SMART GRIDS: FROM TRADITIONAL TO MODERNIZED RESILIENT SYSTEMS

Víctor Escala García

Josep Fanals Batllori

Pol Heredia Julbe

Roger Izquierdo Toro

Palina Nicolas

SMART GRIDS

Master's degree in Energy Engineering

Master's degree in Electric Power Systems and Drives

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Escola Tècnica Superior d'Enginyeria
Industrial de Barcelona

CONTENTS

1. INTRODUCTION

In modern life, electricity has become a necessity and everyone, households like industries, has to be provided whenever needed. The challenge is that electricity cannot be stored and demand fluctuates. In order to do so, since the early 20th century, the electricity grid from producer to consumer has rapidly increased, building a great network. Nevertheless, even if now the grid is able to provide anyone with electricity, in today's context it is not enough anymore. The growing demand and the climate change crisis pushes toward a more efficient use of energy and greener production. The actual grid, mainly being built a century ago when renewable energy was not in the topic, is long as fossil fuel power plants need to be relatively far from cities and villages and weak in numerical control. This leads to a lot of losses in the transmission and distribution lines, up to 10% [1]. As a response to those issues, the smart grid concept emerged. Its definition varies but overall its aim is to improve and modernize the actual grid in order to improve its efficiency and reduce greenhouse effect in a cost effective way [2]. For doing so, smart grids can use faster data communication thanks to numerical meters, implementation of proximate electricity sources such as household PV, new electricity market for prosumers, wiser selection of high voltage lines, etc. The smart grid concept is relatively new and because testing it in real life is expensive it is first modeled and optimized before being implemented. Moreover, to optimize the conventional grid, this latest has to be well understood.

Many previous works have been conducted, mainly starting from 2008 [3]. They mostly focus on the engineering, energy and computer science aspects. Reports in the field of environmental science are about 7% of all the publications and those relating to economical aspects, although their number is increasing, they remain very low. Studies can be separated in two categories, those related to specific elements of smart grids and those optimizing the whole. In the first case, every element of smart grids is discussed: the power generation with renewable energy sources, transmission and distribution (bidirectional flow), storage, end user, microgrid, market, meters and communications. New technologies can be implemented in order to control and heal the system in an automated way. For example, Automatic Voltage Regulation (AVR) keeps the voltage within the limits, Energy Management System (EMS) ensures stability of every operating point, Automatic Generation Control (AGC) performs optimal load distribution among the generating units, Advanced Metering Infrastructure (AMI) regroups the meters devices, Meter Data Management (MDM) allows fast bidirectional communication of the measures and helps the decision-making, and others like Distribution Management System (DMS), Geographical Information System (GIS), Outage Management System (OMS), Wide Area Management System (WAMS), and Demand Side Management (DSM) [4].

On the other hand, research has been done in order to optimize all those elements [5]. Most of the time, optimization is simplified by focusing on a single objective at a time. The main objectives are energy performance, economical performance or environmental protection. Nevertheless, multi-objectives optimizations are more likely to be accurate. The two main multi-objective optimization models are the weighting factor method and the pareto model, based on the necessary and sufficient conditions logic. Then the decision variables and the constraints are to be added. The decision variables are values that decision-makers have to decide and to do so several multi-criteria methods have been established. For example, Gu et al. wrote a couple of widely accepted evaluation methods

and indicators for CCHP microgrid planning [5]. Within the constraints, we can cite the power balance, generation capacity, transmission capacity between a microgrid and a large grid, location of the microgrid energy storage system,... The minimum operational cost is also a strong constraint. Different algorithms regrouping the objectives, the decision variables and the constraints have been written. The most frequently used is the genetic algorithm, based on Darwin's selection principle, it selects within all the solutions the best ones each generation but others such as the simulated annealing method, the particle swarm optimization and others are also frequently used.

In this work, a simplified conventional network is considered. This initial network, composed of 4 loads, 1 nuclear plant, a dismantled gas plant and an interconnection with an external electricity grid, does not answer the requirement of a basic grid, i.e. providing electricity at all time. The goal is thus to improve this network. The first stage is to analyze the existing grid and point out the weaknesses in order to suggest solutions. The second stage consists of the improvement of the system thanks to the addition of lines and voltage increase. The third phase improves the network by integrating renewable energy sources. The fourth phase rehabilitates the dismantled gas plant and reduces energy dependence by integrating storage and control. Finally, the Smart Grid Architecture Model (SGAM) for the control is designed. For every improvement, a contingency analysis is made, i.e. the network reaction in case one element fails.

2. PHASE 1

A system such as the one displayed in Figure 1 is analyzed. The network operates at the transmission level and feds dispersed demand points that symbolyze distribution grids. The grid has an interconnection with a transmission grid, and at the same time, some power is provided by the nuclear power plant. Initially, there are no renewable power plants nor storage systems, which compromises the security of the grid.

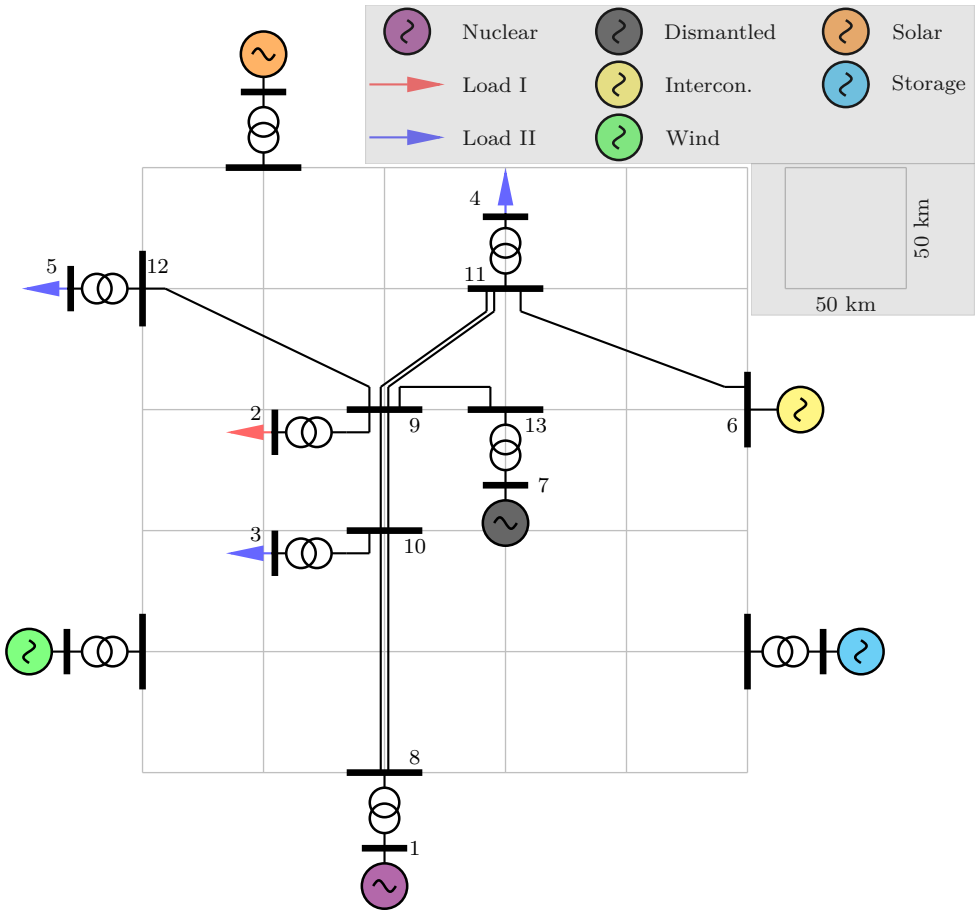

Figure 1. Overview of the network

The first step to analyze the system is to know the demand and the generation profile. In order to model them, the hourly demand and generation data of Spain have been collected [6]. To obtain a typical working day, a statistical analysis has been performed taking into account only the days from the 1st of January to the 31st of March, from Tuesday to Thursday and removing the national holidays. The result then has been normalized. This way, the consumption profile is obtained from the product of the normalized demand and the peak power consumption of 375 MW for load type I and 140 MW for type II. For the generation profile, for simplicity, it has been assumed that the nuclear power plant follows the demand curve, i.e., it is not acting as a constant generator.

Bus / Hour	1	2	3	4	5	6	8	9	10	11	12
0	1.050	0.963	0.989	0.972	0.937	1.000	1.031	0.986	1.000	0.984	0.950
1	1.050	0.974	0.998	0.982	0.952	1.000	1.035	0.995	1.008	0.993	0.963
2	1.050	0.981	1.003	0.988	0.961	1.000	1.037	1.001	1.013	0.998	0.972
3	1.050	0.984	1.005	0.990	0.965	1.000	1.038	1.003	1.015	1.000	0.975
4	1.050	0.985	1.006	0.991	0.966	1.000	1.038	1.004	1.015	1.001	0.976
5	1.050	0.981	1.003	0.988	0.961	1.000	1.037	1.001	1.013	0.998	0.972
6	1.050	0.967	0.992	0.975	0.942	1.000	1.032	0.989	1.003	0.987	0.955
7	1.050	0.938	0.969	0.950	0.905	1.000	1.022	0.965	0.983	0.964	0.920
8	1.050	0.915	0.952	0.931	0.876	1.000	1.014	0.947	0.967	0.946	0.893
9	1.050	0.904	0.944	0.921	0.862	1.000	1.010	0.938	0.960	0.938	0.880
10	1.050	0.900	0.941	0.918	0.856	1.000	1.009	0.935	0.957	0.935	0.875
11	1.050	0.901	0.941	0.919	0.857	1.000	1.009	0.936	0.958	0.935	0.876
12	1.050	0.904	0.944	0.921	0.861	1.000	1.010	0.938	0.960	0.938	0.879
13	1.050	0.906	0.945	0.923	0.864	1.000	1.011	0.939	0.961	0.939	0.882
14	1.050	0.916	0.953	0.931	0.877	1.000	1.014	0.948	0.968	0.947	0.894
15	1.050	0.922	0.957	0.936	0.884	1.000	1.016	0.953	0.972	0.952	0.901
16	1.050	0.925	0.960	0.939	0.889	1.000	1.018	0.955	0.974	0.954	0.905
17	1.050	0.926	0.961	0.940	0.890	1.000	1.018	0.956	0.975	0.955	0.906
18	1.050	0.921	0.957	0.936	0.884	1.000	1.016	0.952	0.972	0.951	0.900
19	1.050	0.903	0.943	0.920	0.860	1.000	1.010	0.937	0.959	0.937	0.878
20	1.050	0.891	0.933	0.910	0.844	1.000	1.005	0.927	0.950	0.927	0.863
21	1.050	0.897	0.939	0.915	0.853	1.000	1.008	0.933	0.955	0.932	0.871
22	1.050	0.921	0.957	0.936	0.884	1.000	1.016	0.952	0.972	0.951	0.900
23	1.050	0.946	0.975	0.957	0.915	1.000	1.025	0.972	0.988	0.970	0.929

Table 1. Voltage profile, in pu, for 24 hours

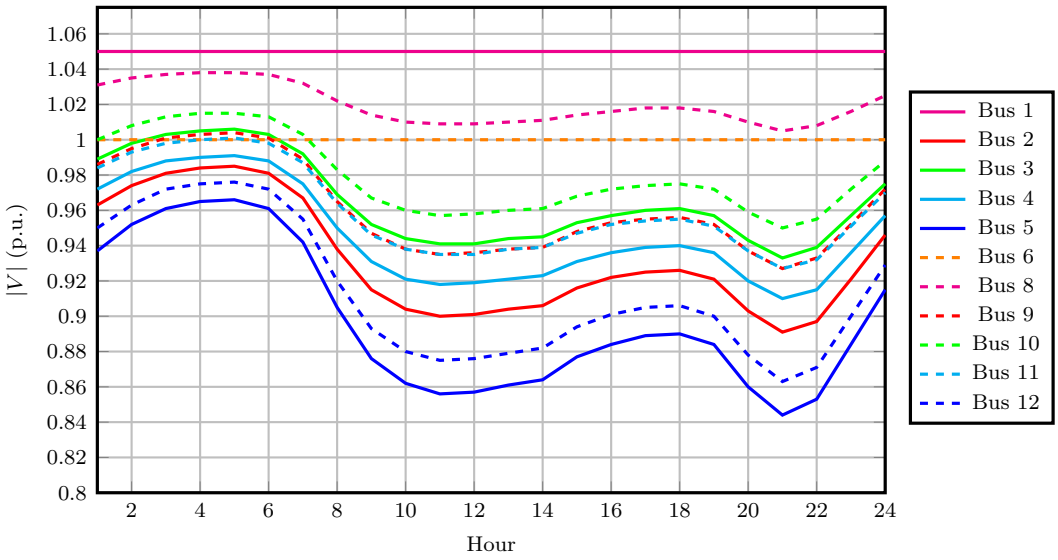

Figure 2. Voltage profile during 24 hours for the initial grid. The low-voltage buses are plotted in solid lines; the high-voltage ones are in dashed lines.

