Master Thesis

**Master's Degree in Electric Power Systems and Drives**

# Integration of an AC-OPF solver in GridCal

January 2024


**Author:** Carlos Alegre Aldeano

**Supervisors:** Marc Cheah Mañé

Josep Fanals i Batllori





ETSEIB

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

UPC

## Abstract

## Resumen

## Resum

# Contents

## List of Figures

# List of Tables

# Glossary

## Symbols

| | |
|---|---|
| $C$ | Capacitance |
| $C_f$ | Shunt filter capacitance |
| $C_s$ | Corrected series filter capacitance |
| $C_s^{'}$ | Series filter capacitance |
| $f_c$ | Cutoff frequency |
| $i_c$ | Shunt current |
| $i_{cs}$ | Series converter current |
| $i_{cf}$ | Shunt Converter current |
| $i_L$ | Load current |
| $i_s$ | Source Voltage |
| $L$ | Inductance |
| $L_f$ | Shunt filter inductance |
| $L_l$ | Line inductance |
| $L_s$ | Corrected series filter inductance |
| $L_s^{'}$ | Series filter inductance |
| $N_T$ | Series transformer turns ratio |
| $P$ | Active power |
| $pf$ | Power factor |
| $P_L$ | Load active power |
| $Q$ | Reactive power |
| $Q_L$ | Load reactive power |
| $Q_{sh}$ | Shunt converter reactive power |
| $R_f$ | Shunt filter resistance |
| $R_l$ | Line resistance |
| $R_s$ | Corrected series filter resistance |
| $R_s^{'}$ | Series filter resistance |
| $\underline{S}_L$ | Load apparent power |
| $u_c$ | Series correction voltage |
| $u_{c,lim}$ | Voltage correction limit |
| $u_{cs}$ | Series converter voltage |
| $u_{cf}$ | Shunt Converter voltage |
| $u_{DC}$ | DC Link Voltage |
| $u_G$ | Grid voltage |
| $u_L$ | Load voltage |
| $u_s$ | Source voltage |
| $x_i$ | Dummy variable |

| | |
|---|---|
| $\underline{x}_i$ | Complex dummy variable |
| $\lvert\underline{x}_i\rvert$ | Magnitude of dummy variable |
| $x_{i,\mathbb{R}}$ | Real component of dummy variable |
| $x_{i,\mathbb{I}}$ | Imaginary component of dummy variable |
| $x_{i,\alpha}$ | Alpha component of dummy variable |
| $x_{i,\beta}$ | Beta component of dummy variable |
| $x_{i,a}$ | Phase a component of dummy variable |
| $x_{i,b}$ | Phase b component of dummy variable |
| $x_{i,c}$ | Phase c component of dummy variable |
| $\theta_{sys}$ | System angle |
| $\omega_{sys}$ | System frequency |

## Acronyms

| | |
|---|---|
| AC | Alternating Current |
| CITCEA | Centre d'Innovació Tecnològica en Convertidors Estàtics i Accionaments |
| DC | Direct Current |
| DSTATCOM | Distribution Static Synchronous Compensator |
| DVR | Dynamic Voltage Regulator |
| FACTS | Flexible AC Transmission Systems |
| IEEE | Institute of Electrical and Electronics Engineers |
| LC | Inductive and Capacitive Low Pass Filter |
| OLTC | On Load Tap Changer |
| PI | Proportional Integral |
| PLL | Phase Locked Loop |
| PSVD | Positive Sequence Voltage Detector |
| PV | Photovoltaic |
| RMS | Root Mean Square |
| SOGI | Second Order Generalized Integrator |
| UPC | Universitat Politècnica de Catalunya |
| UPFC | Unified Power Flow Controller |
| UPQC | Unified Power Quality Conditioner |
| VRE | Variable Renewable Energy |

# Preface

# 1 Introduction

## 1.1 Background

The Optimal Power Flow (OPF) is regarded as a complicated mathematical problem of upmost importance for grid operators. While a grid could operate in very varied conditions, the goal is to pick an optimal operating point that minimizes a given objective function and at the same time respects a set of technical constraints.

As an originally non-convex hard problem, the OPF can take many forms. For instance, an economical dispatch would be the simplest variation in which the power flows are dismissed; the DCOPF considers line flows but only solves for the voltage angles; whereas the ACOPF follows the purest formulation using the complete formulation of the problem. Both of them have their strong and weak points, exchanging computational speed for precision (and even feasibility) of the solution. [**chatzivasileiadis2018optimization**].

Regarding a more technical approach, one would consider the AC-OPF to be the best choice without loosing the ability to perform economic analysis. The straight forward approach of the basic AC-OPF problem is Matpower, an open source package for Matlab that allows the user to perform simulations with specific grid models. It is regarded as the baseline of the AC-OPF formulation for many further works. The solution found, due to the non-convex nature of the problem, is a local optimal point.

There have been many development lines that try to find a global optimal point using relaxations over the constraints of the problem in such a way that the resulting transformed problem is convex where it is posible to find the global optimal point using Interior Point Methods (IPMs from now on). These relaxation methods, unlike the DC aproximations, expand the feasibililty domain of solutions instead of modifying it as seen in FIGURE 1

## 1.2 Objectives

The aim of this project, which is part of the Red Eléctrica de España (REE) SIROCO project, is the implementation of an AC-OPF in GridCal, a Python grid analysis package in development by REE's engineer Santiago Peñate Vera. The starting point of this tool will be the Matpower formulation, with the objective of integrating it and expanding its functionalities.

The first steps will be directed to understanding the formulation and incorporate the solver in Python, adapting the initialization of the problem to the GridCal models, which can be obtained from the matpower models using a built-in parser. During this proces, the formulation has to be as general as possible to allow the introduction of new features.

Once the solver is operative, the new features will be added to the model. These features will be optional and selectable from a GUI. The connection of this software with the GUI is out of the scope of this project, as it will be done by a third party. The additional features are the possibility to optimize controlled transformer setpoints, simple DC links models, reactive generation limitations and the addition of slack variables for operational range that can be violated in exchange of a penalty cost. This last feature improves (and sometimes even allows) the convergence of some grid states that are difficult to solve.

On top of that, the resulting algorithm will be prepared to introduce new features already being discused such as a complete AC/DC modelling, better control models and even the addition of relaxations in case faster studies are required.

## 1.3 Outline

Chapter 1 will describe all the elements of an electrical grid, the parameters that define each of them and the notation used throughout the thesis. The relationships between elements and the physics that determine the operation of an electrical grid are described in Chapter 2. It will be followed in Chapter 3 by the mathematical modelling of the optimization problem that has to be solved to determine optimal operation. Chapter 4 contains all the software structure and algorithms developments, accompanied by some explanations of the GridCal environment. In Chapter 5, benchmarking cases will be shown for each implemented feature.

## 2 Grid Elements

This section describes the elements and information contained in the grid model. The two elements that shape the networks are the buses and lines, which can be also named nodes and branches using a topological naming. On top of some of the buses we can find generators and loads, and on top of some of the lines there can be transformers.

All of these elements have associated information about their working parameters, regarding operation limits, element subtype, monitoring information and other relevant information. Every piece of information needed to run the optimization program over the grid is described in this section.

### 2.1 Buses

The buses, or the nodes of the network, are the points on the grid where generation or consumption of electric power occurs. The generation comes from generators connected to the bus, while the consumption, often refered to as load, comes from the sum of all the users of electrical devices connected to the bus.

When dealing with a transport grid, buses can be seen as the substations that feed an entire neighbourhood, town, or even city. While there is a whole distribution network behind each bus, they are modelled as single points that aggregate all the generation and consumption beneath them. It will be shown later that larger generator units are treated individually since their operating points are typically dictated by the TSO and they have the ability to affect the operation of the grid.

The set of buses of the electrical grid is denoted as $N = \{1, ..., i, ..., j, ..., n\}$, where the index $i$ is used for the $from$ bus, while $j$ is used for the $to$ bus when dealing with an interaction of two buses. Each bus has a state represented by a 2 complex variables, the complex voltage $v_i\angle\theta_i$ and the net apparent power $S_i = P_i + jQ_i$.

The voltage magnitude is expressed in per unit, each grid having its own reference voltage. The phase, expressed in radians, corresponds to the angular difference with respect to the reference bus, also known as the *slack* bus, which is assigned a phase value of 0. The real part of the net apparent power corresponds to the active power, while the imaginary part corresponds to the reactive power. This net power is the balance of the bus's generation and consumption. It must match the power exchange with other buses to accomplish the nodal power balance, which imposes that the net power of the bus should be 0 in equilibrium.

There are several types of bus depending on which variables can be controlled. The most common ones, and the ones that will be considered in the modelling of the grid, are the following:

- The *slack* bus, or reference bus, serves as a voltage magnitude and angle reference. As such, its voltage variables are set as $V_i \angle \theta_i = 1\angle 0°$, using per unit. This type of bus can be regarded as the balancing bus, as it will provide the power necessary to solve any power inbalance when the power flow is solved. Generally, they are the bus with the greatest and most stable generation capacity.

