Università degli Studi di Padova

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
Corso di Laurea in Ingegneria dell'Automazione

Tesi di Laurea Magistrale

Efficient State Estimation in Power Networks for Reactive Power Losses Compensation

Candidato
Alberto Di Vittorio

Relatore
prof. Ruggero Carli

Anno Accademico 2012-2013

Sebbene questa tesi sia solo il frutto degli ultimi miei sei mesi di lavoro, quando viene conclusa, stampata e presa in mano essa non può fare a meno di rappresentare, dentro di te, il simbolo della fine del tuo periodo uiversitario. Un periodo nel quale questo percorso di studi ti ha accompagnato giorno dopo giorno, senza lasciarti mai. Gioie, soddisfazioni, conquiste, e anche qualche grossa batosta. Ebbene, ora sembra davvero arrivato il capolinea. Sperando che questo capolinea non sia altro che il trampolino di lancio per l'entrata nel mondo del lavoro, non posso fare a meno di menzionare e di ringraziare tutte le persone che hanno reso possibile quasto mio traguardo, alcune con un supporto permanente e altre anche con un semplice gesto.

- *Grazie innanzitutto ad Ilaria, la mia fantastica compagna di vita, che ha condiviso con me ogni gioia e ogni traguardo e ha cercato di non farmi perdere autostima, voglia e fiducia in me stesso nei momenti di forte abbattimento;*

- *Grazie a mamma e papà, sempre i primi a credere in me e a spronarmi per tirare fuori il massimo in ogni ambito;*

- *Grazie ai miei nonni Elena, Gianni, Gino e Rita, per avermi fatto sentire un nipote speciale anche quando io non mi sentivo tale.*

- *Grazie a Diego e Wally, non solo semplici compagni universitari ma amici, non solo persone con cui studiare, interrogarsi e migliorarsi, ma con cui condividere una passione;*

- *Grazie a Lele e Diane (e Samuel), compagni e sposi novelli;*

- *Grazie a Tod, per avermi dato molto più di una mano in questa tesi;*

- *Grazie al prof. Ruggero Carli, per essere stato sempre disponibile nei miei confronti durante questo lavoro di tesi;*

- *Grazie a tutti i membri della mitica squadra di calcetto "Futsal3G", Luigino in primis;*

- *Grazie alla città di Lisbona, luogo che per me ricoprirà sempre una posizione speciale;*

- *Grazie a tutti i parenti e gli amici, italiani e non, che in questi cinque anni mi sono stati vicino anche solo con un semplice gesto o pensiero;*

- *Grazie alla musica, vero rifugio di serenità e divertimento che mai arriva alla saturazione;*

- *Grazie a te, perché se stai leggendo questa tesi significa che pensi possa essere un lavoro interessante.*

Contents

Chapter 1

Introduction

1.1 Extended Abstract

In the last decade the electric grid has been undergoing a deep renovation process toward the a so-called *smart grid* featuring larger hosting capability, widespread penetration of renewable energy sources, higher quality of the service and reliability. In particular, the modernization of the low voltage and medium voltage power distribution network consists in the deployment of a large amount of information and communication technologies (ICT), in the form of dispersed measurement, monitoring, and actuation devices. From the environmental point of view, in particular, it is talked about green technology, that is, smart tech capable to exploit sustainable source of energy.

Among the many different aspects of this transition, we focus on the control of the microgenerators inside a smart microgrid ([1], [2]). A microgrid is a portion of the low-voltage power distribution network that is managed autonomously from the rest of the network, in order to achieve better quality of the service, improve efficiency, and pursue specific economic interests. Together with the loads connected to the microgrid (both residential and industrial customers), we also have microgeneration devices (solar panels, combined heat-and-power plants, micro wind turbines, etc.). These devices are connected to the microgrid via electronic interfaces (inverters), whose main task is to enable the injection of the produced power into the microgrid. One example in this sense is the coordinated control of the power inverters of the microgeneration devices connected to the low voltage grid. When properly controlled, these devices can provide valuable *ancillary services* like reactive power compensation, voltage support, automatic generation control, optimal power flow computation, etc. ([3], [4]). In this work we consider the problem of optimal reactive power compensation (RPC). Loads belonging to the microgrid may require a sinusoidal current which is not in phase with voltage. A convenient description for this, consists in saying that they demand reactive power together with active power, associated with out-of-phase and in-phase components of the current, respectively. Reactive power is not a "real" physical power, meaning that there is no energy conversion involved nor fuel costs to produce it. Like active power flows, reactive power flows contribute to power losses on the lines, cause voltage drop, and may lead to grid instability. It is therefore preferable to minimize reactive power flows by producing it as close as possible to the users that need it.

For example, the reactive power compensation strategy proposed in [5] consists in an iterative tuning of the amount of reactive power injected by the microgenerators, with the objective of minimizing power distribution losses across the grid. The proposed procedure requires that microgenerators perform voltage phasor measurements at their own point of connection to the grid. These measurements are then shared with other microgenerators via a communication channel, and processed in a distributed way. Based on the result of this processing, each microgenerators then updates the amount of reactive power injected into the grid, actuating the system. Because of the inherent communication part, this strategy belongs to the wide class of *networked control systems* [7].

One of the main bottleneck in the actuaction of this kind of control strategies in the low voltage power distribution network is the need for accurate voltage phasor measurements across the grid. Specifically, to achieve the aim of control is necessary to handle with the voltage phasor at every node[1] of the grid, namely the **state** of the grid.

Phasor measurement units (PMU) can provide these measurements, but their cost is generally unacceptable for large scale deployment. In particular, time synchronization between different PMUs is a major technological issue, and it is generally tackled via a GPS module that can provide timestamping of the data.

The first contribution of this thesis consists in evaluating the effects of PMU measurement errors for measurement-driven control strategies, adopting the reactive power control proposed in [5] as a prototype. Then we present two distributed estimation algorithms capable of improving the quality of the voltage measurements via exchange of data with other PMUs and via distributed processing of the raw data.

We assume that the power distribution grid is divided into a number of areas. The PMUs beloning to each area transmit their voltage measurements to an *area monitor*. Area monitors can communicate and they are instructed to process the collected measurement in a distributed way.

Similar algorithms have already been proposed in the literature, especially for medium voltage and high voltage power grids, see [6], [7], [8], [9] [10]. The two solutions proposed, however, exhibit some notable original features which make them particularly suited for the scenario of low voltage

[1]A *Node* can represent either an household appliance in a domestic microgrid or an entire house demand in a urban grid.

power distribution grids:

- they only require measurements that can be performed by the devices at their point of connection, instead of power flow and current measurements on the power lines, which are generally not available in low voltage grids;
- the computational effort is very limited and remains constant if the grid grows in size;
- they are completely leader-less (no grid supervisor is present).

In order to present the two proposed distributed algorithms, we first introduce a model for the power grid, which includes a convenient modelling of the measurement errors in which time sync error are explicitly considered. Based on this model, we detail the least-square problem that has to be solved in order to find the maximum-likelyhood estimation of the grid state. For the solution of such optimization problem, we propose two different approaches. The first approach is a distributed implementation of the Alternating Direction Method of Multipliers ([11]). This contribution is of particular interest *per se*: we show how ADMM can be implemented in a scalable way [12], in which every agent is only required to store a portion of the entire state of the systems. The second approach is a distributed Jacobi-like algorithm. The algorithm is completely leader-less, and each monitor has to solve an extremely simple optimization problem, for which a closed form solution is provided.

Finally, it is studied the behaviour of these two estimation algorithm together with the reactive power compensation one. In the last example it is shown how are the estimation algorithm performances in the reactive power compensation for a grid constrained: we added upper and lower bounds to the reactive power that each compensator can exchange with the remaining network ([13]) These box constraints complicate the initial problem, but draw the model closer to the real microgrid. We will show that, using the estimate state, it leads to an optimal behaviour.

1.2 State of the Art

Since Power Networks State Estimation represents the starting point to implement a desirable network control, it has been fully treated in literature. Firstly it has been analyzed through centralized techniques. Afterwards researchers focused on distributed solution since the increasing in network size, the always more relevant computational effort, the networks topology privacy and the robustness to failures become strictly urgent.

The main aim of the estimation is to adequately filter the raw measurements with the purpose to achieve a better knowledge of a desire quantity, namely the state. This is very important due to the fact that measurements could be very noisy. Therefore, they cannot be straightly used to control the network. Indeed, presence of outliers, measurement errors and noisy measures, corrupting the real value of the state, make absolutely unusable the control. As a byproduct, the estimation could be efficiently used to do *fault detection* and *bad data detection*. This is, respectively, to detect fault of the network and to identify particularly bad measures (outliers), for instance, due to instrumentation faults or corruption through the connection lines.

In [14] the authors firstly present the principal electric components to introduce a suitable network model. Secondly, it is fully explained the centralized *weighted least squares* estimation supposing to deal with measurements affected by gaussian noise.

In [15], [16] e [17] it is firstly developed an exact network model, secondly an approximated one and finally the authors deal with the implementation of a centralized static least square estimation modeling the noise as a gaussian random variable. Finally it is suggested how to implement a real-time version of the algorithm proposed and a bad data detection.

In [6] the authors proposed a multi area distributed two-level estimation. Firstly the single area, using just inner measures, estimates its own knowledge of the state. Secondly, a central unit deals with the task of coordinate the single areas estimations via an additional set of pseudo-measurements take by *Phasor Measurement Units* (PMUs). Similar method is described both in [7] and [8].

In [18] it is proposed a technique that, after a preliminary decomposition of the net in smaller subnetworks, place the measurement units with the aim of optimizing their number and costs.

In [9], similar to the two-level implementation of [6], [7] and [8], the author proposes a method to deal with a distributed state estimation via only local measures. Thanks to the exchange of borders information between neighboring areas and a central coordination unit it is finally reached the wide range estimation.

In [10] a complete leader-less algorithm is proposed. Coordination and estimation are carried out only via local exchange of information.

In [11] the authors develop a fully distributed mean square algorithm. This leads a Wireless Sensor Network (WSN), in which the algorithm is tested, to adaptively reach the state estimation with single-hop neighbors exchanges of messages. The optimization problem is solved exploiting the Alternating Direction Method of Multipliers (ADMM).

In [19] an approach able to parallelize optimal power flow is presented. The proposed distributed scheme can be use to coordinate an heterogeneous collection of utilities. Three mathematical decomposition coordination methods are introduced to implement the proposed distributed sheme: the Auxiliary Problem Principle (APP); the Predictor-Corrector Proximal Multiplier Method (PCPM); the Alternating Direction Method (ADM).

In [12] is proposed a modification of the standard Alternating Direction Multiplier Method formulation in order to obtain a scalable version. The resulting algorithm is completely distributed and scalable.

In [5] the authors firstly propose an appropriate model for a low voltage microgrid, secondly they develop a completely distributed algorithm to appropriately command a sub set of microgenerators to achieve an optimal distribution losses minimization.

1.3 Short summary

This thesis is organized as follows:

- Chapter 2 presents the general model for the electric grid on wich we have based the work. Consecutively, is presented a specific low voltage microgrid model suitable either for the estimation topic or for the reactive power compensation formulation.

- Chapter 3 presents the problem to deal with. Specifically it formulates the estimation problem and its importance related to the reactive power compensation. A centralized solution to the problem is developed.

- Chapter 4 introduces and develops two completely distributed and scalable technique to achieve the target. Specifically, firstly an ADMM based solution is developed and proved. Secondly, a Jacobi-like algorithm is proposed.

- Chapter 5 presents a full set of tests to validate the algorithms proposed.

- Chapter 6 gathers the main features of the work done and gives some ideas on what can be done in future.

Notations

A. *State variables*

v_i Magnitude of the voltage at the i^{th} node.

θ_i Phase of the voltage at the i^{th} node.

V Vector containing all voltages' magnitude.

Θ Vector containing all voltages' phase.

i_i Magnitude of the current at the i^{th} node.

ϕ_i Phase of the current at the i^{th} node.

I Vector containing the magnitude of the current.

Φ Vector containing all currents' phase.

x_i Real of the voltage at the i^{th} node.

y_i Imaginary of the voltage at the i^{th} node .

X Vector containing all the real parts.

Y Vector containing all the imaginary parts.

$\mathbf{X} = [X\ Y]^T$ Vector of all the state variables.

B. *Measures*

v_i^m Magnitude of the voltage at the i^{th} node.

θ_i^m Phase of the voltage at the i^{th} node.

V^m Vector containing all the magnitude of the voltages measured.

Θ^m Vector containing all the phases of the voltages measured.

s_i Real of the voltage at the i^{th} node.

r_i Imaginary of the voltage at the i^{th} node.

S Vector of the real parts of the voltage.

R Vector of the imaginary parts of the voltage.

i_i^m Magnitude of the current at the i^{th} node.

ϕ_i^m Phase of the current at the i^{th} node.

h_i Real of the current at the i^{th} node.

k_i Imaginary of the current at the i^{th} node.

H Vector of the real part.

K Vector of the real part.

C. *Standard deviations*

σ_v Standard deviation of the voltage magnitude error.

σ_θ Standard deviation of the voltage phase error.

σ_i Standard deviation of the current magnitude error.

σ_ϕ Standard deviation of the current phase error.

D. *Functions*

$J(\cdot)$ Objective Cost Function.

$f(\cdot)$ Current magnitude nonlinear function of the state.

$g(\cdot)$ Current phase nonlinear function of the state.

$|\cdot|$ Both the absolute value of a quantity or the cardinality of a set depending on the context.

$\cdot^T$ Transpose of a vector or matrix.

$\bar{\cdot}$ Complex conjugate of a complex quantity.

$\cdot^*$ Both complex conjugate and transpose.

E. *Matrixes and Vectors*

I_n Identity matrix $\in \mathbb{R}^{n\mathrm{x}n}$.

$\mathbb{1}$ Vector whose element are all equal to one.

$\mathbb{1}_i$ Canonical vector whose elements are all equal to zero except for that in position i.

Chapter 2

Grid Modeling

In this chapter we introduce in a general way an electric grid, starting from the principal components. Atfer have described a commonly adopted model of an electric grid ([14]) we present the specific model on which we will focus in this thesis ([5]).

2.1 Grid Components

An electric grid consists of a series of electric components such as transmission lines, loads, generators, transformers and capacitors. It is usually assumed the power system to operate in the steady state under balance conditions. This implies that all bus loads and branch flows will be three phase and balanced, all transmission lines are fully transposed and all other series or shut devices are symmetrical in the three phases [14]. These assumptions allow the use of the single phase positive sequence equivalent circuit for modeling the entire system. The following component models are commonly used in representing the network.

2.1.1 Transmission Lines

Transmission lines are usually represented by a two-port π-model characterized by a series impedance of $R + jX$ and a total line charging susceptance of $j2B$ corresponding to the equivalent circuit of figure 2.1

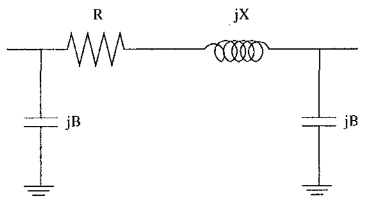


Figure 2.1: Equivalent circuit for a transmission line

2.1.2 Shunt Capacitors or Reactors

Shunt devices are represented by their susceptance at the corresponding bus whose sign determines the type of shunt element. They can be used for voltage and/or reactive power control.

Consider an admittance $Y = jB$ characterized the susceptance B

$$B = \begin{cases} \omega L \\ -\frac{1}{\omega C} \end{cases}$$

the sign of B determines the type of shunt element so a positive value corresponds to a capacitor, alternatively a negative one to a reactor.

2.1.3 Transformers

Transformers can be modeled, as shown in figure 2.2, as series impedance in series with an ideal transformers, where a represents the tap ratio which can be a real value if the transformer is an in phase device or complex if is a phase shifting device; m and k are the buses connected to the transformer.

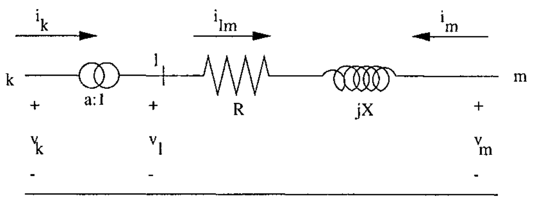


Figure 2.2: Equivalent circuit for a transformers

It is easy to see that the nodal equations of the two-port circuit, for the more general case of a complex value of a, are

$$\begin{bmatrix} i_k \\ i_m \end{bmatrix} = \begin{bmatrix} \frac{y}{|a|^2} & -\frac{y}{\bar{a}} \\ -\frac{y}{a} & y \end{bmatrix} \begin{bmatrix} v_k \\ v_m \end{bmatrix}$$

where $\bar{a}$ is the complex conjugate of a and y represents the admittance of the $l - m$ branch.

2.1.4 Loads and Generators

Loads and Generators are modeled respectively as negative or positive complex power injection and therefore have no effect on the network model.

2.2 Network Model

Modelling each component as above, we can build the general network model of the system, that is the admittance matrix Y describing the Kirchhoff's current law at each bus:

$$I = \begin{bmatrix} i_1 \\ \vdots \\ i_N \end{bmatrix} = \begin{bmatrix} Y_{11} & \cdots & Y_{1N} \\ \vdots & & \vdots \\ Y_{N1} & \cdots & Y_{NN} \end{bmatrix} \begin{bmatrix} v_1 \\ \vdots \\ v_N \end{bmatrix} = YV$$

where i_k is the current injection and v_k is voltage phasor, both at bus k; Y_{km} is the (m, k) entry of the Y matrix representing the total admittance between nodes m and k.

2.3 Smart Grid Model

For our specific problem, the model built is a bit different from the general one, and almost similar to the one described in [5].

2.3.1 Mathematical Preliminaries

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \sigma, \tau)$ be a directed graph, where $\mathcal{V}$ is the set of nodes ($|\mathcal{V}| = n$), $\mathcal{E}$ is the set of edges ($|\mathcal{E}| = r$) and σ, $\tau : \mathcal{E} \to \mathcal{V}$ are two functions such that the edge $e \in \mathcal{E}$ goes from node $\sigma(e)$ to node $\tau(e)$.
Two edges e, e' are said to be *consecutive* if

$$\{\sigma(e), \tau(e)\} \cap \{\sigma(e'), \tau(e')\}$$

is not empty. A *path* is a sequence of consecutive edges. It is possible to describe the graph through its incidence matrix $A \in \mathbb{R}^{r \times n}$ defined as follows:

$$A_{ei} = \begin{cases} \text{-1} & \text{if } i = \sigma(e); \\ 1 & \text{if } i = \tau(e); \\ 0 & \text{otherwise.} \end{cases}$$

A graph is connected if exists a path connecting every pair of nodes. If this is the case the vector $\mathbb{1}$ is the only one owning to the null space $ker(A)$.