Load / Hour	8-10	10-9	9-11	9-12	11-6
0	26.423	19.999	9.388	30.774	63.562
1	24.385	18.464	8.850	28.312	59.190
2	23.089	17.491	8.517	26.761	56.451
3	22.549	17.087	8.381	26.117	55.319
4	22.458	17.019	8.358	26.009	55.130
5	23.082	17.486	8.515	26.752	56.435
6	25.729	19.475	9.203	29.932	62.065
7	30.615	23.176	10.549	35.928	72.752
8	34.100	25.846	11.559	40.331	80.765
9	35.690	27.073	12.031	42.383	84.443
10	36.296	27.542	12.213	43.173	85.847
11	36.166	27.441	12.174	43.002	85.544
12	35.740	27.111	12.046	42.448	84.558
13	35.480	26.910	11.969	42.110	83.956
14	33.997	25.766	11.528	40.198	80.526
15	33.090	25.069	11.262	39.042	78.435
16	32.569	24.669	11.110	38.381	77.234
17	32.392	24.534	11.059	38.158	76.828
18	33.187	25.143	11.290	39.165	78.657
19	35.865	27.208	12.084	42.610	84.847
20	37.641	28.588	12.620	44.945	88.969
21	36.689	27.847	12.332	43.688	86.759
22	33.143	25.110	11.278	39.109	78.557
23	29.319	22.190	10.183	34.319	69.877

Table 2. Percentual loading of the lines for a full day operation

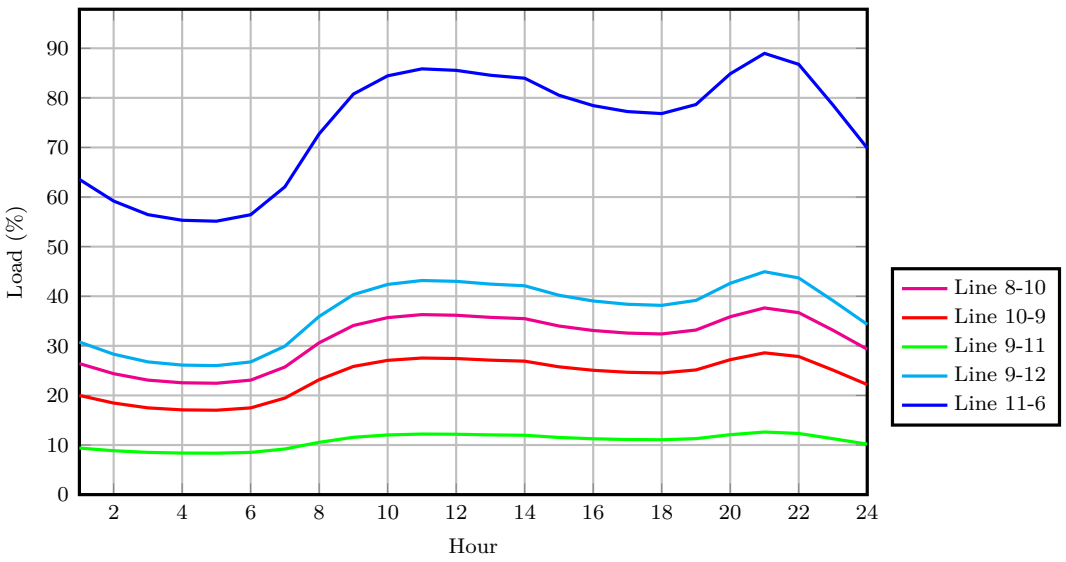

Figure 3. Representation of the percentual loading of the lines during 24 hours

2.1. Operating costs

Regarding the operating costs, some estimations are made in order to assess the influence of importing energy and the impact of faults on lines and transformers.

First, the cost of importing energy depends on the time zone: valley, flat or peak. The analysis that follows considers a working day, which is precisely the date for which the voltages and loading profiles have been shown in Figures 2 and 3 respectively. The cost of importing the energy is mathematically expressed as:

$$C_{imp} = \sum_{k=1}^{n=24} P_{s,k} c(k), \tag{1}$$

where C_{imp} stands for the cost of importing energy for a full day, k denotes the index of a given hour, n the total number of hours in a day, $P_{s,k}$ the energy provided by the slack bus (interconnection point) in MWh at hour k, and $c(k)$ the cost at a certain hour in €/MWh. This last term is equal to 45 €/MWh from 0 to 8 hours, 65 €/MWh from 8 to 10, 14 to 18 and 22 to 24 hours, and 90 €/MWh from 10 to 14 and 18 to 22 hours.

Equation 1 can be treated as a weighting sum. With the generation data obtained from the timeseries power flow, the total importing cost of importing energy becomes 418753.03 €/day, or about 152.84 M€ for a full year. It is important to note that the study related to the cost of importing energy is decoupled from the fault analysis. This is not a hundred percent realistic, because it could be that a switch trips and hence a line or a transformer is disconnected. Then, it could happen that the interconnection has to provide more power. However, since the probabilities are extremely low, they are discarded when computing this cost.

On the other hand, there are the costs due to faults in transformers or lines. About 0.05 failures per km and year are expected in lines, while transformers are meant to fail 0.15 times a year. The penalty for not providing energy is 180 €/MWh. Given that the length of the lines has an impact on its probability of failure, Table 3 shows the length and the subsequent failures per year.

Line	**Length (km)**	**Failures/year**
8-10	100.00	5.00
10-9	50.00	2.50
9-11	70.71	3.54
9-12	111.80	5.59
11-6	111.80	5.59

Table 3. Length and failures per year of all active lines

Figure 3 shows that the line connected to the interconnection point operates at a high load. It is critical to note that if line 8-10 fails, the slack should provide all power, but this would result in exceeding the thermal capacity of the line. Thus, if line 8-10 fails, no power can reach the loads.

In the case of line failures, there is a total disconnection time of 2.5 hours, while for transformers it is 8 hours. The expected time that an element will be disconnected in a year is found by multiplying the aforementioned disconnection time by the number of failures that take place during a year. Table 4 displays the yearly disconnection time and explains the consequences spotted by running

the power flow. This will allow to estimate the penalties due to disconnection.

Element	Disconnection time (h)	Consequences
Line 8-10	12.50	No load served - divergence
Line 9-10	6.25	No load served - divergence
Line 9-11	8.85	Loads at buses 2, 3 and 5 unserved
Line 9-12	13.98	Load at bus 5 unserved
Line 11-6	13.98	No load served
Trafo 1-8	1.20	No load served - divergence
Trafo 2-9	1.20	Load at bus 2 unserved
Trafo 3-10	1.20	Load at bus 3 unserved
Trafo 4-11	1.20	Load at bus 4 unserved
Trafo 5-12	1.20	Load at bus 5 unserved

Table 4. Disconnection time and consequences of losing each element

Once the unserved loads and the associated disconnection times are known, the next step has to do with applying the penalty as follows:

$$C_{discon} \approx \sum_{i=1}^{10} \overline{P}_{uns,i} t_{discon,i} C_p, \tag{2}$$

where C_{discon} is the total disconnection cost, i represents the index of the line or transformer with a total of 10 elements proned to be disconnected (see Table 4), $\overline{P}_{uns,i}$ is the mean unserved power, $t_{discon,i}$ the disconnection time, and C_p the penalty cost to apply. Equation 2 is an approximation in the sense that the unserved power varies according to the time of the day. To not overcomplicate the problem, it has been decided to pick a representative value such as the average.

The application of Equation 2 yields a total yearly penalty cost of 4.99 M€. For the most part, it is due to the disconnection of lines. Meshing more the system would decrease this cost, but on the other side, it would increase the investment cost. Hence, there is a trade-off between cost and reliability.

All the calculations related to costs have not set an inferior limit to the voltages. However, some of them are likely to be unacceptable in reality. The project will proceed to discuss solutions to this issue in the following phases.

2.2. Problem identification

The network modeled presents some serious issues. First, in case of fault, the demand cannot be covered. The network is a ramified line but does not have any interconnection within the system. If a fault occurs, the two branches are not connected and have to support the demand of the remaining part on its own. In the case of the nuclear power plant, it cannot produce enough energy to fulfill all the demand and in the case of the interconnection, if it was to cover all, it would be overloaded.

This leads to the second problem the network faces, there is a risk of overloading. This may happen in case of fault or if the the nuclear power plant shuts down because there is no other source of generation. This could lead to burning hence security and material damage issues. A third drawback is the high impact of the interruptions. As many line are single lines and there are

no multiple connections, only ramifications, a fault has a high chance to directly disconnect the network. Finally, the voltage cannot be kept constant enough. It is usually accepted to fluctuate 10% around the nominal 1 p.u. while in the current transmission network, the voltage reaches almost 0.8 p.u..

2.3. Solution suggestion

As some of the lines present some overloading and demand coverage problems, we suggest improving lines to better ones which are able to transport more power. In order to do so, the critical lines could be changed from single to double lines and/or even change the conductors to thicker cables which allow a larger amount of power flow. Another approach would be to add more lines to the grid to overcome the demand coverage but we must also be aware that when adding new lines to the system we are also increasing the possibility of line failures which may affect interruptibility and cause economic losses to the system.

On the other hand, adding generation points to the system would also help overcome the stated problems in the previous point. If power is more accessible in different locations, the demand can be fulfilled from various points without saturating the most critical lines while evenly distributing the generation. Finally, another solution could be to change the 220 kV existing lines to 400 kV ones in order to allow these to transport higher amounts of power. With this change, only the amount of power transported would be around three times higher than the current one. However, it has to be taken into account that there would have to be an additional transformer to adapt the 220 kV from the interconnection to 400 kV, and the rest of the transformers would have to be replaced to match nominal voltages.

3. PHASE 2

This chapter considers upgrading the system by installing additional lines. In the previous analysis, it was visible that the disconnection of specific lines resulted in an unsolvable problem. Often, no solution was obtained as a result of an exceedingly high demand compared to the available generation. The criteria $N-1$ was therefore not met, yet it is usually treated as a requirement by control center operators [7], [8].

The central aim of this chapter is to reach a robust grid, where a single failure does not compromise the full system. Several solutions were proposed, such as meshing the network, rising the voltage level, etc. This chapter is devoted to exploring the implications of such proposals. First, a more theoretical explanation describing the formulation is presented, and then, the analysis of results takes place. These results include electrical magnitudes as well as economic data. An optimal configuration is finally provided.

3.1. Addition of lines

Recall that the original system depicted in Figure 1 was for the most part a radial system, in the sense that there was no redundancy in the connection of lines. The operating costs are non-negligible since a significant amount of energy cannot be delivered during faults. To combat this issue, the installation of various new lines (in red) is examined, as shown in Figure 4.

By installing new lines, an additional installation cost has to be considered, but a reduction on the operating costs is expected. It is worth mentioning that not all proposed new lines ought to be installed. Rather, by adding one or two of them, the system could already be operating under the desired conditions. There is a cost-benefit tradeoff, so only a few specific new lines will be selected.

3.1.1. Formulation

The formulation of the topology deserves special attention. Mathematically, it is defined as:

$$\{\mathcal{N} \in \mathcal{P}([n]) \mid |\mathcal{N}| = k\}, \tag{3}$$

where $\mathcal{N}$ denotes a subset, that is, possible topology out of the full set of configurations $\mathcal{P}([n])$, $[n] = \{\mathbf{1}_{\sigma_1}, \mathbf{1}_{\sigma_2}, ..., \mathbf{1}_{\sigma_n}\}$ represents the set of elements that cause variations on the topology, and k symbolizes the total amount of possibilities. In the contingency analysis, the set $[n]$ is a function of the state of the elements (mainly, lines). These can be in service or out of service, so can be regarded as boolean variables. The employed notation, of the form $\mathbf{1}_{\sigma_i}$, is 1 if the indicator variable σ_i holds true; otherwise, it is 0. The indicator variable, in the case under study, is simply the state of each particular line.

Note that the whole set $\mathcal{P}$ can be further divided into the set composed of the original lines $\mathcal{A}$, and the set of new lines $\mathcal{B}$. Let j indicate the number of lines in the original set $\mathcal{A}$, and $n-j$ the amount of lines in the new set $\mathcal{B}$. In the particularized case shown in Figure 4, $j = 6$ and $n-j = 8$.

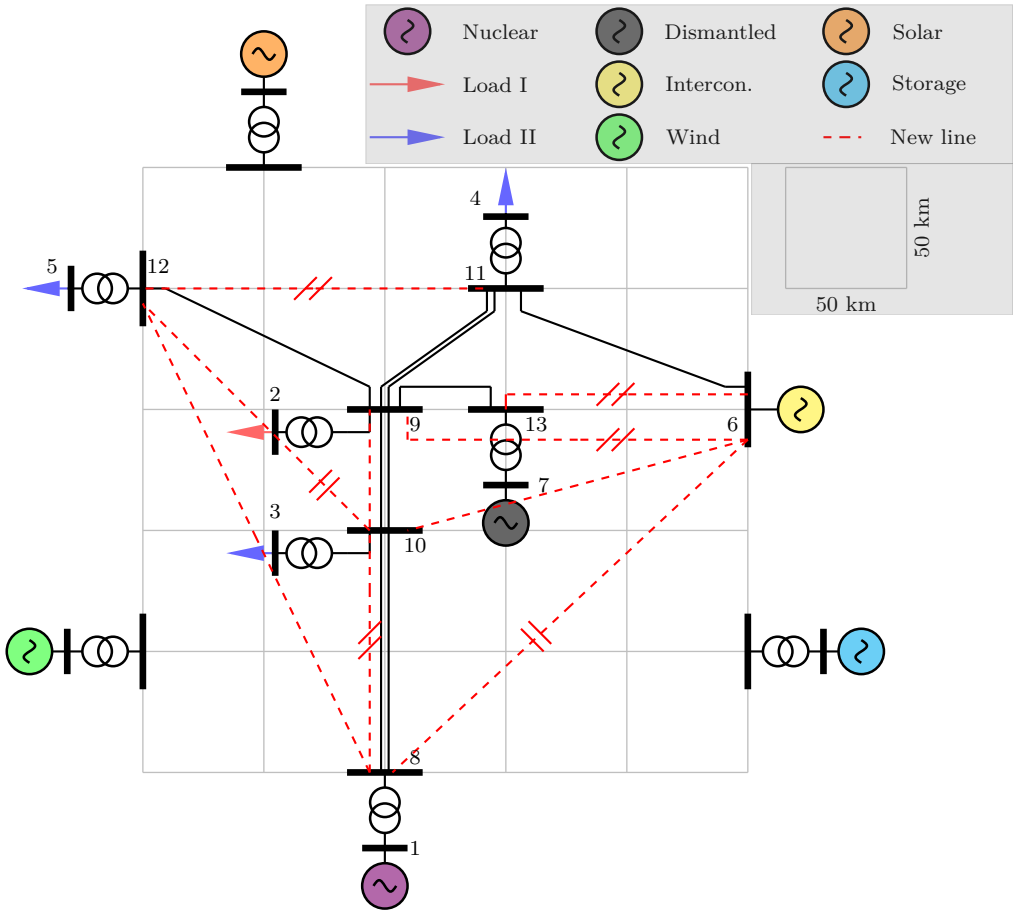

Figure 4. Overview of the network with the addition of lines. Double line if double circuit.

Thus, these subsets are expanded and explicitly become:

$$\begin{cases} \mathcal{A} = \{\mathbf{1}_{\sigma_1}, \mathbf{1}_{\sigma_2}, ..., \mathbf{1}_{\sigma_j}\}, \\ \mathcal{B} = \{\mathbf{1}_{\sigma_{j+1}}, \mathbf{1}_{\sigma_{j+2}}, ..., \mathbf{1}_{\sigma_n}\}. \end{cases} \tag{4}$$

To perform the contingency analysis, we consider that one of the original lines suffers a fault. For instance, if line i ends up disconnected, $\sigma_i \leftarrow$ `false` while the rest of original lines $\sigma_r \leftarrow$ `true` for $r = \{1, 2, ..., i-1, i+1, ..., j\}$.