  Since there can be more than one slack bus in a grid due to being diferent electrical islands, there will be a list of $nslack$ buses called $slack$. It is important to note that each island has to have its own slack bus, otherwise the power flow will not be solvable.

- The PQ buses have known power exchange values. The $P_i$ and $Q_i$ values are forecasted for all the simulation runtime. They normally are buses where there is no generation, and the exchange power is directly given by the demand forecast. The voltage magnitude and angle are unknowns the value of which will be obtained after solving the power flow.

  The list of PQ buses is denoted as $pq$ and has $npq$ elements. Slack buses can be included in this list, and they will only have a fixed voltage phase of 0. The buses that will fit in this list are the ones with different upper and lower voltage limits.

- The PV buses have their active power exchange $P_i$, and their voltage module $v_i$ as controlled magnitudes. Their values are setpoints determined by their operator The reactive power generation and the phase of the bus are unknowns the value of which is obtained after solving the power flow.

  The list of PV buses is denoted as $pv$ and has $npv$ elements. Same as with the PQ buses, slack buses can be included in this list. The buses with equal upper and lower voltage limits will be included in this list, and will have that value as the voltage setpoint.

The goal of the optimization will to obtain the optimal setpoints of this controlled buses, as it will be shown in the optimization formulation.

## 2.2 Lines

The electrical lines connect two buses and allows the transportation of power between them. The power transfer can happen in both directions, although obiously only one at a time. The line has two ends, the 'from' end $f$ and the 'to' end $t$. The lines are identified with the index of each bus at each position and will remain the same independently of the direction of the flow. The sign of the power will be positive when going in the direction 'from' $\rightarrow$ 'to', and negative in the direction 'to' $\rightarrow$ 'from'.

The set of lines is denoted as $M = \{1, ..., k, ..., m\}$, while the set of 'from' and 'to' buses is denoted as $F = \{f_1, ..., f_m\}$ and $T = \{t_1, ..., t_m\}$ respectively. A line is defined used the notation

$k \triangleq (f_k, t_k)$. This line will be modelled as a $\pi$-equivalent line, which will mean that the power transfer will be calculated using the resulting admittance expression and the variables of each bus. The power calculation is derived in following sections.

The lines have a limit over the power they can withstand. This limit, called line rate, has to be checked at both ends of the line, since it is not equal due to line losses. The lines that are monitored and will have this rate applied are the ones with the monitor loading flag set to 1. The list is obtained from the GridCal model of the grid, and will be stored in the $il$ list of indexes containing $nll$ monitored branch elements.

### 2.3 Transformers

The transformers change the level of the voltage between both of its ends. The voltage when going from generation towards the transport grid raises to lower loses, and when arriving to consumption areas lowers again to reduce the risks of manipulation. In the model presented, the transformers are modelled as lines with a transformation of the votlage level. Since per unit is used, this change does not directly imply modification of the voltage module value, but when recovering the actual value in Volts, each bus will have its own nominal voltage level.

In addition to this, the transformers' output voltages are considered adjustable in both module and phase using the called tap variables. Tap module $m_p$ corresponds to the deviation, in per unit, from the nominal transformation ratio, while tap angle $\tau$, in radians, corresponds to the shift added to the voltage phase. The majority of the transformers are considered not controlled, but some of them can be controlled in one or both variables. For that reason, their setpoints can be added as variables of the optimization problem.

The indexing lists containing the branches with controlled transformers is $k_m$ for module controlled transformers and $k_tau$ for phase controlled transformers. Those who have controls for both variables will be in both lists.

## 3 Grid Model

Once all the elements of the grid have been defined, the model of the power flow can be properly explained. The approach chosen is aligned with the models proposed by GridCal, which make use of the Universal Branch Model to describe the admittance matrices of the branches. The power flow model used during this project has been the Bus Injection Model (BIM), which will be calculated at each bus of the grid, and has to be equal to zero for the system to be in equilibrium. This model relates voltages, power transfer, generation and demand, which are the variables of the optimization problem.

### 3.1 Power Flow Equations

We will show the construction of the power flow equations, since some of the intermediate steps can be useful for later purposes. We start from the power flowing through a branch using the Universal Branch Model for the $k$ branch:

$$\begin{bmatrix} i_f \\ i_t \end{bmatrix} = \begin{bmatrix} y_{ff} & y_{ft} \\ y_{tf} & y_{tt} \end{bmatrix} \begin{bmatrix} v_f \\ v_t \end{bmatrix} = \begin{bmatrix} Y_{br} \end{bmatrix} \begin{bmatrix} v_f \\ v_t \end{bmatrix} \tag{1}$$

This equation relates the voltages of the buses connected by the branch and the currents flowing from them. The branch admittance matrix terms can be calculated using the parameters of the branch as follows:

$$Y_{br} = \begin{bmatrix} (y_s + j\frac{b_c}{2})\frac{1}{m_p^2} & -\frac{y_s}{m_p e^{-j\tau}} \\ -\frac{y_s}{m_p e^{j\tau}} & y_s + j\frac{b_c}{2} \end{bmatrix} \tag{2}$$

where $y_s$ is the series admittance calculated as $y_s = \frac{1}{R+jX}$, $b_c$ is the capacitance of the line, $m_p$ is the transformer ratio, and $\tau$ is the phase shift of the transformer.

The expression above contains the information of a single line. In the grid model used, $y_f f$, $y_f t$, $y_t f$ and $y_t t$ are vectors of length $m$ that store these primitives of the admittance matrix for all the branches in the model.

Consider the connectivity matrices $C_f$ and $C_t$, with size $kxn$ that relate the index of the 'from' and 'to' buses to the index of their branch as follows:

$$C_{f_{ki}} = \begin{cases} 1, & \text{for } k == (i,j) \quad \forall j \in \boldsymbol{N} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

$$C_{t_{kj}} = \begin{cases} 1, & \text{for } k == (i,j) \quad \forall i \in \boldsymbol{N} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

Using these connectivities and the primitives, $from$ and $to$ admittance matrices can be constructed. The $to$ admittance matrix contains the elements of the admittance matrix that are connected to the 'to' bus of the branch, while the $from$ admittance matrix contains the elements connected to the 'from' bus of the branch. The bus admittance matrix $Y_{bus}$, which contains all the information needed to calculate the power flow, can then be constructed.

$$
\begin{aligned}
Y_f &= [\boldsymbol{Y_{ff}}]\, C_f + [\boldsymbol{Y_{ft}}]\, C_t \\
Y_t &= [\boldsymbol{Y_{tf}}]\, C_f + [\boldsymbol{Y_{tt}}]\, C_t \\
Y_{\text{bus}} &= C_f^\top Y_f + C_t^\top Y_t + [Y_{sh}]
\end{aligned} \tag{5}
$$

where the brackets indicate that the vector is transformed into a diagonal matrix and the $Y_{sh}$ matrix contains the shunt admittance of each bus, which is a given parameter of the model. The resulting matrices have size *m x n*.

The model used to calculate the power balance of the system is the bus injection model, which is based on the equality between the power injected in a bus and power flowing out of it. The injection can come from generators connected at the bus or lines from buses while the outflow goes to demand connected to the bus or lines that transport energy to other buses.

The vector of complex power injections from lines $S_{bus}$ is given by the following equations:

$$
\boldsymbol{S_{bus}} = [\boldsymbol{V}](Y_{bus}\boldsymbol{V})^* \tag{6}
$$

where $V$ is the vector of the complex voltages of the buses calculated as $\boldsymbol{v}e^{j\boldsymbol{\theta}}$ using element to element multiplication. Again, the notation $[V]$ refers to the diagonal matrix created from the vector $V$. The complete balance can be now calculated as:

$$
\boldsymbol{G^S_{bus}} = \boldsymbol{S_{bus}} - \boldsymbol{S^g_{bus}} + \boldsymbol{S^d_{bus}} = \boldsymbol{0} \quad \text{(equilibrium)} \tag{7}
$$

The notation $G^S$ will be useful later to refer to this balance as an equality constraint of the optimization problem.

One particularity to be considered is that there could be multiple generators connected to the same bus, and each of them has to be considered separatedly since there could be differences in price or operational limits. To do so, $\boldsymbol{S^g}$ is obtained through the use of the *n x ng* connectivity matrix $C_G$:

$$
\boldsymbol{S^g_{bus}} = C_G \boldsymbol{S^g} \tag{8}
$$

In addition to the bus power balance, the line power balance has to be considered to ensure that operational limits of the lines are not exceeded. The power flowing through a line can be calculated using the $from$ and *to* admittance matrices and the voltages of the buses connected to the line. Both sides of the lines have to be considered, since the powers at the ends are not equal and the direction of the flow can change depending on grid conditions.

$$\begin{aligned}
\boldsymbol{V_f} &= C_f \boldsymbol{V} \\
\boldsymbol{V_t} &= C_t \boldsymbol{V} \\
\boldsymbol{S_f} &= [\boldsymbol{V_f}](Y_f \boldsymbol{V})^* \\
\boldsymbol{S_t} &= [\boldsymbol{V_t}](Y_t \boldsymbol{V})^*
\end{aligned} \tag{9}$$

Here it is important to note that the vector $V_f$ will grab multiple times the voltage of the buses that are acting as $from$ of multiple lines.

