Master Thesis

Master's Degree in Electric Power Systems and Drives

Integration of an AC-OPF solver in GridCal

April 2024

Author: Carlos Alegre Aldeano

Supervisors: Marc Cheah Mañé

Josep Fanals i Batllori

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Abstract

Finding an optimal point of operation that reduces the operational cost of a power system while it complies with all the technical limitations is a complex problem that every grid operator has to tackle. To solve this optimization problem, many strategies have been developed, such as the DC-OPF (Direct Current Optimal Power Flow), which performs some simplifications over the power flow equations and constraints to linearize and solve easily the problem, or the AC-OPF (Alternating Current Optimal Power Flow), which is the most exact method, but also more computationally expensive. It encapsulates the economic dispatch, by the means of a cost objective function, and all the technical constraints that the grid has to respect.

The work developed in this thesis presents an AC-OPF solver integrated in GridCal, an open-source power system analysis tool developed by the Spanish TSO, Redeia. This work has been encapsulated under the SIROCO project, an initiative focused on adding and improving functionalities to make GridCal the go-to software when it comes to planning the Spanish network. The solver has been developed in Python, based on the Matpower formulation, and adds important features such as transformer setpoints optimization, DC links, reactive power limitations and some improvements with respect to Matpower's original solver convergence.

Resumen

Encontrar un punto óptimo de funcionamiento que reduzca el coste de operación de un sistema eléctrico y, al mismo tiempo, respete todas las limitaciones técnicas es un problema complejo que todo operador de red debe afrontar. Para resolver este problema de optimización, se han desarrollado muchas estrategias, como el DC-OPF (Direct Current Optimal Power Flow), que realiza algunas simplificaciones sobre las ecuaciones y restricciones del flujo de potencia para linealizar y resolver fácilmente el problema, o el AC-OPF (Alternating Current Optimal Power Flow), que es el método más exacto, pero también más costoso computacionalmente. Tiene en cuenta el despacho económico, mediante una función objetivo de tipo económico, y a su vez tiene en cuenta las restricciones técnicas que debe respetar la red.

El trabajo desarrollado en esta tesis presenta un solucionador de problemas de optimización, adaptado para el AC-OPF e integrado en GridCal, una herramienta de análisis de sistemas eléctricos de código abierto desarrollada por el operador de red español, Redeia. Este trabajo forma parte de un paquete de trabajo dentro del proyecto SIROCO, una iniciativa centrada en añadir y mejorar funcionalidades para hacer de GridCal el software de referencia a la hora de planificar la red española. El solucionador se ha desarrollado en Python, basado en la formulación de Matpower, y añade funcionalidades importantes como la optimización de consignas de transformadores, enlaces de corriente continua, limitaciones de potencia reactiva y algunas mejoras con respecto a la convergencia del solucionador original de Matpower.

Resum

Trobar un punt òptim de funcionament que redueixi el cost d'operació d'un sistema elèctric i a més respecti totes les limitacions tècniques és un problema complex que tot operador de xarxa ha d'afrontar. Per resoldre aquest problema d'optimització, s'han desenvolupat moltes estratègies, com el DC-OPF (Direct Current Optimal Power Flow), que fa algunes simplificacions sobre les equacions i restriccions del flux de potència per linealitzar i resoldre fàcilment el problema, o l'AC-OPF (Alternating Current Optimal Power Flow), que és el mètode més exacte, però també més costós computacionalment. Té en compte el despatx econòmic, mitjançant una funció objectiu de tipus econòmic, i alhora té en compte les restriccions tècniques que ha de respectar la xarxa.

El treball desenvolupat en aquesta tesi presenta un solucionador de problemes d'optimització, adaptat per a l'AC-OPF i integrat a GridCal, una eina d'anàlisi de sistemes elèctrics de codi obert desenvolupada per l'operador de xarxa espanyol, Redeia. Aquest treball forma part d'un paquet de treball dins del projecte SIROCO, una iniciativa centrada a afegir i millorar funcionalitats per fer de GridCal el programari de referència alhora de planificar la xarxa espanyola. El solucionador s'ha desenvolupat a Python, basat en la formulació de Matpower, i afegeix funcionalitats importants com l'optimització de consignes de transformadors, enllaços de corrent continu, limitacions de potència reactiva i algunes millores pel que fa a la convergència del solucionador original de Matpower.

Contents

List of Figures

List of Tables

Glossary

Symbols

α	Scaling factor
Δ	Displacement of a variable
α_d	Step size for multipliers
α_p	Step size for variables
$\boldsymbol{\lambda}$	Equality Lagrange multipliers vector
$\boldsymbol{\mu}$	Inequality Lagrange multipliers vector
γ	Barrier parameter
$\boldsymbol{\theta}$	Voltage angle vector
$\boldsymbol{\tau}$	Tap angles vector
$\mathcal{L}$	Lagrangian function
$\mathcal{L}_x$	Gradient of the Lagrangian
$\mathcal{L}_{xx}$	Hessian of the Lagrangian
$\boldsymbol{\mathcal{V}}$	Voltage modules vector
c_{i_0}	Base cost coefficient for generator i
c_{i_1}	Linear cost coefficient for generator i
c_{i_2}	Quadratic cost coefficient for generator i
$c_{sl_{v_i}}$	Cost for voltage slack variables for bus i
$c_{sl_{s_k}}$	Cost for branch power slack variables for branch k
$\boldsymbol{dc}$	DC branches vector
f	Objective function
f_x	Gradient of the function
f_{xx}	Hessian of the function
$\boldsymbol{fdc}$	DC $from$ buses vector
k	Index of the k-th line of the set
k_m	Vector of indexes of branches with tap modules
k_τ	Vector of indexes of branches with tap angles
i	Index of the i-th bus of the set
$\boldsymbol{il}$	Vector of dispatchable buses
l	Number of branches
m	Number of monitored branches
$\boldsymbol{m_p}$	Tap modules vector
n	Number of buses
n_{DC}	Number of DC links
n_{eq}	Number of equality constraints
n_g	Number of generators
n_{in}	Number of inequality constraints

n_{m_p}	Number of tap modules
n_τ	Number of tap angles
n_{pq}	Number of PQ buses
n_{pv}	Number of PV buses
n_{sl}	Number of slack variables
$\boldsymbol{pq}$	Vector of PQ buses
$\boldsymbol{pv}$	Vector of PV buses
$\boldsymbol{tdc}$	DC to buses vector
y_s	Series admittance bus admittance
$\boldsymbol{sl_{sf}}$	Slack variable for $from$ branch power constraints vector
$\boldsymbol{sl_{st}}$	Slack variable for to branch power constraints vector
$\boldsymbol{sl_{v_{max}}}$	Slack variable for maximum voltage constraints vector
$\boldsymbol{sl_{v_{min}}}$	Slack variable for minimum voltage constraints vector
$\boldsymbol{x}$	Optimization variables vector
C_f	$From$ connectivity matrix
C_g	Generator connectivity matrix
C_t	To connectivity matrix
$\boldsymbol{G}$	Equality constraints vector
$\boldsymbol{G_x}$	Gradient of the equality constraints
G_{xx}	Hessian of the equality constraints
$\boldsymbol{H}$	Inequality constraints vector
$\boldsymbol{H_x}$	Gradient of the inequality constraints
H_{xx}	Hessian of the inequality constraints
J	Jacobian matrix
$\boldsymbol{L}$	Vector of branches
$\boldsymbol{M}$	Vector of monitored branches
P_g	Active power generation vector
Q_g	Reactive power generation vector
R	Resistance
$\boldsymbol{S}$	Complex injection power vector
$\boldsymbol{V}$	Voltage vector
X	Reactance
Y_{bus}	Bus Admittance matrix
Y_f	$From$ branch admittance matrix
Y_t	To branch admittance matrix
$\boldsymbol{Z}$	Inequality slack variables vector

Acronyms

AC	Alternating Current
AC-OPF	Alternating Current Optimal Power Flow
DC	Direct Current
DC-OPF	Direct Current Optimal Power Flow
DSO	Distribution System Operator
FUBM	Flexible Universal Branch Model
GUI	Graphical User Interface
IPM	Interior Point Method
OPF	Optimal Power Flow
TSO	Transmission System Operator

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 DC-OPF considers line flows but only solves for the voltage angles; whereas the AC-OPF 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 [1].

Regarding a more technical approach, one would consider the AC-OPF to be the best choice without losing the ability to perform economic analysis. The straightforward tool to solve the basic AC-OPF problem is Matpower [2], an open source package for Matlab that allows the user to perform simulations with specific grid models. It is regarded as the state-of-the-art of the AC-OPF formulation for many works in the topic. The solution is found using an Interior Point Method, which is an algorithm that can be used to solve non-linear optimization problems with the proper considerations as explained in [3]. Due to the non-convex nature of the problem, the iterations of the algorithm will only find local optimal points, which means the solution is sensible to the initial point chosen and the constraints of the problem [4].

There have been many development lines that try to find a global optimal point using relaxations over the constraints of the problem so that the resulting transformed problem is convex. Once it is transformed, it is possible to find the global optimal point using an IPM. These relaxation methods, unlike the DC approximations, expand the feasibility domain of solutions instead of modifying it as it can be seen in Figure 1.1. The resulting point can be outside the AC problem feasibility domain, but it considers all the possible states unlike a DC aproximation.

1.2 Objectives

The aim of this project, which is part of the Redeia (Spanish TSO) SIROCO project, is the implementation of an AC-OPF solver in GridCal [6], a Python grid analysis package in development by Redeia's engineer Santiago Peñate Vera. This solver will use the traditional formulation, without any simplification, and it is intended to be useful in the decision-making during the planning of the Spanish electrical grid.

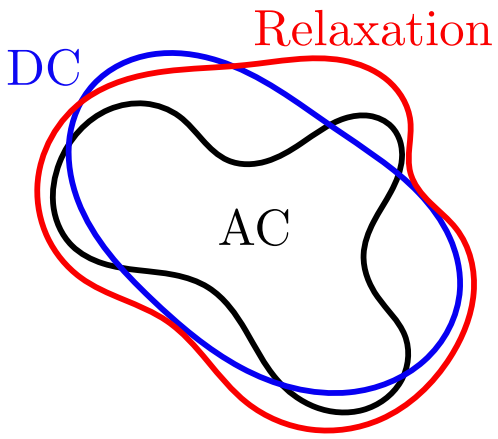

Figure 1.1: Feasibility domain of the AC-OPF problem for different approaches [5].

In Figure 1.2 we can see the plot in GridCal's Graphical User Interface (GUI) of the Spanish grid.

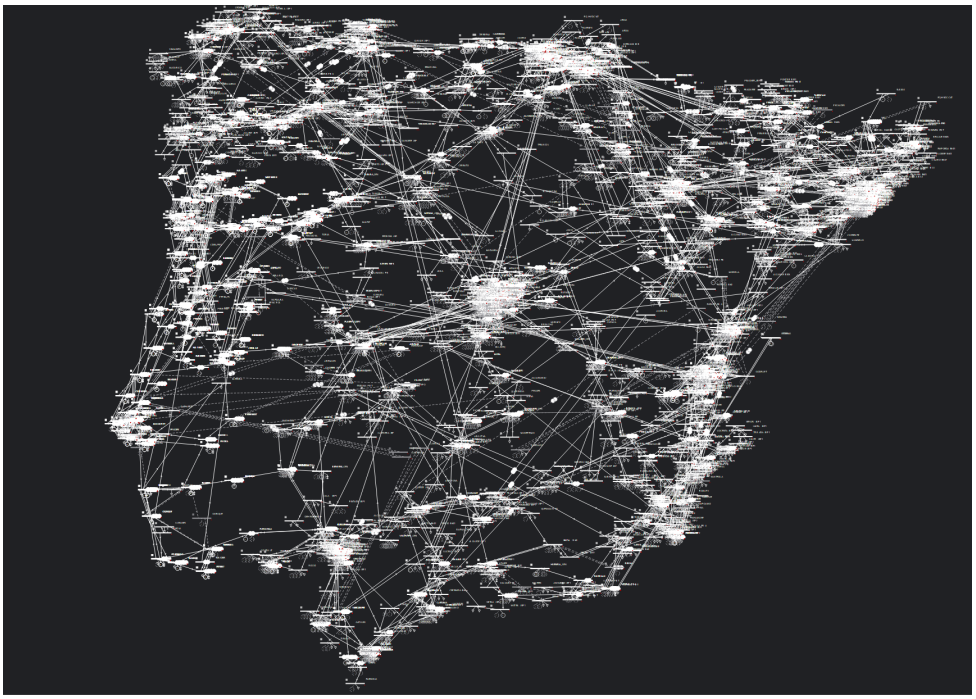

Figure 1.2: Plot of the topology of the Spanish grid in GridCal interface.

The end product needs to be able to solve the AC-OPF problem for similar grids and be able to perform different studies depending on the user's needs. For grids of this size, it is obvious that one should use an appropriate tool to visualize the results with detail, which is the purpose of the whole GridCal package, including its GUI.

The individual objectives of the project are the following:

- Establish the mathematical formulation of the AC-OPF problem from Matpower's work.
- Develop the whole set of equations (economical objective function and technical constraints), including its gradients and Hessians, to build a scalable IPM.
- Implement the solver in GridCal's source code.
- Check performance of the base solver to ensure the results match those from Matpower.
- Add the new features to the solver, such as transformer setpoints optimization, DC links, reactive power limitations.
- Include convergence improvements to the solver, such as bound slack variables and power flow initialization.
- Validate the performance of the solver, proving the scalability on national grids.
- Ensure that future developments can be easily integrated into the solver.

1.3 Outline

This work considers all the steps involved in the development process of this solver, from the mathematical modelling to the software implementation and validation of results.

- Chapter 2 will describe all the elements of an electrical grid, the parameters that define each of them and the notation used throughout the thesis.
- Chapter 3 will describe the relationships between elements and the physics that determine the operation of an electrical grid.
- Chapter 4 will contain the mathematical modelling of the optimization problem that has to be solved to determine optimal operation.
- Chapter 5 will contain all the software structure and algorithms developments, accompanied by some explanations of the GridCal environment.
- Chapter 6 will contain benchmarking cases 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 buses (or nodes if we use the topological convention) and branches that link buses forming a grid. Connected to some buses there can be generators and loads, while devices linking two buses can be lines or 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

Buses, or 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 referred to as load, comes from the sum of all the users of electrical devices connected to the bus.

When dealing with a transmission grid, buses can be seen as the substations that feed an entire neighborhood, 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 $N = \{1, ..., i, ..., n\}$, where the index i will usually be used as an index for vectors such as the voltages or powers, indicating the i-bus value using a subindex. Each bus has a state represented by 2 complex variables, the complex voltage $\mathcal{V}_i \angle \theta_i$ and the complex 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 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. This is typically referred to as the Kirchhoff's current law.

There are several types of bus depending on which variables can be controlled. The most com-

mon 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 an arbitrary value, typically $V_i \angle \theta_i = 1 \angle 0°$ using the per unit. This type of bus can be regarded as the balancing bus, as it will provide the power necessary to solve any power imbalance when the power flow is solved. Generally, it is the bus with best load change response, capable of absorbing or providing power were there are sudden imbalances.

  Since there can be more than one slack bus in a grid due to being different electrical islands operating decoupled from one another, 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.

  There could also be a distributed slack, where there are more than one bus that will respond to power imbalance, although it is not considered in this work.

- PQ buses have known power exchange values. The P_i and Q_i values are known 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.

- 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. The buses with equal upper and lower voltage limits will be included in this list, and will have that value as the voltage setpoint.

Table 2.1: Known variables for the different types of buses in Power Flow calculations.