On the other hand, in the subset $\mathcal{B}$ it is not straightforward to estimate which lines should be connected or disconnected. Perhaps installing a single additional line is enough to meet the $N-1$ criteria, or maybe more lines are required. Hence, it has been decided that all permutations have to be analyzed. Since σ are boolean variables, the total number k of contingencies to simulate are:

$$k = j2^{(n-j)}, \tag{5}$$

so if $j = 6$ and $n - j = 8$, this results in $k = 1536$ different topologies. Thus, a total of 1536 timeseries power flows are computed.

Security analysis in power systems has been traditionaly based on the usage of the DC power flow [9], [10]. It offers the advantage of being non-iterative (and consequently, fast), and allows to determine the influence of faults with the usage of transmission factors. However, its solution is

just an approximation. Since the amount of situations remains reasonable, it has been decided to solve them with the typical Newton-Raphson with the default precision of $1 \cdot 10^{-8}$ MVA [11].

The general procedure to perform the contingency analysis is shown in Algorithm 1.

Algorithm 1: Pseudocode to solve the contingencies

Input: `net` initialized class, j, n
Output: stored results

Generate permutations $\forall \boldsymbol{\sigma}_g$ where $g = [1, 2, ..., 2^{(n-j)}]$
for $i = [1, 2, ..., j]$ **do**
    $\sigma_i \leftarrow$`false`
    $\sigma_r \leftarrow$`true`, where $r \neq i$ and $r \leq j$
    $\mathcal{A} \leftarrow \{\mathbf{1}_{\sigma_1}, \mathbf{1}_{\sigma_2}, ..., \mathbf{1}_{\sigma_j}\}$
    for $g = [1, 2, ..., 2^{(n-j)}]$ **do**
        $[\sigma_{j+1}, \sigma_{j+2}, ..., \sigma_n] \leftarrow \boldsymbol{\sigma}_g$
        $\mathcal{B} \leftarrow \{\mathbf{1}_{\sigma_{j+1}}, \mathbf{1}_{\sigma_{j+2}}, ..., \mathbf{1}_{\sigma_n}\}$
        $\mathcal{N} \leftarrow \mathcal{A} \cup \mathcal{B}$
        `pandapower.timeseries.run_timeseries(`$\mathcal{N}$`,net)`
        Store results

Out of all results extracted, an analysis procedure is required in order to determine which configuration meets the $N-1$ criteria. The requirements are listed below:

- The losses in the lines have to stay below 2%.
- The voltages have to be between a range of ±10%.
- All lines have to remain below 80% of its maximum capacity.

Apart from these requirements, an additional criteria has to do with the costs. These will be critical when choosing the best configuration out of all possible ones. Table 5 displays the costs for the lines, depending on their representative voltage level and the number of circuits.

Type	Mean (€)	Min-max range (€)	Median (€)
380-400 kV, 2 circuits	1060919	579771 - 1401585	1023703
380-400 kV, 1 circuits	598231	302664 - 766802	597841
220-225 kV, 2 circuits	407521	354696 - 461664	437263
220-225 kV, 1 circuits	288289	157926 - 298247	218738

Table 5. Network infrastructure costs for the lines. Data from [12].

It has been decided to calculate the costs with mean value.

3.1.2. Results

Out of all results extracted, an analysis procedure is required in order to determine which configuration meets the $N-1$ criteria. The prechosen configurations are the ones that ensure that in spite of the disconnection of a given element, the power flow remains feasible and the design criterias are met. A total of 39 have satisfied all requirements. Table 6 shows the top 10 cheapest topologies, according to the costs in Table 5.

Identifier	New lines	Infraestructure cost (M€)
19	[6-13, 6-10, 8-9, 10-12]	359.50
214	[6-13, 6-10, 11-12, 8-9]	362.99
77	[6-13, 8-9, 10-12, 9-6]	375.04
49	[6-13, 11-12, 8-9, 9-6]	378.54
80	[6-13, 6-10, 11-12, 8-9, 10-12]	420.63
189	[6-13, 6-10, 8-9, 10-12, 9-6]	420.63
45	[6-13, 6-10, 8-9, 10-12, 8-12]	423.96
70	[6-13, 8-9, 10-12, 9-6]	424.12
157	[6-13, 6-10, 11-12, 8-9, 8-12]	427.46
250	[6-13, 11-12, 8-9, 10-12, 9-6]	436.17

Table 6. Best configurations with the additional lines

The results indicate that the differences in the infrastructure costs are rather minimal. However, installing only four additional lines tends to be the best option. Line 6-13 has been found necessary to ensure a proper operating state of the system under all near-optimal configurations. This makes sense considering that the interconnection usually provides most of the power. Hence, a strong connection to the network is convenient. In fact, the best topology includes two lines connecting the slack bus to the rest of the system are present.

By ensuring that the $N-1$ criteria is met, we guarantee that all power will be provided independently of which element fails. In reality, this matter may be a bit more complex as two lines could fail simultaneously. Nevertheless, the associated cross-probability is extremely low, and hence, neglected. This implies that no operating costs have to be calculated; it is known beforehand that they will be null as all power is provided for the optimal configuration.

3.2. Rising the voltage level

3.2.1. Description

While installing more lines is a technically feasible strategy, it may not be the most optimal in terms of costs since the voltage level of the grid could be increased. The price to pay for such a change is the replacement of the initial transformers of 25/220 kV and 36/220 kV, for transformers that raise the voltage up to 400 kV (thus, 25/380 kV and 36/380 kV).

In practical terms, raising the voltage may not be as simple as replacing four transformers. For instance, the insulators and the switchgear would have to be changed as well. In the worst scenario, the lines in the towers would have to be more separated, which could cause a replacement of the whole set of supports. To keep the problem arguably simple, it is taken into account that the cost of the transformer station is proportional to the voltage — as can be deduced in [12].

The benefits of increasing the votlage become obvious when observing that the power in a three-phase system such as this one is given by:

$$S = \sqrt{3} V_{ll} I, \tag{6}$$

where V_{ll} identifies the line voltage. With this simple relationship, if it is assumed that S remains constant, and V_{ll} increases by a factor of $380/220 \approx \sqrt{3}$, the current would decrease by approx-

imately a factor of $\sqrt{3}$. This means that the active power losses, which are proportional to the square of the current, are roughly reduced by a factor of 3.

Regarding the costs, representative values in €/km are gathered in Table 7. While there is a certain dependence on the type, a generic substation could be taken as reference. Thus, a cost of 42627 €/kV is chosen as an indicator. It is considered that all elements in the substation would have to be replaced to adapt the new voltage level of 380 kV. If there are four substations to be replaced, the total cost for doing so becomes 64.79 M€.

Type	Mean (€/kV)	Min-max range (€/kV)	Median (€/kV)
All substations	42627	24994 - 55508	36755
All GIS substations	46237	29837 - 56017	37449
AIS with 9+ bays	44008	28838 - 56157	41080
AIS with 5-8 bays	35593	19936 - 37251	29021
AIS with 1-4 bays	33192	20276 - 48319	26628

Table 7. Network infrastructure costs for the substation. Data from [12].

3.2.2. Results

The same amount of contingencies as before are computed. Out of the 1536, this time more permutations have become feasible (52 in total). Of course, raising the voltage has a positive effect in this regard, as previously hypothesized. In any case there is a trade-off situation: while the kilometers of new lines may decrease, there is an additional cost due to the new substations. It has to be assessed if the saving due to raising the voltage outweights the cost of installing new substations. Table 8 shows such results for the 10 cheapest options, where the cost of the substations are independent of the topology.

Identifier	New lines	Lines (M€)	Transformers (M€)	Total (M€)
163	[6-13, 8-9, 10-12]	313.92	64.79	378.71
133	[6-13, 11-12, 8-9]	317.41	64.79	382.20
235	[6-13, 8-9, 8-12]	320.75	64.79	385.54
30	[6-13, 8-12, 6-8]	346.07	64.79	410.86
19	[6-13, 6-10, 8-9, 10-12]	359.50	64.79	424.29
214	[6-13, 6-10, 11-12, 8-9]	362.99	64.79	427.78
58	[6-13, 6-10, 8-9, 8-12]	366.33	64.79	431.12
77	[6-13, 8-9, 10-12, 9-6]	375.04	64.79	439.83
169	[6-13, 8-9, 10-12, 8-12]	378.38	64.79	443.17
49	[6-13, 11-12, 8-9, 9-6]	378.54	64.79	443.33

Table 8. Economic results of replacing the substations

A similar conlcusion as before is reached. A new line that connects the interconnection with the dismantled plant is required to meet the planning requirements and achieve an optimal topology. Notice that thanks to the increase in the voltage level, only three new lines should be installed in the optimal configurations. This allows to reduce the cost of these new lines up to 313.92 M€ in the best scenario. However, the extra cost due to the replacement of the transformers implies that the total cost of 378.71 M€ surpasses the minimum cost indicated in Table 6, which is equal to 359.50 M€.

Therefore, even if both options do not differ much in costs, it would still be better to operate around the nominal voltage of 220 kV. This does not mean, however, that installing more lines is the only feasible solution to confront the technical issues. This may have been true some decades ago, but nowadays, power systems can benefit from the penetration of renewables. Renewable sources of generation can be connected near the loads, which allows to reduce the losses and improve the efficiency of the system. A scenario with renewables is covered in the following chapter.

4. PHASE 3

This chapter covers the inclusion of renewables into the initial grid. It has been verified that the system, as it is configured in the beginning, lacks resiliency. One of its major issues is the presence of only two generation units placed far from the loads. This causes some lines to exceed their loading capacity, while voltages' absolute value may be well below 0.9 p.u.. Adding new lines is a possibility, or rather, a must in order to meet the $N-1$ criteria. However, it is hypothetized that the performance of the system can still improve with the installation of a wind farm and a PV power plant.

To verify the above-presented hypothesis, the chapter first presents a general overview of the system, along with the sizing of the renewable power plants. Then, the optimal configuration found in the previous chapter is analyzed with the inclusion of renewables to verify the improvement. Finally, a contingency analysis takes place to assess the $N-1$ criteria.

4.1. Overview

Figure 5 presents the general scheme of the system with the solar and wind plants connected through two parallel lines to nearby buses. Again, new lines that could become suitable for ensuring the robustness of the system are drawn.

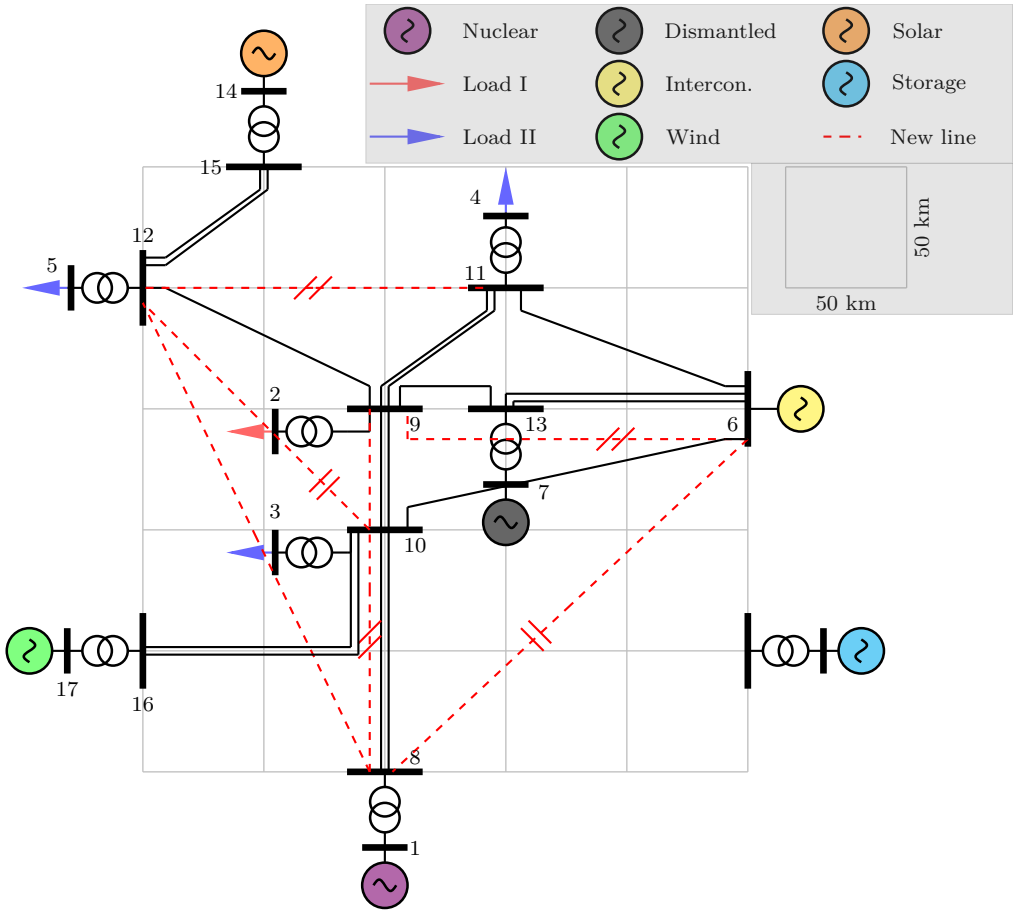

Figure 5. Overview of the network with renewables and the potential addition of lines

The optimal configuration with no renewables, as shown in Table 6, requires the installation of four additional lines: 6-13, 6-10, 8-9, 10-12. The first two are pretty much found in every other topology, as they represent a link with the interconnection. Since the interconnection can theoretically provide any power, it is a good idea to increase the number of associated lines to it. Consequently, lines 6-13 and 6-10 are not considered as potentially attractive new lines, but as already connected lines.

4.2. Renewable integration

In this project, the integration of renewables takes place in the form of solar and wind. The characteristics of each one are particularly detailed. The goal is to characterize their hourly variation of power in relation to the availability of the natural resources. Despite the partial unpredictability that renewables suffer, a representative day in terms of solar irradiance and wind speed has been selected.

4.2.1. Solar PV

First, the PV panels cover an area of 60 ha. This, and some other relevant information is captured in Table 9.

Magnitude	Value	Units
Location	(40.8N; 0.48E)	-
Total area	60	ha
System efficiency	20.7	%
Cell type	Monocrystalline	-
Panel peak power	405	Wp
Panel area	1.95	m^2
Open circuit voltage	37.23	V
Short circuit current	13.87	A

Table 9. Technical characteristics of the PV plant with its panels 405W Deep Blue 3.0 JA Solar Mono [13]

The location has a certain importance to determine the incident irradiation, which is extracted from PVGIS database [14]. In reality, the solar power plant would be located in Tortosa, Catalunya, Spain, according to the given coordinates. The chosen PV panels are of monocrystalline type. Monocrystalline cells generally offer larger efficiencies than polycrystalline cells [15], and although they become more expensive, no cost limitations have been set.