## 3.2 Operational limits

Each element of the grid has its owns operational limits that have to be respected. The limits are stored in lists of length equal to the respective element count. These limits correspond to inequalities of the optimization problem. The notation $H^I$ is used to identify to each diferent limit.

### 3.2.1 Nodal voltage limits

The voltage module limits of the buses are stored in the $v_{max}$ and $v_{min}$ lists, with its value in per unit. The limits are checked using the following equations:

$$\begin{aligned}
\boldsymbol{H^{v_u}} &= \boldsymbol{v} - \boldsymbol{v_{max}} \leq \boldsymbol{0} \quad \text{(upper limit)} \\
\boldsymbol{H^{v_l}} &= \boldsymbol{v_{min}} - \boldsymbol{v} \leq \boldsymbol{0} \quad \text{(lower limit)}
\end{aligned} \tag{10}$$

### 3.2.2 Generation limits

The generation limits apply to both the active and reactive power of the generators. They are stored in the $P_{max}$, $P_{min}$, $Q_{max}$ and $Q_{min}$ lists, with its value in per unit. The limits are checked using the following equations:

$$\begin{aligned}
\boldsymbol{H^{P_u}} &= \boldsymbol{P_g} - \boldsymbol{P_{max}} \leq \boldsymbol{0} \quad \text{(upper limit)} \\
\boldsymbol{H^{P_l}} &= \boldsymbol{P_{min}} - \boldsymbol{P_g} \leq \boldsymbol{0} \quad \text{(lower limit)} \\
\boldsymbol{H^{Q_u}} &= \boldsymbol{Q_g} - \boldsymbol{Q_{max}} \leq \boldsymbol{0} \quad \text{(upper limit)} \\
\boldsymbol{H^{Q_l}} &= \boldsymbol{Q_{min}} - \boldsymbol{Q_g} \leq \boldsymbol{0} \quad \text{(lower limit)}
\end{aligned} \tag{11}$$

### 3.2.3 Transformer limits

The controllable transformers will have their limits for the controllable variables limited. THe limits are similar to the previous:

$$
\begin{aligned}
\boldsymbol{H}^{m_{p_u}} &= \boldsymbol{m_p} - \boldsymbol{m_{p_{max}}} \leq \mathbf{0} \quad \text{(upper limit)} \\
\boldsymbol{H}^{m_{p_l}} &= \boldsymbol{m_{p_{min}}} - \boldsymbol{m_p} \leq \mathbf{0} \quad \text{(lower limit)} \\
\boldsymbol{H}^{\tau_u} &= \boldsymbol{\tau} - \boldsymbol{\tau_{max}} \leq \mathbf{0} \quad \text{(upper limit)} \\
\boldsymbol{H}^{\tau_l} &= \boldsymbol{\tau_{min}} - \boldsymbol{\tau} \leq \mathbf{0} \quad \text{(lower limit)}
\end{aligned}
\tag{12}
$$

### 3.2.4 Line limits

The line limits will be checked considering the rating of each line in per unit. Since the direction of the flow is not known, and to avoid using absolute values, this constraint is considered quadratically.

$$
\begin{aligned}
\boldsymbol{H}^{S_f} &= \boldsymbol{S_f}\boldsymbol{S_f^*} - \boldsymbol{S_{max}^2} \leq \mathbf{0} \\
\boldsymbol{H}^{S_t} &= \boldsymbol{S_t}\boldsymbol{S_t^*} - \boldsymbol{S_{max}^2} \leq \mathbf{0}
\end{aligned}
\tag{13}
$$

### 3.2.5 Bound slack variables

So far, all the constraints described has physical meaning and are related to the grid model. The last set of constraints are related to the optimization problem and are used to improve, or even make posible, the convergence of the solver. The bound slacks are used to allow the solution to surpass some operational limits in order to find a feasible solution. Of course, it should come at a cost and the objective function will be used to penalize this behaviour.

These bound slacks are applied to the voltage limits and to the line limits, allowing sligh over/undervoltages and some overloads. The constraint equations are modified and the slack variables are imposed to be positive. The equations are as follows:

$$
\begin{aligned}
\boldsymbol{H^{v_u}} &= \boldsymbol{v} - \boldsymbol{v_{max}} - \boldsymbol{sl_{v_{max}}} \leq \mathbf{0} \quad \text{(upper limit)} \\
\boldsymbol{H^{v_l}} &= \boldsymbol{v_{min}} - \boldsymbol{v} - \boldsymbol{sl_{v_{min}}} \leq \mathbf{0} \quad \text{(lower limit)} \\
\boldsymbol{H^{S_f}} &= \boldsymbol{S_f S_f^*} - \boldsymbol{S_{max}^2} - \boldsymbol{sl_{sf}} \leq \mathbf{0} \quad \text{(from limit)} \\
\boldsymbol{H^{S_t}} &= \boldsymbol{S_t S_t^*} - \boldsymbol{S_{max}^2} - \boldsymbol{sl_{st}} \leq \mathbf{0} \quad \text{(to limit)} \\
-\boldsymbol{H^{sl_{v_{max}}}} &\leq \mathbf{0} \\
-\boldsymbol{H^{sl_{v_{min}}}} &\leq \mathbf{0} \\
-\boldsymbol{H^{sl_{sf}}} &\leq \mathbf{0} \\
-\boldsymbol{H^{sl_{st}}} &\leq \mathbf{0}
\end{aligned}
\tag{14}
$$

The use of this bound slacks is optional and can be removed from the optimization problem if the user prefers to have a solution in compliance with the stablished limits.

# 4 Optimization Problem

This section describes the optimization model used to solve the AC Optimal Power Flow problem. The mathematical formulation of this problem is presented using abstract notation of the KKT conditions for optimization problems, which later on will regain a more practical form. The implementation of the solver is also described, as well as the Python code used to solve the problem. The general theory of optimization can be found in the book of Boyd and Vandenberghe [**boyd2004convex**], in Nocedal and Wright, and the adaptation of this theory to a more compact formulation can be reviewed in the Matpower Interior Point Solver documentation.

## 4.1 Optimization problem

The general optimization problem involving equality and inequality constraints can be formulated as follows:

$$
\begin{aligned}
\min \quad & f(\boldsymbol{x}) \\
\text{s.t.} \quad & \boldsymbol{G}(\boldsymbol{x}) = \mathbf{0} \\
& \boldsymbol{H}(\boldsymbol{x}) \leq \mathbf{0}
\end{aligned}
\tag{15}
$$

where $\boldsymbol{x} \in \mathbb{R}^n$, being $n$ the number of decision variables, $f(x)$ is the function to be minimized which typically accounts for the generation cost, $\boldsymbol{G}(\boldsymbol{x})$ is a set of equality constraints of the problem, and $\boldsymbol{h}(\boldsymbol{x})$ contains the technical restrictions to respect voltage and line flow limits. The bold notation is used to indicate vectors.

The problem will be solved using the barrier parameter method, which consists of solving a sequence of optimization problems with different values of the barrier parameter $\gamma$. The barrier parameter is an interior point method used to penalize the violation of the inequality constraints in the problem. This algorithm is used to solve the problems by moving through the interior of the feasible region.

The lagrangian of the problem is defined as:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^\top \boldsymbol{G}(\boldsymbol{x}) + \boldsymbol{\mu}^\top (\boldsymbol{H}(\boldsymbol{x}) + \boldsymbol{Z}) - \gamma \sum_{i=1}^{n_i} \log(z_i) \tag{16}$$

where $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are the Lagrange multipliers associated with the equality and inequality constraints respectively.

The KKT conditions are a set of equations that must be satisfied by the solution of the optimization problem. These conditions are a result of imposing that the partial derivatives of the Lagrangian in the $(x, Z, \lambda, \mu)$ variable space equals 0:

$$\begin{aligned}
\nabla_{x,Z,\lambda,\mu}\mathcal{L}(\boldsymbol{x}, \boldsymbol{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \mathbf{0}& \\
\mathcal{L}_x(\boldsymbol{x}, \boldsymbol{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= f_x + \boldsymbol{\lambda}^\top G_x + \boldsymbol{\mu}^\top H_x = \mathbf{0} \\
\mathcal{L}_Z(\boldsymbol{x}, \boldsymbol{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= \boldsymbol{\mu}^\top - \gamma \mathbf{1}_{n_i}^\top [\boldsymbol{Z}]^{-1} = \mathbf{0} \\
\mathcal{L}_\lambda(\boldsymbol{x}, \boldsymbol{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= \boldsymbol{G}^\top(\boldsymbol{x}) = \mathbf{0} \\
\mathcal{L}_\mu(\boldsymbol{x}, \boldsymbol{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= \boldsymbol{H}^\top(\boldsymbol{x}) + \boldsymbol{Z}^\top = \mathbf{0}
\end{aligned} \tag{17}$$

where $\mathbf{1}_{n_i}$ is a vector of ones of length $n_i$, and $\gamma$ is the barrier parameter that will be updated in each iteration of the solver until reaching a value close to zero. The notation $\{_X$ indicates the partial derivative of the given function with respect to the $X$, which is a gradient in case $X$ is a vector of multiple variables.