Bus	$\mathcal{V}$	θ	P_g	Q_g
Slack	Yes	Yes	No	No
PQ	No	No	Yes	Yes
PV	Yes	No	Yes	No

The goal of the optimization will be to obtain the optimal setpoints of these controlled buses, as

it will be shown in the optimization formulation.

2.1.1 Shunt devices

The shunt devices are one-port elements connected to the buses. In the model, the two elements considered are loads and generators, which can be depicted as in Figure 2.1.

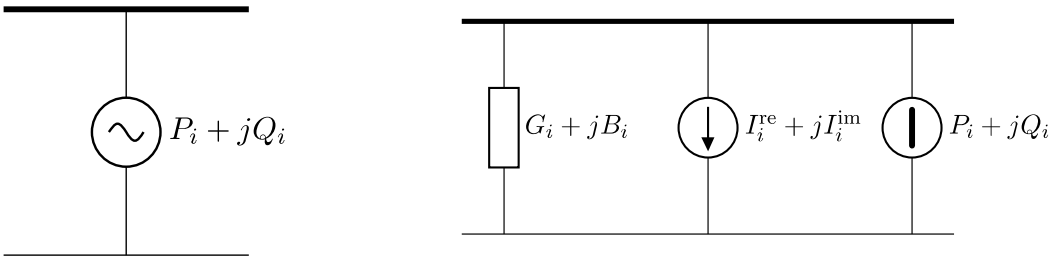

Figure 2.1: Schematic of the generator and load elements.

The model used for the loads is the ZIP model, meaning it is considered as an impedance, intensity and power load. In this project, values obtained from the grid model are the complex power values, and the impedance elements are directly computed in the admittance matrix. The load modelled as a current is not considered in the model.

2.2 Branches

Branches connect two buses and allow the transportation of power between them. The power transfer can happen in both directions, although obviously only one at a time. The line has two ends, the 'from' end f and the 'to' end t. The branches 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 branches is denoted as $L = \{1, ..., k, ..., l\}$, while the set of 'from' and 'to' buses is denoted as $F = \{f_1, ..., f_l\}$ and $T = \{t_1, ..., t_l\}$ respectively. A line is defined used the notation $k \triangleq (f_k, t_k)$. This line will be modelled as a π-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 branches have a limit over the power they can withstand. This limit, called $rate$, has to be checked at both ends of the line, since it is not equal due to active and reactive power losses. Monitored branches will have a constraint that checks that this value is not surpassed, while the rest of the branches may be disregarded as a user choice. The list is obtained from the GridCal model of the grid, and will be stored in the il list of indexes containing m monitored branch

elements.

The general model used is the Flexible Universal Branch Model, represented in Figure 2.2.

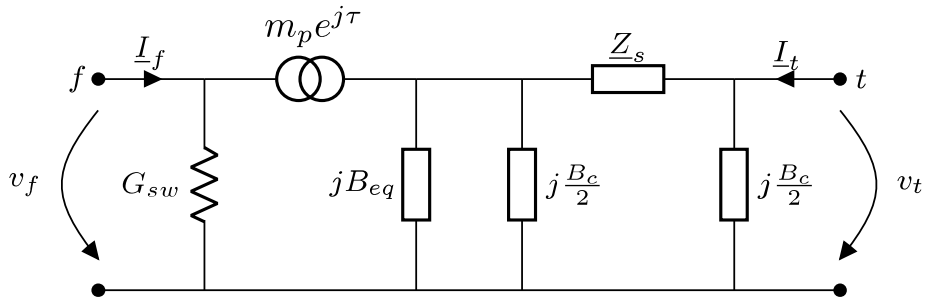

Figure 2.2: Representation of the Flexible Universal Branch Model.

There are two different branch devices: lines and transformers. For lines, the model is the π-equivalent line, disregarding the left part of the model. Transformers are modelled as the full element as seen in Figure 2.2. The relationship between the power flow variables for a branch can be described with the following equation:

$$\begin{bmatrix} i_f \\ i_t \end{bmatrix} = \begin{bmatrix} y_{ff} & y_{ft} \\ y_{tf} & y_{tt} \end{bmatrix} \begin{bmatrix} v_f \\ v_t \end{bmatrix} = Y_{br} \begin{bmatrix} v_f \\ v_t \end{bmatrix} \tag{2.1}$$

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 as follows:

$$Y_{br} = \begin{bmatrix} (y_s + j\frac{b_c}{2})\frac{1}{m_p^2} & -\frac{y_s}{m_p e^{-j\tau}} \\ -\frac{y_s}{m_p e^{j\tau}} & y_s + j\frac{b_c}{2} \end{bmatrix} \tag{2.2}$$

where y_s is the series admittance calculated as $y_s = \frac{1}{R+jX}$, with R and X being the resistance and reactance of the line, b_c is the capacitance of the line, m_p is the transformer ratio, and τ is the phase shift of the transformer. G_{sw} and B_{eq} are included in the shunt element matrix.

2.2.1 Transformers

The transformers change the level of the voltage between both of its ends. The voltage when going from generation towards the transmission grid raises to lower losses, 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 voltage 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 τ, 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 $\boldsymbol{k_m}$ for module controlled transformers and $\boldsymbol{k_\tau}$ for phase controlled transformers. Those who have controls for both variables will be in both lists.

2.2.2 HVDC links

Lossless HVDC lines have been considered in the model. These lines are modelled as a branch element that can transfer active power between two buses without any associated impedance or voltage transformation, meaning the FUBM model considerations are not applied. The only condition applied to these type of lines is that the active power at one end is the opposite of the active power at the other end.

The model presented is really simple, but for the purposes of this work, this is considered enough to have a first step towards the inclusion of hybrid AC/DC models in the future.

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, tap variables, generation and demand, which are the variables of the optimization problem.

The notation used throughout this work is as follows: vectors are denoted in bold (can be both upper or lowercase), matrices are denoted in uppercase, and scalars are denoted in lowercase. Vectors that are transformed in diagonal matrices appear within square brackets. Vectors that slice other vectors using their values as indexes appear after the slice vector within curly brackets.

3.1 Power Flow Equations

The full construction of the power flow equations is shown, since some intermediate steps can be useful for later purposes. The first important equation is the power transfer that occurs between two nodes connected with branch, modelled using the Flexible Universal Branch Model for the k branch as in Equations (2.1) and (2.2). They contain the information of a single line. In the grid model used, y_{ff}, y_{ft}, y_{tf} and y_{tt} are vectors of length l 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 k x l that relate the index of the 'from' and 'to' buses to the index of their branch as follows:

$$C_{f_{ki}} = \begin{cases} 1, & \text{for } k == (i,j) \quad \forall j \in \boldsymbol{N} \\ 0, & \text{otherwise} \end{cases} \tag{3.1}$$

$$C_{t_{kj}} = \begin{cases} 1, & \text{for } k == (i,j) \quad \forall i \in \boldsymbol{N} \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

Using these connectivity matrices 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.

$$\begin{aligned} Y_f &= [\boldsymbol{Y_{ff}}]\, C_f + [\boldsymbol{Y_{ft}}]\, C_t \\ Y_t &= [\boldsymbol{Y_{tf}}]\, C_f + [\boldsymbol{Y_{tt}}]\, C_t \\ Y_{bus} &= C_f^{\top} Y_f + C_t^{\top} \boldsymbol{Y_t} + [\boldsymbol{Y_{sh}}] \end{aligned} \tag{3.3}$$

where the brackets indicate that the vector is transformed into a diagonal matrix and the $[\boldsymbol{Y_{sh}}]$ matrix contains the shunt admittance of each bus, which is calculated for each bus by GridCal modeller. 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 $\boldsymbol{S_{bus}}$ is given by the following equations:

$$\boldsymbol{S_{bus}} = [\boldsymbol{V}](Y_{bus}\boldsymbol{V})^* \tag{3.4}$$

where $\boldsymbol{V}$ is the vector of the complex voltages of the buses calculated as:

$$\boldsymbol{\mathcal{V}} e^{j\boldsymbol{\theta}} \tag{3.5}$$

using element to element multiplication, and being $\mathcal{V}$ the vector of voltage modules and θ the vector of voltage angles. The complete balance can be now calculated as:

$$\boldsymbol{G^S_{bus}} = \boldsymbol{S_{bus}} - \boldsymbol{S^g_{bus}} + \boldsymbol{S^d_{bus}} = \boldsymbol{0} \quad \text{(equilibrium)} \tag{3.6}$$

where $\boldsymbol{S^g_{bus}}$ is the power generated at the buses, and $\boldsymbol{S^d_{bus}}$ the power consumed at the buses. The notation $\boldsymbol{G^S}$ 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 or loads connected to the same bus. For the present work, there is no segregation of loads and the bus demand is obtained directly from GridCal, but it could be consider to model load flexibility and shedding. For generations, the separation has to be done as 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:

$$\boldsymbol{S^g_{bus}} = C_G \boldsymbol{S^g} \tag{3.7}$$

For buses that are part of a DC link, the power balance has to account for the power transferred through the link.

$$\begin{aligned} \boldsymbol{G^S_{bus}}\{\boldsymbol{fdc}\} &+= \boldsymbol{P_{dc}}\{\boldsymbol{dc}\} \\ \boldsymbol{G^S_{bus}}\{\boldsymbol{tdc}\} &-= \boldsymbol{P_{dc}}\{\boldsymbol{dc}\} \end{aligned} \tag{3.8}$$

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.

$$\begin{aligned} \boldsymbol{V_f} &= C_f \boldsymbol{V} \\ \boldsymbol{V_t} &= C_t \boldsymbol{V} \\ \boldsymbol{S_f} &= [\boldsymbol{V_f}](Y_f \boldsymbol{V})^* \\ \boldsymbol{S_t} &= [\boldsymbol{V_t}](Y_t \boldsymbol{V})^* \end{aligned} \tag{3.9}$$

Here it is important to note that the vector $\boldsymbol{V_f}$ will contain the voltage value of the $from$ bus of each line, which means there can be some buses appearing more than one time.

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 $\boldsymbol{H^I}$ is used to identify to each different limit.

3.2.1 Nodal voltage limits

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

$$\begin{aligned} \boldsymbol{H^{v_u}} &= \boldsymbol{\mathcal{V}} - \boldsymbol{V_{max}} \leq \boldsymbol{0} \quad \text{(upper limit)} \\ \boldsymbol{H^{v_l}} &= \boldsymbol{V_{min}} - \boldsymbol{\mathcal{V}} \leq \boldsymbol{0} \quad \text{(lower limit)} \end{aligned} \tag{3.10}$$

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:

$$\begin{aligned} \boldsymbol{H^{P_u}} &= \boldsymbol{P_g} - \boldsymbol{P_{max}} \leq \mathbf{0} \quad \text{(upper limit)} \\ \boldsymbol{H^{P_l}} &= \boldsymbol{P_{min}} - \boldsymbol{P_g} \leq \mathbf{0} \quad \text{(lower limit)} \\ \boldsymbol{H^{Q_u}} &= \boldsymbol{Q_g} - \boldsymbol{Q_{max}} \leq \mathbf{0} \quad \text{(upper limit)} \\ \boldsymbol{H^{Q_l}} &= \boldsymbol{Q_{min}} - \boldsymbol{Q_g} \leq \mathbf{0} \quad \text{(lower limit)} \end{aligned} \tag{3.11}$$

An additional constraint over the generation is the reactive power limitations due to the generator capability curves. These curves can be modelled by piecewise defined function, but in the formulation of the problem they have been considered much simpler, and by default they are not applied. The following constraint models the reactive power limits of the generators:

$$\boldsymbol{H^{ctQ}} = \boldsymbol{Q_g^2} - \boldsymbol{tanmax^2 P_g^2} \leq \mathbf{0} \tag{3.12}$$

where $\boldsymbol{tanmax}$ is the maximum value of the tangent of the angle between the active and reactive power of the generator.

3.2.3 Transformer limits

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

$$\begin{aligned} \boldsymbol{H^{m_{p_u}}} &= \boldsymbol{m_p} - \boldsymbol{m_{p_{max}}} \leq \mathbf{0} \quad \text{(upper limit)} \\ \boldsymbol{H^{m_{p_l}}} &= \boldsymbol{m_{p_{min}}} - \boldsymbol{m_p} \leq \mathbf{0} \quad \text{(lower limit)} \\ \boldsymbol{H^{\tau_u}} &= \boldsymbol{\tau} - \boldsymbol{\tau_{max}} \leq \mathbf{0} \quad \text{(upper limit)} \\ \boldsymbol{H^{\tau_l}} &= \boldsymbol{\tau_{min}} - \boldsymbol{\tau} \leq \mathbf{0} \quad \text{(lower limit)} \end{aligned} \tag{3.13}$$

3.2.4 Line limits

The line limits will be checked only for the monitored branches, which is a subset of size m from the total number of branches l, 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.

$$\begin{aligned} \boldsymbol{H^{S_f}} &= \boldsymbol{S_f S_f^*} - \boldsymbol{S_{max}^2} \leq \mathbf{0} \\ \boldsymbol{H^{S_t}} &= \boldsymbol{S_t S_t^*} - \boldsymbol{S_{max}^2} \leq \mathbf{0} \end{aligned} \tag{3.14}$$

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 possible, 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 behavior.

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

$$\begin{aligned} \boldsymbol{H^{v_u}} &= \boldsymbol{\mathcal{V}} - \boldsymbol{V_{max}} - \boldsymbol{sl_{v_{max}}} \leq \mathbf{0} \quad \text{(upper limit)} \\ \boldsymbol{H^{v_l}} &= \boldsymbol{V_{min}} - \boldsymbol{\mathcal{V}} - \boldsymbol{sl_{v_{min}}} \leq \mathbf{0} \quad \text{(lower limit)} \\ \boldsymbol{H^{S_f}} &= \boldsymbol{S_f S_f^*} - \boldsymbol{S_{max}^2} - \boldsymbol{sl_{sf}} \leq \mathbf{0} \quad \text{(from limit)} \\ \boldsymbol{H^{S_t}} &= \boldsymbol{S_t S_t^*} - \boldsymbol{S_{max}^2} - \boldsymbol{sl_{st}} \leq \mathbf{0} \quad \text{(to limit)} \\ -\boldsymbol{H^{sl_{v_{max}}}} &\leq \mathbf{0} \\ -\boldsymbol{H^{sl_{v_{min}}}} &\leq \mathbf{0} \\ -\boldsymbol{H^{sl_{sf}}} &\leq \mathbf{0} \\ -\boldsymbol{H^{sl_{st}}} &\leq \mathbf{0} \end{aligned} \tag{3.15}$$

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 established 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 Karush-Kuhn-Tucker (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 pseudocode used to solve the problem. The general theory of optimization can be found in the book of Boyd and Vandenberghe [7], with a more detailed explanation of the IPM in Chapter 19 of Nocedal and Wright's book [3], and the adaptation of this theory to a more compact formulation can be reviewed in the Matpower Interior Point Solver documentation [8] [9].

4.1 Optimization problem

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

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & \boldsymbol{G}(\boldsymbol{x}) = \boldsymbol{0} \\ & \boldsymbol{H}(\boldsymbol{x}) \leq \boldsymbol{0} \end{aligned} \tag{4.1}$$

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 the set of n_e equality constraints of the problem, and $\boldsymbol{H}(\boldsymbol{x})$ is the set of n_i inequalities.

The problem will be solved using the barrier parameter method, which consists of solving the optimization problems sequencially with different values of the barrier parameter γ. The barrier parameter is an IPM methodology used to penalize the violation of the inequality constraints in the problem, which are turned into equalities of the form $\boldsymbol{H}(\boldsymbol{x}) + \boldsymbol{Z} = \boldsymbol{0}$ by the addition of the positive slack variables $\boldsymbol{Z}$.