2.3.2 Model

Let us firstly define a **smart grid** (or microgrid depending on its dimensions) as a portion of the power distribution network, described above, that is connected to the power transmission network in one point, the PCC (point of common coupling), and hosts a number of loads and power generators. We consider the grid as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ whose edges $\mathcal{E}$ represent

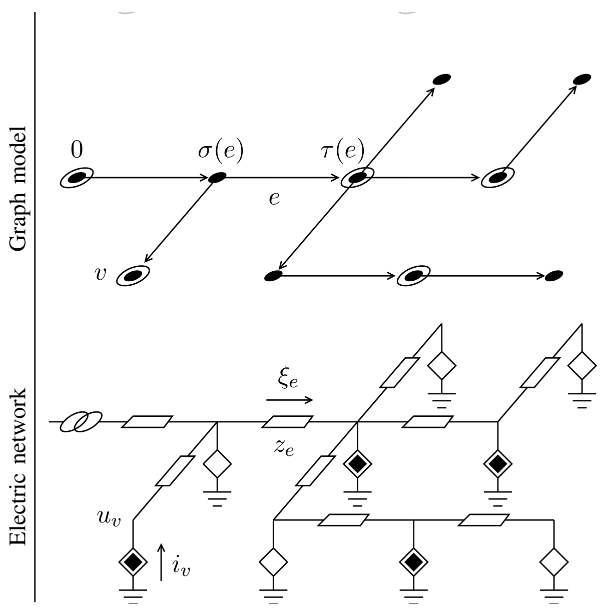


Figure 2.3: Lower Panel: Electric point of view for the grid. Black diamonds represent loads, while white diamonds represent microgenerators.
Upper panel: graph interpretation of the grid. Circled nodes correspond to microgenerators.

the power lines and nodes $\mathcal{V}$ both the loads and generators. Figure 2.3 shows the correspondence between the electric and graph formulation for the grid.

We limit our study to the steady state behavior of the system, as mention above. This let us represent all the signal via a complex number $y = |y|e^{j\angle y}$, since they are sinusoidal wave of the same frequency. The absolute value $|y|$ represents the signal root mean square and the argument $\angle y$ represents its phase with respect to an arbitrary global reference (usually that of the PCC).
The notation introduced above let us define the steady state of the system as:

- $\mathbf{v} = Ve^{j\Theta} \in \mathbb{C}^n$, where $v_i e^{j\theta_i}$ is the complex voltage of the i^{th} node;
- $\mathbf{i} = Ie^{j\Phi} \in \mathbb{C}^n$, where $i_i e^{j\phi_i}$ is the complex current of the i^{th} node;
- $\xi \in \mathbb{C}^r$, where ξ_e is the current flowing in the edge e.

It is useful to highlight the electric component specifically considered in our model clarifying the differences existing between our model and that of section 2.1.

Power lines are commonly described via the π-model characterized by, see figure 2.4,

- $z_e = r_e + jx_e = \frac{1}{g_e + jb_e}$: line impedance;
- y_e^{sh}: shunt admittance.

We neglect the shunt devices and consider only the line impedance z_e.

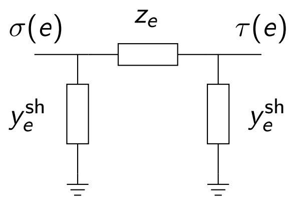


Figure 2.4: Equivalent circuit for a power lines

It is easy to see that the power line is described by the equation

$$\xi_e = \frac{v_{\sigma(e)} - v_{\tau(e)}}{z_e}$$

For every edge e of the graph, we define by z_e the impedance of the corresponding power line. We assume the following.

Assumption 1. *All power lines in the microgrid have the same inductance/resistance ratio, i.e.*

$$ze = e^{j\theta} Z$$

for any e in ξ and for a fixed θ.
Z is a diagonal real-valued matrix, whose elements are $Z_{ee} = |z_e|$.

This assumption is satisfied when the grid is relatively homogeneous, and is reasonable in most practical cases (see for examples the IEEE standard testbeds ([20]). The following physical constraints are satisfied by v, i, and ξ

$$A^T \xi + \mathbf{i} = 0 \tag{2.1}$$

$$A\mathbf{v} + Z\xi = 0 \tag{2.2}$$

where A is the incidence matrix introduced above; $Z = diag\{z_e; e \in \mathcal{E}\}$ represents the diagonal matrix of line impedances.
Equation (2.1) corresponds to Kirchhoff's current law (KCL) at the nodes, while equation (2.2) describes the voltage drop on the edges of the graph.
From (2.1) and (2.2) we can olso obtain

$$\mathbf{i} = A^T Z^{-1} A\mathbf{v} = L\mathbf{v} \tag{2.3}$$

where L represents the *weighted Laplacian* matrix of the graph, the *nodal admittance* matrix in power system analysis.

The *PCC* (point of common coupling) is modeled as a constant voltage generator

$$v_{PCC} = V_N e^{j\theta_0} = V_0 \tag{2.4}$$

where V_N is the nominal voltage and θ_0 is an arbitrary fixed angle. In the power system analysis terminology, node 0 is then a *slack bus* with fixed voltage magnitude and angle.

Differently from PCC, *Loads* are considered to require a given amount of active and reactive power for example depending on the voltage amplitude v_i. Examples of this are constant impedance loads and constant power loads. To describe in a unique way all different loads considered it is useful to exploit the exponential model in which each node (except the PCC) is modeled via a law relating the voltage v_i and current i_i. Specifically

$$v_i \bar{i}_i = s_i \left| \frac{v_i}{V_N} \right|^{\eta_i} \tag{2.5}$$

where s_i is the nominal complex power and η_i is a characteristic parameter. More specifically s_i is the value of the complex power that the node would inject into the grid if the voltage at its point of connection is equal to V_N. The quantities

$$p_v := \Re(s_v) \quad \text{and} \quad q_v := \Im(s_v) \tag{2.6}$$

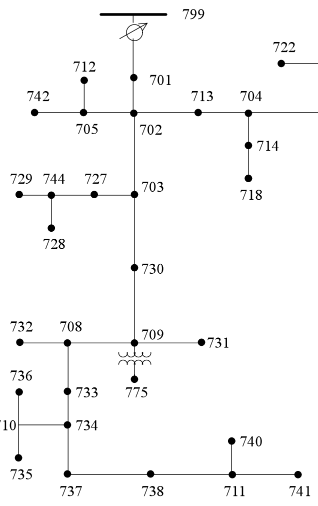
are denoted as *active* and *reactive* power, respectively. The complex power s_v corresponding to grid loads are such that $\{p_v < 0\}$, meaning that positive active power is *supplied* to the device; on the other hand, are such that $\{p_v \geq 0\}$, as positive active power is *injected* into the grid. The parameter η_i identify the device typology: for example constant power, constant current and constant impedance loads are described respectively by $\eta_i = 0, 1, 2$. See that generators fit this model for a parameter $\eta_i = 0$.

Finally, dealing with a low voltage power distribution network, *transformers*, both *tap changers* and *phase shifters*, are neglected.

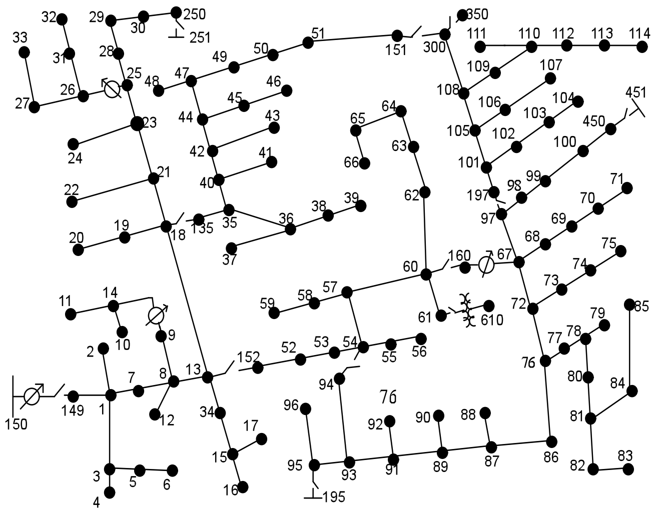
The three equations (2.4), (2.5) and (2.3) individuate a system of non linear equation to be solved to determine the steady state of the grid starting from the knowledge of the grid topology (identify by L) and the power demand required by the nodes. This topic is extensively covered in literature known as *Power Flow Analysis.* To our purpose is completely indifferent how the grid is solved and we will assume on the following to exploit some algorithm that performs it.

2.4 Testing setup: IEEE test Feeders

Here are presented the specific tests setup used through this thesis. All algorithm presented in the following has been tested on either one or both the `IEEE37` or IEEE123 [20] Radial Distribution Test Feeder. More specifically the graphs describing the mentioned test feeder are presented in figure 2.5.



(a) 37 nodes test feeder graph



(b) 123 nodes test feeder graph

Figure 2.5: Test Feeders graphs

Chapter 3

State Estimation and Reactive Power Compensation

This chapter is divided in two parts, both of crucial importance dor this thesis. In the first part we introduce and develop the estimation problem for an electric power grid. In the second one, we provide a formulation for the reactive power compensation problem.

3.1 Model and Estimation Problem Formulation

3.1.1 Model

Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ representing the grid, as described in chapter 2, where $\mathcal{V}$ is the set of n nodes and $\mathcal{E}$ is the set of r edges.
It is assumed that every node can measure its current and voltage divided into magnitude and phase, i.e.,

$$\begin{aligned}
v_i^m &= v_i + e_{v_i}; \qquad e_{v_i} \sim \mathcal{N}(0, \sigma_v^2);\\
\theta_i^m &= \theta_i + e_{\theta_i}; \qquad e_{\theta_i} \sim \mathcal{N}(0, \sigma_\theta^2);\\
i_i^m &= i_i + e_{i_i}; \qquad e_{i_i} \sim \mathcal{N}(0, \sigma_i^2);\\
\phi_i^m &= \phi_i + e_{\phi_i}; \qquad e_{\phi_i} \sim \mathcal{N}(0, \sigma_\phi^2);
\end{aligned}$$

where e_{v_i}, e_{θ_i}, e_{i_i} and e_{ϕ_i} represent the error introduced by the measure itself[1]. All the measurements are assumed to be independent from each other. Collecting all the measurements in vectors one can write

$$\begin{cases}
V^m := V + \mathbf{e}_V;\\
\Theta^m := \Theta + \mathbf{e}_\Theta;\\
I^m := I + \mathbf{e}_I;\\
\Phi^m := \Phi + \mathbf{e}_\Phi
\end{cases}$$

[1] In a real set up every node of the grid is equipped by a PMU(**P**hasor **M**easurement **U**nit).

where, we recall, (see notations page 13)

$$V^m = \begin{bmatrix} v_1^m \\ \vdots \\ v_N^m \end{bmatrix}; V = \begin{bmatrix} v_1 \\ \vdots \\ v_N \end{bmatrix}; e_V = \begin{bmatrix} e_{v_1} \\ \vdots \\ e_{v_N} \end{bmatrix};$$

and where Θ^m, Θ, e_Θ, I^m, I, e_I, Φ^m, Φ and e_Φ are defined similarly.
Now let us define the noise vector $\mathbf{e} = [\mathbf{e}_V\ \mathbf{e}_\Theta\ \mathbf{e}_I\ \mathbf{e}_\Phi]^T$. Then the correlation matrix R for the noise is

$$R = E[\mathbf{e}\mathbf{e}^T] = \begin{bmatrix} \sigma_v^2 I_n & & & \\ & \sigma_\theta^2 I_n & & \\ & & \sigma_i^2 I_n & \\ & & & \sigma_\phi^2 I_n \end{bmatrix}$$

We define the **state** of the grid as the voltage magnitude and phase at every node. Then, it is well known that

$$\begin{cases} V^m = V + \mathbf{e}_V; \\ \Theta^m = \Theta + \mathbf{e}_\Theta; \\ I^m = f(V,\Theta) + \mathbf{e}_I; \\ \Phi^m = g(V,\Theta) + \mathbf{e}_\Phi \end{cases} \tag{3.1}$$

where, generally, $f(\cdot)$ and $g(\cdot)$ represent non linear current's dependance on the state.

Assumption 2. *Measurements are taken all at the same time instant. Therefore there is no synchronization noise.*

In this thesis it is not considered, but it is not very difficult to include also the synchronization noise in this model. (see [21])

3.1.2 Estimation Problem Formulation

In general, we can say that the state estimation problem in an electric grid is reduced as a *Weighted Least Squares Problem* (see [14],[17]). Following this way, we can define this cost function depending on the measurements

$$J(V,\Theta) = \begin{bmatrix} V^m & \Theta^m & I^m & \Phi^m \end{bmatrix} R^{-1} \begin{bmatrix} V^m \\ \Theta^m \\ I^m \\ \Phi^m \end{bmatrix} \tag{3.2}$$

that is, explicitly,

$$\begin{aligned} J(V,\Theta) \quad &= \quad \sum_{i=1}^{n} \Big\{ \frac{1}{\sigma_v^2}(v_i^m - v_i)^2 + \frac{1}{\sigma_\theta^2}(\theta_i^m - \theta_i)^2 + \frac{1}{\sigma_i^2}(i_i^m - f(V,\Theta))^2 + \\ & \quad + \frac{1}{\sigma_\phi^2}(\phi_i^m - g(V,\Theta))^2 \Big\} \end{aligned}$$

Eventually, if measurements of other nature are available, i.e. power (real and reactive) injected, power flow exc..., they can be added to expression in (3.2).
As a matter of fact, we want to underline that a novelty of this work stays just in discarding them and using only current and voltage measures.

In order to *estimate* the state, we have to find the value $(\hat{V}, \hat{\Theta})$ of (V, Θ) that minimizes the objective cost function, that is, to find the solution for the optimization problem

$$\min_{V,\Theta} J(V, \Theta) \tag{3.3}$$

The nonlinear dependance of I and Φ on V and Θ drives the above optimization problem into the *nonlinear unconditioned* class of problems. To solve it we can find a lot of techniques in the literature. For instance, all methods based on the augmented Lagrangian technique.
However, this kind of algorithm may suffer of one of these disadvantages:

- the algorithm does non converge;
- the algorithm converge to a local but not global minima;
- the algorithm converge to the global minima but it requires a very long running time to achieve the convergence.

In this thesis, in order to deal with the optimization problem in (3.3), we pursue an approach based on a suitable linearized model of the electric grid. Interestingly we will see how the linear model leads to a convenient closed form solution.

3.1.3 Model Linearization

In the previous chapter we exploited the relation between complex value current and voltage of the grid

$$\mathbf{i} = L\mathbf{u} \tag{3.4}$$

where $L = A^T Z^{-1} A$ represents the admittance matrix of the grid; being A and Z, respectively, the incidence and inductance matrix of the grid (see chapter 2).

The basic idea to obtain a linear model is based on expressing the quantities of interest as function either of the real or imaginary part of the voltage, instead of the actual representation, based on magnitude and phase. Splitting relation (3.4) into real and imaginary part we can obtain, for a single node, (see page 13)

$$h + jk = [\Re(L) + j\Im(L)] * (s + jr) = [\Re(L)s - \Im(L)r] + j[\Im(L)s + \Re(L)r]$$

that can be rewritten in a matrix form as

$$\begin{bmatrix} h \\ k \end{bmatrix} = \begin{bmatrix} \Re(L) & -\Im(L) \\ \Im(L) & \Re(L) \end{bmatrix} \begin{bmatrix} s \\ r \end{bmatrix}$$

Collecting all nodes values, the whole measures model becomes (see notations page 13)

$$\begin{aligned} \begin{bmatrix} S \\ R \\ H \\ K \end{bmatrix} &= \begin{bmatrix} I_n & 0 \\ 0 & I_n \\ \Re(L) & -\Im(L) \\ \Im(L) & \Re(L) \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} + \begin{bmatrix} \mathbf{e}_S \\ \mathbf{e}_R \\ \mathbf{e}_H \\ \mathbf{e}_K \end{bmatrix} \\ Z \quad &= \qquad \mathcal{H} \quad \cdot \quad \mathbf{X} \quad + \quad \mathbf{e} \end{aligned} \tag{3.5}$$

where $\mathbf{e}_S$, $\mathbf{e}_R$, $\mathbf{e}_H$ and $\mathbf{e}_K$ denotes the noises of the measures with respect to the real and imaginary parts; Z denotes the measures vector and $\mathcal{H}$ denotes the model matrix and $\mathbf{e}$ the noises vector.

Note that (3.5) represents a suitable linear model for the measures with respect to the **new state variables**, that are the real and imaginary part of the nodes voltage.

To manage correctly this new representation, it is necessary to express the noise in a suitable form, starting from the knowledge of the standard deviation of magnitude and phase measures. The noise expressed in this new form is, in fact, textbfno more uncorrelated because, in general, part of the magnitude and phase noise will be reprojected into both real and imaginary part.

In order to understand this fact consider two generic vectors

$$x = \rho e^{j\psi} = \rho(\cos\psi + j\sin\psi);$$

$$\tilde{x} = \tilde{\rho} e^{j\tilde{\psi}} = (\rho + \delta\rho)e^{j(\psi+\delta\psi)} = (\rho + \delta\rho)(\cos(\psi + \delta\psi) + j\sin(\psi + \delta\psi))$$

where, clearly, $\delta\rho$ and $\delta\psi$ represent a sort of error affecting the exact values ρ and ψ. It is possible to rewrite $\tilde{x}$, exploiting some trigonometric relations, as

$$\begin{aligned} \tilde{x} &= \rho(\cos\psi + j\sin\psi)\cos\delta\psi + \delta\rho(\cos\psi + j\sin\psi)\cos\delta\psi - \\ &\quad -\rho(\sin\psi - j\cos\psi)\sin\delta\psi - \delta\rho(\sin\psi - j\cos\psi)\sin\delta\psi \end{aligned}$$

To simplify the expression above we assume $\delta\psi$ small enough and, using the McLaurin expansion for the sine and cosine functions, we get

$$\begin{aligned} \tilde{x} &\simeq \rho(\cos\psi + j\sin\psi)\Big(1 - \frac{\delta\psi^2}{2}\Big) + \delta\rho(\cos\psi + j\sin\psi)\Big(1 - \frac{\delta\psi^2}{2}\Big) - \\ &\quad -\rho(\sin\psi - j\cos\psi)\delta\psi - \delta\rho(\sin\psi - j\cos\psi)\delta\psi \end{aligned}$$

Finally, taking a first order approximation, we get

$$\begin{aligned}\tilde{x} &\simeq \rho(\cos\psi + j\sin\psi) + \delta\rho(\cos\psi + j\sin\psi) - \rho(\sin\psi - j\cos\psi)\delta\psi \\ &\simeq x + (\delta\rho\ \cos\psi - \rho\ \sin\psi\ \delta\psi) + j(\delta\rho\ \sin\psi + \rho\ \cos\psi\ \delta\psi)\end{aligned}$$

that can be rewritten in a matrix form as

$$\tilde{x} \simeq x + \begin{bmatrix}\cos\psi & -\sin\psi \\ \sin\psi & \cos\psi\end{bmatrix}\begin{bmatrix}\delta\rho \\ \rho\ \delta\psi\end{bmatrix} \tag{3.6}$$

Equation (3.6) highlights the approximation exploited to project the error measured in phase and magnitude into real and imaginary component.