The first gradient corresponds to the stationarity condition of the problem, the second gradient corresponds to the complementary slackness, and the other two gradients correspond to the primal feasibility conditions of the problem. These conditions,alongside the dual feasibility condition $Z_i, \mu_i \geq 0 \quad \forall i$, are necessary and sufficient for the solution of the optimization problem. The resulting set of equations will be solved computationally fue to the non-convex nature of the problem. For this type of general problem, the approach chosen is the use of an Interior Point Method (IPM) solver.

## 4.2 Interior Point Method solver

The choice of this type of solver is regarded as the best option to deal with non-linear optimization, as shown in the dedicated chapter 19 in Nocedal and Wright's book. The IPM solver is a type of optimization algorithm that solves the KKT conditions of the problem iteratively. There are multiple algorithms that can be used, although the one presented in this project is the Newton-Raphson. In reference we this algorithm is compared with the Gauss-Seidel algorithm, yielding better scalability for the Newton-Raphson method. Since this algorithm will be used by the Spanish TSO, the scalability of the solver is a key factor to consider. There can be improvements to this method as presented in the literature, but for the purpose of this project, the IPM has been implemented directly using the Newton-Raphson method with a step-control mechanism.

The Newton-Raphson iterative process that will find the roots of the KKT conditions can be described as follows:

**Algorithm 1** Newton-Raphson Iterative Process

1: Initialize $\boldsymbol{x}_0$, tolerance $\epsilon$, and set $k = 0$
2: **while** $||\boldsymbol{f}(\boldsymbol{x}_k)|| > \epsilon$ **do**
3: Compute the Jacobian matrix $J(\boldsymbol{x}_k)$
4: Solve $J(\boldsymbol{x}_k)\Delta\boldsymbol{x}_k = -\boldsymbol{f}(\boldsymbol{x}_k)$ for $\Delta\boldsymbol{x}_k$
5: Apply step contol to $\Delta\boldsymbol{x}_k$
6: Update $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta\boldsymbol{x}_k$
7: $k = k + 1$
8: **end while**

The algorithm will perform several Newton-Raphson steps until the convergence criteria is met, meaning that equations **??** are solved. The structure of this algorithm substituting the submatrices present at the jacobian matrix is as follows:

$$
-\begin{pmatrix} \boldsymbol{\mathcal{L}_{xx}} & \mathbf{0} & \boldsymbol{G}_{\boldsymbol{x}}^{\top}(\boldsymbol{x}) & \boldsymbol{H}_{\boldsymbol{x}}^{\top}(\boldsymbol{x}) \\ \mathbf{0} & [\boldsymbol{\mu}] & \mathbf{0} & [\boldsymbol{Z}] \\ \boldsymbol{G}_{\boldsymbol{x}}(\boldsymbol{x}) & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{H}_{\boldsymbol{x}}(\boldsymbol{x}) & \boldsymbol{I} & \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\Delta x} \\ \boldsymbol{\Delta Z} \\ \boldsymbol{\Delta \lambda} \\ \boldsymbol{\Delta \mu} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\mathcal{L}}_x^{\top} \\ [\boldsymbol{\mu}]\boldsymbol{Z} - \gamma \mathbf{1}_{n_i} \\ \boldsymbol{G}(\boldsymbol{x}) \\ \boldsymbol{H}(\boldsymbol{x}) + \boldsymbol{Z} \end{pmatrix} \tag{18}
$$

where $\boldsymbol{\Delta x}$ represents the variation of $\boldsymbol{x}$ (and similarly to $\boldsymbol{s}$, $\boldsymbol{y}$, $\boldsymbol{z}$), $\boldsymbol{I}$ is the identity matrix, and the term $\boldsymbol{\nabla^2_{xx}\mathcal{L}}$ is the derivative of the first KKT condition, which was already the gradient of the Lagrangian of the problem.

Note that the last of the KKT conditions is not included in the system of equations. This condition will be enforced by the algorithm when choosing the step length of the iteration when updating the variables.

This system of equations can be further reduced using methods that can be found in Nocedal and Wright's book, derived with detail in the MIPS Manual. The relevant equations obtained during the reduction process are the following:

$$\begin{aligned}[\boldsymbol{\mu}]\Delta \boldsymbol{Z} + [\boldsymbol{Z}]\Delta\boldsymbol{\mu} &= -[\boldsymbol{\mu}]\boldsymbol{Z} + \gamma \mathbf{1}_{\boldsymbol{n_i}} \\ [\boldsymbol{Z}]\Delta\boldsymbol{\mu} &= -[\boldsymbol{Z}]\boldsymbol{\mu} + \gamma \mathbf{1}_{\boldsymbol{n_i}} - [\boldsymbol{\mu}]\Delta\boldsymbol{Z} \\ \Delta\boldsymbol{\mu} &= -\boldsymbol{\mu} + [\boldsymbol{Z}]^{-1}(\gamma \mathbf{1}_{\boldsymbol{n_i}} - [\boldsymbol{\mu}]\Delta\boldsymbol{Z}). \\ \boldsymbol{H}_{\boldsymbol{x}\Delta\boldsymbol{X}+\Delta\boldsymbol{Z}} &= -\boldsymbol{H}(\boldsymbol{X}) - \boldsymbol{Z} \\ \Delta\boldsymbol{Z} &= -\boldsymbol{H}(\boldsymbol{X}) - \boldsymbol{Z} - \boldsymbol{H}_{\boldsymbol{x}\Delta\boldsymbol{X}.}\end{aligned} \tag{19}$$

$$\begin{aligned}\boldsymbol{M} &\equiv \boldsymbol{\mathcal{L}}_{\boldsymbol{xx}} + \boldsymbol{H}_{\boldsymbol{x}}^{\top}[\boldsymbol{Z}]^{-1}[\boldsymbol{\mu}]\boldsymbol{H}_{\boldsymbol{x}} \\ &= \boldsymbol{f}_{\boldsymbol{xx}} + \boldsymbol{G}_{\boldsymbol{xx}}(\boldsymbol{\lambda}) + \boldsymbol{H}_{\boldsymbol{xx}}(\boldsymbol{\mu}) + \boldsymbol{H}_{\boldsymbol{x}}^{\top}[\boldsymbol{Z}]^{-1}[\boldsymbol{\mu}]\boldsymbol{H}_{\boldsymbol{x}} \\ \boldsymbol{N} &\equiv \boldsymbol{\mathcal{L}}_{\boldsymbol{x}}^{\top} + \boldsymbol{H}_{\boldsymbol{x}}^{\top}[\boldsymbol{Z}]^{-1}(\gamma \mathbf{1}_{\boldsymbol{n_i}} + [\boldsymbol{\mu}]\boldsymbol{H}(\boldsymbol{x})) \\ &= \boldsymbol{f}_{\boldsymbol{x}}^{\top} + \boldsymbol{G}_{\boldsymbol{x}}^{\top}\boldsymbol{\lambda} + \boldsymbol{H}_{\boldsymbol{x}}^{\top}\boldsymbol{\mu} + \boldsymbol{H}_{\boldsymbol{x}}^{\top}[\boldsymbol{Z}]^{-1}(\gamma \mathbf{1}_{\boldsymbol{n_i}} + [\boldsymbol{\mu}]\boldsymbol{H}(\boldsymbol{x})).\end{aligned} \tag{20}$$

The reduced system of equations using this refactoring is:

$$\begin{bmatrix} \boldsymbol{M} & \boldsymbol{G}_{\boldsymbol{x}}^{\top} \\ \boldsymbol{G}_{\boldsymbol{x}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{x} \\ \Delta\boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{N} \\ -\boldsymbol{G}(\boldsymbol{x}) \end{bmatrix} \tag{21}$$

As it can be seen, the inequality constraint are not included in the system of equations. The solver will firstly solve this smaller matricial problem to obtain the variation of the decision variables and the Lagrange multipliers associated with the equality constraints. Then, the variation of the inequality constraint multipliers and slack will be found using the expressions derived above.

Before proceeding to update the values of the variables, there will be two additional steps to be taken. The first one is to ensure that the dual feasibility condition

$$\boldsymbol{Z_i}, \boldsymbol{\mu_i} \geq 0 \quad \forall i$$

is met. This will be done by using the maximum step length such that the values of the multipliers remain positive. The calculation of this step length is the follwoing:

$$
\begin{aligned}
\alpha_p &= \min\left(\tau \cdot \min\left(-\frac{\boldsymbol{Z}}{\Delta \boldsymbol{Z}}\right), 1\right) \\
\alpha_d &= \min\left(\tau \cdot \min\left(-\frac{\boldsymbol{\mu}}{\Delta \boldsymbol{\mu}}\right), 1\right)
\end{aligned}
\tag{22}
$$

where $\tau$ is a parameter that will be set to a value really close to 1, withouth exceeding it (i.e. 0.9995).