The Lagrangian of the problem is defined by:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^\top \boldsymbol{G}(\boldsymbol{x}) + \boldsymbol{\mu}^\top (\boldsymbol{H}(\boldsymbol{x}) + \boldsymbol{Z}) - \gamma \sum_{i=1}^{n_i} \log(z_i) \tag{4.2}$$

where $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are the Lagrange multipliers associated with the equality and inequality constraints respectively, the superindex $\top$ is used to indicate the transposition of a vector and the elements of the sum z_i correspond to each value of the vector $\boldsymbol{Z}$.

The KKT conditions are a set of equations that must be satisfied by the solution of the opti-

mization problem. These conditions are a result of imposing that the partial derivatives of the Lagrangian in the (x, Z, λ, μ) variable space equals 0, as shown in Equation (4.4).

$$\nabla_{x,Z,\lambda,\mu}\mathcal{L}(\boldsymbol{x}, \boldsymbol{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \mathbf{0} \tag{4.3}$$

$$\begin{aligned}
\boldsymbol{\mathcal{L}_x}(\boldsymbol{x}, \boldsymbol{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= f_x + \boldsymbol{\lambda}^\top G_x + \boldsymbol{\mu}^\top H_x = \mathbf{0} \\
\boldsymbol{\mathcal{L}_Z}(\boldsymbol{x}, \boldsymbol{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= \boldsymbol{\mu}^\top - \gamma \mathbf{1}_{n_i}^\top [\boldsymbol{Z}]^{-1} = \mathbf{0} \\
\boldsymbol{\mathcal{L}_\lambda}(\boldsymbol{x}, \boldsymbol{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= \boldsymbol{G}^\top(\boldsymbol{x}) = \mathbf{0} \\
\boldsymbol{\mathcal{L}_\mu}(\boldsymbol{x}, \boldsymbol{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= \boldsymbol{H}^\top(\boldsymbol{x}) + \boldsymbol{Z}^\top = \mathbf{0}
\end{aligned} \tag{4.4}$$

where $\mathbf{1}_{n_i}$ is a vector of ones of length n_i, and γ is the barrier parameter that will be updated in each iteration of the solver until reaching a value close to zero. The notation $f_{\boldsymbol{X}}$ indicates the gradient of the given function f with respect to the vector $\boldsymbol{X}$.

The first gradient corresponds to the stationary 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 due 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. Since this algorithm will be used by the Spanish TSO for country-sized grids, the scalability of the solver is a key factor to consider. In reference [10], this algorithm is compared with the Gauss-Seidel algorithm, yielding better scalability for the Newton-Raphson method.

Some optimization solver packages such as IPOPT were considered, but creating a tailored solver and integrating it in GridCal was considered the best option, as it avoids depending on third-party software, while also allowing the customization of its behavior.

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 in Algorithm 1.

Algorithm 1 Newton-Raphson Iterative Process

1: Initialize $\boldsymbol{x}_0$, tolerance ϵ, 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 control 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 (4.4) are solved. The expanded formulation of the system solved at each step is:

$$-\begin{pmatrix} \boldsymbol{\mathcal{L}_{xx}} & \mathbf{0} & \boldsymbol{G_x^\top(x)} & \boldsymbol{H_x^\top(x)} \\ \mathbf{0} & [\boldsymbol{\mu}] & \mathbf{0} & [\boldsymbol{Z}] \\ \boldsymbol{G_x(x)} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{H_x(x)} & \boldsymbol{I} & \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\Delta x} \\ \boldsymbol{\Delta Z} \\ \boldsymbol{\Delta\lambda} \\ \boldsymbol{\Delta\mu} \end{pmatrix} = \begin{pmatrix} \mathcal{L}_x^\top \\ [\boldsymbol{\mu}]\boldsymbol{Z} - \gamma\mathbf{1}_{n_i} \\ \boldsymbol{G(x)} \\ \boldsymbol{H(x)} + \boldsymbol{Z} \end{pmatrix} \tag{4.5}$$

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 $\boldsymbol{\mathcal{L}_{xx}}$ is the derivative of the first KKT condition, which was already the gradient of the Lagrangian of the problem. This Hessian matrix is the element that has the most computational cost to be calculated. Its calculation will have to be the most optimized possible to ensure the solver's scalability.

Note that dual feasibility condition $Z_i, \mu_i \geq 0 \quad \forall i$ 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 [3], derived with detail in [8]. The relevant equations obtained during the reduction process are shown in Equations (4.6) and (4.7).

$$\begin{cases} [\boldsymbol{\mu}]\Delta \boldsymbol{Z} + [\boldsymbol{Z}]\Delta \boldsymbol{\mu} & = -[\boldsymbol{\mu}]\boldsymbol{Z} + \gamma \mathbf{1}_{n_i} \\ [\boldsymbol{Z}]\Delta \boldsymbol{\mu} & = -[\boldsymbol{Z}]\boldsymbol{\mu} + \gamma \mathbf{1}_{n_i} - [\boldsymbol{\mu}]\Delta \boldsymbol{Z} \\ \Delta \boldsymbol{\mu} & = -\boldsymbol{\mu} + [\boldsymbol{Z}]^{-1}(\gamma \mathbf{1}_{n_i} - [\boldsymbol{\mu}]\Delta \boldsymbol{Z}). \end{cases} \tag{4.6}$$

$$\begin{cases} \boldsymbol{H}_x \Delta \boldsymbol{X} + \Delta \boldsymbol{Z} & = -\boldsymbol{H}(\boldsymbol{X}) - \boldsymbol{Z} \\ \Delta \boldsymbol{Z} & = -\boldsymbol{H}(\boldsymbol{X}) - \boldsymbol{Z} - \boldsymbol{H}_x \Delta \boldsymbol{X}. \end{cases} \tag{4.7}$$

$$\begin{cases} \boldsymbol{M} & \equiv \boldsymbol{\mathcal{L}_{xx}} + \boldsymbol{H_x^\top}[\boldsymbol{Z}]^{-1}[\boldsymbol{\mu}]\boldsymbol{H_x} \\ & = \boldsymbol{f_{xx}} + \boldsymbol{G_{xx}}(\boldsymbol{\lambda}) + \boldsymbol{H_{xx}}(\boldsymbol{\mu}) + \boldsymbol{H_x^\top}[\boldsymbol{Z}]^{-1}[\boldsymbol{\mu}]\boldsymbol{H_x}. \end{cases} \tag{4.8}$$

$$\begin{cases} \boldsymbol{N} & \equiv \boldsymbol{\mathcal{L}_x^\top} + \boldsymbol{H_x^\top}[\boldsymbol{Z}]^{-1}(\gamma \mathbf{1}_{\boldsymbol{n_i}} + [\boldsymbol{\mu}]\boldsymbol{H}(\boldsymbol{x})) \\ & = \boldsymbol{f_x^\top} + \boldsymbol{G_x^\top}\boldsymbol{\lambda} + \boldsymbol{H_x^\top}\boldsymbol{\mu} + \boldsymbol{H_x^\top}[\boldsymbol{Z}]^{-1}(\gamma \mathbf{1}_{\boldsymbol{n_i}} + [\boldsymbol{\mu}]\boldsymbol{H}(\boldsymbol{x})). \end{cases} \tag{4.9}$$

This transformation will make possible to reduce the size of Equation (4.5) to a system of equations over $(\boldsymbol{x}, \boldsymbol{\lambda})$ instead, and then calculating the terms corresponding to the inequalities using the expressions shown in Equation (4.6). This smaller size reduces the computational time of computing such a large system. The resulting reduced system is the following:

$$\begin{bmatrix} \boldsymbol{M} & \boldsymbol{G_x^\top} \\ \boldsymbol{G_x} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{x} \\ \Delta \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{N} \\ -\boldsymbol{G}(\boldsymbol{x}) \end{bmatrix} \tag{4.10}$$

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, and then it will compute the inequality terms using Equations (4.6) and (4.7).

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 shown at Equation (4.11).

$$\begin{aligned}\alpha_p &= \min\left(\tau \cdot \min\left(-\frac{\boldsymbol{Z}}{\Delta \boldsymbol{Z}}\right), 1\right)\\ \alpha_d &= \min\left(\tau \cdot \min\left(-\frac{\boldsymbol{\mu}}{\Delta \boldsymbol{\mu}}\right), 1\right)\end{aligned} \tag{4.11}$$

where τ is a parameter that will be set to a value really close to 1, without 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 Algorithm 2.

Algorithm 2 Step Control

Initialize $\mathcal{L}_0 = f(\boldsymbol{x}) + \boldsymbol{\lambda}^\top \boldsymbol{G}(\boldsymbol{x}) + \boldsymbol{\mu}^\top \boldsymbol{H}(\boldsymbol{x}) - \gamma \sum_{i=1}^{n_i} \log(Z_i)$, $\alpha = 1$, use its gradient $\boldsymbol{\mathcal{L}_x}$ and hessian $\boldsymbol{\mathcal{L}_{xx}}$ previously used for the Jacobian, and set $\alpha = trust$
while $i < step\_control\_iterations$ **do**
  $\Delta \boldsymbol{x}_1 = \alpha \Delta \boldsymbol{x}$
  $\boldsymbol{x}_1 = \boldsymbol{x} + \Delta \boldsymbol{x}_1$
  Compute $\mathcal{L}_1$ and $\mathcal{L}_{x_1}$
  $\rho = \frac{\mathcal{L}_1 - \mathcal{L}}{\boldsymbol{\mathcal{L}_x} \Delta \boldsymbol{x_1} + 0.5 \Delta \boldsymbol{x_1}^\top \boldsymbol{\mathcal{L}_{xx}} \Delta \boldsymbol{x_1}}$
  if $\rho \leq \rho_{lower}$ **or** $\rho \geq \rho_{upper}$ **then**
    $i = i + 1$
    $\alpha = \alpha / 2$
  end if
end while

This 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:

$$\begin{aligned}\boldsymbol{x} &\leftarrow \boldsymbol{x} + \alpha \alpha_p \Delta \boldsymbol{x}\\ \boldsymbol{Z} &\leftarrow \boldsymbol{Z} + \alpha \alpha_p \Delta \boldsymbol{Z}\\ \boldsymbol{\lambda} &\leftarrow \boldsymbol{\lambda} + \alpha \alpha_d \Delta \boldsymbol{\lambda}\\ \boldsymbol{\mu} &\leftarrow \boldsymbol{\mu} + \alpha \alpha_d \Delta \boldsymbol{\mu}\end{aligned} \tag{4.12}$$

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

$$\gamma \leftarrow \sigma \frac{\boldsymbol{Z}^\top \boldsymbol{\mu}}{n_i} \tag{4.13}$$

where σ is a parameter that will be between 0 and 1. The value of σ will be set to 0.1, as in [8], although Nocedal and Wright propose 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 the algorithm can be stopped. 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 ϵ to consider the problem solved. The later condition can be checked performing direct comparison, while the former two can be calculated as [9]:

$$\begin{aligned} feascond &= \frac{\max\left(\left[\|\mathbf{G}\|_\infty, \max(\boldsymbol{H})\right]\right)}{1 + \max\left(\left[\|\mathbf{x}\|_\infty, \|\mathbf{Z}\|_\infty\right]\right)} \\ gradcond &= \frac{\|\mathcal{L}_\mathbf{x}\|_\infty}{1 + \max\left(\left[\|\lambda\|_\infty, \|\mu\|_\infty\right]\right)} \end{aligned} \tag{4.14}$$

which monitor the compliance of the equality and inequality constraints for the $feascond$ and the gradient of the Lagrangian for the $gradcond$. 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 solving any kind of optimization 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 these equations, as well as the desired initialization and tolerance values. The objective of this project is to solve electrical systems, but any other type of system correctly modelled can also be solved using this solver.

The algorithm is described by the block diagram shown in Figure 4.1, representing the complete algorithm including all the steps. This solver will be called by a parent function that is responsible to model the whole problem and contains the initialization of the variables and the constraints and the functions that compute the value of the objective function, equalities, inequalities, and their Jacobians and Hessians.

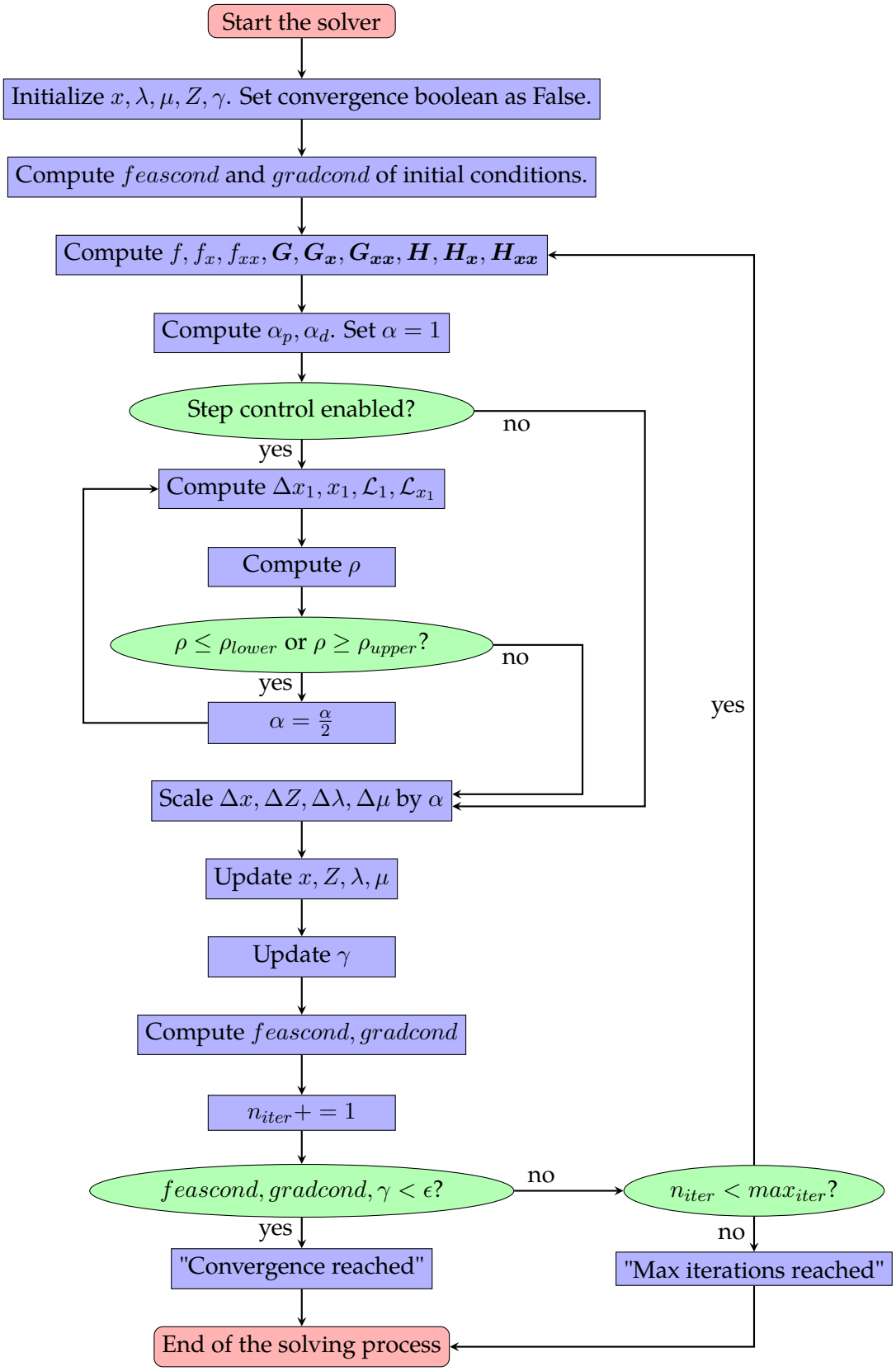

Figure 4.1: Block diagram of the solver.