Managing our error variables in a similar way, we can obtain (see [21]) the new noise correlation matrix:

$$R = \begin{bmatrix}\sigma^2_{\Re(V)}I_n & \sigma_{\Re(V)\Im(V)}I_n & \sigma_{\Re(V)\Re(I)}I_n & \sigma_{\Re(V)\Im(I)}I_n \\ \sigma_{\Im(V)\Re(V)}I_n & \sigma^2_{\Im(V)}I_n & \sigma_{\Im(V)\Re(I)}I_n & \sigma_{\Im(V)\Im(I)}I_n \\ \sigma_{\Re(I)\Re(V)}I_n & \sigma_{\Re(I)\Im(V)}I_n & \sigma^2_{\Re(I)}I_n & \sigma_{\Re(I)\Im(I)}I_n \\ \sigma_{\Im(I)\Re(V)}I_n & \sigma_{\Im(I)\Im(V)}I_n & \sigma_{\Im(I)\Re(I)}I_n & \sigma^2_{\Im(I)}I_n\end{bmatrix}$$

where for all $i \in \{1...n\}$, the diagonal block are equal to

$$\begin{aligned}\sigma^2_{\Re(V)} &= \sigma^2_v\cos^2\theta + \sigma^2_\theta(v^m_i)^2\sin^2\theta; \\ \sigma^2_{\Im(V)} &= \sigma^2_v\sin^2\theta + \sigma^2_\theta(v^m_i)^2\cos^2\theta; \\ \sigma^2_{\Re(I)} &= \sigma^2_i\cos^2\phi + \sigma^2_\phi(i^m_i)^2\sin^2\phi; \\ \sigma^2_{\Im(I)} &= \sigma^2_i\cos^2\phi + \sigma^2_\phi(i^m_i)^2\cos^2\phi;\end{aligned}$$

representing the autocorrelation between quantities. The cross correlation between the real and imaginary part of the voltage is

$$\sigma_{\Re(V)\Im(V)} = \left(\sigma^2_v - \sigma^2_\theta(v^m_i)^2\right)\sin\theta\cos\theta = \sigma_{\Im(V)\Re(V)}.$$

Similarly the correlation between real and imaginary part of the current is

$$\sigma_{\Re(I)\Im(I)} = \left(\sigma^2_i - \sigma^2_\phi(i^m_i)^2\right)\sin\phi\cos\phi = \sigma_{\Im(I)\Re(I)}.$$

3.1.4 Closed Form Solution

Consider the linear model in (3.5), i.e,

$$Z = \mathcal{H}\mathbf{X} + \mathbf{e}$$

It is possible to rewrite the objective cost function $J(V,\Theta)$ as

$$J(\mathbf{X}) = \left[Z - \mathcal{H}\mathbf{X}\right]^T R^{-1}\left[Z - \mathcal{H}\mathbf{X}\right] \tag{3.7}$$

This cost function, being a linear function of the decision variable $\mathbf{X}$, reduces to the classical linear weighted least squares problem. It is well known that, if the matrix $(\mathcal{H}^T R^{-1} \mathcal{H})$ is not singular, then the optimal solution $\hat{\mathbf{X}}$ can be obtain in a closed form as

$$\hat{\mathbf{X}} = (\mathcal{H}^T R^{-1} \mathcal{H})^{-1} \mathcal{H}^T R^{-1} Z \tag{3.8}$$

Dealing with Gaussian additive noise this solution coincides with the *maximum likelihood* estimation.

3.1.5 $\mathcal{H}^T R^{-1} \mathcal{H}$ structure

This matrix does not assume a suitable form to be studied with some algebraic method. Anyway, a deep empirical analysis shows not only its invertibility but moreover its definite positivity as well. This assure the applicability of the algorithm proposed in most of the practical situations. (see [21])

3.2 Reactive Power Compensation

As explained in the Introduction, reactive power compensation is one of the key-problems in which are involved the electric grids. Now we want to present a control strategy similar to the one in [5]. Nevertheless, differently from [5], we will not suppose to have perfect measurements, but instead we want to deal with our noise measurements. Of course we cannot use them directly in the algorithm, because raw data are usually subject to these problems:

- they are too inaccurate;
- they can collect outliers.

This leads to the use of filter data able to let the control algorithm efficiently work.
To this point of view, the estimation represents the filtering of the measurements to achieve a better knowledge of the real state of the grid. It is the first necessary step to deal with a good control strategy.

Let us introduce the optimal reactive power flow problem, exploiting what done in [5], to better understand it.

3.2.1 Problem Formulation

The metric, for the optimality of reactive power flows, is considered to be the active power losses on the power lines. The total active power losses on the edges are then equal to

$$J \triangleq \sum_{e \in \mathcal{E}} |\xi_e|^2 \Re(z_e) = \bar{u}^T \Re(L) u$$

It is assumed to be possible to command only a subset $\mathcal{C} \subset \mathcal{V}$ of the nodes of the grid, named *compensators*. A part of them is assumed to be equipped with some sort of intelligence, as shown in figure 3.1 (u*upper panel*). In addition to this, we can work only on the amount of reactive power injected, as the decision on the amount of active power follows imperative economic criteria.

The problem of optimal reactive power injection at the compensators can be expressed as a quadratic, linearly constrained problem, in the form

$$\min_q \quad J(q), \quad \text{where } J(q) = \frac{1}{2} q^T \mathrm{Re}(\mathbf{X}) q \tag{3.9}$$

$$\text{subject to } \mathbb{1}^T q = 0$$

$$q_v = \mathrm{Im}(s_v), \ v \in \mathcal{V} \setminus \mathcal{C},$$

$\mathrm{Im}(s_v)$, $v \in \mathcal{V} \setminus \mathcal{C}$ being the nominal amount of reactive power injected by the nodes that cannot be commanded.
The challenging part to solving the problem is that each node has only local information.

To appropriately drive the microgenerators, the algorithm provided in [5] implements a distributed optimal reactive power compensation. Specifically the microgenerators are assumed to be organized into overlapping groups, namely *clusters*, each of which coordinated by a cluster header equipped with some intelligence unit (see figure 3.1).

3.2.2 Dual decomposition

In order to derive a control strategy to solve the ORPF problem, we apply the tool of dual decomposition to (3.9). While problem (3.9) might not be convex in general, we rely on the result presented in [22], which shows that zero duality gap holds for the ORPF problems, under some conditions that are commonly verified in practice. Based on this result, we use an approximate explicit solution of the nonlinear equations to derive a *dual ascent algorithm* that can be implemented by the agents. In order to present the approximate solution, we need the following technical lemma.

Lemma 3. *Let $\boldsymbol{L}$ be the complex valued Laplacian $\boldsymbol{L} := A^T \boldsymbol{Z}^{-1} A$. There exists a unique symmetric matrix $\boldsymbol{X} \in \mathbb{C}^{n \times n}$ such that*

$$\begin{cases} \boldsymbol{X}\boldsymbol{L} = I - \mathbb{1}\mathbb{1}_0^T \\ \boldsymbol{X}\mathbb{1}_0 = 0 \end{cases} \tag{3.10}$$

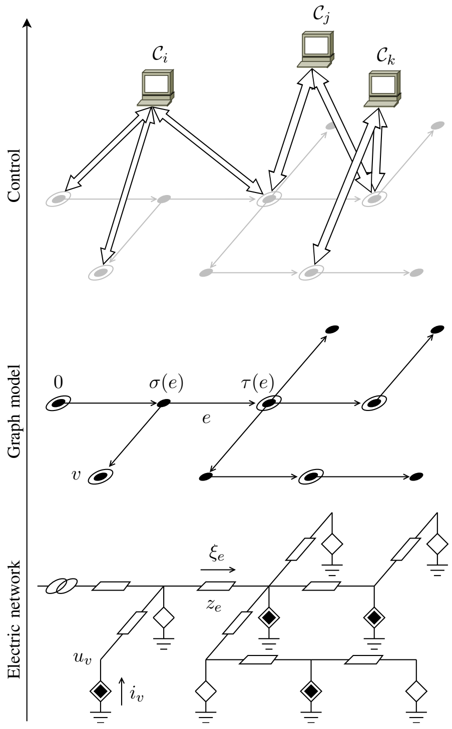


Figure 3.1: Different schematic view of the grid. **upper panel**: division of microgenerators into overlapping clusters to implement the distributed algorithm. **middle panel**: graph representation. Circled nodes represent microgenerators. **lower panel**: circuit representation. Black diamonds are microgenerators and white are loads.

where, we recall, $[\mathbb{1}_0]_v = 1$ *for* $v = 0$*, and 0 otherwise and* I *is the identity matrix.*

This matrix $\mathbf{X}$ depends only on the topology of the microgrid power lines and on their impedance.
The *effective impedance*, Z^{eff}_{uv}, of the power lines for every pair of nodes (u, v) can be represented by the following:

$$|Z|^{\text{eff}}_{uv} = (\mathbb{1}_u - \mathbb{1}_v)^T \mathbf{X} (\mathbb{1}_u - \mathbb{1}_v) \tag{3.11}$$

3.2.3 Distributed algorithm

The algorithm proposed in [5] is based only on a local knowledge, therefore any central controller is not needed. The algorithm can be distributed across the agents of the microgrid, that consist in decomposing the optimization problem into smaller issues.

All the compensators are divided into ℓ possibly overlapping sets $\mathcal{C}_1, \dots, \mathcal{C}_\ell$, with $\bigcup_{i=1}^{\ell} \mathcal{C}_i = \mathcal{C}$ and the nodes of the same set, called *cluster*, are able to communicate to each other, and they are therefore capable of coordinating their actions and sharing their measurements.
The proposed optimization algorithm consists of the following repeated steps:

1. a set $\mathcal{C}_{i(t)}$ is chosen at a certain discrete time $t = 0, 1, 2, \dots$ where $i(t) \in \{1, \dots, \ell\}$;
2. the agents in $\mathcal{C}_{i(t)}$, by coordinating their actions and communicating, determine the new feasible state that minimizes $J(q)$, solving the optimization subproblem in which all the nodes that are not in $\mathcal{C}_{i(t)}$ keep their states constant;
3. the agents in $\mathcal{C}_{i(t)}$ actuate the system by updating their state (the injected reactive power).

Partitioning q as

$$q = \begin{bmatrix} q_{\mathcal{C}} \\ q_{\mathcal{V}\setminus\mathcal{C}} \end{bmatrix}$$

where $q_{\mathcal{C}} \in \mathbb{R}^m$ are the controllable components and $q_{\mathcal{V}\setminus\mathcal{C}} \in \mathbb{R}^{m-n}$ are not controllable. According to this partition of q, it is possible also the partition of the matrix

$$\Re(\mathbf{X}) = \begin{bmatrix} M & N \\ N & Q \end{bmatrix} \tag{3.12}$$

Introduced also the matrices m×m

$$\Omega := \frac{1}{2m} \sum_{h,k \in \mathcal{C}} (\mathbb{1}_h - \mathbb{1}_k)(\mathbb{1}_h - \mathbb{1}_k)^T = I - \frac{1}{m}\mathbf{1}\mathbf{1}^T,$$

$$\Omega_i := \frac{1}{2|\mathcal{C}_i|} \sum_{h,k \in \mathcal{C}_i} (\mathbb{1}_h - \mathbb{1}_k)(\mathbb{1}_h - \mathbb{1}_k)^T =$$

$$= \operatorname{diag}(\mathbb{1}_{\mathcal{C}_i}) - \frac{1}{|\mathcal{C}_i|}\mathbb{1}_{\mathcal{C}_i}\mathbb{1}_{\mathcal{C}_i}^T$$

When the cluster $\mathcal{C}_i$ is fired its nodes perform *th* optimization:

$$q_{\mathcal{C}}^{opt,i} := \arg \min_{q'_{\mathcal{C}} \in q_{\mathcal{C}} + \mathcal{S}_i} J(q'_{\mathcal{C}}, q_{\mathcal{V}\setminus\mathcal{C}}) = q_{\mathcal{C}} - (\Omega_i M \Omega_i)^{\#} \nabla J, \tag{3.13}$$

where

$$\nabla J = M q_{\mathcal{C}} + N q_{\mathcal{V}\setminus\mathcal{C}} = [\operatorname{Re}(\mathbf{X}) q]_{\mathcal{C}} \in \mathbb{R}^m \tag{3.14}$$

is the gradient of $J(q_{\mathcal{C}}, q_{\mathcal{V}\setminus\mathcal{C}})$ with respect to the decision variables $q_{\mathcal{C}}$.
This is computed via a distributed way:
if $h \notin \mathcal{C}_i$ then $\left[q_{\mathcal{C}}^{opt,i}\right]_h = q_h$, if instead $h \in \mathcal{C}_i$ then:

$$\left[q_{\mathcal{C}}^{opt,i}\right]_h = q_h - \sum_{k \in \mathcal{C}_i} \left[(\Omega_i M \Omega_i)^{\#}\right]_{hk} [\nabla J]_k \tag{3.15}$$

The Hessian matrix can be computed a priori thanks only to local knowledge of the mutual effective impedances between pairs compensators. Defining $R^{\text{eff}}_{hk} = \Re(Z^{\text{eff}}_{hk})$ through some computation is obteined that:

$$\Omega_i M \Omega_i = -\frac{1}{2}\Omega_i R^{\text{eff}} \Omega_i. \tag{3.16}$$

Assume that nodes in $\mathcal{C}_i$ can measure the grid voltage at their point of connection. Each agent $k \in \mathcal{C}_i$ compute

$$\nu^{(i)}_k := \frac{1}{|\mathcal{C}_i|} \sum_{v \in \mathcal{C}_i} |u_v||u_k| \sin(\angle u_k - \angle u_v - \theta) \tag{3.17}$$

After some computations is it possible to write the estimate gradient as:

$$[\nabla J]_k = -\cos\theta(\text{Im}(\mathbb{1}^T_k X \bar{s})) \tag{3.18}$$

The iterative algorithm proposed, based on all the above considerations, works as follows: when the cluster $\mathcal{C}_i$ is activated the state of all the system becomes $q_h(t+1) = q_h$ for all $h \notin \mathcal{C}_i$, while the node $h \in \mathcal{C}_i$ will inject the new reactive power

$$q_h(t+1) = q_h - \cos\theta \sum_{k \in \mathcal{C}_i} \left[(\Omega_i R^{\text{eff}} \Omega_i)^{\#}\right]_{hk} \nu^{(i)}_k(t), \tag{3.19}$$

As we know the algorithm can be implemented by the agents of the microgrid in a distributed way. In a preliminary, offline phase, each cluster computes $(\Omega_i R^{\text{eff}} \Omega_i)^{\#}$ then, at every iteration of the algorithm:

- a cluster $\mathcal{C}_i$ is randomly chosen;
- every agent h not belonging to the cluster $\mathcal{C}_i$ holds its injected reactive power constant;
- every agent h belonging to the cluster $\mathcal{C}_i$ senses the grid voltage at its point of connection, computes $\nu^{(i)}_h$, and then updates its injected reactive power according to 3.19.

3.2.4 Estimation and RPC

In order to underline the importance of State Estimation, we want to analyze how the control algorithm is performing using filtered data (instead of raw data). To this purpose, in chapter 5 we will present a lot of tests, using both raw data and estimated data with the same algorithm. We will consider also a slightly different algorithm, proposed in [13], in which are considered upper and lower bounds for the reactive power that each node can share with the others. Thanks to those test, we will be able to show that the performances of the algorithm using the estimated state are comparable to those obtained using the grid real state (with no noise), while raw measurements cause an algorithm behaviour totally undesirable.

Chapter 4

Distributed Estimation Algorithms

We now propose two completely **distributed** and **scalable** algorithms in order to solve the estimation problem presented in the previous chapter. The first one is the *ADMM Estimator*, for which we will prove the convergence to the optimal value, while the second is the *Jacobi-approximate Estimator*, a faster distributed algorithm, very useful if there is the concrete possibility to employ better resources in some grid measurements.

Before describing the distributed algorithms, however, it is important to investigate on why it is preferable to implement this kind of algorithms respect to the centralized ones. There are a lot of advantages, not only from the economical point of view, in managing whatever problem, if it is possible, in a distributed way:

- we can avoid the huge computational effort, growing with the grid size, of a centralized algorithm. Distributed algorithm are much "lighter" from the computational point of view.
- we do not need to employ a central intelligence with which supervise the entire grid, as for a distributed algorithm is sufficient a local knowledge.
- from the robustness point of view, imagine that an error occurs during the running of the centralized algorithm. Then the entire procedure fails and no information will be computed. In a distributed algorithm different procedures are computed separately (often in an asynchronous way), so the robustness is improved a lot.

4.1 Multi Area Decomposition

Let us introduce a decomposition for the grid suitable for the algorithms that are going to be presented.

Let us divide the grid into m non overlapping subset. Each subset represents a microgrid. Adjacent areas are supposed to be connected through tie lines, called *border lines*, as shown in figure 4.1.

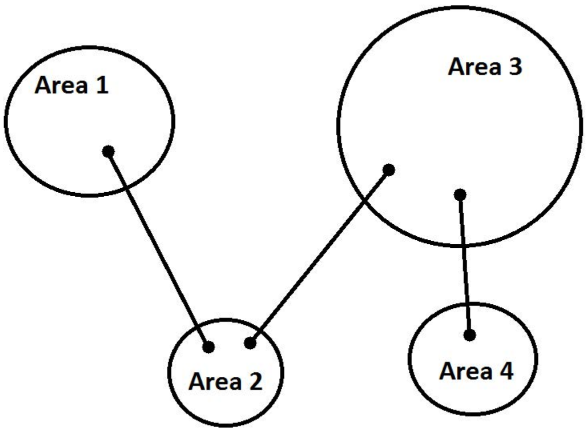


Figure 4.1: Grid divided into non overlapping areas

For each area $a \in [1, ..., m]$ we have:

$\mathbf{X_a}$ internal state;

Z_a internal measures;

L_a internal inductance matrix (describing the internal topology);

Let $\Omega_a \subset [1, ..., m]$ be the subset of *adjacent areas* of a.
Then, for each $b \in \Omega_a$ we have:

Z_{ab} measures of the nodes of area b that direct communicate with some node of area a;

L_{ab} inductance matrix between area a and $b \in \Omega_a$ (describes the communication topology).

Figure 4.2 well explain the quantities introduced.

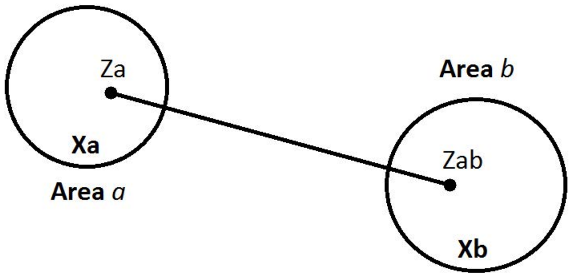


Figure 4.2: Information relating two adjacent areas

4.2 Alternating Direction Multiplier Method

The first solution we propose is based on the Alternating Direction Multiplier Method (ADMM). This is an optimization technique based on the iterative solution of an *augmented Lagrangian* problem.

It is well known from the literature that the the classical ADMM, because of its structure, can be implemented in a distributed way. However, the flow of informations through different areas do not concern only local informations, but global informations. This is a crucial point, that does not make the algorithm scalable.