This step length will be tested to ensure that it is not too large as it could lead to divergence of the algorithm. This will be done by using a step control mechanism that will reduce the step length if the conditions are not met. The step control mechanism will update just the decision variables $x$, using the Lagrangian of the system as a reference. The step control mechanism is described in the following pseudo-code:

Which will reduce all the step sizes if the new lagrangian does not accomplish the condition. If this condition is not activated, then $\alpha = 1$ Now, the values of all the variables and multipliers can be updated with the final values of the step length.

$$
\begin{aligned}
\boldsymbol{x} &\leftarrow \boldsymbol{x} + \alpha\alpha_p \Delta \boldsymbol{x} \\
\boldsymbol{Z} &\leftarrow \boldsymbol{Z} + \alpha\alpha_p \Delta \boldsymbol{Z} \\
\boldsymbol{\lambda} &\leftarrow \boldsymbol{\lambda} + \alpha\alpha_d \Delta \boldsymbol{\lambda} \\
\boldsymbol{\mu} &\leftarrow \boldsymbol{\mu} + \alpha\alpha_d \Delta \boldsymbol{\mu}
\end{aligned}
\tag{23}
$$

The barrier parameter is updated after this new point is calculated using the following expression:

$$
\gamma \leftarrow \sigma \frac{\boldsymbol{Z}^\top \boldsymbol{\mu}}{n_i} \tag{24}
$$

where $\sigma$ is a parameter that will be between 0 and 1. The value of $\sigma$ will be set to 0.1, as in Matpower, although Nocedal and Wright proposes some different approaches with different values of this parameter when the solver approaches the solution.

Finally, the last step of the solver will be to calculate the convergence criteria to determine if there should be another iteration of the algorithm. There are three criteria that will be used to determine if the algorithm: The feasibility condition, the gradient condition and the gamma value. All of them have to be below a tolerance $\epsilon$ to consider the problem solved. The later condition can be checked performing direct comparison, while the former two can be calculated as:

$$
\begin{aligned}
\text{feascond} &= \frac{\max\Big(\big[\|\mathbf{G}\|_\infty, \max(\boldsymbol{H})\big]\Big)}{1+\max\Big(\big[\|\mathbf{x}\|_\infty, \|\mathbf{Z}\|_\infty\big]\Big)} \\
\text{gradcond} &= \frac{\|\mathcal{L}_\mathbf{x}\|_\infty}{1+\max\Big(\big[\|\lambda\|_\infty, \|\mu\|_\infty\big]\Big)}
\end{aligned}
\tag{25}
$$

The algorithm will continue to iterate until the convergence criteria is met, although there is a maximum number of iterations parameter to avoid infinite loops.

## 4.3 Python implementation

The resulting algorithm is implemented in Python as a stand-alone optimization solver. The solver is capable of throwing any kind of problem as long as the objective function, equality constraints, and inequality constraints are provided alongside their gradients and hessians. The user will have to provide a function that returns an object containing this equations, aswell as the desired initialization and tolerance values.

The following pseudo-code will represent the complete algorithm including all the steps. In this project, the function called to calculate the values of the problem equations will be described in the next section.

### 4.3.1 Dealing with sparsity

A small comment to be made is that the solver will work with sparse matrices. Without entering in much detail, the usage of this type of matrices available in the SciPy library will allow the solver to be more efficient when dealing with the large systems that result when modelling power grids that are composed of thousands of buses and lines.

This type of matrices are used to store only the non-zero elements of the matrix, not wasting memory in storing the zeros. For instance, the Jacobian obtained in the IEEE14 test case has a size of 129 x 129, although only 794 of the elements are non-zero, representing 4.8% of the total elements.

The larger the system, the lower this fraction of non-zero values is. A case of 300 buses used as a benchamrk contains 0.3% of non-zero values in the Jacobian. For this reason, the usage of sparse matrices is extremely important to solve country-sized power grids.

# 5 AC-OPF

The Optimal Power Flow (OPF) is regarded as a complicated mathematical problem of upmost importance for grid operators. While a grid could operate in very varied conditions, the goal is to pick an optimal operating point that minimizes a given objective function and at the same time respects a set of technical constraints.

As an originally non-convex hard problem, the OPF can take many forms. For instance, an economical dispatch would be the simplest variation in which the power flows are dismissed; the DCOPF considers line flows but only solves for the voltage angles; whereas the ACOPF follows the purest formulation, which comes at a cost [**chatzivasileiadis2018optimization**]. Relaxations are commonly employed to convexify the problem [**ergun2019optimal**], which makes it easier to obtain a satisfactory feasible solution.

Nonetheless, in this document we abstain ourselves from any oversimplification that deviates from the original problem, and we build an Interior Point Method (IPM) to achieve maximum performance as well as avoiding third-party dependencies such as IPOPT.

As explained earlier, the aim of the project is solving the power flow without simplificating any equation that can give unfeasible results, and that is why the implementation of the solver is done using an IPM. In this chapter, the modelling of the AC-OPF problem as an IPM type of problem is presented. The resulting model will allow the use of this solver for any case study formulated using GridCal. Being a non-convex optimization, the convergence is not guarenteed, although there will be some features that will ease the convergence for grids that have some difficult bounds to satisfy.

## 5.1 Decision variables

The power flow is modelled using polar form for voltages. The first group of variables correspond to the traditional power flow variables. Then, the bound slack variables are introduced for the line flow limits and the voltage limits. The transformer variables are added next, and finally the DC links $from$ powers. The resulting variable vector is:

$$\boldsymbol{x} = [\boldsymbol{\theta}^T, \boldsymbol{\mathcal{V}}^T, \boldsymbol{P_g}^T, \boldsymbol{Q_g}^T, \boldsymbol{sl_{sf}}^T, \boldsymbol{sl_{st}}^T, \boldsymbol{sl_{vmax}}^T, \boldsymbol{sl_{vmin}}^T, \boldsymbol{m_p}^T, \boldsymbol{\tau}^T, \boldsymbol{Pf_{dc}}^T]^T \tag{26}$$

where $\boldsymbol{\theta}$ is the vector of size $N$ of voltage angles, $\boldsymbol{\mathcal{V}}$ is the vector of size $N$ of voltage magnitudes, $\boldsymbol{P_g}$ and $\boldsymbol{Q_g}$ are the vectors of size $Ng$ of active and reactive power generation, $\boldsymbol{sl_{sf}}$ and $\boldsymbol{sl_{st}}$ are the vectors of size $nll$ of slack variables for the line flow limits, $\boldsymbol{sl_v max}$ and $\boldsymbol{sl_{vmin}}$ are the vectors of size $npq$ of slack variables for the voltage limits, $\boldsymbol{tapm}$ and $\boldsymbol{tapt}$ are the vectors of transformer tap positions, sized $ntapm$ and $ntapt$ respectively, and $\boldsymbol{Pfdc}$ is the vector of DC link $from$ powers, sized $ndc$.

In total, there are $2N + 2Ng + 2nll + 2npq + ntapm + ntapt + ndc$ decision variables.

## 5.2 Objective function

Firstly, the objective function is described as the minimization of cost of operation of the grid. This cost is modelled as a quadratic cost with respect to the active power generation, and linear with respect to the limit violation slacks. This means that the generation has to be as cheap as the limitations allow, while this limitation can be surpassed at a greater cost in order to be able to solve complex grid states. For instance, a grid with a high demand state can have some lines slightly overloaded for short periods of time, and exchanging this overload for a penalty can be prefered to not being able to solve the grid state. Of course, the cost has to be such that the overloads are not extreme or common.

The resulting cost function is:

$$\begin{aligned} \min \quad & \sum_{i \in \mathrm{ig}} c_{i_0} + c_{i_1} P_{g_i} + c_{i_2} P_{g_i}^2 \\ & + \sum_{k \in \mathrm{il}} c_{s_k} (sl_{sf_k} + sl_{st_k}) + \sum_{i \in \mathrm{pq}} c_{sl_{v\,i}} (sl_{vmax_i} + sl_{vmin_i}) \end{aligned} \tag{27}$$

where $c_{i_0}$, $c_{i_1}$ and $c_{i_2}$ are the coefficients of the quadratic cost function for the active power generation, $c_{s_k}$ is the cost of the slack variables for the line flow limits, and $c_{sl_{v_i}}$ is the cost of the slack variables for the voltage limits.