4.3.1 Dealing with sparsity

A small comment to be made is that the solver will work with sparse matrices. The usage of this type of matrices available in the SciPy package [11] 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 matrix is 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 benchmark 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

In this section, the development of the AC-OPF model for the IPM solver is presented. As explained earlier, the aim of the project is to solve the power flow without simplification of any equation that can give unfeasible results, while giving the grid operator all the information that can be important to decide the operation of the grid, as for instance how the generation should be distributed among the dispatchable generators in order to have an optimal solution while accomplishing limits, or the setpoints that can be given to the transformers that have controllability.

The development of the model is crucial in order to have a solid model that encapsulates all the constraints of the grid, allowing the easy addition of new constraints that can be considered such as the power curves of the generators, that limit the reactive power generation depending on the active power generation.

The model is developed starting from Matpower's derivation for the traditional power flow variables in [12], and then the additional features developed in this project are introduced using a similar framework.

5.1 Decision variables

The power flow is modelled using polar form, since it is more compact to use compared to the rectangular form. 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:

$$\boldsymbol{x} = [\boldsymbol{\theta}^T, \boldsymbol{\mathcal{V}}^T, \boldsymbol{P_g}^T, \boldsymbol{Q_g}^T, \boldsymbol{sl_{sf}}^T, \boldsymbol{sl_{st}}^T, \boldsymbol{sl_{vmax}}^T, \boldsymbol{sl_{vmin}}^T, \boldsymbol{m_p}^T, \boldsymbol{\tau}^T, \boldsymbol{P_{DC}}^T]^T \tag{5.1}$$

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 n_g of active and reactive power generation, $\boldsymbol{sl_{sf}}$ and $\boldsymbol{sl_{st}}$ are the vectors of size m of slack variables for the line flow limits, $\boldsymbol{sl_{vmax}}$ and $\boldsymbol{sl_{vmin}}$ are the vectors of size n_{pq} of slack variables for the voltage limits, $\boldsymbol{m_p}$ and $\boldsymbol{\tau}$ are the vectors of transformer tap positions, sized n_{m_p} and n_τ respectively, and $\boldsymbol{P_{DC}}$ is the vector of DC link powers, sized n_{DC}.

In total, the number of decision variables can be calculated as:

$$NV = 2n + 2n_g + 2m + 2n_{pq} + n_{m_p} + n_\tau + n_{DC} \tag{5.2}$$

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.

These limitations can be set to be not-strict using slack variables, meaning they can be surpassed at a greater cost in order to be able to solve complex grid states, which can be the case of more realistic scenarios. 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 preferred 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:

$$\begin{aligned} f(\boldsymbol{x}) = & \sum_{i \in \text{ig}} c_{i_0} + c_{i_1} P_{g_i} + c_{i_2} P_{g_i}^2 \\ & + \sum_{k \in \text{M}} c_{s_k} (sl_{sf_k} + sl_{st_k}) + \sum_{i \in \text{pq}} c_{sl_{v_i}} (sl_{vmax_i} + sl_{vmin_i}) \end{aligned} \tag{5.3}$$

where c_{i_0}, c_{i_1} and c_{i_2} are the base, linear and quadratic coefficients of the cost function for the active power generation, c_{s_k} is the cost of the slack variables for the line flow limits, $c_{sl_{v_i}}$ is the cost of the slack variables for the voltage limits and ig, M and pq are the sets of dispatchable generators, monitored lines and PQ buses respectively.

5.2.1 Objective function gradient

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

$$\nabla f(\boldsymbol{x}) = \begin{bmatrix} \mathbf{0}_{\boldsymbol{n}} \\ \mathbf{0}_{\boldsymbol{n}} \\ \boldsymbol{c_1} + 2\boldsymbol{c_2}\boldsymbol{P_g} \\ \mathbf{0}_{\boldsymbol{n_g}} \\ \boldsymbol{c_s} \\ \boldsymbol{c_s} \\ \boldsymbol{c_{sl_v}} \\ \boldsymbol{c_{sl_v}} \\ \mathbf{0}_{\boldsymbol{n_{m_p}}} \\ \mathbf{0}_{\boldsymbol{n_\tau}} \\ \mathbf{0}_{\boldsymbol{n_{DC}}} \end{bmatrix} \tag{5.4}$$

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:

$$\nabla^2 f(\boldsymbol{x}) = \begin{bmatrix} 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 2[\boldsymbol{c_2}] & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 \end{bmatrix} \tag{5.5}$$

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:

$$\boldsymbol{G}(\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{G_P^S} \\ \boldsymbol{G_Q^S} \\ \boldsymbol{G_{slack}^{Th}} \\ \boldsymbol{G_{pv}^V} \end{bmatrix} \tag{5.6}$$

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 equalities for pv buses.

The power balance is calculated in complex form as in Equation (3.6) since the calculations are much more compact, 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 [12], 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:

$$G_X^S = \frac{\partial G^s}{\partial X} = \begin{bmatrix} G_\theta^S & G_V^S & G_P^S & G_Q^S & 0_{n\ x\ sl} & G_{m_p}^S & G_\tau^S & G_{P_{DC}}^S \end{bmatrix} \tag{5.7}$$

The first derivatives of the power balance equations are obtained form Matpower [12] as follows:

$$\begin{aligned} G_\theta^S &= \frac{\partial \boldsymbol{S_{bus}}}{\partial \boldsymbol{\theta}} = [\boldsymbol{I_{bus}^*}]\frac{\partial \boldsymbol{V}}{\partial \boldsymbol{\theta}} + [\boldsymbol{V}]\frac{\partial \boldsymbol{I_{bus}^*}}{\partial \boldsymbol{\theta}} \\ &= \left[\boldsymbol{I_{bus}^*}\right] j[\boldsymbol{V}] + \left[(j\boldsymbol{V})\boldsymbol{I_{bus}^*}\right] \\ &= j[\boldsymbol{V}]\left([\boldsymbol{I_{bus}^*}] - Y_{bus}^*[\boldsymbol{V^*}]\right) \\ G_{\mathcal{V}}^S &= \frac{\partial \boldsymbol{S_{bus}}}{\partial \boldsymbol{\mathcal{V}}} = \left[\boldsymbol{I_{bus}^*}\frac{\partial \boldsymbol{V}}{\partial \boldsymbol{\mathcal{V}}} + [\boldsymbol{V}]\frac{\partial \boldsymbol{I_{bus}^*}}{\partial \boldsymbol{\mathcal{V}}}\right] \\ &= \left[\boldsymbol{I_{bus}^*}\right][\boldsymbol{E}] + [\boldsymbol{V}]Y_{bus}^*[\boldsymbol{E^*}] \\ &= [\boldsymbol{V}]([\boldsymbol{I_{bus}^*}] + Y_{bus}^*[\boldsymbol{V^*}])[\boldsymbol{\mathcal{V}}]^{-1} \\ G_{P_g}^S &= -C_g \\ G_{Q_g}^S &= -jC_g \end{aligned} \tag{5.8}$$

Where $[\boldsymbol{E}] = [\boldsymbol{V}]\boldsymbol{\mathcal{V}}^{-1}$. 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 as seen in Equations (2.1) and (2.2). These primitives are used to calculate the $from$ and to power flowing through a line and its primitives with the following expression:

$$\begin{aligned} \boldsymbol{S_f} &= \boldsymbol{V_f} \boldsymbol{I_f^*} \\ \boldsymbol{S_t} &= \boldsymbol{V_t} \boldsymbol{I_t^*} \\ \boldsymbol{I_f} &= y_{ff} \boldsymbol{V_f} + y_{ft} \boldsymbol{V_t} \\ \boldsymbol{I_t} &= y_{tf} \boldsymbol{V_f} + y_{tt} \boldsymbol{V_t} \end{aligned} \tag{5.9}$$

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

$$\boldsymbol{S_{bus}} = C_f^T \boldsymbol{S_f} + C_t^T \boldsymbol{S_t} \tag{5.10}$$

The equality constraints for the power flow that have to be derived are the ones described by Equation (3.6). Deriving the expressions written above for the admittance primitives, we obtain the following expressions:

$$\begin{aligned} \frac{\partial \boldsymbol{G^S}}{\partial \boldsymbol{m_p}} &= \frac{\partial \boldsymbol{S_{bus}}}{\partial \boldsymbol{m_p}} = Cf^T \frac{\partial \boldsymbol{S_f}}{\partial \boldsymbol{m_p}} + Ct^T \frac{\partial \boldsymbol{S_t}}{\partial \boldsymbol{m_p}} \\ \frac{\partial \boldsymbol{G^S}}{\partial \boldsymbol{\tau}} &= \frac{\partial \boldsymbol{S_{bus}}}{\partial \boldsymbol{\tau}} = Cf^T \frac{\partial \boldsymbol{S_f}}{\partial \boldsymbol{\tau}} + Ct^T \frac{\partial \boldsymbol{S_t}}{\partial \boldsymbol{\tau}} \\ \frac{\partial S_{f_k}}{\partial m_{p_i}} &= V_{f_k} \left(\frac{-2(y_{s_k} V_{f_k})^*}{m_{p_i}^3} + \frac{(y_{s_k} V_{t_k})^*}{m_{p_i}^2 e^{j\tau}} \right) \\ \frac{\partial S_{t_k}}{\partial m_{p_i}} &= V_{t_k} \frac{(y_{s_k} V_{f_k})^*}{m_{p_i}^2 e^{-j\tau_i}} \\ \frac{\partial S_{f_k}}{\partial \tau_i} &= V_{f_k} \frac{j(y_{s_k} V_{t_k})^*}{m_{p_i} e^{j\tau_i}} \\ \frac{\partial S_{t_k}}{\partial \tau_i} &= V_{t_k} \frac{-j(y_{s_k} V_{f_k})^*}{m_{p_i} e^{-j\tau_i}} \end{aligned} \tag{5.11}$$

Each of the individual S_{f_k} and S_{t_k} derivative with respect to each i transformer variable will be stored in the position of their corresponding matrix as $\frac{dS_f}{dm_p}_{ki}$ and $\frac{dS_f}{dm_p}_{ki}$, with size m x n_{m_p} and m x n_τ respectively, with m being the total number of branches of the grid. These matrices will have very few values different from zero, since the transformer control is only present in a few transformers of the grid.

The derivative of $G^S_{P_{DC}}$ 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 n_{DC}. The derivative is obtained as follows:

$$\begin{cases} G^S_{P_{DC}}\{fdc, link\} = 1 \quad \forall link = (fdc, tdc) \in DC \\ G^S_{P_{DC}}\{tdc, link\} = -1 \quad \forall link = (fdc, tdc) \in DC \\ 0 \quad \text{otherwise} \end{cases} \tag{5.12}$$

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

$$G_X = \begin{bmatrix} \boldsymbol{\mathcal{R}(G_X^{S^\top})} \\ \boldsymbol{\mathcal{I}(G_X^{S^\top})} \\ \boldsymbol{G_X^{Th^\top}} \\ \boldsymbol{G_X^{\mathcal{V}^\top}} \end{bmatrix} \tag{5.13}$$

where $\boldsymbol{G_X^S}0$ is separated into its real ($\mathcal{R}$) and imaginary ($\mathcal{I}$) part since the solver needs real valued matrices, 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

The Hessian matrix associated to the equality constraints has the following structure:

$$G^S_{XX} = \begin{bmatrix} G_{\theta\theta} & G_{\theta\mathcal{V}} & G_{\theta P_g} & G_{\theta Q_g} & G_{\theta sl} & G_{\theta m_p} & G_{\theta\tau} & G_{\theta P_{DC}} \\ G_{\mathcal{V}\theta} & G_{\mathcal{V}\mathcal{V}} & G_{\mathcal{V}P_g} & G_{\mathcal{V}Q_g} & G_{\mathcal{V}sl} & G_{\mathcal{V}m_p} & G_{\mathcal{V}\tau} & G_{\mathcal{V}P_{DC}} \\ G_{P_g\theta} & G_{P_g\mathcal{V}} & G_{P_gP_g} & G_{P_gQ_g} & G_{P_gsl} & G_{P_gm_p} & G_{P_g\tau} & G_{P_gP_{DC}} \\ G_{Q_g\theta} & G_{Q_g\mathcal{V}} & G_{Q_gP_g} & G_{Q_gQ_g} & G_{Q_gsl} & G_{Q_gm_p} & G_{Q_g\tau} & G_{Q_gP_{DC}} \\ G_{sl\theta} & G_{sl\mathcal{V}} & G_{slP_g} & G_{slQ_g} & G_{slsl} & G_{slm_p} & G_{sl\tau} & G_{slP_{DC}} \\ G_{m_p\theta} & G_{m_p\mathcal{V}} & G_{m_pP_g} & G_{m_pQ_g} & G_{m_psl} & G_{m_pm_p} & G_{m_p\tau} & G_{m_pP_{DC}} \\ G_{\tau\theta} & G_{\tau\mathcal{V}} & G_{\tau P_g} & G_{\tau Q_g} & G_{\tau sl} & G_{\tau m_p} & G_{\tau\tau} & G_{\tau P_{DC}} \\ G_{P_{DC}\theta} & G_{P_{DC}\mathcal{V}} & G_{P_{DC}P_g} & G_{P_{DC}Q_g} & G_{P_{DC}sl} & G_{P_{DC}m_p} & G_{P_{DC}\tau} & G_{P_{DC}P_{DC}} \end{bmatrix} \tag{5.14}$$

The first set of second derivatives corresponding to the 4 x 4 submatrix block of the Hessian (those that involve only derivation with respect to Power Flow variables $((\theta, \mathcal{V}, P_g, Q_g))$, noted $\boldsymbol{X_{PF}}$) is obtained from [12].

$$
\begin{aligned}
G^S_{XX_{PF}}(\boldsymbol{\lambda}) &= \frac{\partial}{\partial \boldsymbol{X_{PF}}}(\boldsymbol{G^S}^{\top}_{\boldsymbol{X_{PF}}}\boldsymbol{\lambda}) \\
&= \begin{bmatrix} G^S_{\boldsymbol{\theta\theta}}(\boldsymbol{\lambda}) & G^S_{\boldsymbol{\theta}V}(\boldsymbol{\lambda}) & 0 & 0 \\ G^S_{V\boldsymbol{\theta}}(\boldsymbol{\lambda}) & G^S_{VV}(\boldsymbol{\lambda}) & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\
G^S_{\boldsymbol{\theta\theta}}(\boldsymbol{\lambda}) &= \frac{\partial}{\partial \boldsymbol{\theta}}(G^{S^{\top}}_{\boldsymbol{\theta}}\boldsymbol{\lambda}) \\
&= \left[\boldsymbol{V^*}\right]\left(Y^{*\top}_{bus}\left[\boldsymbol{V}\right]\left[\boldsymbol{\lambda}\right] - \left[Y^{*\top}_{bus}[\boldsymbol{V}]\boldsymbol{\lambda}\right]\right) \\
&\quad + \left[\boldsymbol{\lambda}\right]\left[\boldsymbol{V}\right]\left(Y^{*}_{bus}\left[\boldsymbol{V^*}\right] - \left[\boldsymbol{I^*_{bus}}\right]\right) \\
G^S_{\boldsymbol{\mathcal{V}\theta}}(\boldsymbol{\lambda}) &= \frac{\partial}{\partial \boldsymbol{\theta}}(G^{S^{\top}}_{\boldsymbol{\mathcal{V}}}\boldsymbol{\lambda}) \\
&= j[\boldsymbol{\mathcal{V}}]^{-1}([\boldsymbol{V^*}](Y^*_{bus}[\boldsymbol{V}][\boldsymbol{\lambda}] - [Y^{*\top}_{bus}[\boldsymbol{V}]\boldsymbol{\lambda}]) \\
&\quad - [\boldsymbol{\lambda}][\boldsymbol{V}](Y^*_{bus}[\boldsymbol{V^*}] - [\boldsymbol{I^*_{bus}}])) \\
G^S_{\boldsymbol{\mathcal{V}\mathcal{V}}}(\boldsymbol{\lambda}) &= \frac{\partial}{\partial \boldsymbol{\mathcal{V}}}(G^{S^{\top}}_{\boldsymbol{\mathcal{V}}}\boldsymbol{\lambda}) \\
&= [\boldsymbol{\mathcal{V}}]^{-1}\left([\boldsymbol{\lambda}][\boldsymbol{V}]Y^*_{bus}[\boldsymbol{V^*}] + [\boldsymbol{V^*}]Y^*_{bus}[\boldsymbol{V}][\boldsymbol{\lambda}]\right)[\boldsymbol{\mathcal{V}}]^{-1}
\end{aligned}
\tag{5.15}
$$

