y} \tag{A.2}$$

$$\mathbf{Y}_{lm}^{\mathrm{BUS}} := (y_{lm})\mathbf{e}_l \mathbf{e}_l^T - (y_{lm})\mathbf{e}_l \mathbf{e}_m^T \tag{A.3}$$

$$\hat{\mathbf{Y}}_k := \frac{1}{2}\begin{pmatrix} \mathrm{Re}\{\mathbf{Y}_k^{\mathrm{BUS}} + (\mathbf{Y}_k^{\mathrm{BUS}})^{\mathrm{T}}\} & \mathrm{Im}\{(\mathbf{Y}_k^{\mathrm{BUS}})^{\mathrm{T}} - \mathbf{Y}_k^{\mathrm{BUS}}\} \\ \mathrm{Im}\{\mathbf{Y}_k^{\mathrm{BUS}} - (\mathbf{Y}_k^{\mathrm{BUS}})^{\mathrm{T}}\} & \mathrm{Re}\{\mathbf{Y}_k^{\mathrm{BUS}} + (\mathbf{Y}_k^{\mathrm{BUS}})^{\mathrm{T}}\} \end{pmatrix} \tag{A.4}$$

$$\hat{\mathbf{Y}}_{lm} := \frac{1}{2}\begin{pmatrix} \mathrm{Re}\{\mathbf{Y}_{lm}^{\mathrm{BUS}} + (\mathbf{Y}_{lm}^{\mathrm{BUS}})^{\mathrm{T}}\} & \mathrm{Im}\{(\mathbf{Y}_{lm}^{\mathrm{BUS}})^{\mathrm{T}} - \mathbf{Y}_{lm}^{\mathrm{BUS}}\} \\ \mathrm{Im}\{\mathbf{Y}_{lm}^{\mathrm{BUS}} - (\mathbf{Y}_{lm}^{\mathrm{BUS}})^{\mathrm{T}}\} & \mathrm{Re}\{\mathbf{Y}_{lm}^{\mathrm{BUS}} + (\mathbf{Y}_{lm}^{\mathrm{BUS}})^{\mathrm{T}}\} \end{pmatrix} \tag{A.5}$$

$$\overline{\mathbf{Y}}_k := -\frac{1}{2}\begin{pmatrix} \mathrm{Im}\{\mathbf{Y}_k^{\mathrm{BUS}} + (\mathbf{Y}_k^{\mathrm{BUS}})^T\} & \mathrm{Re}\{\mathbf{Y}_k^{\mathrm{BUS}} - (\mathbf{Y}_k^{\mathrm{BUS}})^T\} \\ \mathrm{Re}\{(\mathbf{Y}_k^{\mathrm{BUS}})^T - \mathbf{Y}_k^{\mathrm{BUS}}\} & \mathrm{Im}\{\mathbf{Y}_k^{\mathrm{BUS}} + (\mathbf{Y}_k^{\mathrm{BUS}})^T\} \end{pmatrix} \tag{A.6}$$

$$\overline{\mathbf{Y}}_{lm} := -\frac{1}{2}\begin{pmatrix} \mathrm{Im}\{\mathbf{Y}_{lm}^{\mathrm{BUS}} + (\mathbf{Y}_{lm}^{\mathrm{BUS}})^T\} & \mathrm{Re}\{\mathbf{Y}_{lm}^{\mathrm{BUS}} - (\mathbf{Y}_{lm}^{\mathrm{BUS}})^T\} \\ \mathrm{Re}\{(\mathbf{Y}_{lm}^{\mathrm{BUS}})^T - \mathbf{Y}_{lm}^{\mathrm{BUS}}\} & \mathrm{Im}\{\mathbf{Y}_{lm}^{\mathrm{BUS}} + (\mathbf{Y}_{lm}^{\mathrm{BUS}})^T\} \end{pmatrix} \tag{A.7}$$

$$\mathbf{M}_k := \begin{pmatrix} \mathbf{e}_k \mathbf{e}_k^{\mathrm{T}} & 0 \\ 0 & \mathbf{e}_k \mathbf{e}_k^{\mathrm{T}} \end{pmatrix} \tag{A.8}$$