The novelty of our solution is based on [12] and on [21]. We will show how to implement a local and scalable ADMM algorithm to solve the estimation problem, and moreover we will show a complete proof about its convergence to the solution of the equation (3.8).

Let us firstly introduce the classical ADMM procedure. Afterwards we will present the scalable solution proposed.

4.2.1 ADMM classical

Considering the system 3.5, let us introduce a new notation, more suitable for the ADMM algorithm that we are going to show, imposing

$$m = \begin{bmatrix} m^{(1)} \\ \vdots \\ m^{(N)} \end{bmatrix} \equiv Z; \qquad A = \begin{bmatrix} A_{11} & \cdots & A_{1N} \\ \vdots & & \vdots \\ A_{N1} & \cdots & A_{NN} \end{bmatrix} \equiv \mathcal{H};$$

$$X = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} \equiv \mathbf{X}; \qquad R = \begin{bmatrix} R_1 & & \\ & \ddots & \\ & & R_N \end{bmatrix} \equiv R.$$

It is possible to define the quadratic function, based on these new matrices, more specifically on each row or block of rows,

$$f_i(X) = \big(\sum_{j=1}^{N} A_{ij}x_j - m^{(i)}\big)^T R_i^{-1} \big(\sum_{j=1}^{N} A_{ij}x_j - m^{(i)}\big)$$

Collecting all function f_i, $\forall i = 1 \dots N$, is easy to get

$$F(X) = \sum_{i=1}^{N} f_i(X) = \big(AX - m\big)^T R^{-1} \big(AX - m\big) \tag{4.1}$$

whose optimal solution, $\hat{X}$, is the well known

$$\hat{X} = \big(A^T R^{-1} A\big)^{-1} A^T R^{-1} m$$

Let us define $X^{(i)}$, $i \in [1, \dots, N]$, as the i^{th} copy of the vector X, owning to area i. Thanks to this it is possible to rewrite the minimization problem referred to equation in (4.1) as

$$\underset{X^{(1)} \dots X^{(N)}}{\text{minimize}} \sum_{i=1}^{N} f_i\big(X^{(i)}\big) \qquad s.t. \quad X^{(i)} = X^{(j)} \ \forall j \in \mathcal{N}_i \cup \{i\}$$

where $\mathcal{N}_i$ represents the subset of indices of areas adjacent to area i, but excluding area i itself.

It is then possible to solve the minimization problem through the augmented Lagrangian technique, introducing some redundant bonds that allow us to manage the solution with the ADMM algorithm, as

$$\underset{X^{(1)} \dots X^{(N)}}{\text{minimize}} \sum_{i=1}^{N} f_i\big(X^{(i)}\big) \qquad s.t. \quad X^{(i)} = z_{ij}; \ X^{(i)} = z_{ji} \ \forall j \in \mathcal{N}_i \cup \{i\}$$

The correspondent Lagrangian function is

$$\begin{aligned} \mathcal{L} &= \sum_{i=1}^{N} f_i\big(X^{(i)}\big) + \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i \cup \{i\}} \lambda_{ij}^T \big(X^{(i)} - z_{ij}\big) + \mu_{ij}^T \big(X^{(i)} - z_{ji}\big) + \\ & + \frac{c}{2} \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i \cup \{i\}} ||X^{(i)} - z_{ij}||^2 + ||X^{(i)} - z_{ji}||^2 \end{aligned}$$

This function can be solved through ADMM algorithm, which consists of three main updating steps:

1.
$$\begin{cases} \lambda_{ij}(t) &= \lambda_{ij}(t-1) + c\big(X^{(i)}(t) - z_{ij}(t)\big) \\ \mu_{ij}(t) &= \mu_{ij}(t-1) + c\big(X^{(i)}(t) - z_{ji}(t)\big) \end{cases} \tag{4.2}$$

2.
$$X^{(i)}(t+1) = \underset{X^{(i)}}{\arg\min}\ \mathcal{L}(X, z(t), \lambda(t), \mu(t)) \tag{4.3}$$

3.
$$z_{ij}(t+1) = \underset{z_{ij}}{\arg\min}\ \mathcal{L}(X(t+1), z, \lambda(t), \mu(t)) \tag{4.4}$$

With some manipulations of the updating step (proof is reported in the appendix A) it is possible to rewrite the algorithm in a simpler way consisting of only two updating step:

1.
$$\lambda_{ij}(t) = \lambda_{ij}(t-1) + \frac{c}{2}\big(X^{(i)}(t) - X^{(j)}(t)\big) \tag{4.5}$$

2.
$$X^{(i)}(t+1) = \underset{X^{(i)}}{\arg\min}\ \mathcal{L}(X, \lambda(t)) \tag{4.6}$$

In this algorithm every area estimates the entire grid state, therefore the flow of informations between areas is global and not local.

We are interested, instead, in a kind of algorithm in which every area is able to compute its inner state $X^{(i)}$ in a distributed and local fashion, i.e. only from the knowledge of its adjacent areas state.

4.2.2 ADMM scalable with Projector matrices

The first formulation we propose is the one in [21]. Let us start from the usual state of the art formulation of ADMM minimization problem

$$\underset{X^{(1)}\dots X^{(N)}}{\text{minimize}} \sum_{i=1}^{N} f_i\big(X^{(i)}\big) \quad s.t. \quad X^{(i)} = z_{ij};\ X^{(i)} = z_{ji} \qquad \forall i = 1\dots N,\ \forall j \in \mathcal{N}_i \tag{4.7}$$

We remind that $X^{(1)}, \dots, X^{(N)}$ are local copies of X

$$X = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix}$$

where x_i represents the inner state of area i.

To force the exchange of only local information between adjacent areas we introduce the projector matrix P_i of dimension $(\sum_{i=1}^{N} n_i \times \sum_{i=1}^{N} n_i)$, where n_i represents the dimension of the state of each area i.

$$P_i = \begin{bmatrix} O_{n_1} & & & & \\ & \ddots & & & \\ & & I_{n_i} & & \\ & & & \ddots & \\ & & & & O_{n_N} \end{bmatrix}$$

which is a diagonal matrix with the i^{th} block equal to an identity matrix of dimension n_i and zeros elsewhere. By multiplying $P_i \cdot X$, we are able to extract the only component x_i from the vector X. Indeed,

$$P_i X = \begin{bmatrix} 0 \\ \vdots \\ x_i \\ \vdots \\ 0 \end{bmatrix}$$

Similarly it is possible to define the joint projector

$$P_{ij} = \begin{cases} P_i & i = j \\ P_i + P_j & i \neq j \end{cases}$$

Note that, using the projector matrices, it is possible to rewrite the initial problem as

$$\begin{aligned} &\underset{X^{(1)}\cdots X^{(N)}}{\text{minimize}} \sum_{i=1}^{N} f_i\big(X^{(i)}\big) \qquad (4.8) \\ s.t.\quad &P_{ij}X^{(i)} = P_{ij}z_{ij};\ P_{ij}X^{(i)} = P_{ij}z_{ji} \qquad \forall i = 1 \dots N,\ \forall j \in \mathcal{N}_i \end{aligned}$$

where the P matrices forces to involve only local information between adjacent areas.

From the previous proofs it is known that:

- $\lambda_{ij}(t) = \lambda_{ij}(t-1) + \frac{c}{2}\big(X^{(i)}(t) - X^{(j)}(t)\big)$;
- $z_{ij}(t) = \frac{X^{(i)}(t)+X^{(j)}(t)}{2}$;
- $\lambda_{ij}(t) = \mu_{ij}(t) = -\lambda_{ji}(t) = -\mu_{ji}(t) \ \ \forall t$;

In addition, it is known that $z_{ij} = z_{ji}$, and so it is possible to rewrite the Lagrangian as

$$\mathcal{L}(X,z,\lambda) = \sum_{i=1}^{N} f_i\big(X^{(i)}\big) + \sum_{i=1}^{N} \sum_{j\in\mathcal{N}_i\cup\{i\}} \Big[2\lambda_{ij}^T P_{ij}\big(X^{(i)} - z_{ij}\big) + c(X^{(i)} - z_{ij})^T P_{ij}(X^{(i)} - z_{ij})\Big]$$

By eliminating the constant terms and using the previous equations, we get

$$X^{(i)}(t+1) = \arg\min_{X^{(i)}} \left\{ f_i(X^{(i)}) + \sum_{j\in\mathcal{N}_i\cup\{i\}} X^{(i)T} P_{ij}\big(2\lambda_{ij}(t) + cX^{(i)} - 2cz_{ij}(t)\big) \right\}$$

Finally, through the first order optimality condition, we can obtain the optimal updating step in a closed form solution

$$X^{(i)}(t+1) = \big(A^{(i)T}R_i^{-1}A^{(i)} + cM_i\big)^{-1}\big[A^{(i)T}R_i^{-1}m^{(i)} + U^{(i)}(t) - \Lambda^{(i)}(t)\big]$$

where

$$\begin{aligned} M_i &= \sum_{j\in\mathcal{N}_i\cup\{i\}} P_{ij} \\ \Lambda^{(i)}(t) &= M_i\lambda_{ij}(t) \\ U^{(i)}(t) &= cM_i z_{ij}(t) \end{aligned}$$

Note that

- if $P_i = I \,\forall i = 1 \dots N$, the algorithm turn to be equal to classical implementation (4.7);
- This particular formulation of ADMM is completey local. Indeed, in M_i, U_i, Λ_i is considered only local information, (terms depending on $j \in \mathcal{N}_i$), so the remaining parts of the vector could be neglected in the computation. This makes the algorithm fully scalable and local;

4.2.3 ADMM scalable and compact

By inspecting the P_i matrices, it is easy to note that each one of them is a *sparse* matrix. Indeed the larger part of its components is equal to 0 due to the non-communication between non-adjacent areas. Now we want to re-write this scalable algorithm in a more compact way, considering only the components really involved in the minimization each time. Given

$$X^{(i)} = \begin{bmatrix} x_1^{(i)} \\ \vdots \\ x_N^{(i)} \end{bmatrix}$$

as a local copy of X owning to area i, the quadratic function we have to minimize is

$$f_i(X^{(i)}) = \Big(\sum_{j \in \mathcal{N}_i \cup \{i\}} A_{ij} x_j^{(i)} - m^{(i)} \Big)^T R_i^{-1} \Big(\sum_{j \in \mathcal{N}_i \cup \{i\}} A_{ij} x_j^{(i)} - m^{(i)} \Big)$$

hence we can write

$$f_i(X^{(i)}) = f_i(x_j^{(i)}) \quad \forall j \in \mathcal{N}_i \cup \{i\}$$

We define the vector $Y^{(i)}$, that collects in a compact way all the states useful for area i, that are $x_i^{(i)}$ and all the $x_j^{(i)}$ $\forall j \in \mathcal{N}_i$

$$Y^{(i)} = \begin{bmatrix} x_i^{(i)} \\ \left\{ x_j^{(i)} \right\}_{j \in \mathcal{N}_i} \end{bmatrix}$$

To be precise, this notation does not respect the sorting of the vector components, because $i = 1 \dots N$. Formally we should write

$$Y^{(i)} = \begin{bmatrix} x_{j_1}^{(i)} \\ \vdots \\ x_{j_{|\mathcal{N}_i|+1}}^{(i)} \end{bmatrix} \qquad i \in \{j_1, \dots, j_{|\mathcal{N}_i|+1}\}$$

but the previous notation is much simpler and clear, so we will use that one, generalizing for every $i = 1 \dots N$.

Now it is possible to rewrite the initial problem as

$$\begin{aligned} &\underset{Y^{(1)}\dots Y^{(N)}}{\text{minimize}} \sum_{i=1}^{N} f_i\big(Y^{(i)}\big) \\ s.t.\quad & x_j^{(i)} = x_j^{(j)};\ x_i^{(i)} = x_i^{(j)} \qquad \forall i = 1,\dots,N, \quad \forall j \in \mathcal{N}_i \end{aligned} \tag{4.9}$$

Applying the ADMM procedure to this problem and exploiting the first order optimality conditions, one can obtain the closed form solution, which is equal to

$$Y^{(i)}(t+1) = \Big[\big(\tilde{A}^{(i)T} R_i^{-1} \tilde{A}^{(i)} + c\tilde{M}_i\big)\Big]^{-1} \Big[\tilde{A}^{(i)T} R_i^{-1} m^{(i)} + \tilde{U}^{(i)}(t) - \tilde{\Lambda}^{(i)}(t)\Big]$$

where

- $\tilde{A}$ is obtained due to "compacting" the matrix $A^{(i)}$, considering only non-zero columns, that are A_{ij} columns with $j \in \mathcal{N}_i$.
  (ex. $A^{(i)} = [A_{i1}\ 0\ A_{i3}\ A_{i4}\ 0\ \cdots\ 0] \implies \tilde{A}^{(i)} = [A_{i1}\ A_{i3}\ A_{i4}]$);
- $\tilde{M}_i$ is obtained due to "compacting" the term $M_i = \sum_{j\in\mathcal{N}_i} P_{ij}$ into a matrix of dimension $|\mathcal{N}_i + 1| \times |\mathcal{N}_i + 1|$, with $|\mathcal{N}_i| + 1$ in the position corresponding to index i and 1 in the position corresponding to indexes j, $j \in \mathcal{N}_i$;

The other terms involved in the minimization are

$$\tilde{U}^{(i)}(t) = \begin{bmatrix} u_i^{(i)}(t) \\ \Big\{u_j^{(i)}(t)\Big\}_{j\in\mathcal{N}_i} \end{bmatrix} = \begin{bmatrix} \frac{c}{2}\sum_{j\in\mathcal{N}_i}\big(x_i^{(i)}(t) + x_i^{(j)}(t)\big) \\ \Big\{\frac{c}{2}\big(x_j^{(i)}(t) + x_j^{(j)}(t)\big)\Big\}_{j\in\mathcal{N}_i} \end{bmatrix}$$

$$\tilde{\Lambda}^{(i)}(t) = \begin{bmatrix} \lambda_i^{(i)}(t) \\ \Big\{\lambda_j^{(i)}(t)\Big\}_{j\in\mathcal{N}_i} \end{bmatrix} = \begin{bmatrix} \lambda_i^{(i)}(t-1) + \frac{c}{2}\sum_{j\in\mathcal{N}_i}\big(x_i^{(i)}(t) - x_i^{(j)}(t)\big) \\ \Big\{\lambda_j^{(i)}(t-1) + \frac{c}{2}\big(x_j^{(i)}(t) - x_j^{(j)}(t)\big)\Big\}_{j\in\mathcal{N}_i} \end{bmatrix}$$

In order to show explicitly the distributed and asynchronous properties of this algorithm, we provide the explicit ADMM procedure for this formulation.

First of all we can write the minimization problem adding to the constraints some redundant variables that allow as to manage the solution with the ADMM algorithm:

$$\begin{aligned} &\underset{Y^{(1)}\dots Y^{(N)}}{\text{minimize}} \sum_{i=1}^{N} f_i\big(Y^{(i)}\big) \\ \text{s.t.}\quad & x_i^{(i)} = z_i^{(i,j)};\ x_j^{(i)} = z_j^{(i,j)} \\ & x_i^{(i)} = z_i^{(j,i)};\ x_j^{(i)} = z_j^{(j,i)} \qquad \forall i = 1,\dots,N, \quad \forall j \in \mathcal{N}_i \end{aligned} \tag{4.10}$$

This formulation leads to a Lagrangian function equal to

$$\begin{aligned}\mathcal{L} &= \sum_{i=1}^{N}\Bigg\{ f_i\big(Y^{(i)}\big) + \sum_{j\in\mathcal{N}_i\cup\{i\}}\Big[\lambda_i^{(i,j)}\big(x_i^{(i)} - z_i^{(i,j)}\big) + \lambda_j^{(i,j)}\big(x_j^{(i)} - z_j^{(i,j)}\big)\Big] + \\ &\quad + \sum_{j\in\mathcal{N}_i\cup\{i\}}\Big[\mu_i^{(i,j)}\big(x_i^{(i)} - z_i^{(j,i)}\big) + \mu_j^{(i,j)}\big(x_j^{(i)} - z_j^{(j,i)}\big)\Big] + \\ &\quad + \frac{c}{2}\sum_{j\in\mathcal{N}_i\cup\{i\}}\Big[||x_i^{(i)} - z_i^{(i,j)}||^2 + ||x_j^{(i)} - z_j^{(i,j)}||^2 + ||x_i^{(i)} - z_i^{(j,i)}||^2 + ||x_j^{(i)} - z_j^{(j,i)}||^2\Big]\Bigg\}\end{aligned}$$

The ADMM updating steps are:

1.

$$\begin{cases} \lambda_i^{(i,j)}(t) = \lambda_i^{(i,j)}(t-1) + c\big(x_i^{(i)}(t) - z_i^{(i,j)}(t)\big) \\ \lambda_j^{(i,j)}(t) = \lambda_j^{(i,j)}(t-1) + c\big(x_j^{(i)}(t) - z_j^{(i,j)}(t)\big) \\ \mu_i^{(i,j)}(t) = \mu_i^{(i,j)}(t-1) + c\big(x_i^{(i)}(t) - z_i^{(j,i)}(t)\big) \\ \mu_j^{(i,j)}(t) = \mu_j^{(i,j)}(t-1) + c\big(x_j^{(i)}(t) - z_j^{(j,i)}(t)\big) \end{cases} \tag{4.11}$$

2.

$$Y^{(i)}(t+1) = \underset{Y^{(i)}}{\operatorname{argmin}}\ \mathcal{L}(Y^{(i)}, z(t), \lambda(t), \mu(t)) \tag{4.12}$$

3.

$$z(t+1) = \underset{z}{\operatorname{argmin}}\ \mathcal{L}(Y^{(i)}(t+1), z, \lambda(t), \mu(t)) \tag{4.13}$$

where, in the last two steps, for λ, μ and z we mean respectively all the variables $\lambda_*^{(*,*)}$, $\mu_*^{(*,*)}$ and $z_*^{(*,*)}$ involved in the lagrangian term.

We suppose that node i has in memory:

$$Y^{(i)}(t) = \begin{bmatrix} x_i^{(i)}(t) \\ \Big\{x_j^{(i)}(t)\Big\}_{j\in\mathcal{N}_i} \end{bmatrix};\quad Z^{(i)}(t) := \begin{bmatrix} z_i^{(i,j)}(t) \\ \Big\{z_j^{(i,j)}(t)\Big\}_{j\in\mathcal{N}_i} \end{bmatrix}$$

$$\Lambda^{(i)}(t) = \begin{bmatrix} \lambda_i^{(i,j)}(t) \\ \Big\{\lambda_j^{(i,j)}(t)\Big\}_{j\in\mathcal{N}_i} \end{bmatrix};\quad \mathcal{M}^{(i)}(t) := \begin{bmatrix} \mu_i^{(i,j)}(t) \\ \Big\{\mu_j^{(i,j)}(t)\Big\}_{j\in\mathcal{N}_i} \end{bmatrix}$$

Before doing step (4.11), it is clear that we need a sharing, and hence a communication between neighbour nodes, because of the usage of the terms $z_i^{(j,i)}$ and $z_j^{(j,i)}$, that are in the memory of the node j but not in the i's one.