Let n_p denote the total number of panels, G the irradiance, η the efficiency and S_p the area of a single panel. Then, the total generated power follows:

$$P_{PV} = n_p G \eta S_p. \tag{7}$$

It has to be noted that the irradiance G is varying along the day, and it generally has a peak around noon. The data have been selected for the 15th of February. There are a total of $n_p = 63692$ panels as it has been assumed that the 60 ha surface is the useful one. Figure 6 shows the hourly irrandiance and power of the PV plant. Clearly, they are proportional, just like it has been already shown in Equation 7.

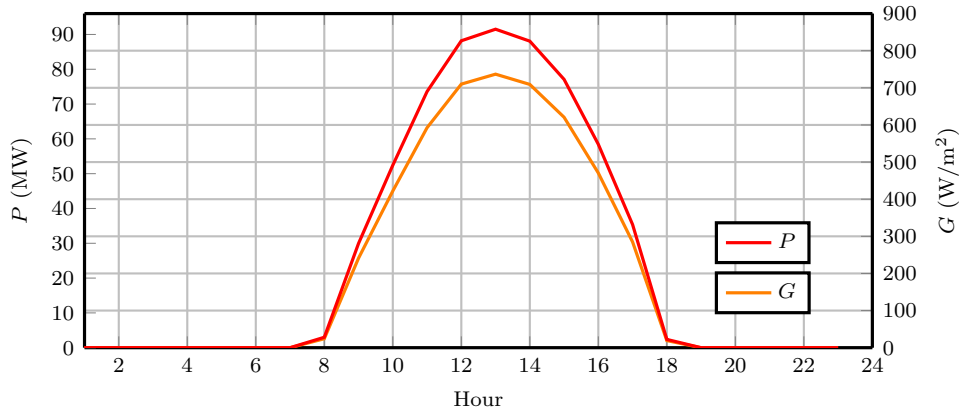

Figure 6. Irradiance and power from the PV plant along a representative day. Data from [14].

The peak in power takes place at hour 13, more or less as expected, and the curves follow a profile that resembles a Gaussian bell. The maximum power generated by the PV plant is 91.5 MW. Although this is a respectable amount, it is roughly one tenth of the load peak. As a consequence, results are not expected to change signficantly. Most likely we will observe slight improvements.

4.2.2. Wind

On the other hand, wind power is proposed as the second renewable alternative. The chosen wind turbines are the Siemens-Gamesa G132-5.0MW. They are rather voluminous turbines, with a rotor diameter of 132 m. Table 10 shows their most notorious technical characteristics.

Magnitude	Value	Units
Location	(42.28N; 3.16E)	-
Total area	80	ha
Nominal power	5.0	MW
Cut-in wind speed	1.5	m/s
Rated wind speed	13.0	m/s
Cut-out wind speed	27.0	m/s
Swept area	13685	m^2

Table 10. Technical characteristics of the wind power plant with its turbines Gamesa G132-5.0MW [16]

The power provided by a wind turbine is given by:

$$P_{wt} = \frac{1}{2}\rho A C_p v^3, \tag{8}$$

where ρ is the air density of approximately 1.225 kg/m^3, A stands for the swept area, C_p is the power coefficient limited to 16/27 (known as Betz's limit [17]), and v represents the wind speed. If the power P_{wt} exceeds the nominal power of the turbine, some power will be curtailed so as not to surpass the machine's ratings. In normal operation, the power coefficient is likely to take values around 0.4.

The available area to install the wind turbines is 80 ha. Nonetheless, wind turbines have to be generously separated one from the others. It has been decided that wind turbines should be separated by a minimum of 5 diameters. Similar separations are adopted in the literature [18]. Thus, a total of two wind turbines are installed. The nominal power of the wind farm becomes

10 MW, which is significantly lower than PV. Despite that, this should not come as a surprise since wind power has a relatively low power density [19].

With this, Figure 7 shows the evolution of the wind speed and the total generated power by the wind farm in an orientative day.

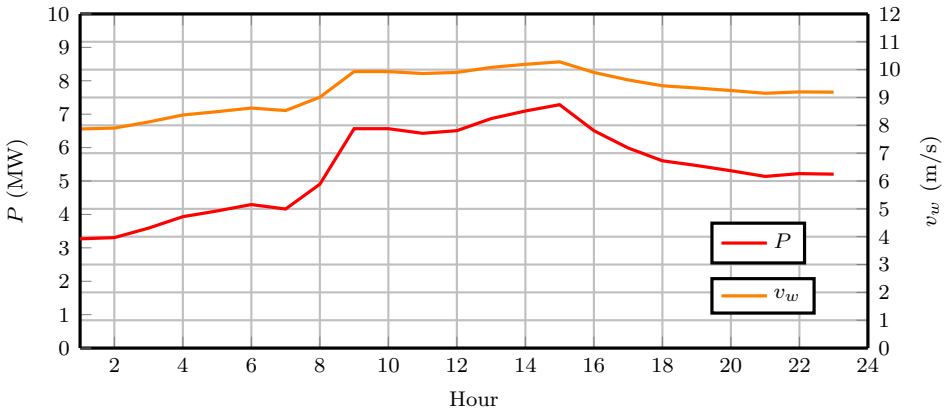

Figure 7. Wind speed and output power from the wind farm. Data from [20].

Notice the cubic relationship between the wind speed and the power. Small increases in wind speed provoke large variations in the otuput power.

4.3. Base case analysis

This base case analysis refers to the assessment of the influence the installation of renewables has on the results. It is compared to the system analzed before with lines 6-10 and 6-13 permanently added. The goal is not to deduce if the grid meets the $N-1$ criteria (most likely some other lines would have to be added), but rather compare if the voltages, the loadings of the lines and the losses suffer a certain improvement.

To perform the comparison, the most extreme values have been gathered. They are shown in Table 11 for both cases (with and without renewables).

Attributes	**Without renewables**	**With renewables**
V_{min} (p.u.)	0.962	0.968
V_{max} (p.u.)	1.050	1.050
Max. load (%)	41.65	40.76
Max. losses (MW)	14.54	14.43
Correct operation?	Yes	Yes

Table 11. Main results to compare between the grid with and without renewables

The results suggest that generally speaking there is little difference in installing or not renewables. The inclusion of the wind farm and the PV power plant cause the minimum voltage to rise a bit, the maximum loading of the lines to decrease about 1% (although without contingencies it is already low) and the maximum power losses experience a tiny reduction. In the two scenarios, the maximum voltage remains 1.05, which indicates there are no overvoltages. Since all magnitudes are inside the allowed limits, the system is operating correctly in both cases.

Adding renewables to the system comes at a cost. First, there has to be an investment of capital for deploying the wind turbines and the PV panels, apart from power electronics equipment and ancilliary components. Then, a couple of new power lines are added to interconnect them to the system. The slight improvement on the electrical magnitudes will most probably fail at justifying this investment.

However, it is important to note that the major advantage of installing renewables is in its environmental impact. If the power from the interconnection is assumed to come from coal and gas power plants, then installing renewables could become a sensible option. This is especially true when taking into account the actual costs of carbon emissions. Currently at 81.65 €/tonne, they have increased by a factor of 3 during the last year [21].

4.4. Contingency analysis

A different story would be the operation with contingencies. Here it is valuable to find out if the presence of distributed renewable generators can help at having to install less extra lines. Hence, the cost could be reduced.

The contingency analysis has been performed similarly to the ones in Tables 6, 8. We take a set of lines that can be potentially installed, and a set of initially powered lines. Contingencies are caused in this latter set to study which new lines have to be connected to meet the requirements. Then, the topologies that ensure the correct operation of the system in spite of which line fails, are selected and their associated costs are calculated. The top 10 most convenient configurations are presented in Table 12.

Identifier	**New lines**	**Infraestructure cost (M€)**
24	[8-9, 10-12]	402.71
12	[8-9, 9-6]	406.21
46	[11-12, 8-9]	406.21
8	[8-9, 8-12]	409.54
0	[11-12, 8-9, 10-12]	463.84
4	[8-9, 10-12, 9-6]	463.84
37	[11-12, 10-12, 9-6]	463.84
63	[8-9, 10-12, 8-12]	467.18
20	[11-12, 8-9, 9-6]	467.34
38	[11-12, 8-9, 8-12]	470.67

Table 12. Best configurations with the additional lines

Again, the optimal configuration is the one where apart from lines 6-10 and 6-13, lines 8-9 and 10-12 are added. This coincides with the conclusion we reached in Table 6. Nevertheless, the costs are now higher due to the addition of two new lines to connect the wind and the solar power plants to the system as such. Introducing renewables is not enough to avoid installing the same lines.

In case lines 8-9 and 10-12 are added, the operational costs would be practically zero because we would be meeting the $N-1$ criteria. To summarize, installing renewables implies an additional cost for the system operator. With time, this could be compensated with the savings in carbon emissions.

5. PHASE 4

While the previous phase included renewables to mitigate the technical problems, another possibility has to do with installing storage systems, rehabilitating a dismantled plant, among others. Certainly, renewables have not been enough to solve the issue of requiring a stronger interconnection. The same number of added lines have been needed in order to meet the $N-1$ criteria. Hence, it is convenient to evaluate if other elements are capable of improving the energetic independence of the system under study, and at the same time, if it becomes economically appealing.

This chapter is structured in the following manner. First, we detail the characteristics of the storage unit, its operation mode and the expected impact on the system. Then, the dismantled plant is evaluated. Focus is placed on its potential environmental impact rather than on its technical aspects. Once these elements have been defined, results are extracted by running the simulation, including the contingency analysis.

Just as an overview, Figure 8 displays the system with the new elements under consideration.

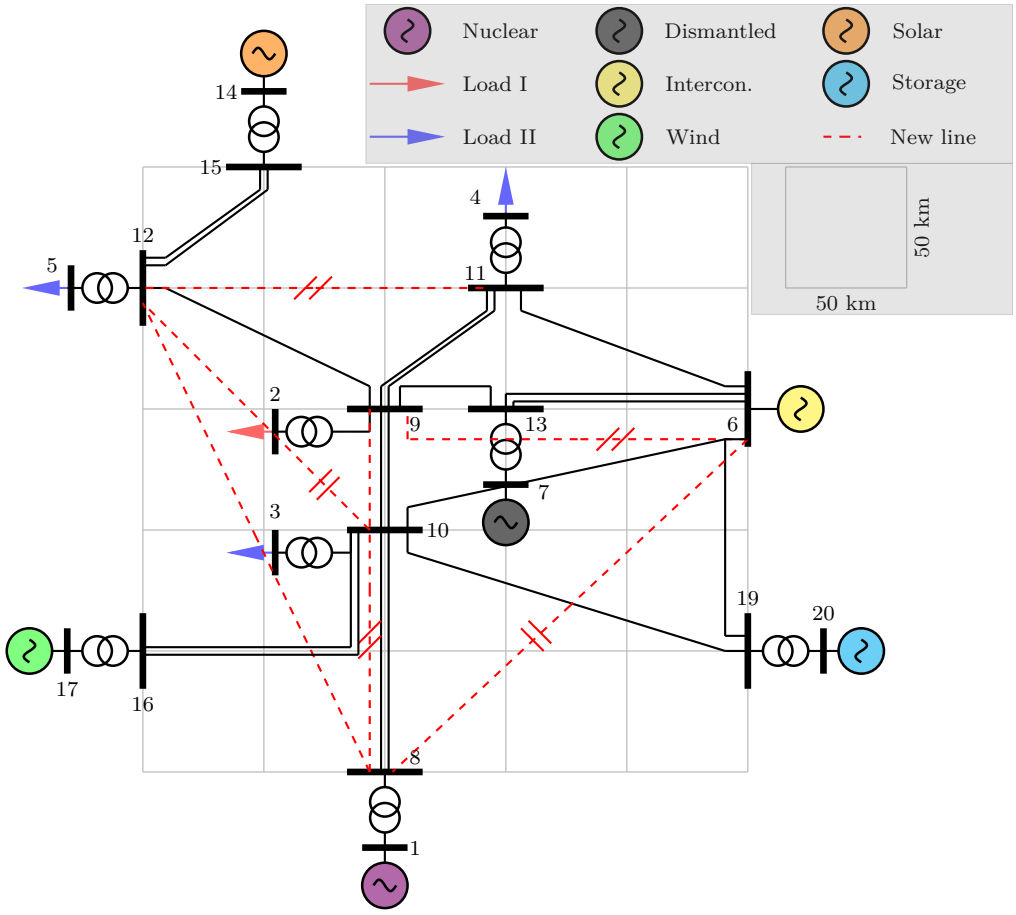

Figure 8. Overview of the network with renewables and the potential addition of lines

5.1. Storage

Storage devices have been conceived as one of the cornerstone class of elements meant to provide flexibility to smart grids [22]. Perhaps in relation to this, electric vehicles (EVs) have been ex-

tensively promoted. Apart from the direct impact on decarbonizing mobility, they can act as a source/sink of energy, which also has an appealing influence on power systems [23]. One of the largest barriers towards adopting batteries has to do with its high investment cost. Although the cost of batteries has been steadily declining for the last two decades [24], there is still a long way to go. Part of the intention regarding this section is to analyze if storage exerts a positive effect on the power flow of the power system under study.

Energy storage options can take many forms. For instance, fast-response devices such as supercapacitors provide peaks of power, yet their accumulated energy is rather low. The same characteristics apply to flywheels, although they store the energy mechanically, not electrically. Another option, likely the most favorable one, are batteries. They store the energy chemically, can provide energy for a sustained period of time, and tend to be easily controlled with power converters. This project conceives the storage unit as a lithium-ion battery, since it offers an attractive trade-off between lifetime, power and energy denstiy, and flexibility of operation, despite its high cost.

Table 13 shows the most notorious characteristics of the chosen lithium-ion battery, along with the corresponding converter.

Magnitude	**Units**	**Value**
Capacity	MWh	50
Peak power	MW	10
Round-trip battery η	%	92
Maximum DOD	%	80
Converter η	%	96
Lifetime	cycles	3000
Working temperature	°C	-20/55
Specific energy	Wh/kg	133
Orientative LCOE	€/kWh	0.11/0.66

Table 13. Storage system characteristics. DOD: depth of discharge, LCOE: levelized cost of energy. Data from [25]–[27].

Having a lifetime of 3000 cycles means that it is not appropriate to charge and discharge the battery several times per day. If that were to be the case, the energy storage system would last for just a few years at most. On the contrary, if the battery is charged and discharged once per decade, its lifetime could approach a decade. For this reason, the proposed power profile to follow is represented in Figure 9.