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, with the expressions for a given branch k and its connected buses (f_k, t_k) 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{5.16}
$$

Crossed partial derivative $\frac{\partial^2}{\partial m_{p_i} \partial \tau_i}$ are considered for those lines with transformers that controls both variables. To compose the submatrix, the following expression is used for all of these transformers:

$$\begin{aligned} \frac{dS_{bus}}{dm_p dm_p}_{ii} &= \Re\left(\frac{\partial^2 S_{f_k}}{\partial m_{p_i} \partial m_{p_i}} \lambda_f + \frac{\partial^2 S_{t_k}}{\partial m_{p_i} \partial m_{p_i}} \lambda_t\right) \\ &+ \Im\left(\frac{\partial^2 S_{f_k}}{\partial m_{p_i} \partial m_{p_i}} \lambda_{f+N} + \frac{\partial^2 S_{t_k}}{\partial m_{p_i} \partial m_{p_i}} \lambda_{t+N}\right) \end{aligned} \tag{5.17}$$

Since we have to take into account 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:

$$\boldsymbol{H}(\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{H^{S_f}} \\ \boldsymbol{H^{S_t}} \\ \boldsymbol{H^{V_u}} \\ \boldsymbol{H^{P_u}} \\ \boldsymbol{H^{Q_u}} \\ \boldsymbol{H^{V_l}} \\ \boldsymbol{H^{P_l}} \\ \boldsymbol{H^{Q_l}} \\ \boldsymbol{H^{sl}} \\ \boldsymbol{H^{m_{p_u}}} \\ \boldsymbol{H^{m_{p_l}}} \\ \boldsymbol{H^{\tau_u}} \\ \boldsymbol{H^{\tau_l}} \\ \boldsymbol{H^{Q_{max}}} \\ \boldsymbol{H^{P_{DC_u}}} \\ \boldsymbol{H^{P_{DC_l}}} \end{bmatrix} \tag{5.18}$$

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 have their value squared.

$$\boldsymbol{H_f^S} = [\boldsymbol{S_f^*}]\boldsymbol{S_f} - \boldsymbol{S_{max}^2} \leq 0 \tag{5.19}$$

$$\begin{aligned} H_X^f &= 2(\Re([\boldsymbol{S_f}])\Re(S_X^f) + \Im([\boldsymbol{S_f}])\Im(S_X^f)) \\ H_{XX}^f(\boldsymbol{\mu}) &= 2\Re(S_{XX}^f([\boldsymbol{S_f^*}]\boldsymbol{\mu}) + {S_X^f}^T[\boldsymbol{\mu}]S_X^f) \end{aligned} \tag{5.20}$$

And similarly for the H^t *to* constraints. The derivatives with respect of the PF variables are obtained from [12] again:

$$\begin{aligned} S_{\boldsymbol{\theta}}^f &= \left[\boldsymbol{I_f^*}\right]\frac{\partial \boldsymbol{V_f}}{\partial \boldsymbol{\theta}} + \left[\boldsymbol{V_f}\right]\frac{\partial \boldsymbol{I_f^*}}{\partial \boldsymbol{\theta}} \\ &= \left[\boldsymbol{I_f^*}\right] jC_f[\boldsymbol{V}] + [C_f\boldsymbol{V}](j\bar{Y_f}[\boldsymbol{V}])^* \\ &= j\left([\boldsymbol{I_f^*}]C_f[\boldsymbol{V}] - [C_f\boldsymbol{V}]Y_f^*[\boldsymbol{V^*}]\right) \\ S_{\boldsymbol{\mathcal{V}}}^f &= \left[\boldsymbol{I_f^*}\right]\frac{\partial \boldsymbol{V_f}}{\partial \boldsymbol{\mathcal{V}}} + \left[\boldsymbol{V_f}\right]\frac{\partial \boldsymbol{I_f^*}}{\partial \boldsymbol{\mathcal{V}}} \\ &= \left[\boldsymbol{I_f^*}\right] C_f[\boldsymbol{E}] + [C_f\boldsymbol{V}]Y_f^*[\boldsymbol{E^*}] \\ S_{\boldsymbol{P_g}}^f &= 0 \\ S_{\boldsymbol{Q_g}}^f &= 0 \end{aligned} \tag{5.21}$$

$$\begin{aligned} S_{\boldsymbol{\theta\theta}}^f(\boldsymbol{\mu}) &= \frac{\partial}{\partial \boldsymbol{\theta}}\left(S_{\boldsymbol{\theta}}^f\right)^T\boldsymbol{\mu} \\ &= [\boldsymbol{V^*}]{Y_f^*}^\top[\boldsymbol{\mu}]C_f[\boldsymbol{V}] + [\boldsymbol{V}]C_f^\top[\boldsymbol{\mu}]Y_f^*[\boldsymbol{V^*}] \\ &- [{Y_f^*}^\top[\boldsymbol{\mu}]C_f\boldsymbol{V}][\boldsymbol{V^*}] - [C_f^T[\boldsymbol{\mu}]Y_f^*\boldsymbol{V^*}][\boldsymbol{V}] \\ S_{\boldsymbol{\mathcal{V}\theta}}^f(\boldsymbol{\mu}) &= \frac{\partial}{\partial \boldsymbol{\theta}}\left(S_{\boldsymbol{\mathcal{V}}}^f\right)^\top\boldsymbol{\mu} \\ &= j[\boldsymbol{\mathcal{V}}]^{-1}([\mathbf{V^*}]{Y_f^*}^\top[\boldsymbol{\mu}]C_f[\mathbf{V}] - [\mathbf{V}]C_f^\top[\boldsymbol{\mu}]Y_f^*[\mathbf{V^*}] \\ &- [{Y_f^*}^\top[\boldsymbol{\mu}]C_f\mathbf{V}][\mathbf{V^*}] + [C_f^\top[\boldsymbol{\mu}]Y_f^*\mathbf{V^*}][\mathbf{V}]) \\ S_{\boldsymbol{\mathcal{V}\mathcal{V}}}^f(\boldsymbol{\mu}) &= \frac{\partial}{\partial \boldsymbol{\mathcal{V}}}\left(S_{\boldsymbol{\mathcal{V}}}^f\right)^\top\boldsymbol{\mu} \\ &= [\boldsymbol{\mathcal{V}}]^{-1}\left([\mathbf{V^*}]{Y_f^*}^\top[\boldsymbol{\mu}]C_f[\mathbf{V}] + [\mathbf{V}]C_f^\top[\boldsymbol{\mu}]Y_f^*[\mathbf{V^*}]\right)[\boldsymbol{\mathcal{V}}]^{-1} \end{aligned} \tag{5.22}$$

The derivatives with respect to the tap variables have been derived using $\boldsymbol{H}$, the expressions for $\boldsymbol{S_f}$, $\boldsymbol{S_t}$ and its derivatives obtained in the power balance derivatives in Equations (5.11).

6 Results

After implementing the software in the GridCal package, now it is time to test how this solver performs when compared to already available solutions. The benchmark used for this comparison will be the Matpower solver [2], an open-source package that has been used in many works as a reference for the AC-OPF problem. The objective is to have a similar runtime and number of iterations for the base AC-OPF problem (the model already used by Matpower's solver), and to see how well does it scale when the new features are added.

The grids used for this benchmark are a combination of small grids with some interesting configurations that allow some testing of the new features, and larger grids that will be used to see if the solver is capable to solve them and what is the computational cost of doing so. The largest grid is the Great Britain national grid (an adaptation), which can give some insight on how the solver performs in a real-world scenario.

The results obtained have been obtained in a computer with the following specifications:

- CPU: Intel Core i7-1165G7 @ 2.80GHz
- RAM: 16GB DDR4
- GPU: Intel Iris Xe Graphics
- OS: Windows 11
- IDE: PyCharm
- Python version: 3.10
- Matpower version - Python version: PyPower 5.1.16 [2] [13]

And the tolerance requested for the convergence conditions is 10^{-6}.

There are two possible initializations for the solver, which are the following:

- **Power flow initialization:** The solver uses GridCal's built-in power flow with the default values introduced in the grid model as the initial values for the variables. The results of the power flow solver are used as the starting point for the optimization variables.
- **Basic initialization:** This is the initialization used by Matpower, which sets the variables at the center of the domain. Although it seems simpler, for some cases it could be better to ensure all the variables are not too close to some of their limits.

The use of bound slacks has also been tested for both initialization options, yielding a total of 4

different combinations per case. The results will be shown in the following subsections.

6.1 Error evolution

The first comparison carried out is the error evolution for the different cases. The graphs portray the error at each iteration for the different initialization settings, and will be compared to the error obtained by Matpower's solver. The error is the maximum value of the expressions (4.14) at each iteration. This error corresponds to the maximum value of the convergence conditions, which are calculated for each derivative of the Lagrangian, and including the γ value. The error should always tend to decrease, although sometimes it does so really slowly, or it even goes up for some iterations. This could mean that the path taken by the interior point method is exceeding some limits, and the resulting new point is worse than the previous. Excessive increases of the error are normally avoided by the step control implemented in the solver, although it does not prevent these increases in problems that are ill-conditioned or even unsolvable.

6.1.1 Case 9-bus

The first case is a 9-bus system [14] which can be useful to see the behavior of the algorithm in a small system, while it can be portrayed as shown in Figure 6.1 with sufficient clarity in this project using the GridCal GUI.

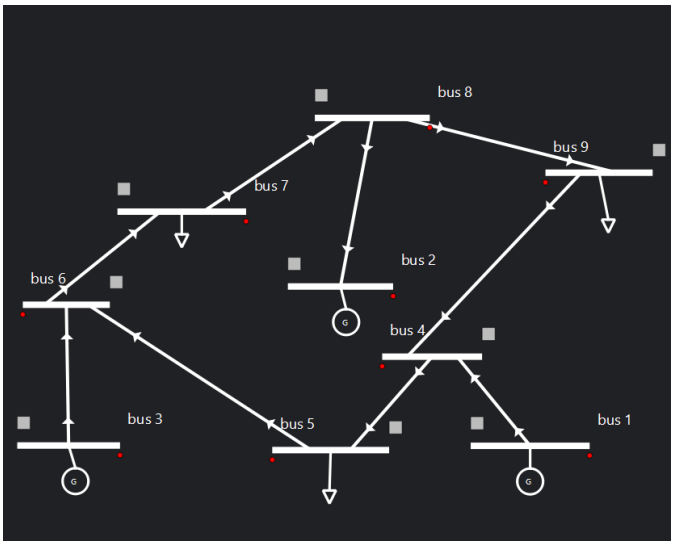

Figure 6.1: Schematic of the 9-bus grid.

It can be seen in Figure 6.2 that for such a small grid, there is not a significant difference between the different initialization options. The error evolution is very similar for all of them, and the error obtained is very close to the one obtained by Matpower, being the option without bound slacks slightly better as the vector of variables is considerably smaller. If there are no issues with the grid, the initialization with bound slacks is not necessary for this case.

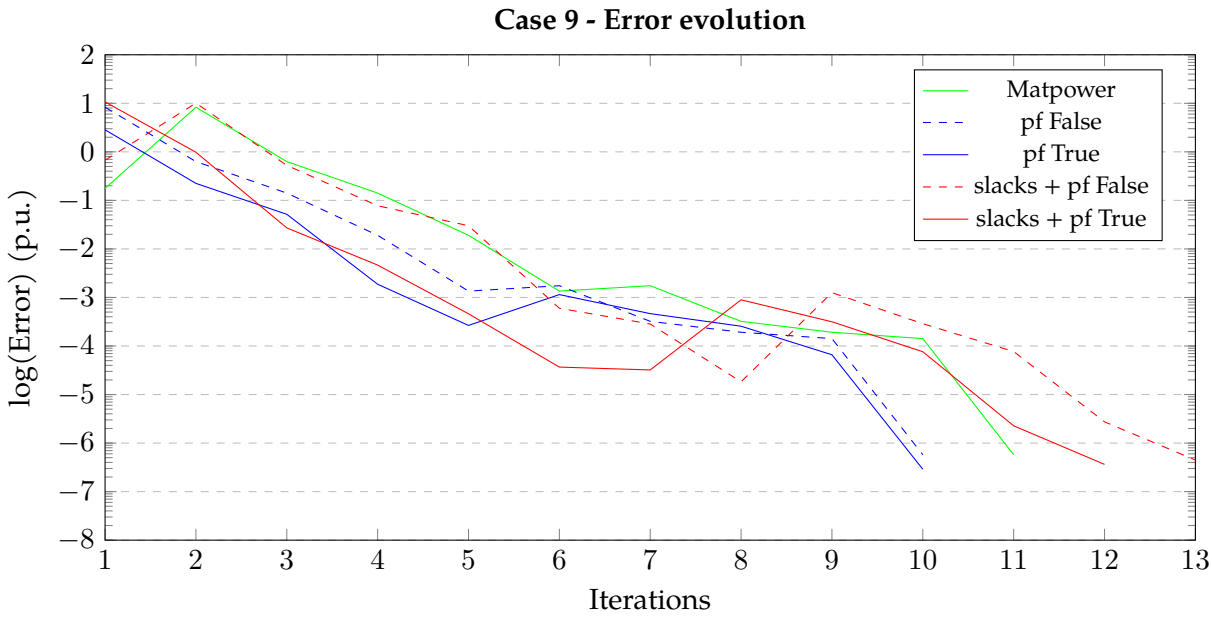

Figure 6.2: Error evolution for the Case 9 system for different initialization options.

The result can be visualized in GridCal's GUI as shown in Figure 6.3.

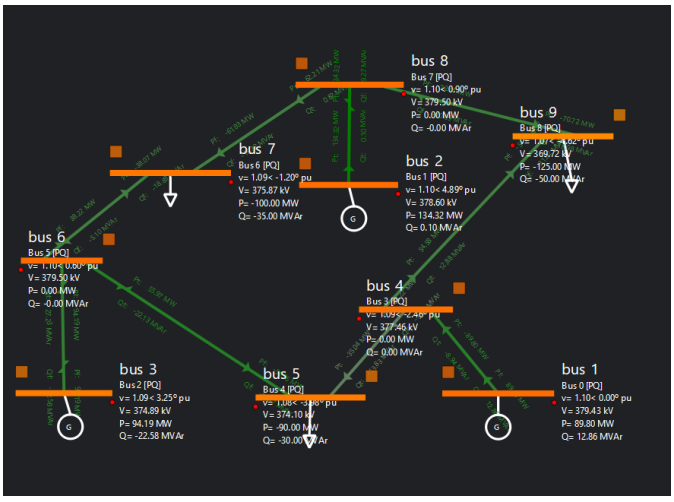

Figure 6.3: Plot of the solution of Case 9.