Thanks to this, by inspecting the lagrangian function it is easy to note that the minimization step in (4.12) can be done asynchronously by the algorithm, because the terms involved in the minimization belong all to the area i.

Finally, before doing step (4.13), asynchronously as before, we need a sharing between neighbour areas again. Indeed, considering for example the minimization in which are involved the terms $z_j^{(i,j)}$, we have

$$z_j^{(i,j)}(t+1) = \underset{z_j^{(i,j)}}{\operatorname{argmin}}\ \mathcal{L}(Y(t+1), z_j^{(i,j)}, \lambda(t), \mu(t))$$

Discarding terms that are not depending on $z_j^{(i,j)}$, the minimization becomes

$$z_j^{(i,j)}(t+1) = \underset{z_j^{(i,j)}}{\operatorname{argmin}}\Big\{\lambda_j^{(i,j)}(t)\Big(x_j^{(i)}(t+1) - z_j^{(i,j)}(t+1)\Big) + \frac{c}{2}||x_j^{(i)}(t+1) - z_j^{(i,j)}(t+1)||^2 + \\ +\mu_j^{(j,i)}(t)\Big(x_j^{(j)}(t+1) - z_j^{(i,j)}(t+1)\Big) + \frac{c}{2}||x_j^{(j)}(t+1) - z_j^{(i,j)}(t+1)||^2\Big\}$$

exploiting the first order optimality condition, we obtain

$$z_j^{(i,j)}(t+1) = \frac{x_j^{(i)}(t+1) + x_j^{(j)}(t+1)}{2} - \frac{\lambda_j^{(i,j)}(t) + \mu_j^{(j,i)}(t)}{2c}$$

From this it is clear that we need to know the terms $\{x_j^{(j)}(t+1)\}_{j\in\mathcal{N}_i}$ and $\{\mu_j^{(j,i)}(t)\}_{j\in\mathcal{N}_i}$. With a similar process it is easy to demonstrate that in the computation of the variables $z_j^{(j,i)}$ it is necessary to know the terms $x_j^{(j)}(t+1)$ and $\lambda_j^{(j,i)}(t)$. Therefore we need a sharing of these terms, because they are not in the memory of the node i.

We have shown a fundamental property: this algorithm works completely in ad asynchronous way, hence the minimization steps are completely separable in a distributed way.

4.2.4 Convergence of ADMM scalable

In this subsection we will show that the ADMM scalable converges to the optimal solution for the problem 4.1. The proof exploits the work done in [23], which treats problems in the form

$$\begin{aligned} \underset{x,z}{\text{minimize}} \quad & F(x) + G(z) \\ \text{subject to} \quad & Ax + Bz = c \end{aligned} \tag{4.14}$$

with variables $x \in \mathbb{R}^n$ and $z \in \mathbb{R}^m$, where $A \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{p\times m}$ and $c \in \mathbb{R}^p$.

We know that, from [23], under these assumptions the algorithm converges to the optimal value for (4.14), that is:

$$p^* = \inf\big\{F(x) + G(z) \mid Ax + Bz = c\big\}$$

After formulating the augmented Lagrangian

$$\mathcal{L}_\rho(x,z,\lambda) = F(x) + G(z) + \lambda^T(Ax + Bz - c) + (\rho/2)||Ax + Bz - c||_2^2$$

we can write the ADMM iterations, that are

$$\begin{aligned} \lambda^k &:= \lambda^{k-1} + \rho(Ax^k + Bz^k - c) \\ x^{k+1} &:= \operatorname*{argmin}_x \mathcal{L}_\rho(x, z^k, \lambda^k) \\ z^{k+1} &:= \operatorname*{argmin}_z \mathcal{L}_\rho(x^{k+1}, z, \lambda^k) \end{aligned}$$

where $\rho > 0$.

Therefore, in order to prove the convergence in our particular case of study, it is sufficient to show that our problem is rewritable in in the form (4.14).

Proof. First of all it is easy to note that in our particular case of study F and G are closed, proper and convex (in fact, quadratic). In particular we have $G(z) \equiv 0$ and

$$F(x) = \sum_{i=1}^N f_i(x) = \big(Ax - m\big)^T R^{-1} \big(Ax - m\big) \tag{4.15}$$

We start from the compact formulation of ADMM

$$\begin{aligned} &\underset{Y^{(1)}\dots Y^{(N)}}{\text{minimize}} \sum_{i=1}^N f_i\big(Y^{(i)}\big) \\ &\text{s.t.} \quad x_j^{(i)} = x_j^{(j)};\ x_i^{(i)} = x_i^{(j)} \qquad \forall i = 1,\dots,N, \quad \forall j \in \mathcal{N}_i \end{aligned} \tag{4.16}$$

where, we remind,

$$Y^{(i)} = \begin{bmatrix} x_i^{(i)} \\ \left\{x_j^{(i)}\right\}_{j\in\mathcal{N}_i} \end{bmatrix}$$

Let us consider the constraints in (4.10). There are, for each node i:

- $|\mathcal{N}_i|$ constraints for $x_i^{(i)} = z_i^{(i,j)}$ (one for each $j \in \mathcal{N}_i$);
- $|\mathcal{N}_i|$ constraints for $x_i^{(i)} = z_i^{(j,i)}$ (one for each $j \in \mathcal{N}_i$);
- $|\mathcal{N}_i|$ constraints for $x_j^{(i)} = z_j^{(i,j)}$ (one for each $j \in \mathcal{N}_i$);
- $|\mathcal{N}_i|$ constraints for $x_j^{(i)} = z_j^{(j,i)}$ (one for each $j \in \mathcal{N}_i$).

Let us define

$$\bar{Z}^{(i)} = \begin{bmatrix} \left\{ z_i^{(i,j)} \right\}_{j\in\mathcal{N}_i} \\ \left\{ z_i^{(j,i)} \right\}_{j\in\mathcal{N}_i} \\ \left\{ z_j^{(i,j)} \right\}_{j\in\mathcal{N}_i} \\ \left\{ z_j^{(j,i)} \right\}_{j\in\mathcal{N}_i} \end{bmatrix}$$

$\bar{Z}^{(i)}$ has dimension $4|\mathcal{N}_i| \times 1$. Let us define moreover

$$A^{(i)} = \left[\begin{array}{c|ccc} 1 & 0 & & \\ \vdots & & \ddots & \\ 1 & & & 0 \\ \hline 1 & 0 & & \\ \vdots & & \ddots & \\ 1 & & & 0 \\ \hline 0 & 1 & & \\ \vdots & & \ddots & \\ 0 & & & 1 \\ \hline 0 & 1 & & \\ \vdots & & \ddots & \\ 0 & & & 1 \end{array}\right] ; \qquad B^{(i)} = -I_{4|\mathcal{N}_i|}$$

Each block of $A^{(i)}$ has dimension $|\mathcal{N}_i| \times |\mathcal{N}_i| + 1$. Using these new variables it is possible to write in a compact way all the $4|\mathcal{N}_i|$ constraints of (4.10):

$$A^{(i)}Y^{(i)} + B^{(i)}\bar{Z}^{(i)} = \underline{0} \tag{4.17}$$

By collecting all $A^{(i)}$, $B^{(i)}$, $Y^{(i)}$ and $\bar{Z}^{(i)}$ we can define

$$A = \begin{bmatrix} A^{(1)} \\ \vdots \\ A^{(N)} \end{bmatrix} ; \qquad B = \begin{bmatrix} B^{(1)} \\ \vdots \\ B^{(N)} \end{bmatrix}$$

$$x = \begin{bmatrix} Y^{(1)} \\ \vdots \\ Y^{(N)} \end{bmatrix} ; \qquad z = \begin{bmatrix} \left\{ z_1^{(1,j)} \right\}_{j\in\mathcal{N}_1} \\ \left\{ z_1^{(j,1)} \right\}_{j\in\mathcal{N}_1} \\ \vdots \\ \left\{ z_N^{(N,j)} \right\}_{j\in\mathcal{N}_N} \\ \left\{ z_N^{(j,N)} \right\}_{j\in\mathcal{N}_N} \end{bmatrix} \tag{4.18}$$

and obtain directly the formulation $Ax + Bz = c$ as in 4.14, with $c = \underline{0}$. $\square$

4.3 Jacobi approximate algorithm

Before seeing the reactive power compensation algorithms with the ADMM estimation, we want to present briefly a new approximate Jacobi-like algorithm, distributed and faster than ADMM. This algorithm, differently from ADMM, does not converge to the optimal solution, but it can have a high practical relevance if we are able to make some measurements more accurate than the others.

For a deeper mathematical analysis of the Jacobi algorithm we remind to [21].

Consider a linear model of the grid similar of that introduced in subsection 3.1.3. Suppose the grid divided into m areas according with section 4.1. It is possible to suitably sort the state vector, composed of the real and imaginary part of the voltage nodes, as

$$\mathbf{X} = \begin{bmatrix} \mathbf{X_1} \\ \mathbf{X_2} \\ \vdots \\ \mathbf{X_m} \end{bmatrix} = \begin{bmatrix} X_1 \; Y_1 \mid X_2 \; Y_2 \mid \cdots \mid X_m \; Y_m \end{bmatrix}^T$$

This lets to highlight the single area state. Accordingly, the inductance matrix L becomes

$$L = \begin{bmatrix} L_{11} & L_{12} & \cdots & L_{1m} \\ L_{21} & L_{22} & \cdots & L_{2m} \\ & & \vdots & \\ L_{m1} & L_{m2} & & L_{mm} \end{bmatrix}$$

where L_{ij} represents the part of L concerning area i and j; it identifies the communication edges as well as the admittance line values.
Similarly, we have that the measures Z and noise $\mathbf{e}$ can be expressed as

$$Z = \begin{bmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_m \end{bmatrix} = \begin{bmatrix} S_1 \\ R_1 \\ H_1 \\ K_1 \\ -- \\ \vdots \\ -- \\ S_m \\ R_m \\ H_m \\ K_m \end{bmatrix} ; \quad \mathbf{e} = \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \vdots \\ \mathbf{e}_m \end{bmatrix} = \begin{bmatrix} \mathbf{e}_{S_1} \\ \mathbf{e}_{R_1} \\ \mathbf{e}_{H_1} \\ \mathbf{e}_{K_1} \\ -- \\ \vdots \\ -- \\ \mathbf{e}_{S_m} \\ \mathbf{e}_{R_m} \\ \mathbf{e}_{H_m} \\ \mathbf{e}_{K_m} \end{bmatrix}$$

It follows a correlation matrix R, for the noise vector $\mathbf{e}$, equal to

$$R = E[\mathbf{e}\mathbf{e}^T] = \begin{bmatrix} R_1 & & & \\ & R_2 & & \\ & & \ddots & \\ & & & R_m \end{bmatrix}$$

R_i has a structure similar to that seen in section 3.1.3 but it concernes only measurements taken from the same area.

Accordingly to the state sorting, the matrix of model (3.5) will be characterized by a similar structure. Specifically

$$\mathcal{H} = \begin{bmatrix} \mathcal{H}_1 \\ \vdots \\ \mathcal{H}_m \end{bmatrix}$$

Thanks to this sorting it is then possible to outline a specific linear model for every area $i \in [1, \dots, m]$ of the form

$$Z_i = \mathcal{H}_i \mathbf{X} + \mathbf{e}_i = \mathcal{H}_{ii}\mathbf{X_i} + \sum_{j \neq i} \mathcal{H}_{ij}\mathbf{X_j} + \mathbf{e}_i;$$

Note that the matrix model is equal to

$$\begin{aligned} \mathcal{H}_i &= \begin{bmatrix} \mathcal{H}_{i1} & \cdots & \mathcal{H}_{ii} & \cdots & \mathcal{H}_{im} \end{bmatrix} \\ &= \left[\begin{array}{cc|c|cc|c|cc} 0 & 0 & & I_{n_i} & 0 & & 0 & 0 \\ 0 & 0 & \cdots & 0 & I_{n_i} & \cdots & 0 & 0 \\ \Re(L_{i1}) & -\Im(L_{i1}) & & \Re(L_{ii}) & -\Im(L_{ii}) & & \Re(L_{im}) & -\Im(L_{im}) \\ \Im(L_{i1}) & \Re(L_{i1}) & & \Im(L_{ii}) & \Re(L_{ii}) & & \Im(L_{im}) & \Re(L_{im}) \end{array}\right] \end{aligned}$$

whose block $\mathcal{H}_{ii}$ is relative to the inner state and blocks $\mathcal{H}_{ij}$ to other areas state.

Distributed Solution

In section 3.1.4 it has been shown that the optimal global solution to the centralized problem is equal to

$$\hat{\mathbf{X}} = \left(\mathcal{H}^T R^{-1} \mathcal{H}\right)^{-1} \mathcal{H}^T R^{-1} Z$$

Exploiting the Jacobi procedure, the closed form solution, in a Jacobi point of view, can be rewritten as

$$\hat{\mathbf{X}}_\mathbf{i}(t+1) = \left(\mathcal{H}_{ii}^T R_{ii}^{-1} \mathcal{H}_{ii}\right)^{-1} \mathcal{H}_{ii}^T R_{ii}^{-1} \Big(Z_i - \sum_{j \neq i} \mathcal{H}_{ij} \hat{\mathbf{X}}_\mathbf{j}(t) \Big) \tag{4.19}$$

The iterative step described by equation (4.19) shows how the single area state estimation depends on its inner measures and its border nodes estimation.

This let us handling only with local informations and to implement a completely distributed algorithm:

1. each area i receive the border estimated state form adjacent areas $j \in \Omega_i$;
2. estimates its inner state using equation 4.19;
3. sends to area j the estimated value of node $i \in \Omega_j$.

The idea that make this algorithm "competitive" in estimation scenario is to measure with a more accurate instrument the *border nodes* of each area. In this way we reduce the error in border measurements. As they are crucial for a correct estimation, we will show a big set of simulation showing that this algorithm performs a very good estimation and therefore has a good behaviour applied to the reactive power compensation algorithm.

Chapter 5

Testing results

In this chapter we are going to validate the algorithms previously described, applying them to the reactive power compensation algorithms proposed in [5] and in [13]. In order to do this, we presents a lot of simulations run over the `IEEE37 Test Feeder` [20], whose graph is shown in figure 5.1

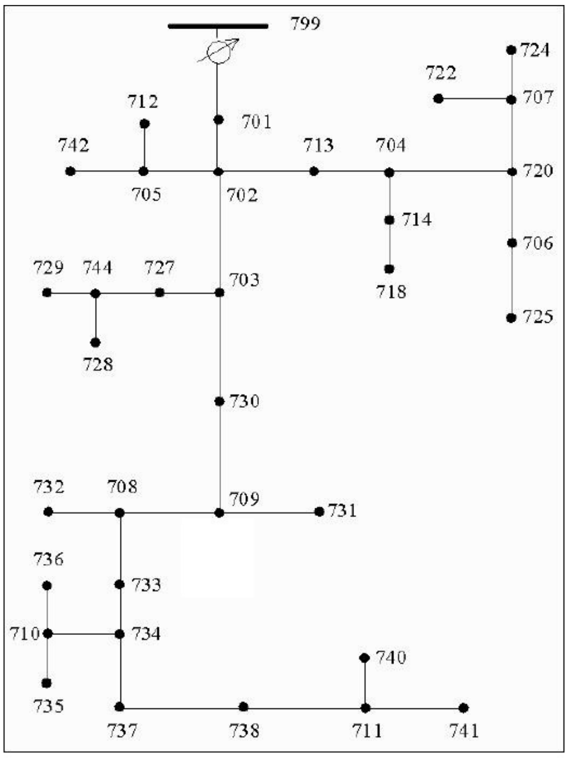


Figure 5.1: IEEE37 test feeder

For testing our algorithms, this grid has been divided according to the *multi area decomposition* presented in figure 4.1. In particular, we divided the graph into 4 non-overlapping areas as in figure 5.2, with one compensator (in red) for each area.

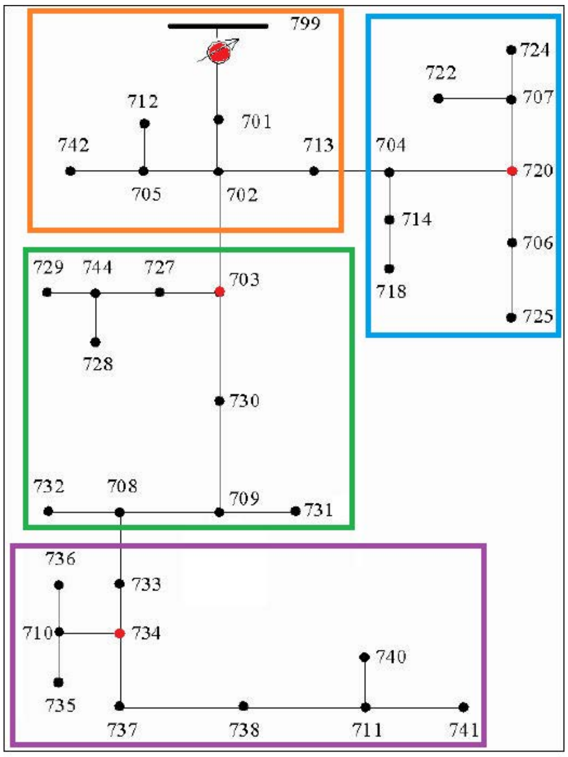


Figure 5.2: multi area decomposition of IEEE37

All tests that are going to be presented have been executed using MATLAB© R2011b on a $Win7$ based computer with *core* $i5$ quad-core processor clocking at 2.27GHz and 4GB of RAM.

5.1 Noise sizing

Let us assumed to measure both voltage and current at every node[1], divided into amplitude and phase. Such measurements are generated by calculating a solution of the power flow problem (the complex voltage and current at every node) and then corrupting it with a Gaussian additive noise. Hence every node is characterized by a collection of values equal to:

$$\begin{cases} v_m = v + e_v; & e_v \sim \mathcal{N}(0, \sigma_V^2); \\ \theta_m = \theta + e_\theta; & e_\theta \sim \mathcal{N}(0, \sigma_\Theta^2); \\ i_m = i + e_i; & e_i \sim \mathcal{N}(0, \sigma_I^2); \\ \phi_m = \phi + e_\phi; & e_\phi \sim \mathcal{N}(0, \sigma_\Phi^2); \end{cases}$$

[1]In a real implementation every node is equipped with a PMU, which takes the measurements.

More specifically, as default standard deviation values, it is assumed that:

- $\sigma_V = 10^{-2} PCC_{VoltageAmplitue}[Volt]$: it means to let the measures to have a standard deviation equal to 1% the PCC voltage amplitude. To better understand it for a PCC voltage amplitude of $4.16KV$ it means to have a measurement error of $\simeq 10V$ on average;
- $\sigma_I = 10^{-2} i_{max}[Ampere]$: as seen for the voltage measures, it means to let the current amplitude measurement being equal to 1% of the maximum current on average;
- $\sigma_\theta = \sigma_\phi = 10^{-2}[rad]$: for a $50Hz$ signal it means to measure a phase with an almost maximum error of $\simeq 100\mu s$.