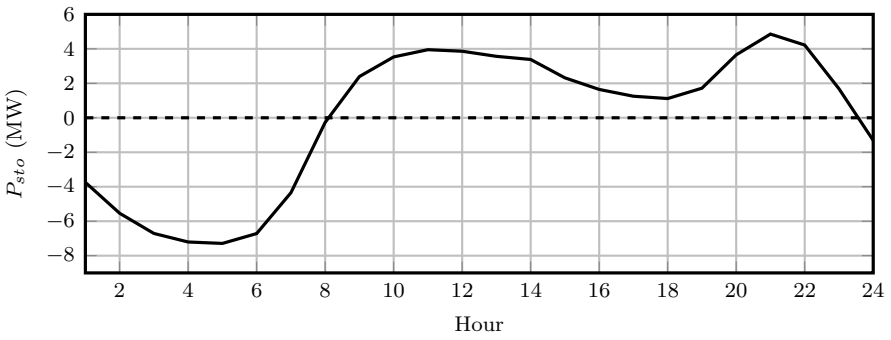

Figure 9. Daily charge and discharge profile for the battery system

This profile resembles the power demand of the loads. This dependence has been established intentionally. In previous analysis, it has been found that during peak hours (around 8 p.m.) the loading of lines is most extreme, and also, the voltages are closer to the lower limit of 0.9 p.u.. Therefore, in general terms, the battery should be discharged during the day and charged at night. According to the chosen sign criteria, positive powers indicate generation, while negative powers denote consumption.

Since the daily demand profile amongst consecutive days do not experience significant differences, it is convenient to always start and finish the day with the same state of charge (SOC). To achieve this, the accumulated area under the curve (in Figure 9, between the solid line and the dashed line) is kept at $E \cdot DOD = 40$ MWh. Besides, the sum of the positive and the negative area yields zero, as desired.

5.2. Dismantled plant

Another possibility to consider in order to reduce the dependence on the interconnection consists of employing the once dismantled power plant placed in bus 7 as indicated in Figure 8. The hypothesis is that this dismantled plant is powered by some kind of fossil fuel (coal, diesel, gas, etc.) or renewable source such as biomass. Thus, even if it could have a positive impact on the electrical magnitudes, its associated environmental impact has to be kept in check.

We start by exploring the diverse energy sources that could power it to extract conclusions regarding costs, emissions, and geopolitical aspects:

- Coal: it is the most polluting of the aforementioned fuels. Even though its emission factor largely depends on the type of coal (lignite, bituminous...), it is in the range of 300 to 360 g CO_2/kWh [28]. Coal power plants are mostly used in countries with availability of natural resources, which is not the case of Spain.

- Diesel: or more generally petroleum, these power plants are more a rarity than an actual common choice. They are suitable to be installed in microgrids or islanded systems thank to their flexibility and suitability for hybrid systems where they act as backup units [29]. Since their chemical proportion between hydrogen and carbon is larger than in the case of coal, its emission factor becomes lower; it is around 270 g CO_2/kWh [28].

- Natural gas: it is cleaner than the two previous fuels, with an emission factor of 180 g CO_2/kWh approximately [28]. Power plants powered by natural gas usually take the form of the so-called combined cycle gas turbine (CCGT) power plant. During the previous decades, Spain promoted the installation of combined cycle power plants, which nowadays are unused for the most part [30]. Probably the largest disadvantage of these power plants is the high cost of natural gas, which has fluctuated extensively during the last year [31].

- Biomass: it is considered a renewable source with carbon neutral emissions [28], since the CO_2 emitted during combustion has previously been absorbed from the atmosphere. Then, the emission factor of biomass is 0 gCO_2/kWh. Nowadays, there are already some examples

where the existing facilities have been converted from coal or natural gas to biomass [32].

When it comes to the implementation of the program, it makes no difference to consider one technology or another, as only the output power is relevant. However, this chapter also focuses on the emissions. Therefore, the results section analyzes the results for coal, diesel and natural gas.

The chosen power generation profile for the dismantled plant is depicted in Figure 10. Compared to storage or the demand, here the peaks are more exaggerated. The underlying idea is that the system is more in need of generation during peak hours. And if demand increases, generation should increase by a larger factor to keep the losses and loadings of the lines at satisfactory values. A peak power of 50 MW has been considered, which corresponds to the most critical hour of the day. The plant is supposed to stop operating at night, and its voltage is maintained at 1.02 p.u. since it acts as a PV node.

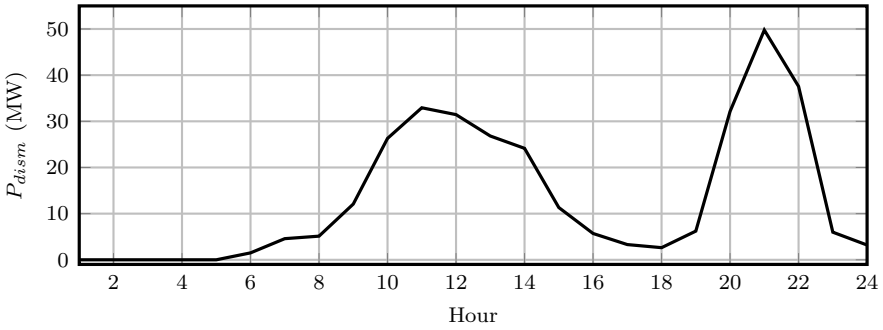

Figure 10. Daily generation profile of the dismantled plant

With storage and the generation of the dismantled plant defined, the simulation can be ran again. The next sections explore the results and draw conclusions from them.

5.3. Base case results

The base case considers the grid as shown in Figure 8, with the black lines representing actual connections. The red lines are only taken into account for the contingency analysis. In any case, it is expected that the presence of the battery and an additonal plant should contribute to improving the electrical magnitudes, at least slightly. Table 11 shows this is precisely the case.

Attributes	**Phase 2**	**Phase 3**	**Phase 4**
V_{min} (p.u.)	0.962	0.968	0.971
V_{max} (p.u.)	1.050	1.050	1.050
Max. load (%)	41.65	40.76	39.63
Max. losses (MW)	14.54	14.43	13.89
Correct operation?	Yes	Yes	Yes

Table 14. Main results to compare between the grid without renewables, with renewables, and with storage and the dismantled plant

The results tend to be the ones we could already expect. The minimum voltage rises a bit thanks to the inclusion of more generation sources, which control the voltages at their respective buses and hence ensure the operation around nominal voltages. Similarly, the loadings of the lines are

reduced a bit, being the most loaded line at only 39.63%. There could be reasons to think that the system is oversized, because lines do not approach their limits of 80%. While these claims would be full of validity, faults have to be taken into account as well. Redundant connections are a must if the criteria $N-1$ has to be met.

The maximum losses are again reduced. A clear pattern can be spotted from comparing the results in the three phases. Going from phase 2 to phase 4, results improve each time the grid becomes stronger and more decentralized. These are in essence some of the core characteristics of smart grids. Another central goal of smart grids has to do with decarbonizing the system, which is described below.

Regarding emissions, it is known beforehand that storage, renewables and nuclear do not emit CO_2 gases or equivalent during their operation. The only generation sources to account for emissions are the interconnection and the dismantled plant. In this base case, the dismantled plant produced 332 MWh, whereas the interconnection provided 4686 MWh. The interconnection can be assumed to have an emission factor of 190 kg CO_2-eq/MWh as in the case of Spain [33], [34].

With this, the total emissions for a full-day operation are gathered in Table 15 depending on the technology of the dismantled power plant.

Fuel	**Dismantled** (tCO_2-eq)	**Interconnection** (tCO_2-eq)	**Total** (tCO_2-eq)
Coal	108.56	890.34	998.90
Diesel	89.64	890.34	979.98
Gas	59.76	890.34	950.10
Biomass	0.00	890.34	890.34

Table 15. Total daily emissions depending on the scenario

The conclusion we reach is that even if there is a non-negligible difference in the emissions caused by the dismantled plant, its overall influence is not that relevant. The interconnection accounts for 90 to 95% of the total CO_2 emissions. The only counteract to that is the integration of nuclear and renewable power plants, which produce about 10.06 GWh daily. This is about twice the energy imported from the slack bus, so in proportion, about 2/3 of the total energy come from clean sources. This is equivalent to saying that the electricity emissions factor of the whole system is roughly 63 kg CO_2-eq/MWh. This is a more than acceptable value that most European countries do not reach (see [35]).

Finally, an economic comparison can be performed according to the related costs of each energy source. These costs include the net present value of the unit-cost of electricity over the lifetime of a system, known as Levelized Cost of Electricity (LCOE), and the CO_2 emission costs. Considering the LCOE of each energy source [36]–[39] and the current CO_2 price [40], the biomass power plant is the cheapest option since it has no CO_2 emission costs associated. It is 16.32% cheaper than natural gas, followed by coal with 35.97%, and diesel with 65.21%.

5.4. Contingency analysis

Just as in previous phases, a contingency analysis is performed to verify if new lines are required, and if so, which ones should be. The marked red lines shown in Figure 8 are the potentially added lines, while the black ones are the alreay established ones. Table 16 presents the top 10 best configurations under faults.

Identifier	**New lines**	**Infraestructure cost (M€)**
33	[8-9]	419.49
24	[8-9, 10-12]	477.12
12	[8-9, 9-6]	480.62
46	[11-12, 8-9]	480.62
8	[8-9, 8-12]	483.96
57	[8-9, 6-8]	505.94
0	[8-9, 10-12]	538.25
4	[8-9, 10-12, 9-6]	538.25
37	[11-12, 10-12, 8-12]	538.25
63	[8-9, 10-12, 8-12]	541.59

Table 16. Best configurations with the additional lines with storage and a dismantled plant

The results indicate that, contrary to previous configurations, installing only the additional line from buses 8 to 9 is enough to meet the $N-1$ criteria. This is certainly a positive result, because in all other cases, at least two lines had to be added. Notice, however, that most configurations are the same as in Table 12. This makes sense since the topology is basically identical, with just minor variations due to the added storage and the dismantled plant.

The total installation cost increases a bit respect to the case where only renewables were added. Of course, new lines have been included to connect the storage unit to the system. Yet, the conclusion has not changed. Moving towards a smart grid, which includes a larger penetration of renewables, batteries, and other elements, supposes a large investment cost. The question is if this invesment ends up being worth it. Multiple reasons could be added into the equation, but what is for sure is that there is a climate emergency to combat.

6. CODE

```
import pandapower as pp
import pandas as pd
import numpy as np
import pandapower.control as control
import pandapower.networks as nw
import pandapower.timeseries as timeseries
import itertools
from pandapower.timeseries.data_sources.frame_data import DFData
from pandapower.plotting import simple_plot
import time

import sys
import codecs

from line_param_calc import calc_line


pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)

def initialize_net(path_bus, path_geodata, path_line, path_demand, path_busload, path_generation
    , path_busgen, path_trafo, rene=False, path_solar_prof=None, path_wind_prof=None):
    """
    initialize the grid from the .csv files

    :param path_bus: path to the bus .csv file
    :param geodata: path to the geodata .csv file
    :param path_line: path to the line .csv file
    :param path_demand: path to the normalized demand .csv file
    :param busload: path to the bus-load look up table .csv file
    :param path_generation: path to the normalized generation .csv file
    :param busgen: path to the bus-generator look up table .csv file
    :param trafo: path to the trafo .csv file
    :param rene: boolean for renewables
    :return: the net class
    """

    def create_bus(path_bus, path_geodata):
        """
        adapts the data from the bus file (if needed)

        :param path_bus:
        :param path_geodata:
        :return: the net with the buses added
        """

        df_bus = pd.read_csv(path_bus)
        df_geodata = pd.read_csv(path_geodata)

        net.bus = df_bus

        # adapt geodata
        for ll in range(len(df_geodata)):
            indx_bus = pp.get_element_index(net, "bus", df_geodata['name'][ll])
            df_geodata['name'][ll] = indx_bus

        net.bus_geodata = df_geodata

        return net


    def create_line(path_line):
        """
        adapts the data from the line file