Something that can catch the eye is the orange color of the bus, which means they are near their upper voltage limit. This is due to the optimization trying to raise the voltage value as high as the limit allows to reduce the current flowing through the lines, which reduces quadratically the losses.

6.1.2 Case Pegase89

The next grid studied is a case with 89 buses obtained from the Pegase database [15], with an interesting result shown in Figure 6.4.

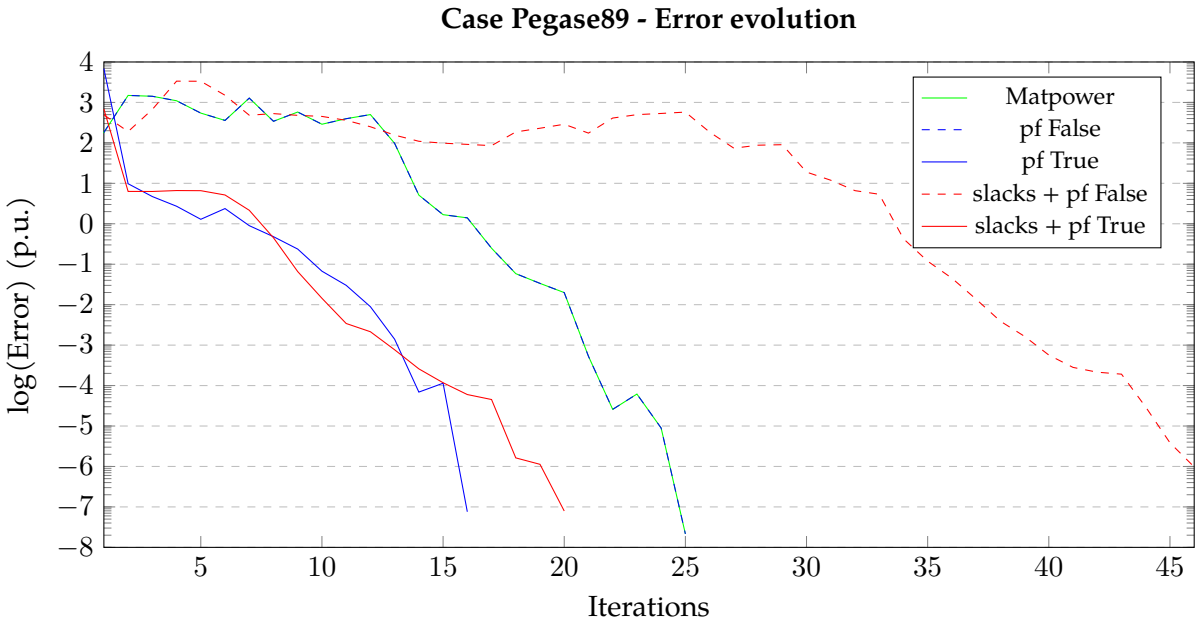

Figure 6.4: Error evolution for the Pegase89 system for different initialization options.

Firstly, it should be noted that the solution without bound slacks and with the power flow initialization disabled follows the exact same evolution as the Matpower solver. This did not happen in the previous case, but it seems to be due to the GridCal parser processing of the model. For some cases during the developing of the software, some grids were unsolvable due to some of its parameters not being directly specified in the model. For instance, some line ratings were not specified, and while the Matpower solver would then ignore some of these limits, the GridCal parser would not be able to process it in a correct way. To solve this, the parser has been adjusted to set a default value in case the lines are monitored but the rating is specified. Many other small nuances during the model loading could be the source of this difference in the convergence.

The other relevant observation that can be done is that the power flow initialization is doing a great job in improving the convergence of the problem. In a real scenario situation, the user would input a grid model that is already in a feasible state, although it may not be optimal, but it might be close to what an optimum generation profile would be. In this case, the OPF using an initialization with power flow takes almost half of the iterations in the case without bound slacks, and a third in the case with bound slacks.

One last observation is that bound slacks are not useful in this case either. The error evolution is very similar to the one without bound slacks, and the number of variables is considerably larger,

which will make the problem a bit heavier to run as it will be shown in a later subsection.

6.1.3 Case 300

The next case is a 300-bus system [16], which is a considerably larger grid than the previous ones. The error evolution for each initialization is shown in Figure 6.5.

Figure 6.5: Error evolution for the 300-bus system for different initialization options.

The performance of the solver developed in this work outclasses the one of Matpower in this case. Considering that the solver is based on the same principles, this means that the GridCal modelled problem is better conditioned than the one modelled by Matpower, which is why the solver has been developed in such an environment.

6.1.4 Case GB

Last case shown is a country-sized 2223-bus system that has been provided directly by Redeia, which models the Great Britain grid. Error evolutions are shown in Figure 6.6.

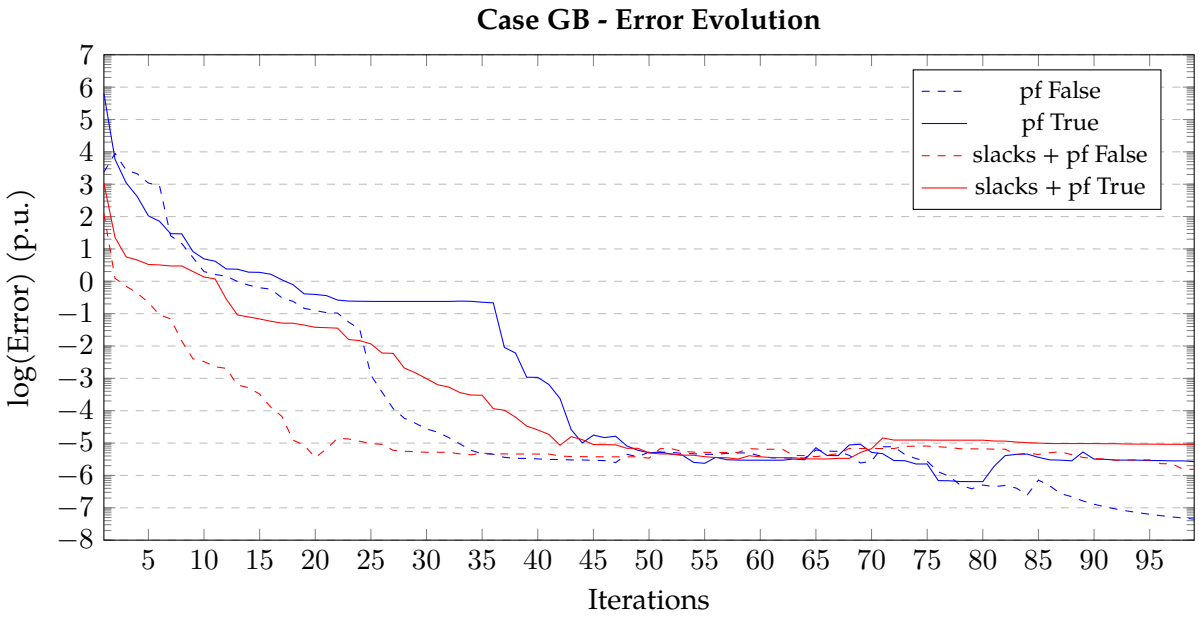

Figure 6.6: Error evolution for the Great Britain grid for different initialization options.

In this case, there is no comparative with the Matpower solver since the data is stored in a file that can only be managed in GridCal, and the model is not public. The results show a really promising evolution for a real use case, obtaining a low error in a few iterations. The slack initialization option can be preferred in this case, which is to be expected since it is a case that has not been tested before, and it might correspond to a difficult grid state (meaning the limits could be a bit strict, some generators/lines could be disabled or the load profile could be high).

Despite the good performance during the first steps, it stagnates when asking for lower tolerance values, a side effect of adding a lot of variables to the problem.

6.2 Effect of tap variables control - Case IEEE14

Once having seen the performance of the solver in some study cases, it is time to analyze the effect of the tap variables control. This control is a feature that allows the solver to manage the tap variables of the transformers in the grid, which are the ones that can be modified to control the voltage levels or the power transfer. This control can be done in two ways: by setting the tap variables as fixed variables or by setting them as optimization variables. The first option is the one that has been used in the previous subsections, and the second one is the one that is going to be tested in this one.

The grid used is the IEEE14 model grid [17], which is a 14-bus grid that includes transformers. Firstly, the error evolution is compared in Figures 6.7 and 6.8 to see the effect of adding this tap variables.

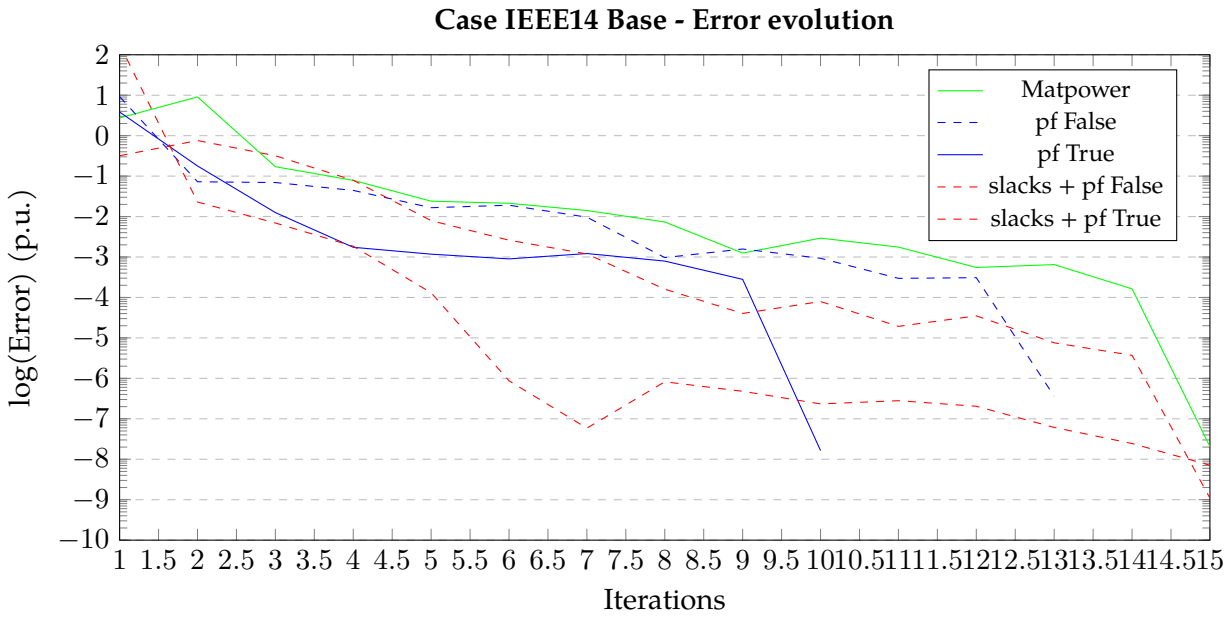

Figure 6.7: Error evolution for the Case IEEE14 for different initialization options.

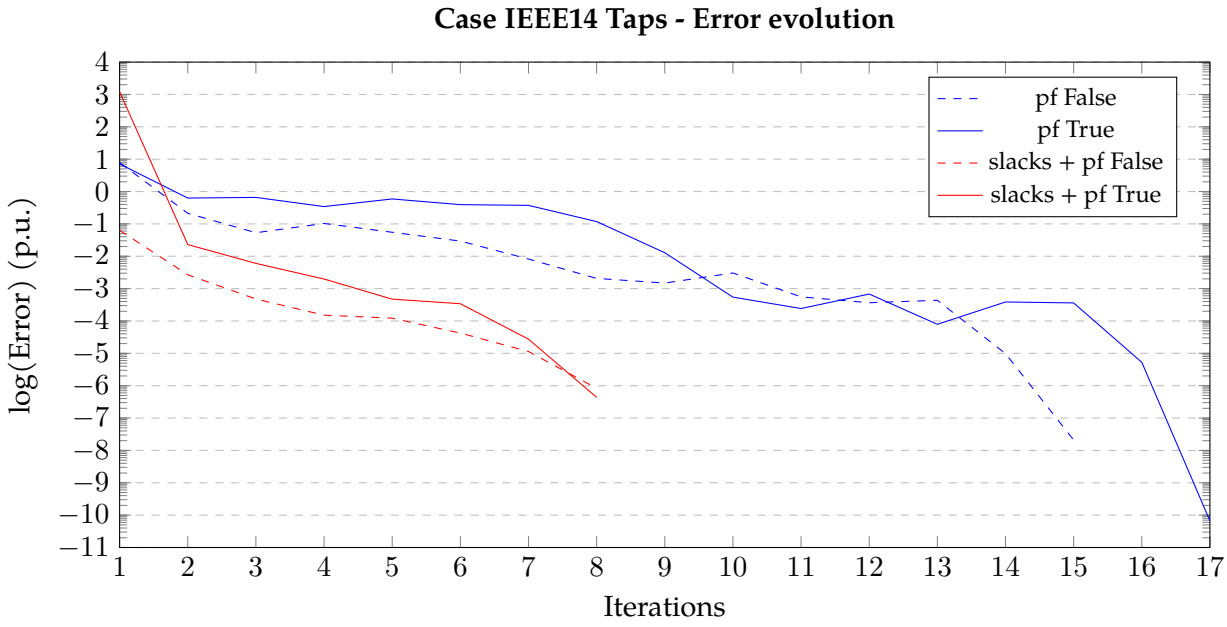

Figure 6.8: Error evolution for the Case IEEE14 with tap variables for different initialization options.

There are two important remarks obtained from these error evolutions. Firstly, the case is solved in fewer iterations in the version of the solver developed in this project. Secondly, the usage of bound slacks improves convergence for the case with tap variables. A more detailed analysis is done for the solution values for this case to analyze the impact of adding new optimization

variables. For simplicity, the comparison will be made only for the cases with power flow initialization and without bound slacks, as they add to the cost function and the interest of this analysis is just testing the effect of using tap optimization.

Figures 6.9 and 6.10 show the values of the solution for the bus voltages and generators. Note that there are no differences between the case solved with Matpower and the developed software. The case with controlled tap variable has some slight differences, specially regarding the voltage magnitude. The impact in the generation is also noticeable for the reactive power generation, which has shifted mostly from generator 1 to generator 3.

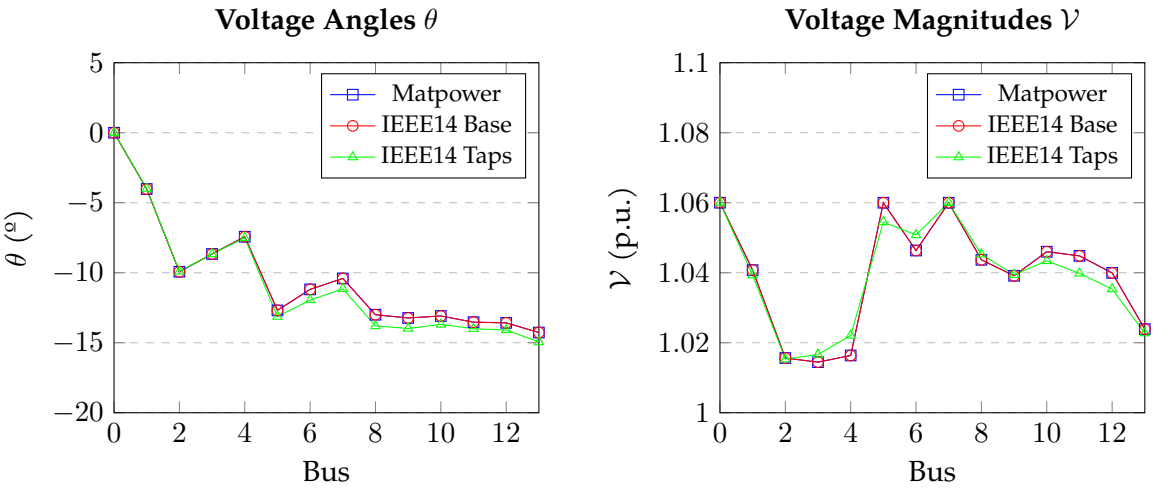

Figure 6.9: Voltage comparison for the Case IEEE14 considering tap variables.