### 5.2.1 Objective function gradient

The gradient of the objective function results in the following vector:

$$\nabla f(\boldsymbol{x}) = \begin{bmatrix} \mathbf{0}_{\boldsymbol{N}} \\ \mathbf{0}_{\boldsymbol{N}} \\ \boldsymbol{c_1} + 2\boldsymbol{c_2}\boldsymbol{P_g} \\ \mathbf{0}_{\boldsymbol{Ng}} \\ \boldsymbol{c_s} \\ \boldsymbol{c_s} \\ \boldsymbol{c_{sl_v}} \\ \boldsymbol{c_{sl_v}} \\ \mathbf{0}_{\boldsymbol{ntapm}} \\ \mathbf{0}_{\boldsymbol{ntapt}} \\ \mathbf{0}_{\boldsymbol{ndc}} \end{bmatrix} \tag{28}$$

where the subindex $\mathbf{0}_{\boldsymbol{n}}$ indicates a vector of zeros of size $n$.

#### 5.2.2 Objective function hessian

The hessian of the objective function is a diagonal matrix, with the following structure:

$$\nabla^2 f(\boldsymbol{x}) = \begin{bmatrix} 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 2c_2 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 \end{bmatrix} \tag{29}$$

with only the diagonal elements corresponding to the active power generation variables are non-zero.

### 5.3 Equality constraints

The vector of equality constraints $G$ for the AC-OPF problem has the following structure:

$$\boldsymbol{G}(\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{G_P^S} \\ \boldsymbol{G_Q^S} \\ \boldsymbol{G_{slack}^{Th}} \\ \boldsymbol{G_{pv}^V} \end{bmatrix} \tag{30}$$

where $\boldsymbol{G_P^S}$ and $\boldsymbol{G_Q^S}$ are the active and reactive nodal power balances, $\boldsymbol{G_{slack}^{Th}}$ is the slack phase reference, and $\boldsymbol{G_{pv}^{V}}$ are the voltage limits for $pv$ buses.

The power balance is calculated in complex form as in and then it is split into active and reactive power balances, since the values have to be real numbers for the solver.

To obtain the jacobian and hessian matrices, the $G$ vector needs to be derived twice, including the associated multiplier in the second derivative term. The derivatives are obtained following the Matpower documentation [**zimmerman2011matpower**], directly using the derivatives with respect to the voltage and power variables, and developing with a similar philosophy the derivatives with respect to the additional variables that are introduced in the model.

### 5.3.1 First derivatives of G

The vector of partial derivatives of the equality constraints $G_X^S$ is:

$$G_X^s = \frac{\partial G^s}{\partial X} = \begin{bmatrix} G_\theta^s & G_V^s & G_P^s & G_Q^s & \mathbf{0}_{\boldsymbol{nsl}} & G_{m_p}^S & G_\tau^S & G_{Pf_{dc}}^S \end{bmatrix}$$

where the first derivatives of the power balance equations are obtained form Matpower [**zimmerman2011matpow** as follows:

$$\begin{aligned}
G_\theta^s &= \frac{\partial s_{\text{bus}}}{\partial \theta} = [I_{\text{bus}}^*]\frac{\partial V}{\partial \theta} + [V]\frac{\partial I_{\text{bus}}^*}{\partial \theta} \\
&= \left[I_{\text{bus}}^*\right] j[V] + \left[(jV)I_{\text{bus}}^*\right] \\
&= j[V]\left([I_{\text{bus}}^*] - Y_{\text{bus}}^*[V^*]\right) \\
G_V^s &= \frac{\partial s_{\text{bus}}}{\partial V} = \left[I_{\text{bus}}^*\frac{\partial V}{\partial V} + [V]\frac{\partial I_{\text{bus}}^*}{\partial V}\right] \\
&= \left[I_{\text{bus}}^*\right][E] + [V]Y_{\text{bus}}^*[E^*] \\
&= [V]([I_{\text{bus}}^*] + Y_{\text{bus}}^*[V^*])[\mathcal{V}]^{-1} \\
G_{P_g}^S &= -C_g \\
G_{Q_g}^S &= -jC_g
\end{aligned}$$

The derivatives with respect to the tap variables have been derived with the following expressions, starting from the admittance primitives which are the quantities that depend on the tap variables:

$$\begin{aligned}
y_{ff} &= \frac{y_s}{m_p^2} \\
y_{ft} &= \frac{-y_s}{m_p e^{-j\tau}} \\
y_{tf} &= \frac{-y_s}{m_p e^{j\tau}} \\
y_{tt} &= y_s
\end{aligned} \tag{31}$$

Where $y_s = \frac{1}{R+jX}$. This primitives are used to calculate the $from$ and $to$ power flowing through a line and its primitives with the following expression:

$$
\begin{aligned}
S_f &= V_f I_f^* \\
S_t &= V_t I_t^* \\
I_f &= y_{ff} V_f + y_{ft} V_t \\
I_t &= y_{tf} V_f + y_{tt} V_t \\
\frac{\partial S_f}{\partial m_p} &= V_f \frac{\partial I_f^*}{\partial m_p} = V_f \left( \frac{-2(y_s V_f)^*}{m_p^3} + \frac{(y_s V_t)^*}{m_p^2 e^{j\tau}} \right) \\
\frac{\partial S_t}{\partial m_p} &= V_t \frac{\partial I_t^*}{\partial m_p} = V_t \frac{(y_s V_f)^*}{m_p^2 e^{-j\tau}}
\end{aligned}
\tag{32}
$$

To obtain the power injection per bus, we compose the $S$ vector using the connectivity matrices as follows:

$$
S_{bus} = C_f^T S_f + C_t^T S_t \tag{33}
$$

The equality constraints for the power flow can be written as:

$$
G^S = S_{bus} + S_{load} - C_g S_g \tag{34}
$$

The first derivatives that appear from the new variables only affect the $S_{bus}$. Deriving the expressions written above for the primitive admittances, we obtain the following expressions:

$$
\begin{aligned}
\frac{\partial G^S}{\partial m_p} &= \frac{\partial S_{bus}}{\partial m_p} = Cf^T \frac{\partial S_f}{\partial m_p} + Ct^T \frac{\partial S_t}{\partial m_p} \\
\frac{\partial G^S}{\partial \tau} &= \frac{\partial S_{bus}}{\partial \tau} = Cf^T \frac{\partial S_f}{\partial \tau} + Ct^T \frac{\partial S_t}{\partial \tau} \\
\frac{\partial S_{f_k}}{\partial m_{p_i}} &= V_{f_k} \left( \frac{-2(y_{s_k} V_{f_k})^*}{m_{p_i}^3} + \frac{(y_{s_k} V_{t_k})^*}{m_{p_i}^2 e^{j\tau}} \right) \\
\frac{\partial S_{t_k}}{\partial m_{p_i}} &= V_{t_k} \frac{(y_{s_k} V_{f_k})^*}{m_{p_i}^2 e^{-j\tau_i}} \\
\frac{\partial S_{f_k}}{\partial \tau_i} &= V_{f_k} \frac{j(y_{s_k} V_{t_k})^*}{m_{p_i} e^{j\tau_i}} \\
\frac{\partial S_{t_k}}{\partial \tau_i} &= V_{t_k} \frac{-j(y_{s_k} V_{f_k})^*}{m_{p_i} e^{-j\tau_i}}
\end{aligned}
\tag{35}
$$

Each of the individual $S_{f_k}$ and $S_{t_k}$ derivative with respect to each $i$ transformer variable will be stored in a matrix $\frac{dS_f}{dm_p}_{ki}$ and $\frac{dS_f}{dm_p}_{ki}$, with size $L$ x $ntapm$ and $L$ x $ntapt$ respectively.

The derivative $G^S_{Pf_{dc}}$ can be obtained using the indexing of the buses participant of a DC link, which are listed in the lists $fdc$ and $tdc$. Each link DC is also indexed in the list $dc$, of length $ndc$. The derivative is obtained as follows:

$$\begin{cases} G^S_{Pf_{dc}}[fdc, link] = 1 \quad \text{if} \quad \forall link = (fdc, tdc) \in dc \\ G^S_{Pf_{dc}}[tdc, link] = -1 \quad \text{if} \quad \forall link = (fdc, tdc) \in dc \\ 0 \quad \text{otherwise} \end{cases} \tag{36}$$

The complete matrix $G_X$ can be obtained concatenating the following submatrices:

$$G_X = \begin{bmatrix} \mathcal{R}(G^S) \\ \mathcal{I}(G^S) \\ G^{Th}_X \\ G^{\mathcal{V}}_X \end{bmatrix} \tag{37}$$

where $G^{Th}$ and $G^{\mathcal{V}}$ are the slack phase and voltage magnitude equalities, and their derivatives have 1 in the fixed buses positions and 0 otherwise.