```

```
66          :param path_line:
67          :return: the net with the lines added
68          """
69
70          df_line = pd.read_csv(path_line)
71          for _, line in df_line.iterrows():
72              from_bus = pp.get_element_index(net, "bus", line.from_bus)
73              to_bus = pp.get_element_index(net, "bus", line.to_bus)
74
75              rr, xx, cc, imax = calc_line(line.a,
76                                           line.b,
77                                           line.c,
78                                           line.d,
79                                           line.e,
80                                           line.max_i,
81                                           int(line.parallel),
82                                           line.Rca,
83                                           line.Dext,
84                                           line.kg)
85
86              pp.create_line_from_parameters(net,
87                                             from_bus,
88                                             to_bus,
89                                             length_km=line.length,
90                                             r_ohm_per_km=rr,
91                                             x_ohm_per_km=xx,
92                                             c_nf_per_km=cc,
93                                             max_i_ka=imax,
94                                             name=line.name_l,
95                                             parallel=line.parallel,
96                                             in_service=line.in_service)
97
98          return net
99
100
101     def create_load(path_demand, path_busload, path_bus):
102         """
103         adapts the load files
104
105         :param path_demand:
106         :param path_busload:
107         :param path_bus:
108         :return: the net with the loads added
109         """
110
111         df_demand = pd.read_csv(path_demand)
112         df_busload = pd.read_csv(path_busload)
113         df_bus = pd.read_csv(path_bus)
114
115         # create basic load dataframe
116         # find the bus index of each load
117         load_indx = []
118         for _, load in df_busload.iterrows():
119             bus_load = pp.get_element_index(net, "bus", load.bus)
120             load_indx.append(bus_load)
121
122         load_indx = pd.DataFrame(load_indx)
123         load_indx = load_indx.rename(columns={0: "bus"})
124
125         # load name and peak power
126         load_name = df_busload['bus']
127         load_pmw = df_busload['p_mw']
128         load_qmvar = df_busload['q_mvar']
129
130         # merge in a full dataframe
131         headers = ["name", "bus", "p_mw", "q_mvar"]
132         df_load = pd.concat([load_name, load_indx, load_pmw, load_qmvar], axis=1)
```

```
133         df_load.columns.values[0] = "name"
134
135         # create time series from the basic load df
136         Nt = len(df_demand)
137         Nl = len(df_load)
138         pmw_ts = np.zeros((Nt, Nl), dtype=float)
139         qmvar_ts = np.zeros((Nt, Nl), dtype=float)
140         for i in range(Nt):  # number of time periods
141             pmw_ts[i,:] = df_load['p_mw'][:] * df_demand['norm'][i]
142             qmvar_ts[i,:] = df_load['q_mvar'][:] * df_demand['norm'][i]
143
144         # form loads as a static picture (initial time)
145         for ll in range(len(df_busload)):
146             pp.create_load(net, bus=load_indx['bus'][ll], p_mw=pmw_ts[0, ll], q_mvar=qmvar_ts[0,
        ll], name=load_name[ll], index=int(ll))
147
148         # timeseries
149         df_pload_ts = pd.DataFrame(pmw_ts, index=list(range(Nt)), columns=net.load.index)
150         df_qload_ts = pd.DataFrame(qmvar_ts, index=list(range(Nt)), columns=net.load.index)
151         ds_pload_ts = DFData(df_pload_ts)
152         ds_qload_ts = DFData(df_qload_ts)
153         const_load = control.ConstControl(net, element='load', element_index=net.load.index,
        variable='p_mw', data_source=ds_pload_ts, profile_name=net.load.index)
154         const_load = control.ConstControl(net, element='load', element_index=net.load.index,
        variable='q_mvar', data_source=ds_qload_ts, profile_name=net.load.index)  # add the
        reactive like this?
155
156         return net
157
158
159     def create_generator(path_generation, path_busgen, path_bus):
160         """
161         adapts the generation files
162
163         :param path_generation:
164         :param path_busgenerator:
165         :param path_bus:
166         :return: the net with the generators added
167         """
168
169         df_generation = pd.read_csv(path_generation)
170         df_busgen = pd.read_csv(path_busgen)
171         df_bus = pd.read_csv(path_bus)
172
173         # create basic generator dataframe
174         # find the bus index of each gen
175         gen_indx = []
176         for _, gen in df_busgen.iterrows():
177             bus_gen = pp.get_element_index(net, "bus", gen.bus)
178             gen_indx.append(bus_gen)
179
180         gen_indx = pd.DataFrame(gen_indx)
181         gen_indx = gen_indx.rename(columns={0: "bus"})
182
183         # load name and peak power
184         gen_name = df_busgen['bus']
185         gen_pmw = df_busgen['p_mw']
186         gen_vpu = df_busgen['vm_pu']
187
188         # merge in a full dataframe
189         headers = ["name", "bus", "p_mw", "vm_pu"]
190         df_gen = pd.concat([gen_name, gen_indx, gen_pmw, gen_vpu], axis=1)
191         df_gen.columns.values[0] = "name"
192
193         # create time series from the basic load df
194         Nt = len(df_generation)
195         Ng = len(df_gen)
```

```
196         pmw_ts = np.zeros((Nt, Ng), dtype=float)
197         for i in range(Nt):  # number of time periods
198             pmw_ts[i,:] = df_gen['p_mw'][:] * df_generation['norm'][i]
199 
200         # gen structure for 1 t
201         for ll in range(len(df_busgen)):
202             pp.create_gen(net, bus=gen_indx['bus'][ll], p_mw=pmw_ts[0, ll], vm_pu=gen_vpu[ll],
        name=gen_name[ll], index=int(ll))
203 
204 
205         # timeseries
206         df_gen_ts = pd.DataFrame(pmw_ts, index=list(range(Nt)), columns=net.gen.index)
207         ds_gen_ts = DFData(df_gen_ts)
208         const_gen = control.ConstControl(net, element='gen', element_index=net.gen.index,
        variable='p_mw', data_source=ds_gen_ts, profile_name=net.gen.index)
209 
210         return net
211 
212 
213     def create_generator_rene(path_generation, path_busgen, path_bus):
214         """
215         adapts the generation files
216 
217         :param path_generation:
218         :param path_busgenerator:
219         :param path_bus:
220         :return: the net with the generators added
221         """
222 
223         df_generation = pd.read_csv(path_generation)
224         df_busgen = pd.read_csv(path_busgen)
225         df_bus = pd.read_csv(path_bus)
226         # df_solar_prof = pd.read_csv(path_solar_profile)
227         # df_wind_prof = pd.read_csv(path_wind_profile)
228 
229         # create basic generator dataframe
230         # find the bus index of each gen
231         gen_indx = []
232         for _, gen in df_busgen.iterrows():
233             bus_gen = pp.get_element_index(net, "bus", gen.bus)
234             gen_indx.append(bus_gen)
235 
236         gen_indx = pd.DataFrame(gen_indx)
237         gen_indx = gen_indx.rename(columns={0: "bus"})
238 
239         # load name and peak power
240         gen_name = df_busgen['bus']
241         gen_pmw = df_busgen['p_mw']
242         gen_vpu = df_busgen['vm_pu']
243 
244         # merge in a full dataframe
245         headers = ["name", "bus", "p_mw", "vm_pu"]
246         df_gen = pd.concat([gen_name, gen_indx, gen_pmw, gen_vpu], axis=1)
247         df_gen.columns.values[0] = "name"
248 
249 
250         # create time series from the basic load df
251         Nt = len(df_generation)
252         Ng = len(df_gen)
253 
254 
255         pmw_ts = np.zeros((Nt, Ng), dtype=float)
256         for gg in range(Ng):
257             for i in range(Nt):  # number of time periods
258                 # pmw_ts[i,gg] = df_gen['p_mw'][gg] * df_generation['norm'][i]
259                 pmw_ts[i,gg] = df_gen['p_mw'][gg] * df_generation.iloc[i,gg+1]
260 
```

```
261             pp.create_gen(net, bus=gen_indx['bus'][gg], p_mw=pmw_ts[0, gg], vm_pu=gen_vpu[gg],
        name=gen_name[gg], index=int(gg))  # take t=0
262
263             # gen structure for 1 t
264             # for ll in range(len(df_busgen)):
265                 # pp.create_gen(net, bus=gen_indx['bus'][ll], p_mw=pmw_ts[0, ll], vm_pu=gen_vpu[
        ll], name=gen_name[ll], index=int(ll))
266
267
268         # timeseries
269         df_gen_ts = pd.DataFrame(pmw_ts, index=list(range(Nt)), columns=net.gen.index)
270         ds_gen_ts = DFData(df_gen_ts)
271         const_gen = control.ConstControl(net, element='gen', element_index=net.gen.index,
        variable='p_mw', data_source=ds_gen_ts, profile_name=net.gen.index)
272
273         return net
274
275
276     def create_intercon(path_bus):
277         """
278         defines the interconnection (slack bus)
279
280         :param path_bus:
281         :return: the net with the interconnection added
282         """
283
284         df_bus = pd.read_csv(path_bus)
285
286         # find the slack index
287         slack_indx = 0
288         for ll in range(len(df_bus)):
289             # slack_indx = pp.get_element_index(net, "bus", bb.name)
290             if df_bus['name'][ll] == 'intercon':
291                 slack_indx = pp.get_element_index(net, "bus", df_bus['name'][ll])
292
293         pp.create_ext_grid(net, slack_indx, vm_pu=1.0, va_degree=0)
294
295         return net
296
297
298     def create_trafo(path_trafo):
299         """
300         defines the transformers
301
302         :param path_trafo:
303         :return: the net with the transformers added
304         """
305
306         df_trafo = pd.read_csv(path_trafo)
307
308         # for trafo in df_trafo:
309         for _, trafo in df_trafo.iterrows():
310             hv_bus = pp.get_element_index(net, "bus", trafo.hv_bus)
311             lv_bus = pp.get_element_index(net, "bus", trafo.lv_bus)
312
313             pp.create_transformer_from_parameters(net,
314                                                   hv_bus,
315                                                   lv_bus,
316                                                   trafo.sn_mva,
317                                                   trafo.vn_hv_kv,
318                                                   trafo.vn_lv_kv,
319                                                   trafo.vkr_percent,
320                                                   trafo.vk_percent,
321                                                   trafo.pfe_kw,
322                                                   trafo.i0_percent,
323                                                   in_service=trafo.in_service)
324
```

```
325         return net
326
327
328     # create empty network
329     net = pp.create_empty_network()
330
331     # buses
332     net = create_bus(path_bus, path_geodata)
333
334     # lines
335     net = create_line(path_line)
336
337     # loads
338     net = create_load(path_demand, path_busload, path_bus)
339
340     # gens
341     # if rene is False:
342         # net = create_generator(path_generation, path_busgen, path_bus)
343     # else:
344     net = create_generator_rene(path_generation, path_busgen, path_bus)
345
346     # interconnection
347     net = create_intercon(path_bus)
348
349     # trafos
350     net = create_trafo(path_trafo)
351
352     return net
353
354
355
356 def get_Plosses(net, prnt=False):
357     """
358     returns the active power losses
359
360     :param net: full grid
361     :param prnt: to print the value
362     :return: total active power losses
363     """
364
365     Pll = sum(net.res_line['pl_mw'])
366     if prnt is True:
367         print('The active power losses are: ', Pll, 'MW')
368
369     return Pll
370
371
372 def get_max_loading(net, prnt=False):
373     """
374     returns the maximum loading in percentage
375
376     :param net: full grid
377     :param prnt: to print the value
378     :return: maximum percentual loading of the lines
379     """
380
381     L_max = max(net.res_line['loading_percent'])
382     if prnt is True:
383         print('The maximum loading is: ', L_max, '%')
384
385     return L_max
386
387
388 def run_store_timeseries(net, identifier):
389     """
390     run the timeseries and store the data
391
```

```
392     :param net: full grid
393     :param identifier: the name to append to the .xlsx file
394     :return: nothing, just store in .xlsx
395     """
396
397     ow = timeseries.OutputWriter(net, output_path="./Results/Cases/Case_"+identifier,
        output_file_type=".xlsx")
398
399     ow.log_variable('res_bus', 'vm_pu')
400     ow.log_variable('res_line', 'loading_percent')
401     ow.log_variable('res_line', 'pl_mw')
402     ow.log_variable('line', 'parallel')
403     ow.log_variable('load', 'p_mw')  # try to read the load to calculate then the efficiency
404     # timeseries.run_timeseries(net)
405     timeseries.run_timeseries(net, continue_on_divergence=True)
406
407     # store other data, like state of the lines...
408     df_lines_states = pd.DataFrame([net.line['name'], net.line['from_bus'], net.line['to_bus'],
        net.line['in_service']])
409     df_lls = df_lines_states.T
410     df_lls.to_excel("./Results/Cases/Case_"+identifier+"/lines_states.xlsx")
411
412     # store diagnostic
413     diagn = pp.diagnostic(net, report_style='compact')
414     # df_diagn = pd.DataFrame(diagn)
415     df_diagn = pd.DataFrame.from_dict(list(diagn))
416     df_diagn.to_excel("./Results/Cases/Case_"+identifier+"/diagnostic.xlsx")
417
418     return ()
419
420
421 def perms(n_extra_lines):
422     """
423     generate the permutations
424
425     :param n_extra_lines: number of added lines to combine
426     :return: list of permutations
427     """
428     # lst = ['True', 'False'] * n_extra_lines
429     lst = [True, False] * n_extra_lines
430     perms_all = set(itertools.permutations(lst, int(n_extra_lines)))
431     perms_all = list(perms_all)
432
433     return perms_all
434
435
436 def run_contingencies_ts(path_bus, path_geodata, path_line, path_demand, path_busload,
        path_generation, path_busgen, path_trafo, n_extra_lines=0):
437     """
438     run contingencies by disconnecting lines for now
439
440     :param path: path of the datafiles to create the grid
441     :param n_extra_lines: number of added lines to combine, the last ones in the .csv
442     :return: store in xlsx files
443     """
444     perms_extra = perms(n_extra_lines)
445
446     net_ini = initialize_net(path_bus, path_geodata, path_line, path_demand, path_busload,
        path_generation, path_busgen, path_trafo)
447
448     n_cases = len(perms_extra)
449     n_lines = len(net_ini.line)
450
451     # disconnect the initial lines and run the cross cases with connecting the others
452     for jj in range(n_lines - n_extra_lines):
453         net_2 = pp.pandapowerNet(net_ini)
454         net_2.line['in_service'][jj] = False
```

```
455
456         # evaluate the state by connecting the extra lines
457         for kk in range(n_cases):
458             # copy the net
459             net_3 = pp.pandapowerNet(net_2)
460
461             # change state of the line (True/False)
462             net_3.line['in_service'][n_lines - n_extra_lines:] = perms_extra[kk][:]
463
464             # run timeseries and store
465             run_store_timeseries(net_3, str(jj) + '_' + str(kk))
466             # run_store_timeseries(net_3, str(jj))
467             # run_store_timeseries(net_3, str(hash(str(jj) + '_' + str(kk))))
468
469     # return ()
470     return n_lines, n_extra_lines, n_cases
471
472
473 def process_contingencies(n_lines, n_extra_lines, n_cases):
474     """
475     merge all excels into one but different sheets
476
477     :param n_lines: number of total lines
478     :param n_extra_lines: number of added lines
479     :param n_cases: resulting total number of cases
480     :return: nothing, just store
481     """
482
483     dd0 = pd.DataFrame([])
484
485     nnx = n_cases * (n_lines - n_extra_lines)
486     f0_vpu = dd0.to_excel("./Results/All_vpu_" + str(nnx) + ".xlsx")
487     f0_load = dd0.to_excel("./Results/All_load_" + str(nnx) + ".xlsx")
488     f0_pl = dd0.to_excel("./Results/All_pl_" + str(nnx) + ".xlsx")
489     f0_diag = dd0.to_excel("./Results/All_diag_" + str(nnx) + ".xlsx")
490     f0_line = dd0.to_excel("./Results/All_line_" + str(nnx) + ".xlsx")
491     f0_parallel = dd0.to_excel("./Results/All_parallel_" + str(nnx) + ".xlsx")
492     f0_pmw = dd0.to_excel("./Results/All_pmw_" + str(nnx) + ".xlsx")
493
494
495     w_vpu = pd.ExcelWriter("./Results/All_vpu_" + str(nnx) + ".xlsx")
496     w_load = pd.ExcelWriter("./Results/All_load_" + str(nnx) + ".xlsx")
497     w_pl = pd.ExcelWriter("./Results/All_pl_" + str(nnx) + ".xlsx")
498     w_diag = pd.ExcelWriter("./Results/All_diag_" + str(nnx) + ".xlsx")
499     w_line = pd.ExcelWriter("./Results/All_line_" + str(nnx) + ".xlsx")
500     w_parallel = pd.ExcelWriter("./Results/All_parallel_" + str(nnx) + ".xlsx")
501     w_pmw = pd.ExcelWriter("./Results/All_pmw_" + str(nnx) + ".xlsx")
502
503     for jj in range(n_lines - n_extra_lines):
504         for kk in range(n_cases):
505             fold_path = "./Results/Cases/Case_" + str(jj) + "_" + str(kk)
506             f1_vpu = pd.read_excel(fold_path + "/res_bus/vm_pu.xlsx")
507             f1_load = pd.read_excel(fold_path + "/res_line/loading_percent.xlsx")
508             f1_pl = pd.read_excel(fold_path + "/res_line/pl_mw.xlsx")
509             f1_diag = pd.read_excel(fold_path + "/diagnostic.xlsx")
510             f1_line = pd.read_excel(fold_path + "/lines_states.xlsx")
511             f1_parallel = pd.read_excel(fold_path + "/line/parallel.xlsx")
512             f1_pmw = pd.read_excel(fold_path + "/load/p_mw.xlsx")
513
514             f1_vpu.to_excel(w_vpu, sheet_name=str(jj) + '_' + str(kk))
515             f1_load.to_excel(w_load, sheet_name=str(jj) + '_' + str(kk))
516             f1_pl.to_excel(w_pl, sheet_name=str(jj) + '_' + str(kk))
517             f1_diag.to_excel(w_diag, sheet_name=str(jj) + '_' + str(kk))
518             f1_line.to_excel(w_line, sheet_name=str(jj) + '_' + str(kk))
519             f1_parallel.to_excel(w_parallel, sheet_name=str(jj) + '_' + str(kk))
520             f1_pmw.to_excel(w_pmw, sheet_name=str(jj) + '_' + str(kk))
521
```

```
522
523     w_vpu.save()
524     w_load.save()
525     w_pl.save()
526     w_diag.save()
527     w_line.save()
528     w_parallel.save()
529     w_pmw.save()
530
531     return ()
532
533
534
535 def find_optimal_config(path_diagN, path_lineN, path_loadN, path_plN, path_vpuN, path_parallelN,
        path_pmwN, n_lines, n_extra_lines, n_cases, exclude_lines):
536     """
537     Find the optimal configuration of lines considering convergence, losses, voltages, and costs
538
539     :param path_diagN: path of the diagnostic N cases
540     :param path_lineN: path of the line N cases
541     :param path_loadN: path of the loading of the lines N cases
542     :param path_plN: path of the losses through the lines N cases
543     :param path_vpuN: path of the voltages N cases
544     :param path_parallelN: path of the number of parallel lines, N cases
545     :param path_pmwloadN: path of the MW of load in the buses, N cases
546
547     :return: optimal case
548     """
549
550     diag = pd.read_excel(path_diagN, sheet_name=None)
551     line = pd.read_excel(path_lineN, sheet_name=None)
552     load = pd.read_excel(path_loadN, sheet_name=None)
553     pl = pd.read_excel(path_plN, sheet_name=None)
554     vpu = pd.read_excel(path_vpuN, sheet_name=None)
555     parallel = pd.read_excel(path_parallelN, sheet_name=None)
556     pmwload = pd.read_excel(path_pmwN, sheet_name=None)
557
558
559     on_off_all_lines = []
560     vec_config = []
561     for name, sheet in diag.items():
562         # if len(sheet) == 0 or name[0] not in exclude_lines:  # if no diagnostic errors
563         if True or name[0] not in exclude_lines:  # if no diagnostic errors
564         # if True:
565             on_off_lines = line[name]['in_service']
566             on_off_all_lines.append(on_off_lines)
567             n_parallel = parallel[name]
568             n_parallel_subset = n_parallel.loc[0, 0:].T  # only the first row, all are equal
569
570             # loadings
571             loadings = load[name]
572             loadings_subset = loadings.loc[0:, 0:]
573             conditional_loading = loadings_subset[loadings_subset[:] > 80].isnull().values.all()
574             # if True, good, we are below 80%
575
576             # vpu
577             vpuss = vpu[name]
578             vpuss_subset = vpuss.loc[0:, 0:]
579             conditional_vpu1 = vpuss_subset[vpuss_subset[:] > 1.10].isnull().values.all()
580             conditional_vpu2 = vpuss_subset[0.01 < vpuss_subset[:]][vpuss_subset[:] < 0.90].
        isnull().values.all()  # to avoid the 0.0
581             # if both are True, we are good
582
583             # active power losses
584             pls = pl[name]
585             pls_subset = pls.loc[0:, 0:]
586             pmws = pmwload[name]
```

```
            pmw_subset = pmws.loc[0:, 0:]

            ok_losses = True
            kk = 0
            while ok_losses is True and kk < 24:  # calculate efficiency at any time
                P_load_all = sum(pmw_subset.loc[kk,:])
                P_loss_all = sum(pls_subset.loc[kk,:])
                P_gen_all = P_load_all + P_loss_all
                eff = P_load_all / P_gen_all
                if eff < 0.98:
                    ok_losses = False
                kk += 1
                # print('Losses: Pload, Ploss, Pgen, eff: ')
                # print(P_load_all, P_loss_all, P_gen_all, eff)

            # if ok_losses is True, we are good

            full_condition = conditional_loading and conditional_vpu1 and conditional_vpu2 and ok_losses
            if full_condition is True:
                print(name)
                vec_config.append(name)

            # print('Condition: load, vpu1, vpu2, loss: ')
            # print(conditional_loading, conditional_vpu1, conditional_vpu2, ok_losses)

    df_configs = pd.DataFrame(vec_config)
    df_configs.to_excel("./Results/OK_configs.xlsx")

    return ()


def select_best(path_configs, path_res_lines, path_ini_lines, n_lines, n_extra_lines, n_cases):
    """
    Find the single optimal configuration, also considering costs

    :param path_configs: path to the OK_configs.xlsx file
    :param path_res_lines: path to the solution for all lines, to read the sheet and states
    :param path_ini_lines: to know the distances
    :return: data for the optimal configuration
    """

    # data costs of the lines
    c_2line = 407521  # euro/km
    c_1line = 288289  # euro/km

    # store costs
    lengths = pd.read_csv(path_ini_lines)['length']
    circuits = pd.read_csv(path_ini_lines)['parallel']
    costs = []
    for ll in range(len(circuits)):
        if circuits.loc[ll] == 1:
            costs.append(c_1line * lengths.loc[ll])
        elif circuits.loc[ll] == 2:
            costs.append(c_2line * lengths.loc[ll])

    costs = np.array(costs)

    # use lines operating state
    res_configs = pd.read_excel(path_configs)
    res_lines = pd.read_excel(path_res_lines, sheet_name=None)
    name_configs_prev = list(res_configs.iloc[:,1])

    # check if the same _yy is in all lines
    configs_all = []
    for xx in name_configs_prev:
        namex = xx.split('_')[1]
```

```
653         configs_all.append(namex)
654
655     nice_configs = set(configs_all)
656     n_count_max = 0
657     n_config = '00'
658
659     # look max number of occurrences
660     for nn in nice_configs:
661         occurr = configs_all.count(nn)
662         if occurr > n_count_max:
663             n_count_max = occurr
664
665     # store the names of the configurations always valid
666     valid_configs = []
667     if n_count_max == n_lines - n_extra_lines:
668         for nn in nice_configs:
669             occurr = configs_all.count(nn)
670             if occurr == n_count_max:
671                 valid_configs.append(nn)
672
673
674
675
676     # only check 0_xx to get the data
677     costs_configs = []
678     dic_configs = {}
679     for name, sheet in res_lines.items():
680         nxx = name.split('_')[1]
681         if nxx in valid_configs:
682             on_off_lines = np.array(res_lines[name]['in_service'])
683             c_total = np.dot(on_off_lines, costs)
684             dic_configs[nxx] = c_total
685
686
687     names_final = np.array(list(dic_configs.keys()))
688     costs_final = np.array(list(dic_configs.values()))
689     # print(dic_configs.keys())
690     # print(dic_configs.values())
691     print(names_final)
692     print(costs_final)
693
694     # final_configs = pd.DataFrame([np.array(dic_configs.keys()), np.array(dic_configs.values())
        ], columns=['configs', 'costs'])
695     final_configs = pd.DataFrame([names_final, costs_final])
696     # final_configs = pd.DataFrame([np.transpose(names_final), np.transpose(costs_final)])
697     print(final_configs)
698
699     final_configs.transpose().to_excel("./Results/OPT_configs.xlsx")
700
701     return ()
702
703
704
705
706
707
708 if __name__ == "__main__":
709
710     time_start = time.time()
711
712
713     # ------------- Inputs -------------
714     # load paths
715     # path_bus = 'Datafiles/phII/bus1.csv'
716     # path_geodata = 'Datafiles/phII/geodata1.csv'
717     # path_line = 'Datafiles/phII/line2.csv'
718     # path_demand = 'Datafiles/phII/demand1.csv'
```

```
719     # path_busload = 'Datafiles/phII/bus_load1.csv'
720     # path_generation = 'Datafiles/phII/generation1.csv'
721     # path_busgen = 'Datafiles/phII/bus_gen1.csv'
722     # path_trafo = 'Datafiles/phII/trafo1.csv'
723
724     # rising voltage level
725     # path_bus = 'Datafiles/phII_380kV/bus1.csv'
726     # path_geodata = 'Datafiles/phII_380kV/geodata1.csv'
727     # path_line = 'Datafiles/phII_380kV/line2.csv'
728     # path_demand = 'Datafiles/phII_380kV/demand1.csv'
729     # path_busload = 'Datafiles/phII_380kV/bus_load1.csv'
730     # path_generation = 'Datafiles/phII_380kV/generation1.csv'
731     # path_busgen = 'Datafiles/phII_380kV/bus_gen1.csv'
732     # path_trafo = 'Datafiles/phII_380kV/trafo1.csv'
733
734     # --------------- phase 3 ---------------
735     # phase with rene
736     # path_bus = 'Datafiles/phIII/bus1.csv'
737     # path_geodata = 'Datafiles/phIII/geodata1.csv'
738     # path_line = 'Datafiles/phIII/line1reduced.csv'
739     # path_demand = 'Datafiles/phIII/demand1.csv'
740     # path_busload = 'Datafiles/phIII/bus_load1.csv'
741     # path_trafo = 'Datafiles/phIII/trafo1.csv'
742
743     # rene
744     # path_generation = 'Datafiles/phIII/generation_all.csv'
745     # path_busgen = 'Datafiles/phIII/bus_gen1.csv'
746
747     # no rene
748     # path_generation = 'Datafiles/phIII/generation1.csv'
749     # path_busgen = 'Datafiles/phIII/bus_gen1_norene.csv'
750
751     # -------------- phase 4 --------------
752     path_bus = 'Datafiles/phIV/bus1.csv'
753     path_geodata = 'Datafiles/phIV/geodata1.csv'
754     path_line = 'Datafiles/phIV/line1reduced.csv'
755     path_demand = 'Datafiles/phIV/demand1.csv'
756     path_busload = 'Datafiles/phIV/bus_load1.csv'
757     path_trafo = 'Datafiles/phIV/trafo1.csv'
758     path_generation = 'Datafiles/phIV/generation_all.csv'
759     path_busgen = 'Datafiles/phIV/bus_gen1.csv'
760
761
762
763
764
765
766
767     # ------------- Running -------------
768     # define net
769     # net = initialize_net(path_bus, path_geodata, path_line, path_demand, path_busload,
         path_generation, path_busgen, path_trafo)
770     net = initialize_net(path_bus, path_geodata, path_line, path_demand, path_busload,
         path_generation, path_busgen, path_trafo, rene=True)
771
772     # run and store timeseries data, for only 1 case
773     run_store_timeseries(net, '_00storage')
774
775     # run contingencies
776     n_lines, n_extra_lines, n_cases = run_contingencies_ts(path_bus, path_geodata, path_line,
         path_demand, path_busload, path_generation, path_busgen, path_trafo, n_extra_lines=6)
777     process_contingencies(n_lines, n_extra_lines, n_cases)
778
779     # n_lines = 16
780     # n_extra_lines = 6
781     # n_cases = 64
782
```

```
783     print(n_lines, n_extra_lines, n_cases)
784
785
786     # ------------- Processing -------------
787     # store
788     nxx = n_cases * (n_lines - n_extra_lines)
789     path_diagN = 'Results/All_diag_' + str(nxx) + '.xlsx'
790     path_lineN = 'Results/All_line_' + str(nxx) + '.xlsx'
791     path_loadN = 'Results/All_load_' + str(nxx) + '.xlsx'
792     path_plN = 'Results/All_pl_' + str(nxx) + '.xlsx'
793     path_vpuN = 'Results/All_vpu_' + str(nxx) + '.xlsx'
794     path_parallelN = 'Results/All_parallel_' + str(nxx) + '.xlsx'
795     path_pmwN = 'Results/All_pmw_' + str(nxx) + '.xlsx'
796
797     exclude_lines = [6,7]
798
799     find_optimal_config(path_diagN, path_lineN, path_loadN, path_plN, path_vpuN, path_parallelN,
          path_pmwN, n_lines, n_extra_lines, n_cases, exclude_lines)
800
801     path_configs = 'Results/OK_configs.xlsx'
802     path_lineN = 'Results/All_line_' + str(nxx) + '.xlsx'
803     path_line_ini = 'Datafiles/phIV/line1reduced.csv'
804
805     select_best(path_configs, path_lineN, path_line_ini, n_lines, n_extra_lines, n_cases)
806
807     end_time = time.time()
808     print(end_time - time_start, 's')
```