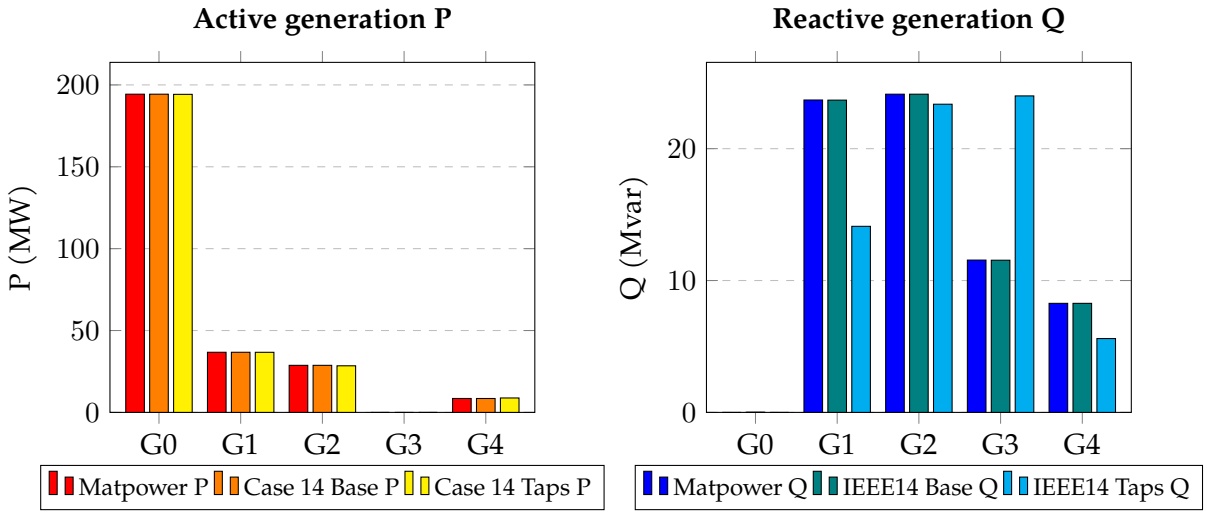

Figure 6.10: Generation comparison for Case IEEE14.

Table 6.1 shows the setpoints for the tap variables of the transformers in the grid as obtained with the solver developed. Note that the transformers that only have one variable controlled

have a N/C (Non-Controlled) as a solution.

Table 6.1: Tap variable setpoints for the transformers of the corresponding branches in IEEE14.

Controlled trafos	**m_p (p.u)**	τ (º)
Branch 17	0.965876	0.754872
Branch 18	N/C	1.365530
Branch 19	0.974728	N/C

The important result is shown in Table 6.2, which shows the impact of being able to control the tap variables setpoints in the cost of operation. Of course, this impact may not seem big in this case, representing a reduction of cost of only 0.03%. However, this result has only considered a small grid, without using properly modelled generation costs, and only controlling three transformers. And still, considering the total expenses of a country-sized grid as well as this result being an instant photo of a given grid state, it can translate in substantial savings in the long-term.

Table 6.2: Comparison of costs between different cases.

	Matpower	**Case 14 Base**	**Case 14 Taps**
Cost (€/MWh)	8081.53	8081.53	8078.85

6.3 Effect of reactive control

Using the same base case, now the test is performed over the reactive power control. The limit is set as a maximum of a power factor of 0.8, which is translated to a maximum reactive power generation of 75% of the active power generation. The constraint is applied with respect to the absolute value of Q, meaning this constraint applies to both the injection and consumption of reactive power. Figure 6.11 shows the effect in the voltage values. There is a noticeable drop in magnitude in some of the buses.

In Figure 6.12, it can be seen that generator 2, which previously had a reactive generation of 24.12 Mvar, now has a generation of 20.96 Mvar, which corresponds exactly to the 75% of the active power generation. This generator being at its limit for reactive power generation means the other generator have to contribute to reactive generation, impacting the voltage levels in the grid.

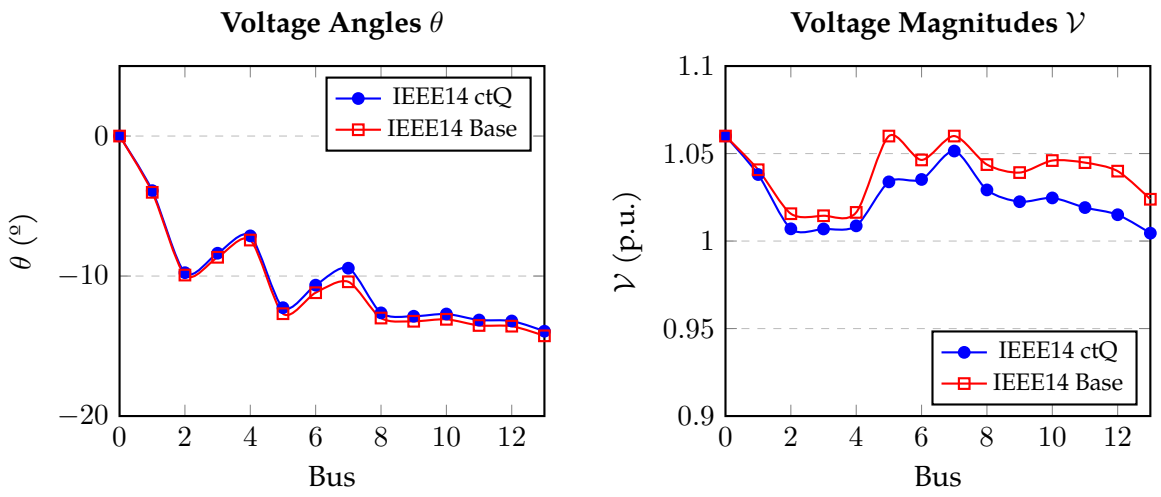

Figure 6.11: Voltage comparison for the Case IEEE14 with and without reactive control.

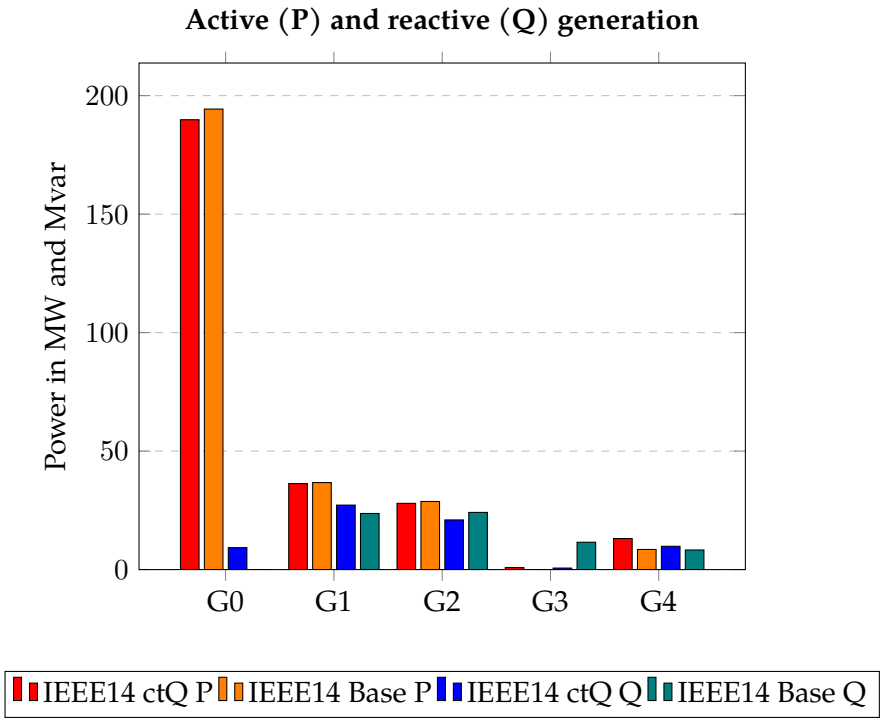

Figure 6.12: Generation comparison for the Case IEEE14 with and without reactive control.

The effect of this constraint can be seen in the cost value of the simulation. Table 6.3 shows an increase of around 0.075% in the cost of operation, which again can be significant in a bigger grid.

Table 6.3: Cost for Case IEEE14 with Q control.

	IEEE14 Base	IEEE14 ctQ
Cost (€/MWh)	8081.53	8087.82

6.4 DC link and dual price study

The following study is performed over a small grid with two islands of three buses each. Both of them have the same configuration and elements, but their generators will have different costs profiles as follows:

$$\begin{aligned} c_{g1.0}(P_{g1.0}) &= 1 \cdot P_{g1.0} + 2 \cdot P_{g1.0}^2 \\ c_{g1.1}(P_{g1.1}) &= 1 \cdot P_{g1.1} + 3 \cdot P_{g1.1}^2 \\ c_{g2.0}(P_{g2.0}) &= 1 \cdot P_{g2.0} + 1.5 \cdot P_{g2.0}^2 \\ c_{g2.1}(P_{g2.1}) &= 1 \cdot P_{g2.1} + 1 \cdot P_{g2.1}^2 \end{aligned} \tag{6.1}$$

The grid is firstly tested without interconnection, obtaining the results for each island isolated. Then, a lossless DC link is added between the two islands. In Figure 6.13, the comparison between the generation profiles can be seen.

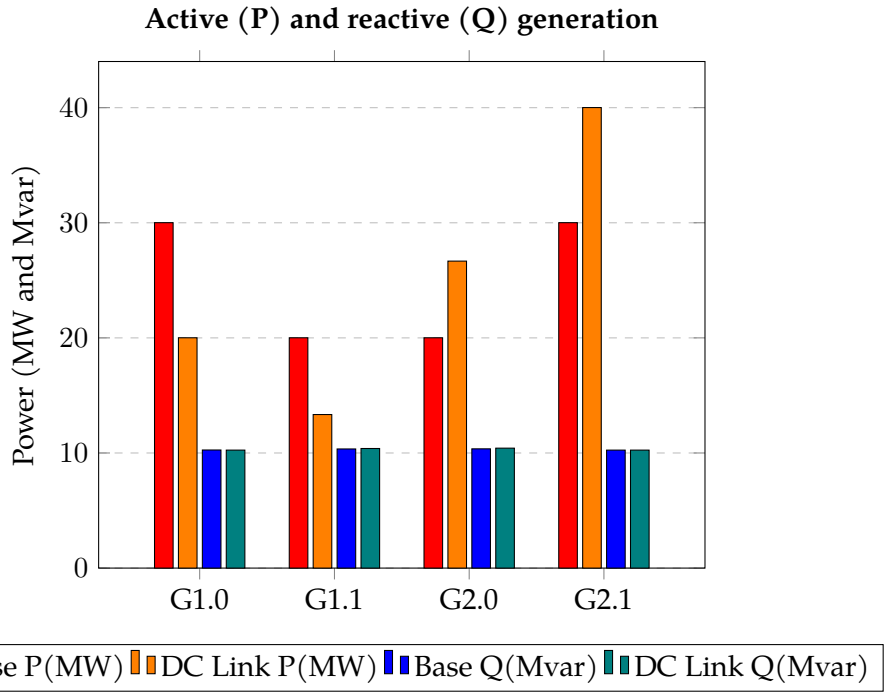

Figure 6.13: Generation comparison for the island case with and without DC link.

As expected, after both grids are connected, the generators of the second island become prevalent as their prices are lower than the first island, and the power is transferred through this DC link with a total power of $P_{DC} = 16.67$MW. The interconnection has a great impact in the dual price as seen in Figure 6.14.

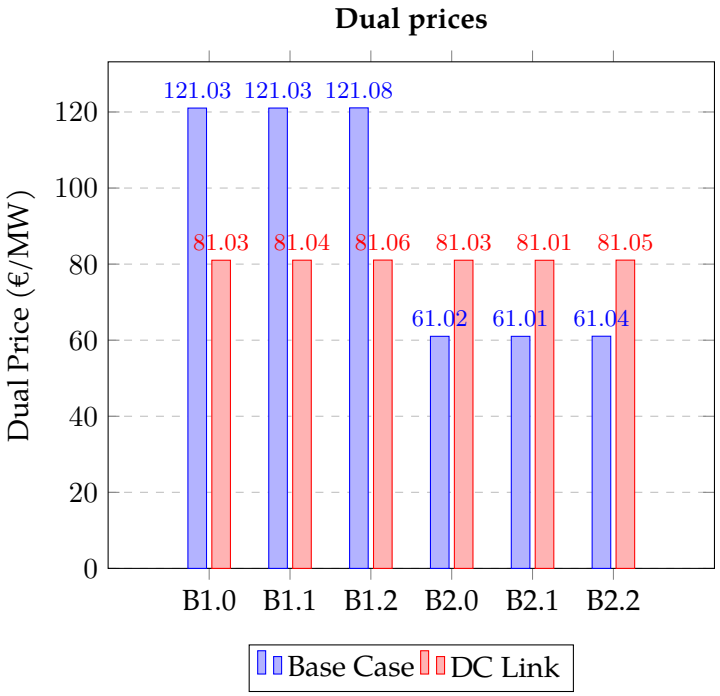

Figure 6.14: Dual price comparison for the case with and without DC link.

These results are really important to understand the meaning of the dual price, which for the optimization problem are equal to the value of the equality multipliers associated to the nodal power balance of each bus. These values indicate the rate of increase in cost that would be caused by demanding an additional infinitesimal amount of power at that bus. The following calculations can be done to check which is the cost of generating an additional MW of power per generation, using their cost function and the obtained setpoint from the optimization solution:

$$\begin{aligned}
\frac{\partial c_{g1.0}}{\partial P_{g1.0}}_{Base} &= 1 + 4 \cdot P_{g1.0} = 1 + 4 \cdot 30.0067 = 121.027 \\
\frac{\partial c_{g1.1}}{\partial P_{g1.1}}_{Base} &= 1 + 6 \cdot P_{g1.1} = 1 + 6 \cdot 20.0056 = 121.034 \\
\frac{\partial c_{g2.0}}{\partial P_{g2.0}}_{Base} &= 1 + 3 \cdot P_{g2.0} = 1 + 3 \cdot 20.0056 = 61.017 \\
\frac{\partial c_{g2.1}}{\partial P_{g2.1}}_{Base} &= 1 + 2 \cdot P_{g2.1} = 1 + 2 \cdot 30.0067 = 61.013
\end{aligned} \tag{6.2}$$

$$\begin{aligned}
\frac{\partial c_{g1.0}}{\partial P_{g1.0}}_{DC} &= 1 + 4 \cdot P_{g1.0} = 1 + 4 \cdot 20.0062 = 81.025 \\
\frac{\partial c_{g1.1}}{\partial P_{g1.1}}_{DC} &= 1 + 6 \cdot P_{g1.1} = 1 + 6 \cdot 13.3392 = 81.035 \\
\frac{\partial c_{g2.0}}{\partial P_{g2.0}}_{DC} &= 1 + 3 \cdot P_{g2.0} = 1 + 3 \cdot 26.6750 = 81.025 \\
\frac{\partial c_{g2.1}}{\partial P_{g2.1}}_{DC} &= 1 + 2 \cdot P_{g2.1} = 1 + 2 \cdot 40.0057 = 81.012
\end{aligned} \tag{6.3}$$

As it can be seen, the values of dual prices for buses with generation are equal to the derivative of those generators connected. For buses with no generation, the price is slightly above since it has to account for the additional power injected due to the losses during the power transfer.