$$\mathbf{M}_{lm} := \begin{pmatrix} (\mathbf{e}_l - \mathbf{e}_m)(\mathbf{e}_l - \mathbf{e}_m)^{\mathrm{T}} & 0 \\ 0 & (\mathbf{e}_l - \mathbf{e}_m)(\mathbf{e}_l - \mathbf{e}_m)^{\mathrm{T}} \end{pmatrix} \tag{A.9}$$

$$\mathbf{w} := (\mathbf{v}^{\mathrm{Re}}, \mathbf{v}^{\mathrm{Im}}) \tag{A.10}$$

then we get that

$$p_k = \mathrm{Tr}\{\hat{\mathbf{Y}}_k \mathbf{w}\mathbf{w}^{\mathrm{T}}\} \tag{A.11}$$

$$q_k = \mathrm{Tr}\{\overline{\mathbf{Y}}_k \mathbf{w}\mathbf{w}^{\mathrm{T}}\} \tag{A.12}$$

$$p_{lm} = \mathrm{Tr}\{\hat{\mathbf{Y}}_{lm} \mathbf{w}\mathbf{w}^{\mathrm{T}}\} \tag{A.13}$$

$$|s_{lm}|^2 = (\mathrm{Tr}\{\hat{\mathbf{Y}}_{lm} \mathbf{w}\mathbf{w}^{\mathrm{T}}\})^2 + (\mathrm{Tr}\{\overline{\mathbf{Y}}_{lm} \mathbf{w}\mathbf{w}^{\mathrm{T}}\})^2 \tag{A.14}$$

$$|v_k|^2 = \mathrm{Tr}\{\mathbf{M}_k \mathbf{w}\mathbf{w}^{\mathrm{T}}\} \tag{A.15}$$

$$|v_l - v_m|^2 = \mathrm{Tr}\{\mathbf{M}_{lm} \mathbf{w}\mathbf{w}^{\mathrm{T}}\}. \tag{A.16}$$

The OPF problem can now be written equivalently as

$$\begin{aligned}
\text{minimize} \quad & \sum_{k\in\mathcal{G}} f_k^{\mathrm{G}}(\mathrm{Tr}\{\hat{\mathbf{Y}}_k\mathbf{W}\} + p_k^{\mathrm{L}}) && && \text{(A.17a)}\\
\text{subject to} \quad & p_k^{\min} - p_k^{\mathrm{L}} \le \mathrm{Tr}\{\hat{\mathbf{Y}}_k\mathbf{W}\} \le p_k^{\max} - p_k^{\mathrm{L}} && k\in\mathcal{G} && \text{(A.17b)}\\
& q_k^{\min} - q_k^{\mathrm{L}} \le \mathrm{Tr}\{\overline{\mathbf{Y}}_k\mathbf{W}\} \le q_k^{\max} - q_k^{\mathrm{L}} && k\in\mathcal{G} && \text{(A.17c)}\\
& \mathrm{Tr}\{\hat{\mathbf{Y}}_k\mathbf{W}\} = -p_k^{\mathrm{L}} && k\in\mathcal{N}\setminus\mathcal{G} && \text{(A.17d)}\\
& \mathrm{Tr}\{\overline{\mathbf{Y}}_k\mathbf{W}\} = -q_k^{\mathrm{L}} && k\in\mathcal{N}\setminus\mathcal{G} && \text{(A.17e)}\\
& (v_k^{\min})^2 \le \mathrm{Tr}\{M_k\mathbf{W}\} \le (v_k^{\max})^2 && k\in\mathcal{N} && \text{(A.17f)}\\
& (\mathrm{Tr}\{\hat{\mathbf{Y}}_{lm}\mathbf{W}\})^2 + (\mathrm{Tr}\{\overline{\mathbf{Y}}_{lm}\mathbf{W}\})^2 \le (s_{lm}^{\max})^2 && (l,m)\in\mathcal{L} && \text{(A.17g)}\\
& \mathrm{Tr}\{\hat{\mathbf{Y}}_{lm}\mathbf{W}\} \le p_{lm}^{\max} && (l,m)\in\mathcal{L} && \text{(A.17h)}\\
& \mathrm{Tr}\{\mathbf{M}_{lm}\mathbf{W}\} \le (\Delta V_{lm}^{\max})^2 && (l,m)\in\mathcal{L} && \text{(A.17i)}\\
& W = \mathbf{w}\mathbf{w}^T && && \text{(A.17j)}
\end{aligned}$$