These values are chosen as standard values, it is just a convention. However, simulations have been executed for several values, greater and smaller than the standard ones, to highlight the algorithms performances in different situations.

It should be underlined that using smaller values of standard deviations implies better or equal but absolutely not worse performances of the algorithms. Similarly, using greater values of standard deviations implies worse or equal but absolutely not better performances.

In order to a better comprehension and to make the reader able to match immediately the performances, tests will be grouped according to the standard deviation values, considering both algorithms in the same analysis.

5.2 Performances of the algorithms

5.2.1 About Centralized Estimation

The centralized estimation is the starting point of our study. It represents the optimal global solution to the estimation problem, that is the solution to the equation 3.8. This estimator assumes the presence of a central unit over the entire network, which is able to collect and process all nodes measurements.

In this thesis we are not going to analyze the behaviour of the central estimation (for centralized simulations see [21]), we will just consider it as the optimal solution X_{opt}, and we will use it in order to test the performances of the different distributed estimations. In fact, the variable introduced to understand immediately the estimation performance is $||X_{opt} - X_{estim}||$, where X_{estim} means the entire state obtained by collecting the distributed solutions. This variable is our choice to express the distance between our distributed solution and the optimal one. Formally, we have:

$$||X_{opt} - X_{estim}|| = \sqrt{\sum_{i=1}^{n} (x_i^{opt} - x_i^{estim})^2}$$

where x_i^{opt} and x_i^{estim} are the single components respectively of X_{opt} and of X_{estim} and n is the number of agents composing the state involved in the estimation.

5.2.2 Tests setup

As mentioned above, we will consider both algorithms in the same moment, in order to better understand the comparison of their performances. For the ADMM estimation we will consider three times same algorithm, each time with a different (increasing) number of iterations, to observe how the solution can change simply reducing the number of ADMM iterations. Tests proposed will consider:

- ADMM estimation with 10 iterations;
- ADMM estimation with 100 iterations;
- ADMM estimation with 1000 iterations;
- Jacobi-like algorithm with 2 iterations;

The last choice is imposed, because in every simulation we noted that the state computed using the Jacobi algorithm, after the second iteration, remains constant whatever is the number of iterations. This behaviour is probably due to Matlab approximation. Executing the simulation, indeed, a warning inform us that results can be inaccurate because of the structure of the error matrix R.

The following tests are made up in this way: we will first present the behaviour only of the estimation algorithm, analyzing the distance between the optimal solution. Then we will apply the 4 different configurations to the reactive power compensation algorithm, in order to see if the compensation is acceptable or not.

To be precise, we have to highlight a convention applied in RPC algorithm. In order to better plotting the results of the compensation with ADMM and JACOBI, we supposed to compute the initial losses after having estimated the state. If you want to know the initial losses before estimating, you can consider the value related to noisy measurements. As the measurements trend is sometimes completely undesirable, we will zoom the most significant part of the graphics. Therefore in some cases it will not be possible to see the measurements trend, or part of it.

5.2.3 Performances for default values of noise standard deviations

Let us compare our algorithms for default values of standard deviations (see section 5.1). Considering only the ADMM algorithm, in figure 5.3 we can find the performances for each number of iterations.

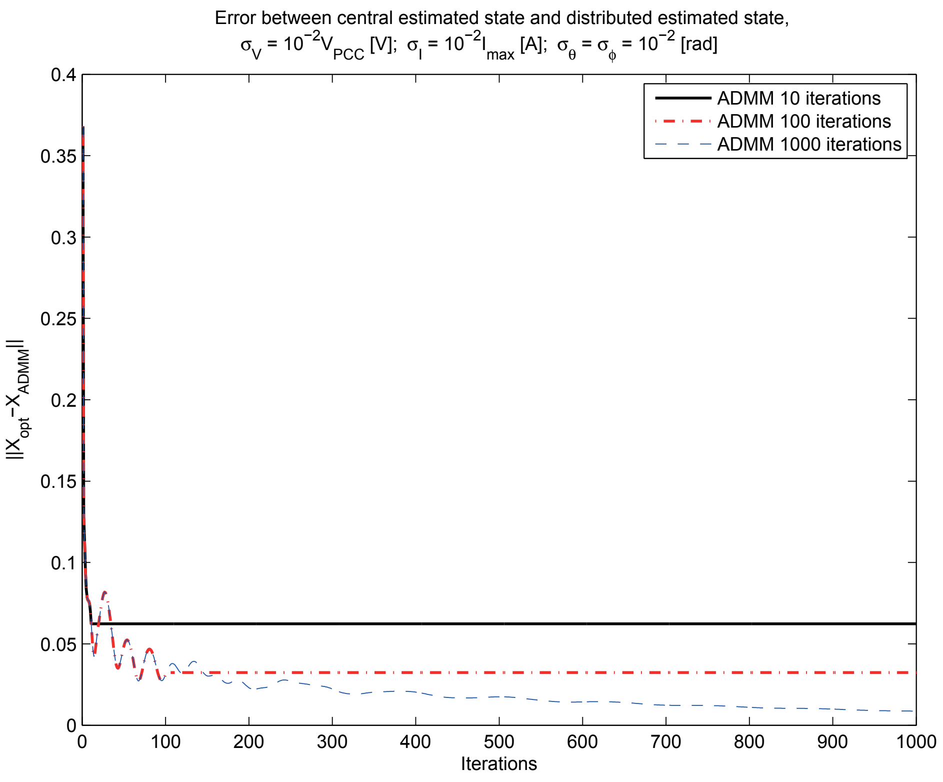


Figure 5.3: Distance from the optimal value - ADMM estimation

From the proof proposed in chapter 4, we know that the error plotted in this figure converges to 0 for $T \to \infty$. However, for $T = 1000$ iterations, we can note that the error is already very close to 0.

In order to compare the ADMM algorithm with the Jacobi-like one, we have to make a slightly different comparison. Remembering the structure of the Jacobi algorithm, indeed, we supposed a standard deviation ten times smaller in the border nodes. Therefore, the optimal centralized solution is obviously different from the ADMM one. Hence, in order to compare in a correct way the 2 algorithms, we decided to plot a distance from the *real* state of the network. In particular, we choose:

$$\log_{10}(||X - X_{estim}||)$$

where X means the *real* state of the network (the state with no noise).

As you can notice from the figure 5.4, Jacobi-like algorithm with just two iterations estimates the state better than the 100 iterations ADMM, but worse than 1000 iterations algorithm.

Let us then apply these estimation techniques to the reactive power compensation algorithm explained in section 3.2. Results are plotted in figure 5.5.

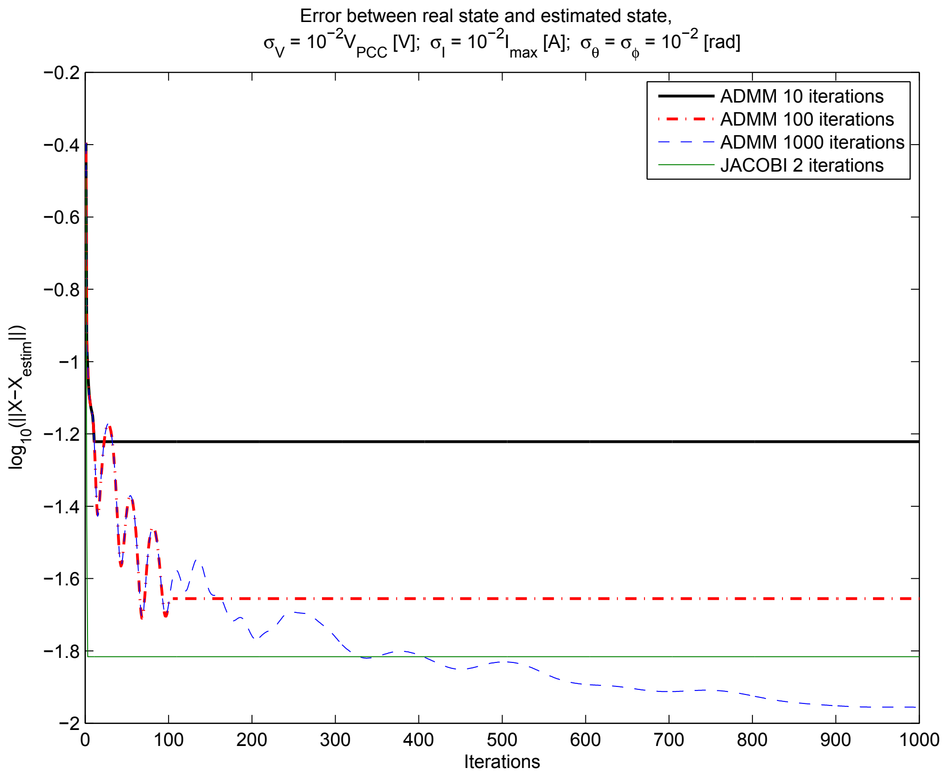


Figure 5.4: Distance from the real state - ADMM and JACOBI algorithms

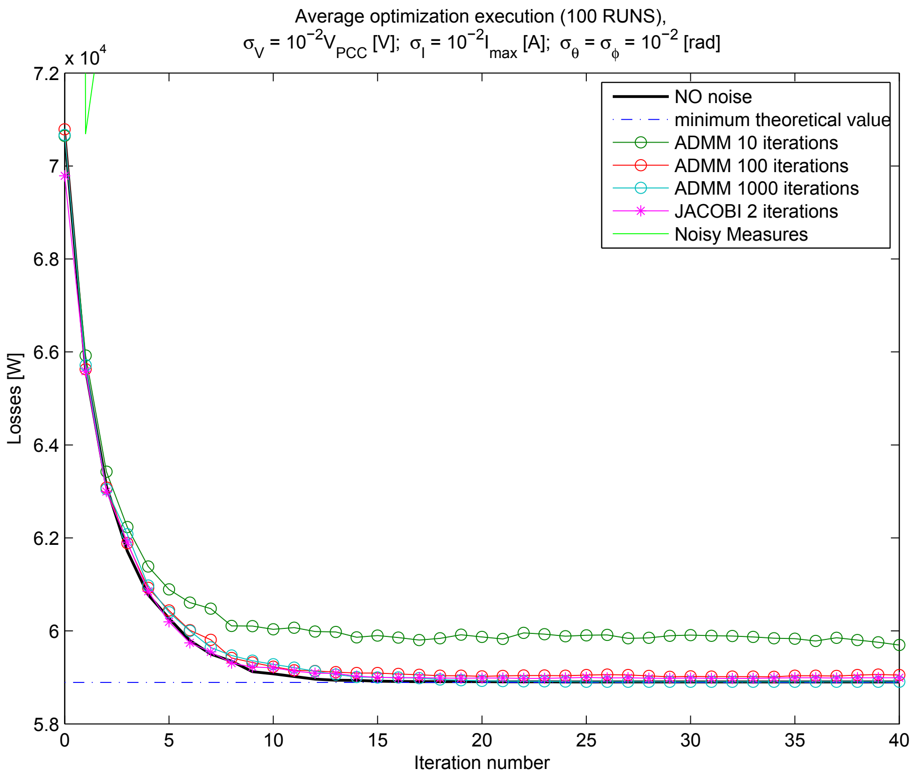


Figure 5.5: Reactive power compensation - ADMM and JACOBI algorithms

	Losses before optimization [kW]	Losses after optimization [kW] (optimal = 58.903)	Losses reduction respect to no-noise init. losses (optimal = 16.68 %)
Real State	70.692	58.903	16.68 %
ADMM 10 iter.	70.649	59.695	15.56 %
ADMM 100 iter.	70.676	59.027	16.50 %
ADMM 1000 iter.	70.670	58.907	16.67 %
JACOBI (2 iter.)	69.788	58.993	16.55 %
Noisy Measures	2420.718	83.840	-18.60 %

Table 5.1: Losses before and after optimization for different data set.

As you can notice from the table, Every data set presents an acceptable behaviour, except for the noisy measurements. As we expected looking estimation performances in figure 5.4, JACOBI algorithm is better than ADMM$_{100}$ but worse than ADMM$_{1000}$, which is pretty equivalent to the no-noise data set.

Let us now introduce a unique variable $\sigma := \sigma_V = \sigma_I = \sigma_\theta = \sigma_\phi$. Now we are going to analyze the performances of the algorithms for different values of σ, from 10^{-4} to 10^{-1}.

All the following tests are compared to the theoretical minimum value for the reactive power losses, that is 58.903 [kW].

Computational effort

It is important to deepen a bit more about the computational effort belonging to these algorithms. Considering a real electric grid, indeed, it is reasonable to suppose network state to have a dynamic evolution. Because of this, if we want to apply directly these algorithms to a dynamic-state, it is necessary to complete the estimation process very fast, before that the state changes due to the dynamic evolution. Considering our algorithms, we have:

- $t_{ADMM_{10}} \simeq 9\ ms$;
- $t_{ADMM_{100}} \simeq 28\ ms$;
- $t_{ADMM_{1000}} \simeq 230\ ms$;
- $t_{JACOBI_2} \simeq 5\ ms$;

From these values we can immediately note the fastness of the Jacobi algorithm. However, we do not have to forget that for using this algorithms we need a more precise measuring instruments, capable of reducing the error of one order of magnitude.

5.2.4 Performances for $\sigma = 10^{-4}$

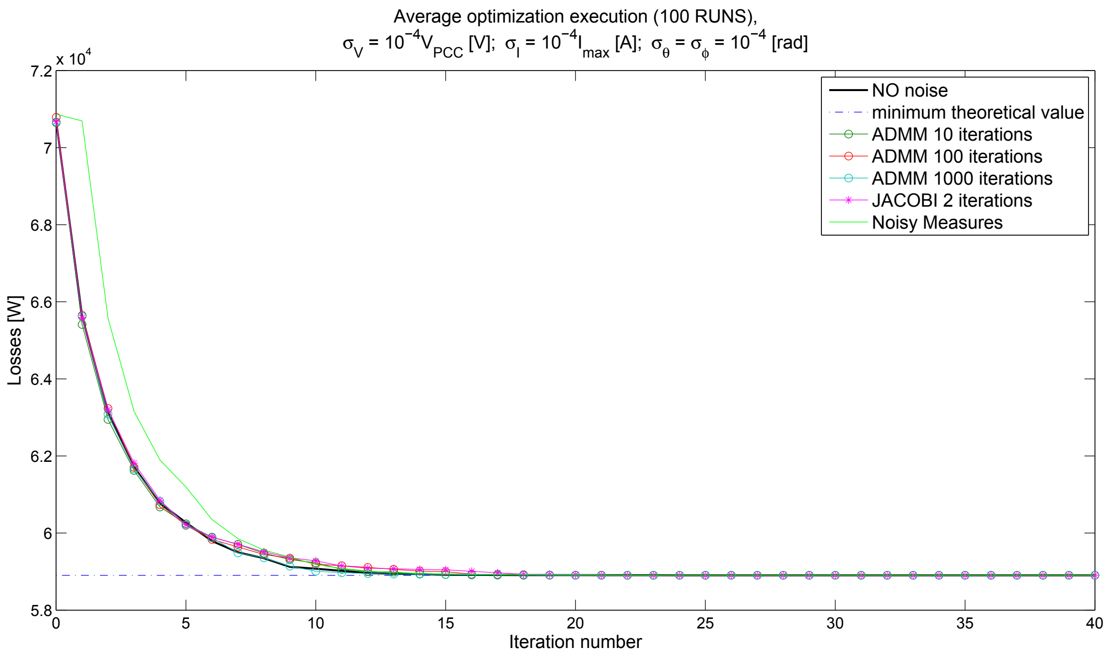


Figure 5.6: Reactive power compensation for $\sigma = 10^{-4}$

	Losses before optimization [kW]	Losses after optimization [kW]	Losses reduction respect to no-noise init. losses
Real State	70.692	58.903	16.68 %
ADMM 10 iter.	70.649	58.903	16.68 %
ADMM 100 iter.	70.789	58.903	16.68 %
ADMM 1000 iter.	70.670	58.903	16.68 %
JACOBI (2 iter.)	70.676	58.903	16.68 %
Noisy Measures	70.867	58.906	16.67 %

Table 5.2: Losses before and after optimization for different data set.

As you can notice, for $\sigma = 10^{-4}$ it is not necessary any algorithm, because raw data provide a performance pretty similar to the real state. The theoretical minimum is however reached by any algorithm.

5.2.5 Performances for $\sigma = 10^{-3}$

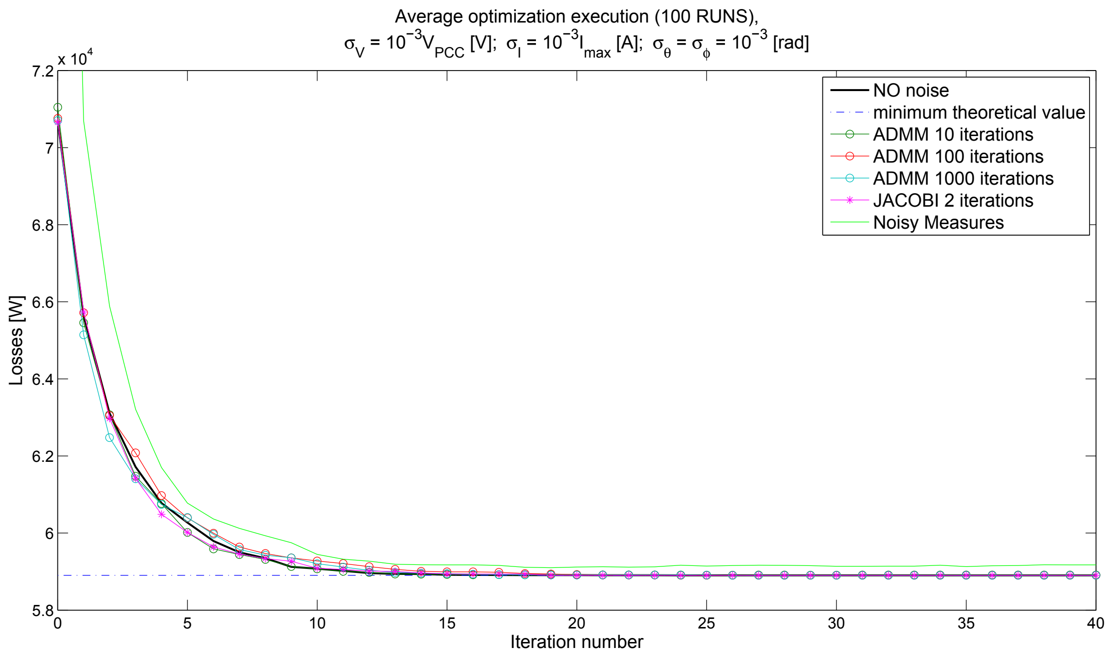


Figure 5.7: Reactive power compensation for $\sigma = 10^{-3}$

	Losses before optimization [kW]	Losses after optimization [kW]	Losses reduction respect to no-noise init. losses
Real State	70.692	58.903	16.68 %
ADMM 10 iter.	71.046	58.912	16.66 %
ADMM 100 iter.	70.756	58.904	16.67 %
ADMM 1000 iter.	70.703	58.903	16.68 %
JACOBI (2 iter.)	70.660	58.903	16.68 %
Noisy Measures	70.867	59.174	16.29 %

Table 5.3: Losses before and after optimization for different data set.