Finally, the cost of operation has obviously been reduced after connecting both islands as seen in Table 6.4. The difference is significant, representing a reduction of 11.5% in the cost of operation. This example shows the economical benefit of having an interconnection to a neighboring grid as the INELFE interconnection between Spain and France, apart from the technical benefits in terms of frequency stability, power support during faults, etc.

Table 6.4: Costs for Case 2 islands.

	Base Case 2 islands	**2 islands + DC link**
Cost (€/MWh)	4602.23	4102.12

6.5 Runtime comparison

Lastly, the execution times of all the cases with each different initialization combination have been measured. All measurements have been reproduced in the same conditions, repeating 5 times each execution to have a larger sample size, although all the values had low deviation. Results are shown in Figures 6.15 and 6.16, separating the larger case for better visualization.

Overall, the solver seems to be on the same level of performance as Matpower in the cases where a comparison could be made. For smaller cases, Matpower has a slight advantage, while for bigger cases the initialization with a power flow solution greatly improves the performance of the solver. The slack initialization is slower in most of the cases as expected, but in some cases it performs at the same level as the power flow initialization with no slacks. Bear in mind that this option is not expected to be used for improved speed, rather for improved convergence.

An important remark to be made is that the power flow execution time is not included in the runtime of the solver, as it is assumed that the user would be able to calculate it once with the built-in power flow solver, and then use it directly for all the desired studies. Another remark

is that this comparison is made with the Python adaptation of Matpower, while the original implementation runs in Matlab/Octave and should be quicker.

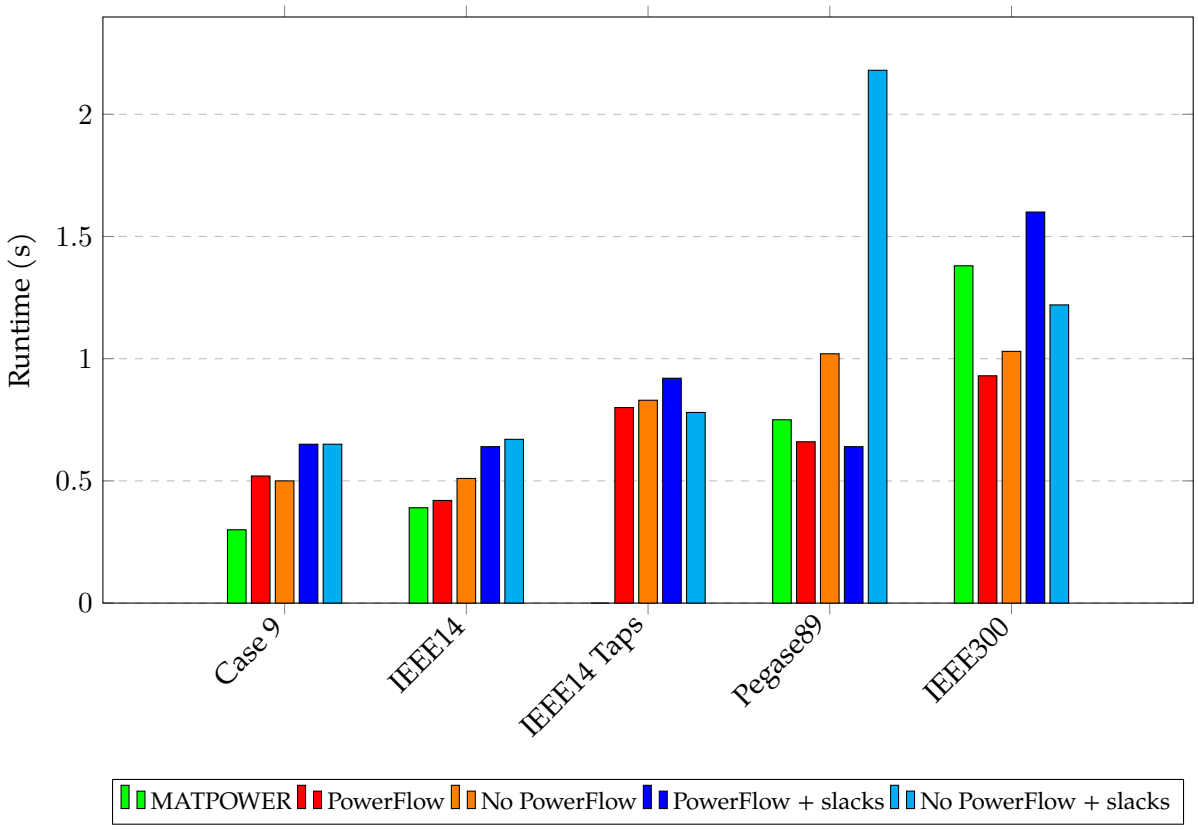

Figure 6.15: Runtime comparison for different cases.

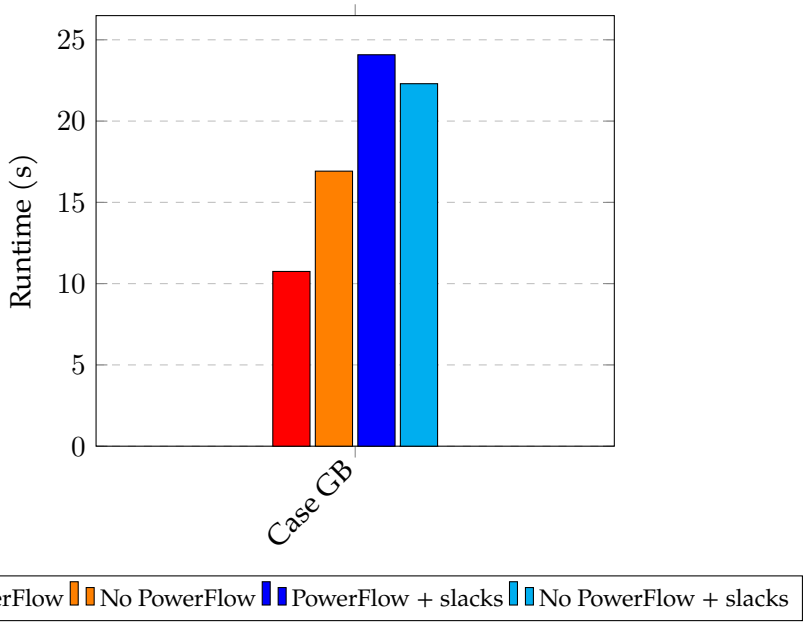

Figure 6.16: Runtime comparison for the GB case.

7 Conclusion

The work developed in this thesis has been focused on the implementation of an AC-OPF solver in GridCal. The results shown in Chapter 6 demonstrate the solver's capabilities and the impact of the features implemented in addition to the preexisting works in the topic.

The standard AC-OPF problem as implemented in Matpower has been successfully replicated, incorporating the option to start from a solution of the power flow with the effect of reducing the necessary iterations to reach convergence in some cases.

The addition of transformer setpoints optimization has also been tested, showing that adding these tap variables in the optimization problem yields a working point that lowers the costs of that found without considering them.

DC links have been added in a simplistic form in the model as a tool to perform quick analysis between islands connected through a DC line.

Reactive power limitations have also been added to the model as a simple relation to also have the possibility to consider such constraints in the optimization problem.

Overall, the implementation of this solver in the GridCal environment can be considered a success. As a tool tailored for Redeia's needs, the whole GridCal package is expected to be used for the future planning of the Spanish electrical grid, which involves the usage of the solver developed in this project, and due to the open-source nature of the package, other TSOs or DSOs could also benefit from it.

7.1 Further Work

The solver holds a lot of potential for enhancement, with many possible improvements of the existing features as well as future additions of new functionalities. Some of the future works that are being discussed are the following:

- Related to raw performance, some intermediate calculations for the gradients and Hessians could be optimized to reduce the time needed to solve the problem. This involves using the sparse structures used in the process of creating the Jacobian and Hessian matrices, although it is not a trivial task and requires a deep understanding of the data structure involved.

- The constraints related to reactive power limitations and DC links have been added in a primitive way, since they were considered as secondary implementations which were nice to have, but still lack some information needed to correctly model them.

- For the reactive power constraints, the grid model will have to include the parametrized equation for their generation curves to be able to obtain the analytic derivatives necessary to include in the gradients and Hessians of the optimization problem.

- For the DC links, the model will need a better implementation of the converters that connect the ends of the line, which will require adequate modelling of the control that they follow, the losses of the link and the power flow equations that are used to calculate the power transfer. It is possible that a full AC/DC solver would be needed to correctly model the power flow of the link.

- The addition of relaxations to the problem is also a possibility if there is a need to solve a great amount of cases for planning purposes, as this approach would increase the speed of the solver. It would come at the cost of precision, but for some cases it could be a good trade-off.

- Time-series studies could be considered by introducing new constraints and elements that have to be modelled considering a time variable, such as batteries and its state of charge, flexibility of loads, generation forecasting for renewable energies...

Acknowledgments

I want to thank Marc Cheah and Josep Fanals, my thesis supervisors from UPC and eRoots respectively, for the opportunity to develop this project in collaboration with Redeia and for their guidance and support through the process. I'd also like to thank Santiago Peñate for his help in developing the tool and integrating it in GridCal in collaboration with Josep and me.

A Environmental, Social, and Gender Impact

The present chapter considers the impact that this project has in the society.

A.1 Environmental Impact

The environmental impact of this project is positive, as it contributes to the development of a tool that will help to optimize the Spanish electrical grid. This optimization will allow the grid to operate more efficiently, reducing the amount of energy that is wasted and minimizing the environmental impact of the grid. By optimizing the grid, it will be possible to reduce the amount of energy that is generated from fossil fuels, which will help to reduce greenhouse gas emissions and combat climate change. In addition, the optimization of the grid will help to reduce the amount of energy that is lost during transmission, which will help to reduce the overall energy consumption of the grid. This will help to reduce the environmental impact of the grid and contribute to the development of a more sustainable energy system.

A.2 Social Impact

The social impact of this project is also positive, as it will help to improve the reliability and stability of the Spanish electrical grid. By optimizing the grid, it will be possible to reduce the risk of blackouts and improve the overall reliability of the grid. This will help to ensure that the grid can meet the energy needs of the population and provide a reliable source of electricity. In addition, the optimization of the grid will help to reduce the cost of electricity for consumers, as it will help to reduce the amount of energy that is wasted and improve the overall efficiency of the grid. This will help to reduce the financial burden on consumers and contribute to the development of a more sustainable energy system.

B Time Planning

Figure B.1 shows the temporal evolution of the various tasks that have constituted the project.

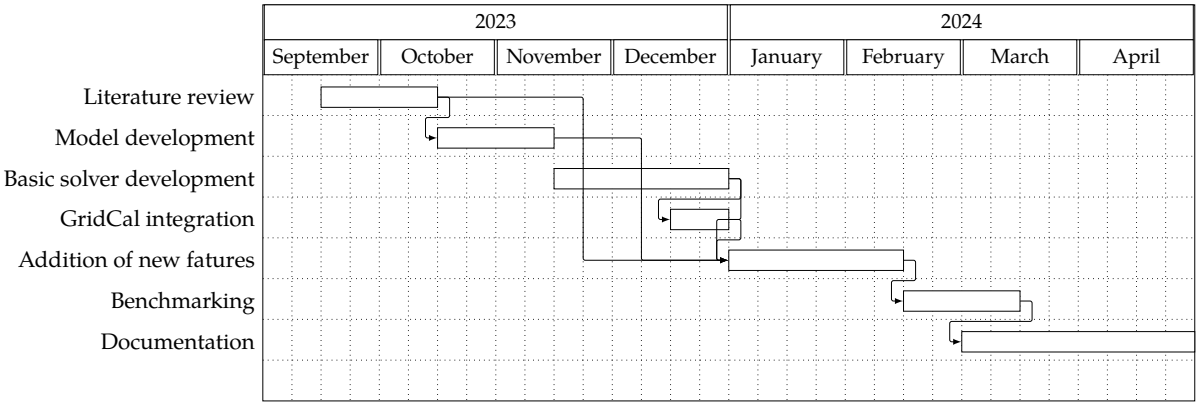

Figure B.1: Gantt Chart of the project.

C Budget

The costs associated to the project are those that have to bee counted for the contract with Redeia, which considering it is an open-source implementation, only consider the consultancy workforce of the employees working on the project. In this case, there has been one full-time development engineer and one part-time supervisor (10-hour week). From Figure B.1 we can see that the project has lasted 32 weeks, although only 26 of them can be considered as working weeks of the project since the documentation work is outside of the contract. The complete costs are detailed in Table C.1.

Table C.1: Total Costs.

Concept	**Unit cost (€/h)**	**Quantity (h)**	**Total (€)**
Development engineer	25.00	1,040	26,000.00
Supervisor	30.00	260	7,800.00
Total			33,800.00

References

[1] S. Chatzivasileiadis, "Optimization in modern power systems," *Lecture Notes. Tech. Univ. of Denmark. Available online: https://arxiv. org/pdf/1811.00943. pdf*, 2018.

[2] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, "Matpower: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Transactions on power systems*, vol. 26, no. 1, pp. 12–19, 2010.

[3] J. Nocedal and S. J. Wright, *Numerical optimization*. Springer, 1999.

[4] W. A. Bukhsh, A. Grothey, K. I. McKinnon, and P. A. Trodden, "Local solutions of the optimal power flow problem," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 4780–4788, 2013.

[5] C. Coffrin and L. Roald, "Convex relaxations in power system optimization: A brief introduction," *arXiv preprint arXiv:1807.07227*, 2018.

[6] S. Peñate, "Gridcal.," *GitHub. Available: https://github. com/SanPen/GridCal*, 2019.

[7] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[8] R. D. Zimmerman and H. Wang, "Matpower interior point solver mips 1.4 user's manual," 2020.

[9] H. Wang, C. E. Murillo-Sanchez, R. D. Zimmerman, and R. J. Thomas, "On computational issues of market-based optimal power flow," *IEEE Transactions on power Systems*, vol. 22, no. 3, pp. 1185–1193, 2007.

[10] H. Abaali, T. Talbi, and R. Skouri, "Comparison of newton raphson and gauss seidel methods for power flow analysis," *International Journal of Energy and Power Engineering*, vol. 12, no. 9, pp. 627–633, 2018.

[11] P. Virtanen *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[12] R. D. Zimmerman, "Ac power flows, generalized opf costs and their derivatives using complex matrix notation," *MATPOWER Tech. Note*, vol. 2, pp. 60–62, 2010.

[13] R. Lincoln, "Pypower 5.1.16," *PYPOWER/api*, 2023.

[14] J. H. Chow, *Time-scale modeling of dynamic networks with applications to power systems*. Springer, 1982, p. 70.

[15] C. Josz, S. Fliscounakis, J. Maeght, and P. Panciatici, "Ac power flow data in matpower and qcqp format: Itesla, rte snapshots, and pegase," *arXiv preprint arXiv:1603.01533*, 2016.

[16] M. Adibi, F. Alvarado, and R. Christie, "Power systems test case archive," *University of Washington*, 1993.

[17] I. Dabbagchi and R. Christie, "Power systems test case archive," *University of Washington*, 1993.