Listing 6.1. Main code in Python with the Pandapower library

```
 1 import numpy as np
 2
 3 def calc_line(a, b, c, d, e, immax, npar, Rca, Dext, kgg):
 4     """
 5     calculate r, x, c, and return also Imax
 6
 7     :param a: horizontal distance between A1 and C2
 8     :param b: horizontal distance between B1 and B2
 9     :param c: horizontal distance between C1 and A2
10     :param d: vertical distance between A1 and B1
11     :param e: vertical distance between B1 and C1
12     :param immax: max current in A
13     :param npar: number of parallel lines (1 or 2)
14     :param Rca: ac resistance in ohm/km
15     :param Dext: external diameter in mm
16     :param kg: factor of roughly 0.8
17     :return: r, x, c, imax
18     """
19
20     def single_line(a, b, immax, Rca, Dext, kgg):
21         """
22         calculate the R, X, C parameters, also return Imax
23
24         :param a: horizontal distance between A and C
25         :param b: vertical distance between A and B
26         :param immax: max current in A
27         :param Rca: ac resistance in ohm/km
28         :param Dext: external diameter in mm
29         :param kg: factor of roughly 0.8
30
31         :return: R, X, C, Imax, in the units desired by pandapower
32         """
33
34         # cardinal: https://www.elandcables.com/media/38193/acsr-astm-b-aluminium-conductor-
       steel-reinforced.pdf
35         # 54 Al + 7 St, Imax = 888.98 A
36
```

```
37          w = 2 * np.pi * 50  # rad / s
38          Imax = immax * 1e-3  # kA
39          # Stot = 547.3 * 1e-6  # m2, the total section
40          # R_ac_75 = 0.07316 * 1e-3  # ohm / m
41          # kg = 0.809  # from the slides in a 54 + 7
42
43          R_ac_75 = Rca * 1e-3  # ohm / m, should we correct by temperatures?
44          Stot = np.pi * Dext ** 2 / 4 * 1e-6  # m2, the total section
45          kg = kgg
46
47          r = np.sqrt(Stot / np.pi)  # considering the total section
48
49          dab = np.sqrt((a / 2) ** 2 + b ** 2)
50          dbc = np.sqrt((a / 2) ** 2 + b ** 2)
51          dca = a
52
53          GMD = (dab * dbc * dca) ** (1 / 3)
54          GMR = kg * r
55          RMG = r
56
57          L = 4 * np.pi * 1e-7 / (2 * np.pi) * np.log(GMD / GMR)  # H / m
58
59          C = 2 * np.pi * 1e-9 / (36 * np.pi) / np.log(GMD / RMG)  # F / m
60
61          # in the units pandapower wants
62          R_km = R_ac_75 * 1e3  # ohm / km
63          X_km = L * w * 1e3  # ohm / km
64          C_km = C * 1e9 * 1e3  # nF / km
65
66          return R_km, X_km, C_km, Imax
67
68
69      def double_line(a, b, c, d, e, immax, Rca, Dext, kgg):
70          """
71          calculate the R, X, C parameters, also return Imax
72
73          :param a: horizontal distance between A1 and C2
74          :param b: horizontal distance between B1 and B2
75          :param c: horizontal distance between C1 and A2
76          :param d: vertical distance between A1 and B1
77          :param e: vertical distance between B1 and C1
78          :param immax: max current in A
79          :param Rca: ac resistance in ohm/km
80          :param Dext: external diameter in mm
81          :param kgg: factor of roughly 0.8
82          :return: R, X, C, Imax, in the units desired by pandapower
83          """
84
85          # cardinal: https://www.elandcables.com/media/38193/acsr-astm-b-aluminium-conductor-
        steel-reinforced.pdf
86          # 54 Al + 7 St, Imax = 888.98 A
87
88          w = 2 * np.pi * 50  # rad / s
89          Imax = immax * 1e-3 * 2  # kA, for the full line, x2
90          # Stot = 547.3 * 1e-6  # m2, the total section
91          # R_ac_75 = 0.07316 * 1e-3  # ohm / m
92          # kg = 0.809  # from the slides in a 54 + 7
93
94          R_ac_75 = Rca * 1e-3  # ohm / m, should we correct by temperatures?
95          Stot = np.pi * Dext ** 2 / 4 * 1e-6  # m2, the total section
96          kg = kgg
97
98
99
100         r = np.sqrt(Stot / np.pi)  # considering the total section
101
102         da1b1 = np.sqrt((b / 2 - a / 2) ** 2 + d ** 2)
```

```
103         da1b2 = np.sqrt((a / 2 + b / 2) ** 2 + d ** 2)
104         da2b1 = np.sqrt((c / 2 + b / 2) ** 2 + e ** 2)
105         da2b2 = np.sqrt((b / 2 - c / 2) ** 2 + e ** 2)
106
107         db1c1 = np.sqrt((b / 2 - c / 2) ** 2 + e ** 2)
108         db1c2 = np.sqrt((b / 2 + a / 2) ** 2 + d ** 2)
109         db2c1 = np.sqrt((b / 2 + c / 2) ** 2 + e ** 2)
110         db2c2 = np.sqrt((b / 2 - a / 2) ** 2 + d ** 2)
111
112         dc1a1 = np.sqrt((a / 2 - c / 2) ** 2 + (d + e) ** 2)
113         dc1a2 = c
114         dc2a1 = a
115         dc2a2 = np.sqrt((a / 2 - c / 2) ** 2 + (d + e) ** 2)
116
117         dab = (da1b1 * da1b2 * da2b1 * da2b2) ** (1 / 4)
118         dbc = (db1c1 * db1c2 * db2c1 * db2c2) ** (1 / 4)
119         dca = (dc1a1 * dc1a2 * dc2a1 * dc2a2) ** (1 / 4)
120
121         rp = kg * r
122
123         da1a2 = np.sqrt((a / 2 + c / 2) ** 2 + (d + e) ** 2)
124         db1b2 = b
125         dc1c2 = np.sqrt((c / 2 + a / 2) ** 2 + (d + e) ** 2)
126
127         drap = np.sqrt(rp * da1a2)
128         drbp = np.sqrt(rp * db1b2)
129         drcp = np.sqrt(rp * dc1c2)
130
131         dra = np.sqrt(r * da1a2)
132         drb = np.sqrt(r * db1b2)
133         drc = np.sqrt(r * dc1c2)
134
135         GMD = (dab * dbc * dca) ** (1 / 3)
136         GMR = (drap * drbp * drcp) ** (1 / 3)
137         RMG = (dra * drb * drc) ** (1 / 3)
138
139         L = 4 * np.pi * 1e-7 / (2 * np.pi) * np.log(GMD / GMR)  # H / m
140
141         C = 2 * np.pi * 1e-9 / (36 * np.pi) / np.log(GMD / RMG)  # F / m
142
143         # in the units pandapower wants
144         R_km = R_ac_75 / 2 * 1e3  # ohm / km, like 2 resistances in parallel
145         X_km = L * w * 1e3  # ohm / km
146         C_km = C * 1e9 * 1e3  # nF / km
147
148         return R_km, X_km, C_km, Imax
149
150     if npar == 1:
151         rr, xx, cc, imm = single_line(a, b, immax, Rca, Dext, kgg)
152     elif npar == 2:
153         rr, xx, cc, imm = double_line(a, b, c, d, e, immax, Rca, Dext, kgg)
154     else:
155         print('Error: number of parallel lines is not 1 nor 2')
156
157     return rr, xx, cc, imm
158
159 # rr, xx, cc, ii = double_line(11, 2, 4, 5, 6, 1000)
160 # print(rr, xx, cc, ii)
```