For $\sigma = 10^{-3}$ all data sets up to ADMM_{10} are well performing, only noisy measurements deviate a bit from the optimal value.

5.2.6 Performances for $\sigma = 5 \cdot 10^{-3}$

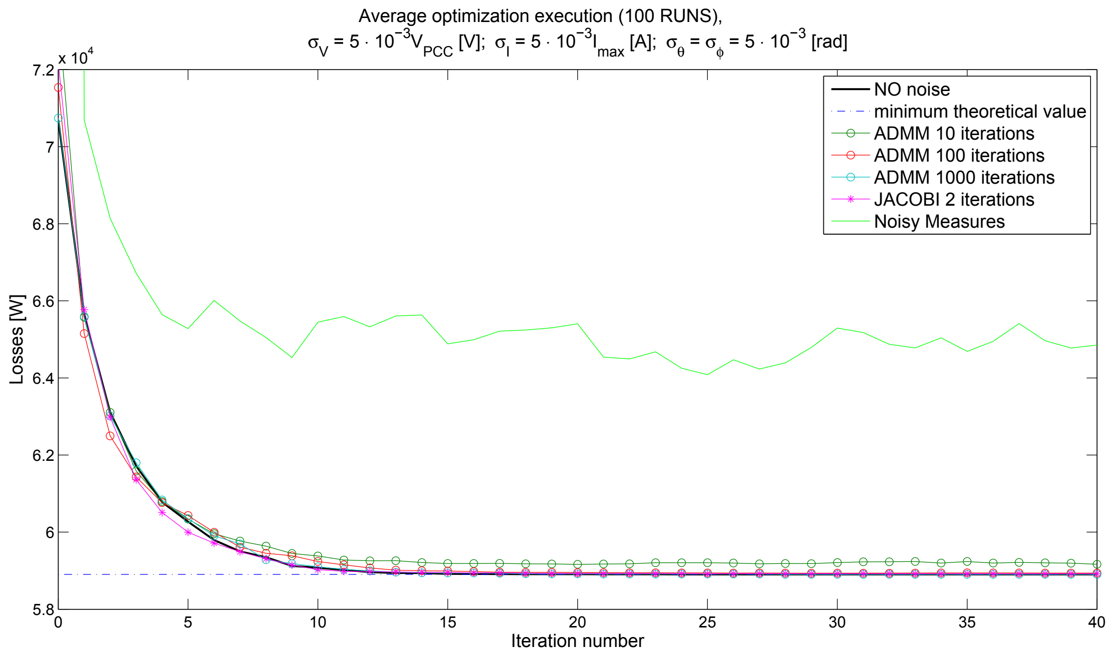


Figure 5.8: Reactive power compensation for $\sigma = 5 \cdot 10^{-3}$

	Losses before optimization [kW]	Losses after optimization [kW]	Losses reduction respect to no-noise init. losses
Real State	70.692	58.903	16.68 %
ADMM 10 iter.	73.466	59.166	16.30 %
ADMM 100 iter.	71.537	58.937	16.63 %
ADMM 1000 iter.	70.743	58.905	16.67 %
JACOBI (2 iter.)	72.201	58.920	16.65 %
Noisy Measures	713.206	64.854	8.26 %

Table 5.4: Losses before and after optimization for different data set.

As you can notice, already with $\sigma = 5 \cdot 10^{-3}$ the use of raw data is not desirable, while both ADMM and JACOBI algorithm can provide a reactive power compensation in a successful way, of course proportionally with the number of ADMM iterations.

5.2.7 Performances for $\sigma = 3 \cdot 10^{-2}$

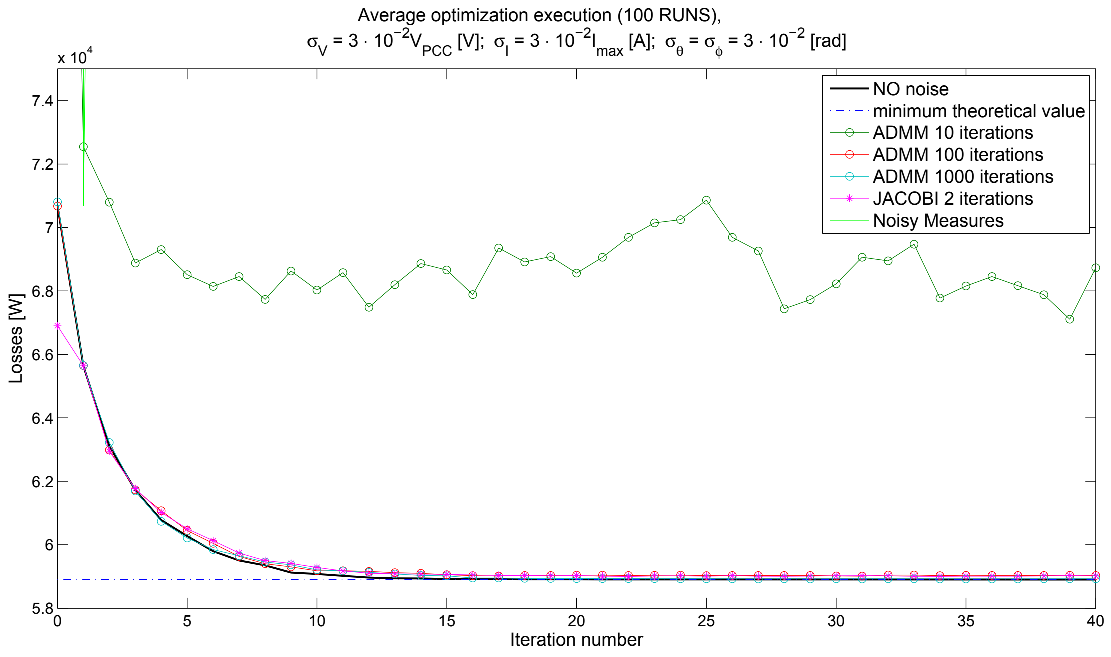


Figure 5.9: Reactive power compensation for $\sigma = 3 \cdot 10^{-2}$

	Losses before optimization [kW]	Losses after optimization [kW]	Losses reduction respect to no-noise init. losses
Real State	70.692	58.903	16.68 %
ADMM 10 iter.	73.466	68.728	2.78 %
ADMM 100 iter.	70.789	59.050	16.47 %
ADMM 1000 iter.	70.810	58.926	16.64 %
JACOBI (2 iter.)	66.902	59.016	16.52 %
Noisy Measures	19820.694	290.399	-310.79 %

Table 5.5: Losses before and after optimization for different data set.

For $\sigma = 3 \cdot 10^{-2}$, value slightly bigger than the default, it is clear that, in addition to raw data, even ADMM$_{10}$ become useless. On the contrary, ADMM$_{100}$, ADMM$_{1000}$ and JACOBI algorithms still present a good performance.

5.2.8 Performances for $\sigma = 5 \cdot 10^{-2}$

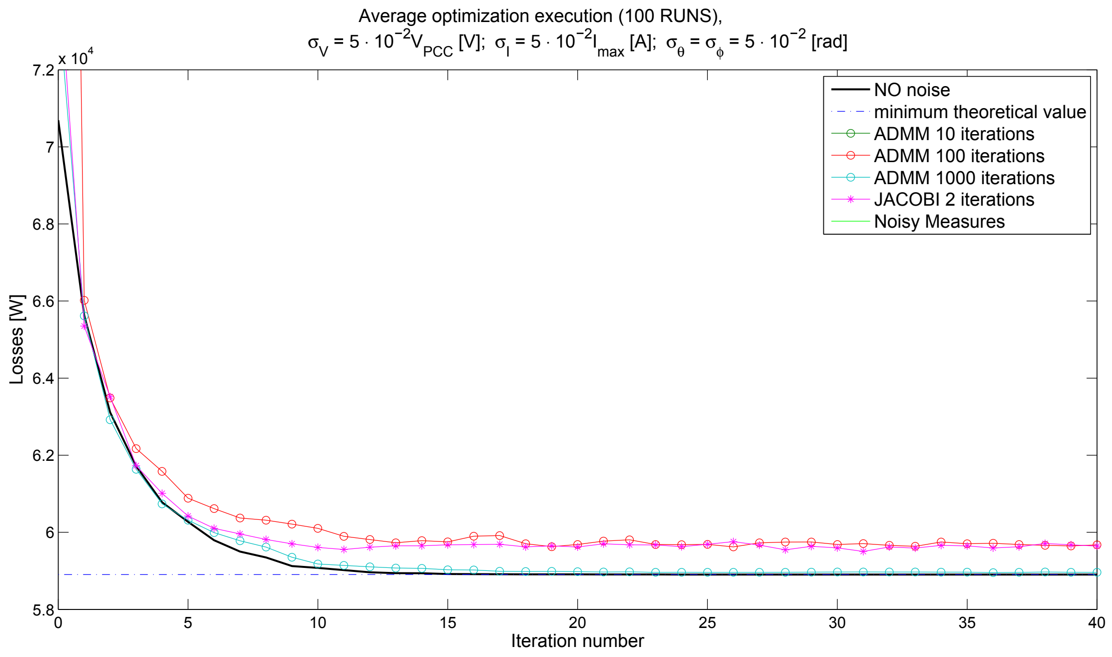


Figure 5.10: Reactive power compensation for $\sigma = 5 \cdot 10^{-2}$

	Losses before optimization [kW]	Losses after optimization [kW]	Losses reduction respect to no-noise init. losses
Real State	70.692	58.903	16.68 %
ADMM 10 iter.	263.347	109.571	-55.00 %
ADMM 100 iter.	114.320	59.673	15.59 %
ADMM 1000 iter.	73.854	58.961	16.59 %
JACOBI (2 iter.)	74.917	59.642	15.63 %
Noisy Measures	52022.472	646.674	-814.78 %

Table 5.6: Losses before and after optimization for different data set.

Plot referred to $\sigma = 5 \cdot 10^{-2}$ is zoomed in the most significant region, discarding the trends of ADMM$_{10}$ and of noisy measurements. We can note that JACOBI and ADMM$_{100}$ have a pretty similar performance. The only algorithm which assures a compensation greater than 16% is ADMM$_{1000}$.

5.2.9 Performances for $\sigma = 10^{-1}$

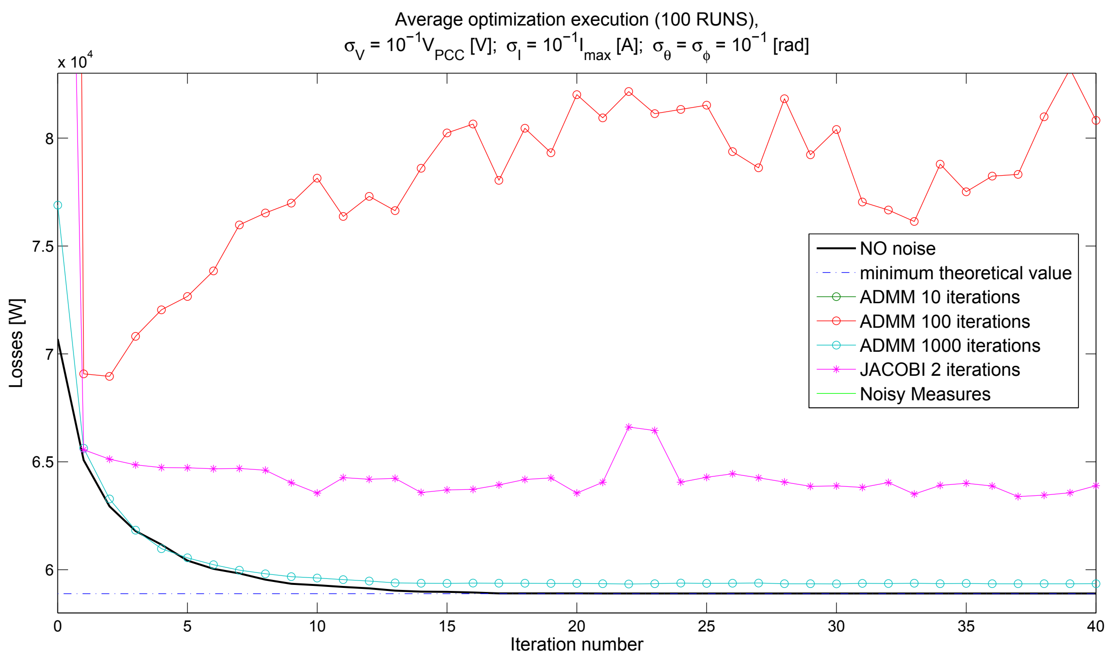


Figure 5.11: Reactive power compensation for $\sigma = 10^{-1}$

	Losses before optimization [kW]	Losses after optimization [kW]	Losses reduction respect to no-noise init. losses
Real State	70.692	58.903	16.68 %
ADMM 10 iter.	263.347	290.176	-310.48 %
ADMM 100 iter.	114.320	80.818	-14.32 %
ADMM 1000 iter.	73.854	59.357	16.03 %
JACOBI (2 iter.)	74.917	63.898	9.61 %
Noisy Measures	$1.7692 \cdot 10^5$	∞	–

Table 5.7: Losses before and after optimization for different data set.

For $\sigma = 5 \cdot 10^{-2}$ we obtain very interesting results. Considering the large magnitude of the error (ten times bigger than the standard values), it is clear that, except for the ADMM$_{1000}$, every algorithm fails in the reactive power compensation. However, it is interesting to note the difference between the ADMM algorithm with 100 iterations and the JACOBI one. While for $\sigma = 5 \cdot 10^{-2}$ the behaviours are pretty identical, with $\sigma = 10^{-1}$ they are both unusable, but the JACOBI algorithm performs much better than the ADMM_{100}.

Moreover, we have to underline the importance of the state estimation, looking the trend of the ADMM_{1000}: despite of this huge error, the reactive power compensation is performed with a distance of only 0.65% from the compensation with the real state, that is the minimum theoretical value for the reactive power losses.

In this case, $\sigma = 10^{-1}$, we want to show the correctness of the estimation process with the logarithmic plot, as done for default values of noise standard deviations:

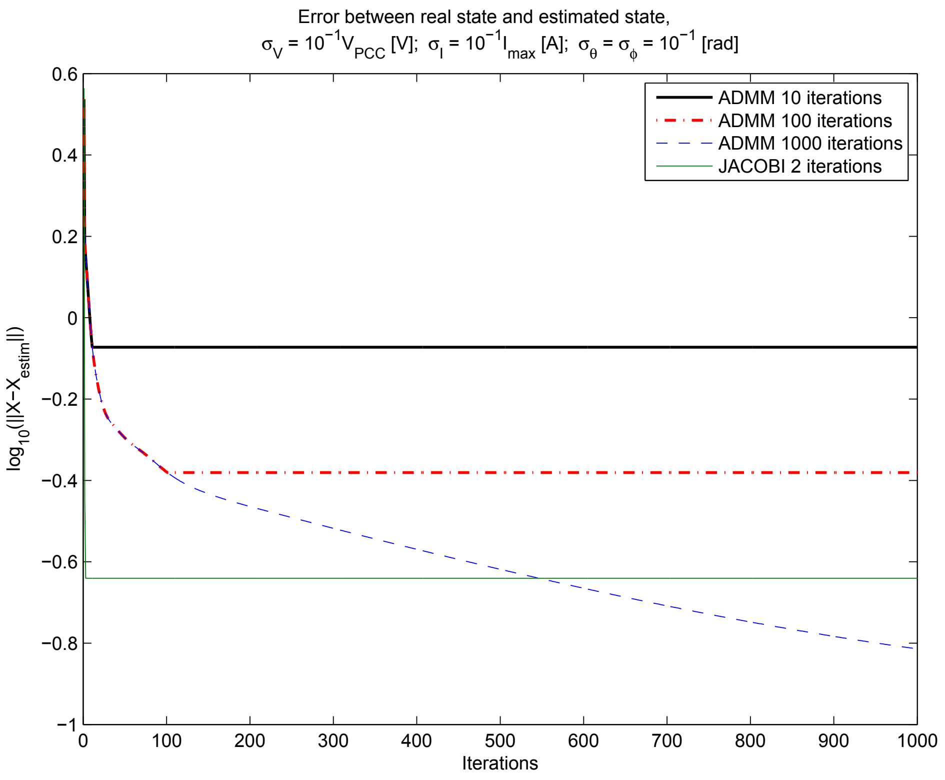


Figure 5.12: Distance from the real state - ADMM and JACOBI algorithms

This figure presents results that are consistent with the performances obtained with the application to the reactive power compensation algorithm.

5.2.10 Jacobi algorithm on a 6-areas divided grid

In order to show the good behaviour of JACOBI algorithm for a whatever distributed configuration of the network, we want to present some tests based on a different grid-setup. Let us consider the `IEEE37` divided in 6 areas as in figure 5.13.

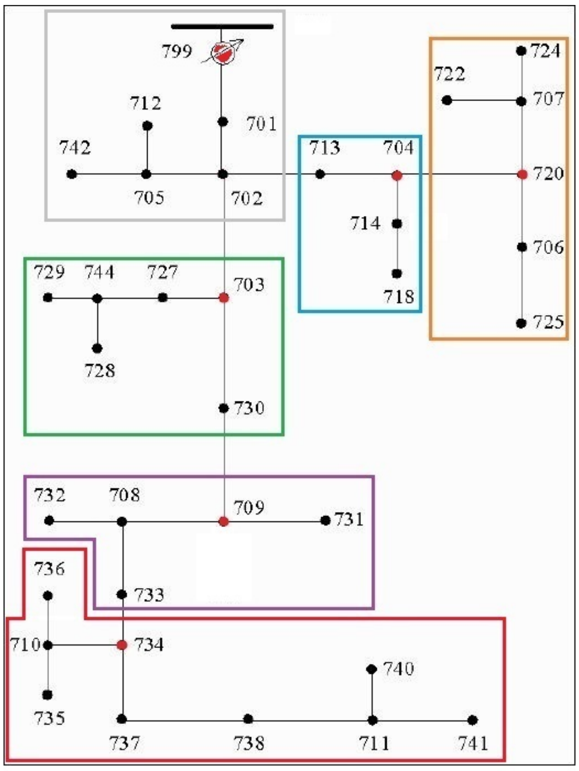


Figure 5.13: ieee37 divided in 6 areas

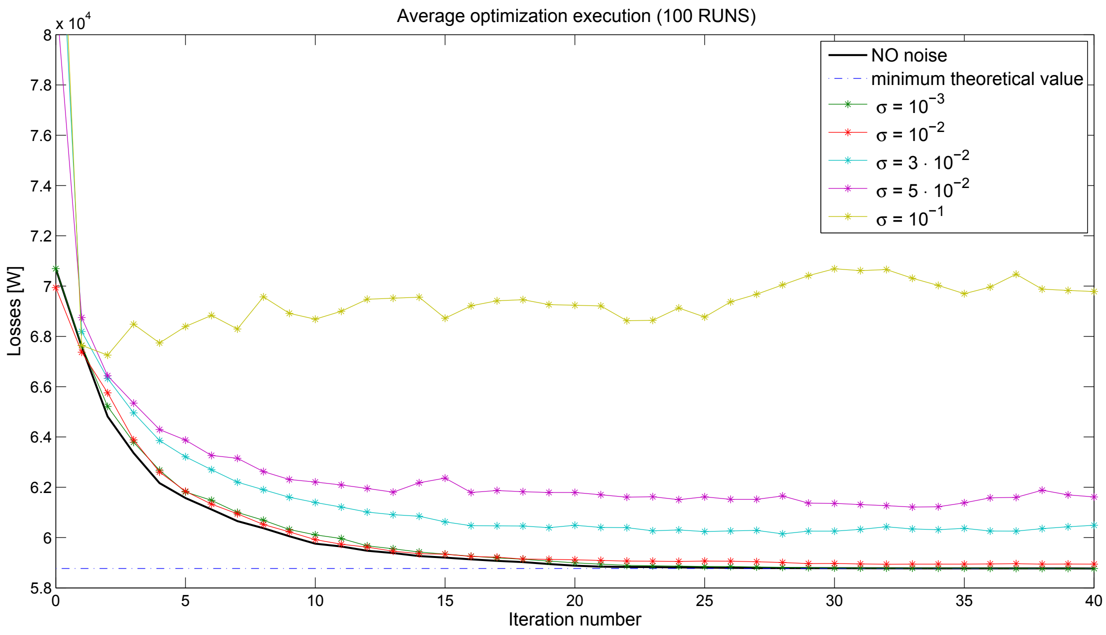


Figure 5.14: Reactive power compensation for different value of σ

This setup presents one compensator for each area, and so 6 compensators (nodes in red). Let us compare the performances of the algorithm for different values of σ. Minimum theoretical value for losses is 58.769 [kW]. Results are shown in figure 5.14

	Losses before optimization [kW]	Losses after optimization [kW]	Losses reduction respect to no-noise init. losses
Real State	70.692	58.769	16.87 %
$\sigma = 10^{-3}$	70.692	58.771	16.86 %
$\sigma = 10^{-2}$	69.940	58.943	16.62 %
$\sigma = 3 \cdot 10^{-2}$	88.980	60.487	14.44 %
$\sigma = 5 \cdot 10^{-2}$	81.780	61.615	12.84 %
$\sigma = 10^{-1}$	91.313	69.782	1.29 %

Table 5.8: Losses before and after optimization for different data set.