### 5.3.2 Second derivatives of G

$$\begin{bmatrix} G_{\theta\theta} & G_{\theta\mathcal{V}} & G_{\theta P_g} & G_{\theta Q_g} & G_{\theta sl} & G_{\theta m_p} & G_{\theta\tau} & G_{\theta Pf_{dc}} \\ G_{\mathcal{V}\theta} & G_{\mathcal{V}\mathcal{V}} & G_{\mathcal{V}P_g} & G_{\mathcal{V}Q_g} & G_{\mathcal{V}sl} & G_{\mathcal{V}m_p} & G_{\mathcal{V}\tau} & G_{\mathcal{V}Pf_{dc}} \\ G_{P_g\theta} & G_{P_g\mathcal{V}} & G_{P_gP_g} & G_{P_gQ_g} & G_{P_gsl} & G_{P_gm_p} & G_{P_g\tau} & G_{P_gPf_{dc}} \\ G_{Q_g\theta} & G_{Q_g\mathcal{V}} & G_{Q_gP_g} & G_{Q_gQ_g} & G_{Q_gsl} & G_{Q_gm_p} & G_{Q_g\tau} & G_{Q_gPf_{dc}} \\ G_{sl\theta} & G_{sl\mathcal{V}} & G_{slP_g} & G_{slQ_g} & G_{slsl} & G_{slm_p} & G_{sl\tau} & G_{slPf_{dc}} \\ G_{m_p\theta} & G_{m_p\mathcal{V}} & G_{m_pP_g} & G_{m_pQ_g} & G_{m_psl} & G_{m_pm_p} & G_{m_p\tau} & G_{m_pPf_{dc}} \\ G_{\tau\theta} & G_{\tau\mathcal{V}} & G_{\tau P_g} & G_{\tau Q_g} & G_{\tau sl} & G_{\tau m_p} & G_{\tau\tau} & G_{\tau Pf_{dc}} \\ G_{Pf_{dc}\theta} & G_{Pf_{dc}\mathcal{V}} & G_{Pf_{dc}P_g} & G_{Pf_{dc}Q_g} & G_{Pf_{dc}sl} & G_{Pf_{dc}m_p} & G_{Pf_{dc}\tau} & G_{Pf_{dc}Pf_{dc}} \end{bmatrix} \tag{38}$$

The first set of second derivatives in the upper left (Power Flow variables $((\theta, \mathcal{V}, P_g, Q_g))$, noted $X_{PF}$) is obtained from the Technical Note 2 of Matpower [**zimmerman2011matpower**].

$$
\begin{aligned}
G^S_{XX_{PF}}(\lambda) &= \frac{\partial}{\partial X_{PF}}({G^S}^\top_{X_{PF}}\lambda) \\
&= \begin{bmatrix} G^S_{\theta\theta}(\lambda) & G^S_{\theta V}(\lambda) & 0 & 0 \\ G^S_{V\theta}(\lambda) & G^S_{VV}(\lambda) & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\
G^S_{\theta\theta}(\lambda) &= \frac{\partial}{\partial\theta}({G^S}^\top_{\theta}\lambda) \\
&= \left[V^*\right]\left(Y^{*\top}_{\text{bus}}\left[V\right]\left[\lambda\right] - [Y^{*\top}_{\text{bus}}[V]\lambda]\right) \\
&\quad + \left[\lambda\right]\left[V\right]\Big(Y^*_{\text{bus}}\left[V^*\right] - \left[I^*_{\text{bus}}\right]\Big) \\
G^S_{V\theta}(\lambda) &= \frac{\partial}{\partial\theta}(G^s_V\lambda) \\
&= j[\mathcal{V}]^{-1}([\mathbf{V}^*](\mathbf{Y}^*_{\text{bus}}[\mathbf{V}][\lambda] - [\mathbf{Y}^{*\top}_{\text{bus}}[V]\lambda]) \\
&- [\lambda][V](\mathbf{Y}^*_{bus}[V^*] - [I^*_{bus}])) \\
G^S_{\mathcal{V}\mathcal{V}}(\lambda) &= \frac{\partial}{\partial\mathcal{V}}({G^S}^\top_{\mathcal{V}}\lambda) \\
&= [\mathcal{V}]^{-1}\left([\lambda][\mathbf{V}]\mathbf{Y}^*_{\text{bus}}[\mathbf{V}^*] + [\mathbf{V}^*]\mathbf{Y}^*_{\text{bus}}[\mathbf{V}][\lambda]\right)[\mathcal{V}]^{-1}
\end{aligned}
\tag{39}
$$

The rest of the second derivatives have been obtained following a similar strategy to add the multipliers. The derivation starts again with the $from$ and $to$ expressions, and is as follows:

$$
\begin{aligned}
\frac{\partial^2 S_{f_k}}{\partial m_{p_i}\partial m_{p_i}} &= V_{f_k}\left(\frac{6(y_{s_k}V_{f_k})^*}{m_{p_i}^4} - \frac{2(y_{s_k}V_{t_k})^*}{m_{p_i}^3 e^{j\tau_i}}\right) \\
\frac{\partial^2 S_{t_k}}{\partial m_{p_i}\partial m_{p_i}} &= V_{t_k}\frac{-2(y_{s_k}V_{f_k})^*}{m_{p_i}^3 e^{-j\tau_i}} \\
\frac{\partial^2 S_{f_k}}{\partial \tau_i\partial \tau_i} &= V_{f_k}\frac{(y_{s_k}V_{t_k})^*}{m_{p_i} e^{j\tau_i}} \\
\frac{\partial^2 S_{t_k}}{\partial \tau_i\partial \tau_i} &= V_{t_k}\frac{(y_{s_k}V_{f_k})^*}{m_{p_i} e^{-j\tau_i}} \\
\frac{\partial^2 S_{f_k}}{\partial m_{p_i}\partial \tau_i} &= V_{f_k}\frac{-j(y_{s_k}V_{t_k})^*}{m_{p_i}^2 e^{j\tau_i}} \\
\frac{\partial^2 S_{t_k}}{\partial m_{p_i}\partial \tau_i} &= V_{t_k}\frac{j(y_{s_k}V_{f_k})^*}{m_{p_i}^2 e^{-j\tau_i}} \\
\frac{\partial S_{f_k}}{\partial m_{p_i}\partial v_{f_k}} &= \frac{V_{f_k}}{v_{f_k}}\left(\frac{-4(y_{s_k}V_{f_k})^*}{m_{p_i}^3} + \frac{(y_{s_k}V_{t_k})^*}{m_{p_i}^2 e^{j\tau_i}}\right) \\
\frac{\partial S_{t_k}}{\partial m_{p_i}\partial v_{f_k}} &= \frac{V_{t_k}}{v_{f_k}}\frac{(y_{s_k}V_{f_k})^*}{m_{p_i}^2 e^{-j\tau_i}} \\
\frac{\partial S_{f_k}}{\partial m_{p_i}\partial v_{t_k}} &= \frac{V_{f_k}}{v_{f_t}}\left(\frac{-2(y_{s_k}V_{f_k})^*}{m_{p_i}^3} + \frac{(y_{s_k}V_{t_k})^*}{m_{p_i}^2 e^{j\tau_i}}\right) \\
\frac{\partial S_{t_k}}{\partial m_{p_i}\partial v_{t_k}} &= \frac{V_{t_k}}{v_{f_t}}\frac{(y_{s_k}V_{f_k})^*}{m_{p_i}^2 e^{-j\tau_i}} \\
\frac{\partial S_{f_k}}{\partial \tau_i\partial v_{f_k}} &= \frac{V_{f_k}}{v_{f_k}}\frac{j(y_{s_k}V_{t_k})^*}{m_{p_i} e^{j\tau_i}} \\
\frac{\partial S_{t_k}}{\partial \tau_i\partial v_{f_k}} &= \frac{V_{t_k}}{v_{f_k}}\frac{-j(y_{s_k}V_{f_k})^*}{m_{p_i} e^{-j\tau_i}} \\
\frac{\partial S_{f_k}}{\partial \tau_i\partial v_{f_k}} &= \frac{V_{f_k}}{v_{f_t}}\frac{j(y_{s_k}V_{t_k})^*}{m_{p_i} e^{j\tau_i}} \\
\frac{\partial S_{t_k}}{\partial \tau_i\partial v_{f_k}} &= \frac{V_{t_k}}{v_{f_t}}\frac{-j(y_{s_k}V_{f_k})^*}{m_{p_i} e^{-j\tau_i}} \\
\frac{\partial S_{f_k}}{\partial m_{p_i}\partial \theta_{f_k}} &= V_{f_k}\frac{j(y_{s_k}V_{t_k})^*}{m_{p_i}^2 e^{j\tau_i}} \\
\frac{\partial S_{t_k}}{\partial m_{p_i}\partial \theta_{f_k}} &= V_{t_k}\frac{-j(y_{s_k}V_{f_k})^*}{m_{p_i}^2 e^{-j\tau_i}} \\
\frac{\partial S_{f_k}}{\partial m_{p_i}\partial \theta_{t_k}} &= V_{f_k}\frac{-j(y_{s_k}V_{t_k})^*}{m_{p_i}^2 e^{j\tau_i}} \\
\frac{\partial S_{t_k}}{\partial m_{p_i}\partial \theta_{t_k}} &= V_{t_k}\frac{j(y_{s_k}V_{f_k})^*}{m_{p_i}^2 e^{-j\tau_i}} \\
\frac{\partial S_{f_k}}{\partial \tau_i\partial \theta_{f_k}} &= V_{f_k}\frac{-(y_{s_k}V_{t_k})^*}{m_{p_i} e^{j\tau_i}} \\
\frac{\partial S_{t_k}}{\partial \tau_i\partial \theta_{f_k}} &= V_{t_k}\frac{-(y_{s_k}V_{f_k})^*}{m_{p_i} e^{-j\tau_i}} \\
\frac{\partial S_{f_k}}{\partial \tau_i\partial \theta_{f_k}} &= V_{f_k}\frac{(y_{s_k}V_{t_k})^*}{m_{p_i} e^{j\tau_i}} \\
\frac{\partial S_{t_k}}{\partial \tau_i\partial \theta_{f_k}} &= V_{t_k}\frac{(y_{s_k}V_{f_k})^*}{m_{p_i} e^{-j\tau_i}}
\end{aligned}
\tag{40}
$$