where the variables are $\mathbf{w}$ and $\mathbf{W}$. Constraint (A.17j) is equivalent to requiring $\mathbf{W}$ to be semidefinite and having 1 matrix. Since $\mathbf{w}$ only appears in constraint (A.17j) we replace it by

$$\mathrm{rank}(\mathbf{W}) = 1. \tag{A.18}$$

Note that both the objective function, (A.17a), and constraint (A.17g) are quadratic. This needs to modified before we get a SDP formulation with the additional rank constraint (A.18). For that purpose we use Schur's complement formula which says that for a symmetric matrix

$$M = \begin{pmatrix} A & B \\ B^T & C \end{pmatrix} \tag{A.19}$$

we have that $M \succeq 0$ if and only if $C \succeq 0$ and $A - BC^{-1}B^T \succeq 0$. By using Schur's complement formula we get that (A.17g) is equivalent to:

$$\begin{bmatrix} -(S_{lm,\max})^2 & \mathrm{Tr}\{\mathbf{Y}_{lm}W\} & \mathrm{Tr}\{\overline{\mathbf{Y}}_{lm}W\} \\ \mathrm{Tr}\{\mathbf{Y}_{lm}W\} & -1 & 0 \\ \mathrm{Tr}\{\overline{\mathbf{Y}}_{lm}W\} & 0 & -1 \end{bmatrix} \preceq 0. \tag{A.20}$$

Note that we can put the objective function as constraint if we set:

$$f_k^{\mathrm{G}}(\mathrm{Tr}\{\hat{\mathbf{Y}}_k\mathbf{w}\mathbf{w}^{\mathrm{T}}\} + p_k^{\mathrm{D}}) \le \alpha_k \qquad k\in\mathcal{G} \tag{A.21}$$

where we define a new objective function

$$\sum_{k\in\mathcal{G}} \alpha_k. \tag{A.22}$$

If we use Schur's complement formula we get that (A.21) is equivalent to:

$$\begin{bmatrix} a_k^{(1)}\mathrm{Tr}\{\mathbf{Y}_k\mathbf{W}\} - \alpha_k + a_k^{(2)} + a_k^{(1)}p_k^{\mathrm{L}} & \sqrt{a_k^{(0)}}\mathrm{Tr}\{\mathbf{Y}_k\mathbf{W}\} + \sqrt{a_k^{(0)}}p_k^{\mathrm{L}} \\ \sqrt{a_k^{(0)}}\mathrm{Tr}\{\mathbf{Y}_k\mathbf{W}\} + \sqrt{a_k^{(0)}}p_k^{\mathrm{L}} & -1 \end{bmatrix} \preceq 0 \tag{A.23}$$

Then we can then formulate the OPF probelm as an semidefinate programs with rank constraint:

$$\begin{aligned}
\text{minimize} \quad & \sum_{k\in\mathcal{G}} \alpha_k && && \text{(A.24a)}\\
\text{subject to} \quad & \text{Equation A.23} && k\in\mathcal{G} && \text{(A.24b)}\\
& p_k^{\min} - p_k^{\mathrm{L}} \le \mathrm{Tr}\{\hat{\mathbf{Y}}_k\mathbf{W}\} \le p_k^{\max} - p_k^{\mathrm{L}} && k\in\mathcal{G} && \text{(A.24c)}\\
& q_k^{\min} - p_k^{\mathrm{L}} \le \mathrm{Tr}\{\overline{\mathbf{Y}}_k\mathbf{W}\} \le q_k^{\max} - q_k^{\mathrm{L}} && k\in\mathcal{G} && \text{(A.24d)}\\
& \mathrm{Tr}\{\hat{\mathbf{Y}}_k\mathbf{W}\} = -p_k^{\mathrm{L}} && k\in\mathcal{G} && \text{(A.24e)}\\
& \mathrm{Tr}\{\overline{\mathbf{Y}}_k\mathbf{W}\} = -q_k^{\mathrm{L}} && k\in\mathcal{G} && \text{(A.24f)}\\
& (v_k^{\min})^2 \le \mathrm{Tr}\{M_k\mathbf{W}\} \le (v_k^{\max})^2 && k\in\mathcal{N} && \text{(A.24g)}\\
& \text{Equation A.20} && (l,m)\in\mathcal{L} && \text{(A.24h)}\\
& \mathrm{Tr}\{\hat{\mathbf{Y}}_{lm}\mathbf{W}\} \le P_{lm}^{\max} && (l,m)\in\mathcal{L} && \text{(A.24i)}\\
& \mathrm{Tr}\{\mathbf{M}_{lm}\mathbf{W}\} \le (\Delta V_{lm}^{\max})^2 && (l,m)\in\mathcal{L} && \text{(A.24j)}\\
& \mathrm{rank}(\mathbf{W}) = 1 && && \text{(A.24k)}\\
& \mathbf{W} \succeq 0, && && \text{(A.24l)}
\end{aligned}$$