Looking at the table and comparing results with the ones obtained before with a 4-areas configuration, we can state that performances are substantially equivalent.

5.2.11 Performances adding saturation constraints to each power line

Now we want to apply ADMM and JACOBI estimation strategies to a reactive power compensation algorithms with saturation constraints, treated in [13]. In this algorithm we added power saturation constraints to each power line. In [13] it has been demonstrated that in this new configuration the algorithm does not reach always the optimal value because it can stop in a local minimum, different from the global one. Then it has been implemented a *multi-hop* strategy in order to reach anyway the optimal value. We will show that the problem of non-convergence is present even with the estimation algorithms, and moreover that ADMM and JACOBI performances with the *multi-hop* strategy are pretty equal to the no-noise state ones.

We will analyze the algorithms for default value of noise standard deviations, that is $\sigma = 10^{-2}$, in the 4-areas configuration of `IEEE 37` test feeder.

Let us start highlighting the behaviour without the *multi-hop* strategy. Every algorithm does not reach the theoretical minimum value o f reactive power losses, as shown in figure 5.15. We did not consider the performance using raw measurements because it is completely undesirable.

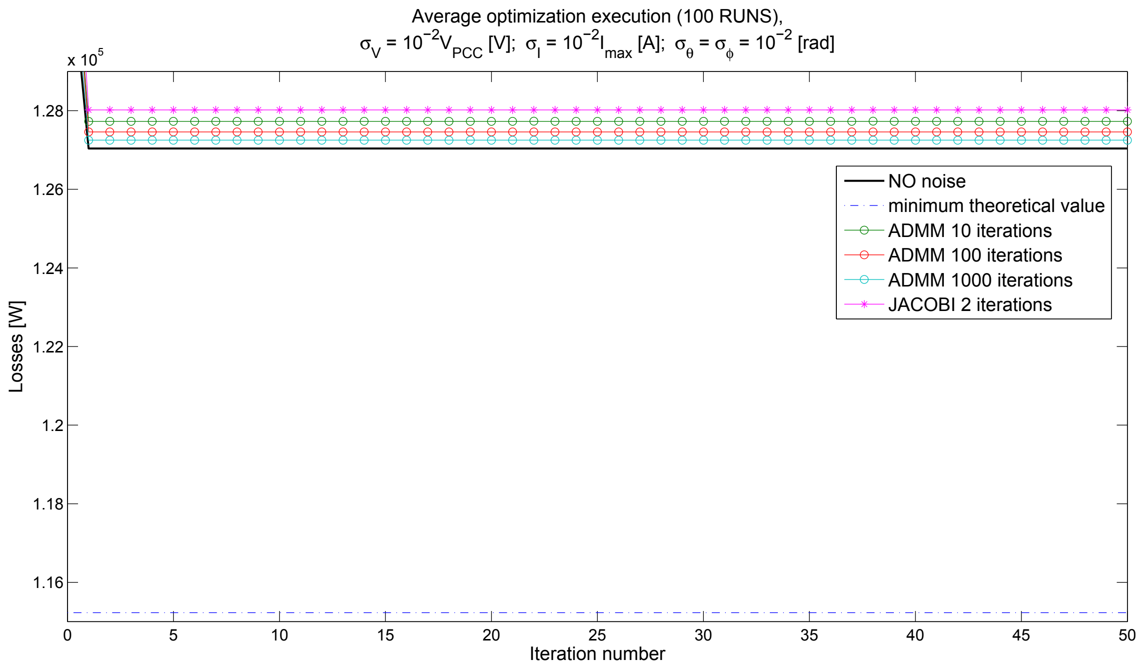


Figure 5.15: Behaviour with saturation constraints on power lines

	Losses before optimization [kW]	Losses after optimization [kW] (optimal = 115.227)	Losses reduction respect to no-noise init. losses (optimal = 14.45 %)
Real State	134.693	127.040	5.68 %
ADMM 10 iter.	134.498	127.728	5.17 %
ADMM 100 iter.	134.391	127.460	5.37 %
ADMM 1000 iter.	134.047	127.250	5.53 %
JACOBI (2 iter.)	136.588	128.020	4.95 %

Table 5.9: Losses before and after optimization with saturation constraints

Introducing the *multi-hop* strategy, instead, both algorithms improve their performances and approach the optimal value, as shown in figure 5.16.

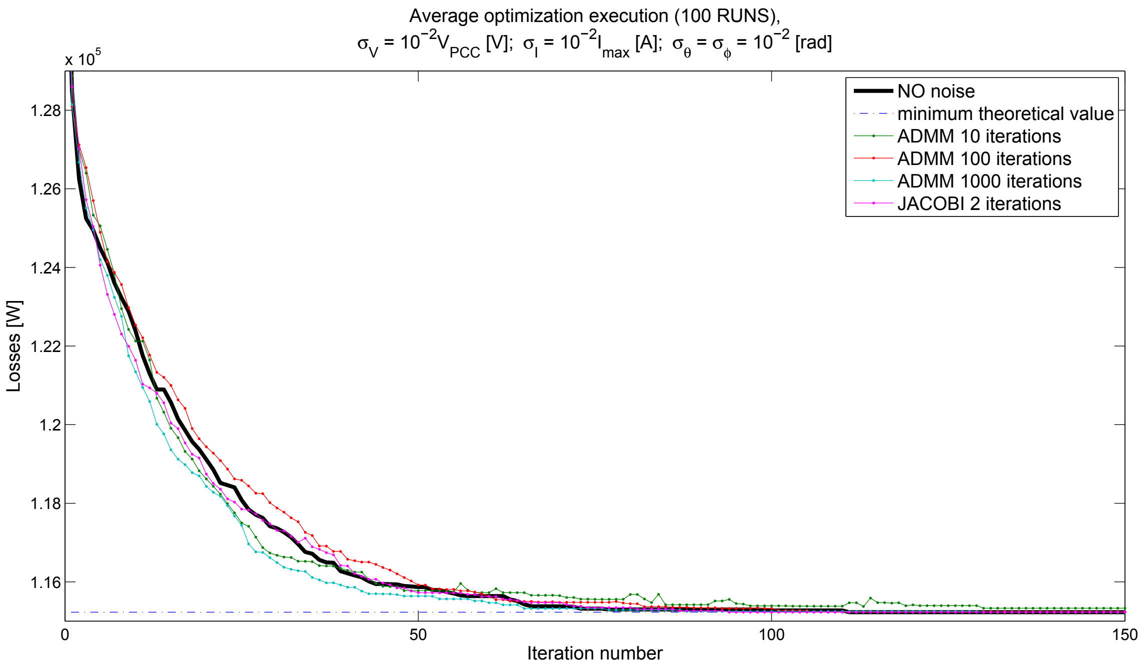


Figure 5.16: Behaviour with saturation constraints and *multi-hop* strategy

	Losses before optimization [kW]	Losses after optimization [kW] (optimal = 115.227)	Losses reduction respect to no-noise init. losses (optimal = 14.45 %)
Real State	134.693	115.227	14.45 %
ADMM 10 iter.	136.052	115.330	14.38 %
ADMM 100 iter.	136.973	115.240	14.44 %
ADMM 1000 iter.	134.565	115.229	14.45 %
JACOBI (2 iter.)	137.285	115.227	14.45 %

Table 5.10: Losses before and after optimization with multi-hop strategy

Chapter 6

Conclusion

6.1 Results achieved

In this thesis we deal with the problem of estimation in low voltage power networks. This is well known to be the starting point for controlling a network since raw measurements are too inaccurate to work with. Specifically, we proposed two separate algorithms capable of solving the estimation problem in a distributed and scalable way. In particular we formally proved that the first one, based on the ADMM, converges to the optimal centralized solution for the estimation problem.

Focusing on advantages and disadvantages of these algorithms, we have:

ADMM algorithm	JACOBI-like algorithm
completely scalable and distributed	completely scalable and distributed
converges to the maximum likelihood solution	leads only to an approximate solution
high computational effort and time to converge	very low computational effort and time to converge
deals with the synchronization noise [21]	does not deal with the synchronization noise
	requires some more accurate measurements on the network

Table 6.1: Advantages and disadvantages of ADMM and JACOBI algorithms

In addition to this, a specific power network control algorithm has been tested to underline the importance of estimation. Specifically, a power losses reduction through reactive power compensation algorithm, with and without saturation constraints ([5],[13]) has been chosen as prototype. The simulations show how using the estimated state, respect to the raw measurements, improves much more the performances, making them comparable with those obtained using the exact state value. The measurements, instead, leads to an undesirable behaviour.

6.2 Possible future developments

This thesis can be an interesting starting point to analyze in a deeper way power networks with noisy measurements. We suggest some future developments that can be inspected starting from our results:

1. Study how the dynamic behaviour of a power grid can affect these result, and eventually find a solution to dynamically estimate the state of the network, i.e. with extended Kalman filter.

2. In order to apply the Jacobi-like algorithm to a real netweork, it can be interesting to focus on measuring instruments available at the moment, understand if it is possible to buy instruments capable of measuring the voltages with a fixed order of magnitude of the error, and which is eventually the price of these instruments.

3. Compare the algorithms presented in this thesis with other distributed algorithms that are rising in the literature, i.e. Newton-Raphson based algorithms, in order to understand which algorithm is best suited for a particular controlling problem.

Appendix A

Proof of equations 4.5 and 4.6

Lemma 4. *For each time instant t, the updating step of z_{ij} in the ADMM algorithm is*

$$z_{ij}(t+1) = \frac{1}{2c}\Big(\lambda_{ij}(t) + \mu_{ji}(t) + c\big(X^{(i)}(t+1) + X^{(j)}(t+1)\big)\Big) \tag{A.1}$$

Proof. Starting from 4.4 we have:

$$\begin{aligned} z_{ij}(t+1) &= \arg\min_{z_{ij}} \mathcal{L}(X(t+1), z, \lambda(t), \mu(t)) \\ &= \arg\min_{z_{ij}} \Big\{-(\lambda_{ij}^T(t) + \mu_{ji}^T(t))z_{ij} + \frac{c}{2}\big[||X^{(i)}(t+1) - z_{ij}||^2 + ||X^{(j)}(t+1) - z_{ij}||^2\big]\Big\} \end{aligned}$$

because all the other components does not depend on z_{ij}.

Through the first order optimality condition we have

$$-\lambda_{ij}(t) - \mu_{ji}(t) - c\big(X^{(i)}(t+1) - z_{ij}\big) - c\big(X^{(j)}(t+1) - z_{ij}\big) = 0$$

and hence

$$z_{ij}(t+1) = \frac{1}{2c}\Big(\lambda_{ij}(t) + \mu_{ji}(t) + c\big(X^{(i)}(t+1) + X^{(j)}(t+1)\big)\Big)$$

□

Lemma 5. *In general it is true that*

$$\lambda_{ij}(-1) = -\mu_{ji}(-1) \Longrightarrow \lambda_{ij}(t) = -\mu_{ji}(t) \ \forall t$$

Proof. The proof is done by induction. The base case is trivial, due to a simple initialization $\lambda_{ij}(-1) = -\mu_{ji}(-1) = 0$. Then, substituting the expression of $z_{ij}(t)$ obtained in A.1 in 4.2 we have

$$\begin{cases} \lambda_{ij}(t) &= \lambda_{ij}(t-1) + \frac{c}{2}\big(X^{(i)}(t) - X^{(j)}(t)\big) - \frac{1}{2}\big(\lambda_{ij}(t-1) + \mu_{ji}(t-1)\big) \\ \mu_{ji}(t) &= \mu_{ji}(t-1) + \frac{c}{2}\big(X^{(j)}(t) - X^{(i)}(t)\big) + \frac{1}{2}\big(\lambda_{ij}(t-1) + \mu_{ji}(t-1)\big) \end{cases}$$

By summing these equations we obtain

$$\lambda_{ij}(t) + \mu_{ji}(t) = \lambda_{ij}(t-1) + \mu_{ji}(t-1) + \frac{c}{2} \cdot 0.$$

Applying the inductive hypothesis, the thesis follows.

□

Now, by substituting this equivalence in the z_{ij} equation we obtain

$$z_{ij}(t) = \frac{X^{(i)}(t) + X^{(j)}(t)}{2}$$

Because of this, by simply substituting this expression in equations 4.2, 4.3 and 4.4, we can discard both z_{ij} and μ_{ij} terms, obtaining the new formulation for the ADMM algorithm described in equations 4.5 and 4.6.

Bibliography

[1] J. A. Lopes, C. L. Moreira, and A. G. Madureira, "Defining control strategies for microgrids islanded operation," *IEEE Trans. Power Syst.*, vol. 21, no. 2, pp. 916–924, May 2006.

[2] T. C. Green, C. L. Prodanović, and A. G. Madureira, "Control of inverter-based micro-grids," *Electr. Pow. Syst. Res.*, vol. 77, no. 9, pp. 1204–1213, Jul 2007.

[3] F. Katiraei and M. R. Iravani, "Power management strategies for a microgrid with multiple distributed generation units," *IEEE Trans. Power Syst.*, vol. 21, no. 4, pp. 1821–1831, Nov 2006.

[4] M. Prodanović, K. De Brabandere, J. Van den Keybus, T. Green, and J. Driesen, "Harmonic and reactive power compensation as ancillary serivces in inveerter-based distributed generation," *IET Gener. Transm. Distrib.*, vol. 1, no. 3, pp. 432–438, 2007.

[5] S. Bolognani and S. Zampieri, "A distributed control strategy for reactive power compensation in smart microgrids," *arXiv*, vol. arXiv:1106.5626v2 [math.OC], Oct. 2011. [Online]. Available: http://arxiv.org/abs/1106.5626

[6] G. Valverde and V. Terzja, "Pmu-based multi-area state estimation with low data exchange," *Innovative Smart Grid Technologies Conference Europe (ISGT Europe), 2010 IEEE PES*, pp. 1–7, October 2010.

[7] L. Zhao and A. Abur, "Multiarea state estimation using synchronized phasor measurements," *Ieee Transaction on Power System*, vol. 20, no. 2, pp. 611–617, May 2005.

[8] W. Jiang, V. Vittal, and G. T. Heydt, "A distributed state estimator utilizing synchronized phasor measurements," *Ieee Transaction on Power System*, vol. 22, no. 2, pp. 563–571, May 2007.

[9] G. N. Korres, "A distributed multiarea state estimation," *Ieee Transaction on Power System*, vol. 26, no. 1, pp. 73–84, February 2011.

[10] A. J. Conejo, S. de la Torre, and M. Canas, “An optimization approach to multiarea state estimation,” *Ieee Transaction on Power System*, vol. 22, no. 1, pp. 213–221, February 2007.

[11] G. Mateos, I. D. Schizas, and G. B. Giannaakis, “Performance analysis of the consensus-based distributed lms algorithm,” *EURASIP Journal on Advances in Signal Processing*, October 2009.

[12] T. Erseghe, “A distributed and scalable processing method based upon admm,” *Ieee Signal Processing Letters*, vol. 19, pp. 563–566, 2012.

[13] S. Bolognani, A. Carron, A. Di Vittorio, and D. Romeres, “Distributed multi-hop reactive power compensation in smart micro-grids subject to saturation constraints,” 2012.

[14] A. Abur and A. G. Exposito, *Power System State Estimation, theory and implementation.* Marcel Dekker, 2004.

[15] F. C. Schweppe and D. B. Rom, “Power system static-state estimation, part i: Exact model,” *Transaction on Power Apparatus and Systems*, vol. Pas-89, no. 1, pp. 120–125, January 1970.

[16] ——, “Power system static-state estimation, part ii: Approxiamte model,” *Transaction on Power Apparatus and Systems*, vol. Pas-89, no. 1, pp. 125 – 130, January 1970.

[17] ——, “Power system static-state estimation, part iii: Implementation,” *Transaction on Power Apparatus and Systems*, vol. Pas-89, no. 1, pp. 130 – 135, January 1970.

[18] C. Rakpenthai, S. Premrudeepreechacharn, S. Uatrongjit, and N. R. R. Watson, “Measurement placement for power system state estimation using decomposition technique,” *Electric Power System Research*, vol. 75, pp. 41–49, 2005.

[19] B. H. Kim and R. Baldick, “A comparison of distributed optimal power flow algorithm,” *Ieee Transaction on Power System*, vol. 15, no. 2, pp. 599–604, May 2000.

[20] W. H. Kersting, “Radial distribution test feeders,” *Transaction on Power Systems*, vol. 6, no. 3, pp. 975–985, August 1991.

[21] M. Todescato, “Distributed algorithms for state estimation in a low voltage distribution network,” Master’s thesis, Oct 2012.

[22] J. Lavaei and S. H. Low, “Zero duality gap in optimal power flow problem,” *IEEE Trans. on Power Syst.*, 2011.

[23] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*, 2010, vol. 3. [Online]. Available: www.stanford.edu/~boyd/papers/pdf/admm_distr_stats.pdf