Listing 6.2. Code for the calculation of lines

BIBLIOGRAPHY

[1] Council of European Energy Resources, *Report on power losses*, https://www.ceer.eu/documents/104400/-/-/09ecee88-e877-3305-6767-e75404637087, Accessed: 2021-12-14, 2017.

[2] M. Ahat, S. B. Amor, M. Bui, A. Bui, G. Guérard, and C. Petermann, "Smart grid and optimization," 2013.

[3] I. Vakulenko, L. Saher, O. Lyulyov, and T. Pimonenko, "A systematic literature review of smart grids," in *E3S Web of Conferences*, EDP Sciences, vol. 250, 2021, p. 08 006.

[4] I. Alotaibi, M. A. Abido, M. Khalid, and A. V. Savkin, "A comprehensive review of recent advances in smart grids: A sustainable future with renewable energy resources," *Energies*, vol. 13, no. 23, p. 6269, 2020.

[5] K. Gao, T. Wang, C. Han, J. Xie, Y. Ma, and R. Peng, "A review of optimization of microgrid operation," *Energies*, vol. 14, no. 10, p. 2842, 2021.

[6] Red Eléctrica de España, *Sistema de información del operador del sistema (esios). perfiles de demanda y generación.* https://www.esios.ree.es/es?locale=en, Accessed: 2021-10-25.

[7] Q. Wang and J. D. McCalley, "Risk and "n-1" criteria coordination for real-time operations," *IEEE Transactions on Power Systems*, vol. 28, no. 3, pp. 3505–3506, 2013.

[8] A. Gourtani, H. Xu, D. Pozo, and T.-D. Nguyen, "Robust unit commitment with n-1 security criteria," *Mathematical Methods of Operations Research*, vol. 83, no. 3, pp. 373–408, 2016.

[9] B. Stott, J. Jardim, and O. Alsaç, "Dc power flow revisited," *IEEE Transactions on Power Systems*, vol. 24, no. 3, pp. 1290–1300, 2009.

[10] F. Capitanescu, J. M. Ramos, P. Panciatici, D. Kirschen, A. M. Marcolini, L. Platbrood, and L. Wehenkel, "State-of-the-art, challenges, and future trends in security constrained optimal power flow," *Electric Power Systems Research*, vol. 81, no. 8, pp. 1731–1741, 2011.

[11] L. Thurner, A. Scheidler, F. Schafer, J. H. Menke, J. Dollichon, F. Meier, S. Meinecke, and M. Braun, "Pandapower - an open source python tool for convenient modeling, analysis and optimization of electric power systems," *IEEE Transactions on Power Systems*, 2018, ISSN: 0885-8950. DOI: 10.1109/TPWRS.2018.2829021. [Online]. Available: https://arxiv.org/abs/1709.06743.

[12] Agency for the Cooperation of Energy Regulators (ACER), *Uic report - electricity infraestructure. 2015.*

[13] Autosolar, *Pallet paneles 405w deep blue 3.0 ja solar mono*, https://autosolar.es/panel-solar-24-voltios/pallet-paneles-405w-deep-blue-30-ja-solar-mono, Accessed: 2021-12-01.

[14] European Comission, *Photovoltaic geographical information system (pvgis)*, https://ec.europa.eu/jrc/en/pvgis, Accessed: 2021-12-01.

[15] N. Kumar, P. Yadav, and S. Chandel, "Comparative analysis of four different solar photovoltaic technologies," in *2015 International Conference on Energy Economics and Environment (ICEEE)*, IEEE, 2015, pp. 1–6.

[16] Wind Turbine Models, *Gamesa g132-5.0mw*, https://en.wind-turbine-models.com/turbines/768-gamesa-g132-5.0mw, Accessed: 2021-12-01.

[17] M. Ragheb, "Wind energy conversion theory, betz equation," *Wind Energie*, 2014.

[18] J. Bartl, F. Pierella, and L. Sætrana, "Wake measurements behind an array of two model wind turbines," *Energy Procedia*, vol. 24, pp. 305–312, 2012.

[19] V. Smil, *Power density: a key to understanding energy sources and uses*. MIT press, 2015.

[20] NASA Prediction of Worldwide Energy Resources, *The power project*, https://power.larc.nasa.gov/, Accessed: 2021-12-01.

[21] Investing, *Carbon emissions futures overview*, https://www.investing.com/commodities/carbon-emissions, Accessed: 2021-12-07.

[22] B. P. Roberts and C. Sandberg, "The role of energy storage in development of smart grids," *Proceedings of the IEEE*, vol. 99, no. 6, pp. 1139–1144, 2011.

[23] V. Monteiro, H. Gonçalves, and J. L. Afonso, "Impact of electric vehicles on power quality in a smart grid context," in *11th International Conference on Electrical Power Quality and Utilisation*, IEEE, 2011, pp. 1–6.

[24] M. S. Ziegler and J. E. Trancik, "Re-examining rates of lithium-ion battery technology improvement and cost decline," *Energy & Environmental Science*, vol. 14, no. 4, pp. 1635–1651, 2021.

[25] F. Díaz-González, A. Sumper, and O. Gomis-Bellmunt, *Energy storage in power systems*. John Wiley & Sons, 2016.

[26] F. Lo Franco, A. Morandi, P. Raboni, and G. Grandi, "Efficiency comparison of dc and ac coupling solutions for large-scale pv+ bess power plants," *Energies*, vol. 14, no. 16, p. 4823, 2021.

[27] M. M. Rahman, A. O. Oni, E. Gemechu, and A. Kumar, "Assessment of energy storage technologies: A review," *Energy Conversion and Management*, vol. 223, p. 113 295, 2020, ISSN: 0196-8904.

[28] Catalan Office for Climate Change, *Practical guide for calculating greenhouse gas (ghg) emissions*, https://canviclimatic.gencat.cat/web/.content/04_ACTUA/Com_calcular_emissions_GEH/guia_de_calcul_demissions_de_co2/190301_Practical-guide-calculating-GHG-emissions_OCCC.pdf, Accessed: 2021-12-10.

[29] M. S. Ismail, M. Moghavvemi, and T. Mahlia, "Techno-economic analysis of an optimized photovoltaic and diesel generator hybrid power system for remote houses in a tropical climate," *Energy conversion and management*, vol. 69, pp. 163–173, 2013.

[30] AleaSoft, *The combined cycle gas turbines and the wind energy in the spanish electricity mix*, https://aleasoft.com/combined-cycle-gas-turbines-wind-energy-spanish-electricity-mix/, Accessed: 2021-12-10.

[31] Trading Economics, *Natural gas usd/mmbtu*, https://tradingeconomics.com/commodity/natural-gas, Accessed: 2021-12-10.

[32] Biomass Magazine, *Grontmij converts power plant from natural gas to biomass*, http://biomassmagazine.com/articles/12122/grontmij-converts-power-plant-from-natural-gas-to-biomass, Accessed: 2021-12-14.

[33] Red Eléctrica de España, *Emissions from the spanish electricity system have fallen by 30 million tonnes in the last 5 years*, https://www.ree.es/en/press-office/news/featured-story/2020/06/emissions-spanish-electricity-system-have-fallen-30-million-tonnes-last-5-years, Accessed: 2021-12-11.

[34] C. C. Spork, A. Chavez, X. Gabarrell Durany, M. K. Patel, and G. Villalba Méndez, "Increasing precision in greenhouse gas accounting using real-time emission factors: A case study of electricity in spain," *Journal of Industrial Ecology*, vol. 19, no. 3, pp. 380–390, 2015.

[35] European Environmental Agency, *Greenhouse gas emission intensity of electricity generation in europe*, https://www.eea.europa.eu/ims/greenhouse-gas-emission-intensity-of-1, Accessed: 2021-12-11.

[36] IRENA, *Renewable power generation costs in 2020*, https://www.irena.org/publications/2021/Jun/Renewable-Power-Costs-in-2020, Accessed: 2021-12-14.

[37] S. Roussanaly, M. Vitvarova, R. Anantharaman, D. Berstad, B. Hagen, J. Jakobsen, V. Novotny, and G. Skaugen, "Techno-economic comparison of three technologies for pre-combustion co 2 capture from a lignite-fired igcc," *Frontiers of Chemical Science and Engineering*, vol. 14, no. 3, pp. 436–452, 2020.

[38] Freeing Energy, *Energy fact – solar, wind, and nat gas are, by far, the cheapest ways to generate electricity*, https://www.freeingenergy.com/facts/lcoe-cost-comparison-solar-nuclear-wind-g108/, Accessed: 2021-12-14.

[39] Lazard, *Lazard's levelized cost of energy analysis*, https://www.lazard.com/perspective/levelized-cost-of-energy-2017/, Accessed: 2021-12-14.

[40] Ember, *Daily carbon prices*, https://ember-climate.org/data/carbon-price-viewer/, Accessed: 2021-12-14.