where the variables are $\mathbf{W}$ and α_k for $k\in\mathcal{G}$.

Bibliography

[1] A. Bednarz, N. Bean, and M. Roughan. Hiccups on the road to privacy-preserving linear programming. pages 117–120, Chicago, IL, USA, November.

[2] Alice Bednarz. *Methods for Two-Party Privacy Preserving Linear Programming*. PhD thesis, The University of Adelaide, 2012.

[3] Alex R-Borden Bernard C-Lesieutre, Daniel K-Molzahn and Christopher L. DeMarco. Examining the limits of the application of semidefinite programming to power flow problems. In *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*, pages 1492 – 1499, Sept. 2011.

[4] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, January 2011.

[5] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[6] Jorge Jardim Brian Stott and Ongun Alsac. Dc power flow revisited. *IEEE Transactions on power systems*, 2009.

[7] J. Carpentier. Contribution à l'étude du dispatching économique. *Bulletin de la Société Française des Electriciens Ser 8*, 3, August 1962.

[8] Kui Ren Cong Wang and Jia Wang. Secure and practical outsourcing of linear programming in cloud computing. In *IEEE INFOCOM*, 2011.

[9] Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

[10] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.0 beta. `http://cvxr.com/cvx`, September 2012.

[11] Balho H. Kim and Ross Baldick. Coarse-grainded distributed optimal power flow. *IEEE Transactions on Power Systems*, 12:932–939, May 1997.

[12] J. Lavaei and S. H. Low. Zero duality gap in optimal power flow problem. *IEEE Trans. Power Systems*, 27(1):92–107, 2012.

[13] Anya Castillo Mary B. Cain, Richard O'Neil. History of optimal power flow and formulations. 2012.

[14] K. S. Pandya and S.K. Joshi. A survay of optimal power flow methods. *Journal of Theoretical and Applied Information Technology*, 2008.

[15] James Paris. *Framework for Non-Intrusive Load Monitoring and Diagnostics*. PhD thesis, Massachusetts Institute of Technology, 2006.

[16] Elias Leake Quinn. Privacy and the New Energy Infrastructure. Technical report, Center for Energy and Environmental Security (CEES), University of Colorado, Fall 2008.

[17] Carlos E. Murillo-Sánchez R. D. Zimmerman and R. J. Thomas. Matpower: Steady-state operations, planning and analysis tools for power systems research and education. *Power Systems, IEEE Transactions on*, 26(1):12 – 19, Feb. 2011.

[18] Mary B. Cain Richard O'Neil, Anya Castillo. The iv formulation and linear approximation of the ac optimal power flow problem. 2012.

[19] Hadi Saadat. *Power system Analysis*. PSA Publishing, third edition, 2010.

[20] Almir Mutapcic Stehpen Boyd. Subgradient methods. 2007.

[21] Almir Mutapcic Jacob Mattingley Stehpen Boyd, Lin Xiao. Notes on decomposition methods. 2007.

[22] Lieven Vandenberghe Stehpen Boyd. Subgradients. 2008.

[23] Tomas Toft. Solving linear programs using multiparty computation. 2009.

[24] Pradeep Chathuranga Weeraddana, George Athanasiou, Martin Jakobsson, Carlo Fischione, and John S. Barras. Per-se privacy preserving distributed optimization. *CoRR*, abs/1210.3283, 2012.

[25] Ray D. Zimmerman and Carlos E. Murillo-Sánchez. Matpower 4.1 user's manual. Des. 2011.