We have to consider the crossed partial derivative $\frac{\partial^2}{\partial m_{p_i} \partial \tau_i}$ for those lines with transformer that controlls both variables. To compose the hessian matrix, we perform as the following example for all the crossed derivatives:

$$\begin{aligned}\frac{dS_{bus}}{dmdm_{ii}} &= \Re(\frac{\partial^2 S_{f_k}}{\partial m_{p_i} \partial m_{p_i}} \lambda_f + \frac{\partial^2 S_{t_k}}{\partial m_{p_i} \partial m_{p_i}} \lambda_t) \\ &+ \Im(\frac{\partial^2 S_{f_k}}{\partial m_{p_i} \partial m_{p_i}} \lambda_{f+N} + \frac{\partial^2 S_{t_k}}{\partial m_{p_i} \partial m_{p_i}} \lambda_{t+N})\end{aligned} \tag{41}$$

Since we have to take into account for the active and reactive power constraints, we have to divide the expression into real and imaginary and then use the corresponding multiplier. We compose the resulting hessian by concatenating the complete matrix as in the previous case.

## 5.4 Inequality constraints

We proceed similarly to the equality constraints. The vector of inequality constraints $H$ for the AC-OPF problem has the following structure:

$$\boldsymbol{H}(\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{H}^{\boldsymbol{S_f}} \\ \boldsymbol{H}^{\boldsymbol{S_t}} \\ \boldsymbol{H}^{\boldsymbol{V_u}} \\ \boldsymbol{H}^{\boldsymbol{P_u}} \\ \boldsymbol{H}^{\boldsymbol{Q_u}} \\ \boldsymbol{H}^{\boldsymbol{V_l}} \\ \boldsymbol{H}^{\boldsymbol{P_l}} \\ \boldsymbol{H}^{\boldsymbol{Q_l}} \\ \boldsymbol{H}^{\boldsymbol{sl}} \\ \boldsymbol{H}^{\boldsymbol{m_{p_u}}} \\ \boldsymbol{H}^{\boldsymbol{m_{p_l}}} \\ \boldsymbol{H}^{\boldsymbol{\tau_u}} \\ \boldsymbol{H}^{\boldsymbol{\tau_l}} \\ \boldsymbol{H}^{\boldsymbol{Q_{max}}} \\ \boldsymbol{H}^{\boldsymbol{Pf_{dc_u}}} \\ \boldsymbol{H}^{\boldsymbol{Pf_{dc_l}}} \end{bmatrix} \tag{42}$$

The first derivatives of the power flows are the most complex ones to get, since they will be derived from the $from$ and $to$ powers, and the constraints has its value squared.

$$H_f^S = [S_f^*]S_f - S_{max}^2 \tag{43}$$

$$\begin{aligned} H_X^f &= 2(\Re([S_f])\Re(S_X^f) + \Im([S_f])\Im(S_X^f)) \\ H_{XX}^f(\mu) &= 2\Re(S_{XX}^f([S_f^*]\mu) + {S_X^f}^T[\mu]S_X^f) \end{aligned} \tag{44}$$

And similarly for the $H_t$ The derivatives with respect of the PF variables are obtained from the Matpower documentation [**zimmerman2011matpower**] again:

$$\begin{aligned} S_\theta^f &= \left[I_f^*\right]\frac{\partial V_f}{\partial\theta} + \left[V_f\right]\frac{\partial I_f^*}{\partial\theta} \\ &= \left[I_f^*\right]jC_f[V] + [C_fV](jY_f[V])^* \\ &= j\left([I_f^*]C_f[V] - [C_fV]Y_f^*[V^*]\right) \\ S_\mathcal{V}^f &= \left[I_f^*\right]\frac{\partial V_f}{\partial\mathcal{V}} + \left[V_f\right]\frac{\partial I_f^*}{\partial\mathcal{V}} \\ &= \left[I_f^*\right]C_f[E] + [C_fV]Y_f^*[E^*] \\ S_{P_g}^f &= 0 \\ S_{Q_g}^f &= 0 \end{aligned} \tag{45}$$

$$\begin{aligned} S_{\theta\theta}^f(\mu) &= \frac{\partial}{\partial\theta}\left(S_\theta^f\right)^T\mu \\ &= [V^*]Y_f^{*\top}[\mu]C_f[V] + [V]C_f^\top[\mu]Y_f^*[V^*] \\ &- [Y_f^{*\top}[\mu]C_fV][V^*] - [C_f^T[\mu]Y_f^*V^*][V] \\ S_{\mathcal{V}\theta}^f(\mu) &= \frac{\partial}{\partial\theta}\left(S_\mathcal{V}^f\right)^\top\mu \\ &= j[\mathcal{V}]^{-1}([\mathbf{V}^*]Y_f^{*\top}[\mu]C_f[\mathbf{V}] - [\mathbf{V}]C_f^\top[\mu]Y_f^*[\mathbf{V}^*] \\ &- [Y_f^{*\top}[\mu]C_f\mathbf{V}][\mathbf{V}^*] + [C_f^\top[\mu]Y_f^*\mathbf{V}^*][\mathbf{V}]) \\ S_{\mathcal{V}\mathcal{V}}^f(\mu) &= \frac{\partial}{\partial\mathcal{V}}\left(S_\mathcal{V}^f\right)^\top\mu \\ &= [\mathcal{V}]^{-1}\left([\mathbf{V}^*]Y_f^{*\top}[\mu]C_f[\mathbf{V}] + [\mathbf{V}]C_f^\top[\mu]Y_f^*[\mathbf{V}^*]\right)[\mathcal{V}]^{-1} \end{aligned} \tag{46}$$

The derivatives with respect to the tap variables have been derived using the expressions obtained in the derivation of the power balance derivatives and the general expression of the H, the expressions for $S_f$, $S_t$ and its derivatives is shown in the section above

# 6 Results

## 6.1 Optimal gap

## 6.2 Runtime

## 6.3 Solution recovery

# Acknowledgments

# A Environmental, Social, and Gender Impact

# B Time Planning

# C Budget

## C.1 Equipment

The costs of machinery and digital tools required in the project development appear in Table **??**.

Table 1: Equipment Costs.

| Concept | Unit cost (€) | Quantity | Total (€) |
|---|---|---|---|
| Personal computer | 1000.00 | 1 | 1000.00 |
| Matlab individual annual license | 2000.00 | 1 | 2000.00 |
| **Total** | | | 3000.00 |

## C.2 Human resources

The working hours spent on the thesis and related work are captured in Table **??**. It also includes the cost linked to the supervision process.

Table 2: Human Resources Costs.

| Concept | Unit cost (€/h) | Quantity (h) | Total (€) |
|---|---|---|---|
| Research | 25.00 | 100 | 2500.00 |
| Code development | 25.00 | 400 | 10000.00 |
| Testing | 25.00 | 150 | 3750.00 |
| Writing | 25.00 | 100 | 2500.00 |
| Supervision | 30.00 | 50 | 1500.00 |
| **Total** | | | 20250.00 |

## C.3 Total budget

The total budget formed by aggregating equipment and human resources is shown in Table **??**. No Value Added Tax (VAT) is considered in the budget.

Table 3: Total Budget of the Thesis.

| Concept | Total (€) |
|---|---|
| Equipment | 3000.00 |
| Human resources | 20250.00 |
| **Total** | 23250.00 |

2nd of July, 2023

Titouan Delorme

# Bibliography


[1] D. K. Molzahn, I. A. Hiskens, *et al.*, “A survey of relaxations and approximations of the power flow equations,” *Foundations and Trends® in Electric Energy Systems*, vol. 4, no. 1-2, pp. 1–221, 2019.

[2] R. A. Jabr, “Radial distribution load flow using conic programming,” *IEEE transactions on power systems*, vol. 21, no. 3, pp. 1458–1459, 2006.

[3] C. Coffrin, H. L. Hijazi, and P. Van Hentenryck, “The qc relaxation: A theoretical and computational study on optimal power flow,” *IEEE Transactions on Power Systems*, vol. 31, no. 4, pp. 3008–3018, 2015.