The Netherlands Illuminated: Designing and Simulating a Model of the Dutch Transmission Grid

Thesis by
Andy Zhang, Kyrian Rahimatulla

In Partial Fulfillment of the Requirements for the
Degree of
BSc Electrical Engineering

DELFT UNIVERSITY OF TECHNOLOGY
Postbus 5, 2600 AA, Delft, The Netherlands

2025
Defended June 25

ABSTRACT

Public communication regarding power grid policy is difficult due to the unintuitive behavior of electricity. This thesis presents the development of an easily understandable yet accurate model of the Dutch electricity network. The topology, production and load of the contemporary grid is studied using publicly available data. An accurate model of the grid is designed using stochastic modeling and data analysis techniques. An attempt is made to simulate the model using DC power flow techniques. The software is developed to be compatible with the Illuminator system, which can visualize the simulation in a physical model. The ultimate implementation of the model was unsuccessful. The design process and attempted simulation of the model is documented in this thesis.

PREFACE

This Electrical Engineering Bachelor's thesis was written as part of a collaborative project by Bachelor Graduation Project Group A: Illuminator. This thesis is just a portion of the collaborative effort to develop a visually intuitive hands-on model for simulating power systems. This was easily the largest engineering project we have ever done. It was a hard yet valuable and rewarding experience for us both.

Firstly, we want to thank the head instructor Dr. Ing. Ioan Lager for his work organizing the BAP course, planning the examinations, dealing with our emails, and making all of this possible. We would also like to thank our supervisor, Dr. Milos Cvetkovic for proposing the project, lending us much of his knowledge, and supporting us whenever he could despite his busy schedule. On that note, we would like to extend our sincerest gratitude towards Jort Groen and Despoina Georgiadi. As part of the Illuminator development team, they were under no obligation to provide us with the amount of help that they did. Nevertheless, their guidance, support, and knowledge made a significant impact on our understanding of Illuminator, and helped us overcome challenges along the way. Finally, we would like to thank our colleges Jane Libier, Roos Bonouvrie, Bram Dorland and Alje Vermeer. It was a great privilege to have them on our team. Their smiles, conversations and laughter made this project all the more rewarding.

Andy Zhang and Kyrian Rahimatulla
Delft, June 2025

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

The electrical grid is one of the most important pieces of public infrastructure. Yet, despite its ubiquitous presence in everyday life, electricity remains an elusive concept for the general public to understand. Unlike water, its flow can't be heard. Unlike traffic, its congestion can't be seen. Unlike rail, its mismanagement isn't cared for. This is a great problem.

The production of electricity contributes 36% of current global greenhouse gas (GHG) emissions[1]. Combating climate change requires substantial emission reductions in the global electricity system. This means urgent, decisive policy changes on the national level. Such action necessitates broad support from the populous.

It is becoming ever more important for the public to understand the reasoning behind policies around the electrical system. This bachelor graduation project aims to develop a system that serves that exact purpose. Illuminator[2] is a visually intuitive, easy to use, plug-and-play model of an electrical power system. It is designed to help the public understand how the electrical grid functions on a national level. The Illuminator project was done by three subgroups: A1, A2 and A3. The development of the hardware and software capabilities of Illuminator are discussed in the theses of subgroups A1[3] and A2[4]. The goal of this subgroup is to put their work to practical use. In this thesis, the behavior of the Dutch electrical grid will be studied and modeled. Then, an attempt will be made to use the Illuminator system to simulate and visualize the modeled grid. Finally, the results of the model and simulation will be discussed.

1.1 Problem Definition

The Dutch government has stated[5] that their current priorities regarding the grid include:

- Reducing carbon emissions in line with European targets.
- Preventing grid congestion during the energy transition.
- Increasing energy sovereignty by reducing reliance on imports.

These aspects are abstract and lead to questions. How do emissions compare between different sources? How does grid congestion happen? How much electricity does the Netherlands import? Answering these questions involves complicated calculations that are unreasonable to expect from the average person. But these mechanisms are used in the drafting of national policy. The understanding of these mechanisms needs to become more accessible through intuitive visualization. This project aims to develop a tool that can help with communicating these mechanics through intuitive but accurate visualizations based on official data.

The greatest obstacle to our goal are the limitations on publicly accessible data. This may limit the accuracy of the model. However, that will not effect our aspired goal to show that the creation of such a model is possible.

1.2 State-of-the-Art Analysis

In the case of the Dutch grid, several models already exist. Specifically ones that model power flow include: TenneT themselves[6], Koirala et al[7], and Zomerdijk et al[8]. The last one is noteworthy as it is similar to our case. Their model is based on the current electric grid and only uses publicly available data. The other two models are meant for predicting the future development of the grid. Regardless,

all of these models are primarily meant for scientific purposes and not for policy communication. None of them can visualize the data in a way that is easy to understand.

Some visual grid simulation tools also exist such as DIgSILENT PowerFactory[9] and ETAP Power Simulator [10] but these are again meant for professionals. They may visually display the data, but not in a way that is intuitive for the average person. They also do not account for emissions at all.

Perhaps the best tool available is the site electricitymaps.com [11]. It has an easy to use user interface and presents electricity production and emissions data in an intuitive way. However, this website only shows electricity transfer between countries. It is not capable of showing the network within The Netherlands.

1.2.1 Illuminator

Illuminator [2] is an energy system demonstrator in active development at TU Delft. Illuminator aims to bridge the gap between physically accurate simulators and intuitive visual demonstrators. It is a Python software library built atop another Python library called Mosaik. Mosaik [12] is a general Smart Grid simulation framework specifically designed to be used to build more application-specific simulators.

Illuminator simulates the control and power flow between various grid components, such as loads and wind turbines. Some models, like the wind turbine, receive data from CSV files. In the case of the wind turbine this is wind speed. These models then calculate their power output and send that to the controller. The controller aggregates the total supply and demand and determines how much power must be drawn. One of the more advanced things that the Illuminator controller can do is load shifting. This is a process in which the controller automatically halts the charging of consumer devices when the power over the grid connection is close to its maximum capacity. Vice versa when the connection has capacity to spare.

A limitation of Illuminator is that the controllers are only capable of handling a centralized topology. These are topologies that have a single node where all of the power must flow through. This is despite Mosaik having the capability for dealing with other configurations. This limitation is an obstacle to the goal of designing an accurate demonstrator. A solution will need to be found.

Chapter 2

PROGRAM OF REQUIREMENTS

The requirements of this project can be loosely split into a modeling part and a simulation part. The modeling part concerns the creation of a mathematical model of an electrical grid, based on real world behavior. The simulation part concerns the implementation of that model using Illuminator software. The requirements for these two parts are treated separately.

2.1 Model Requirements

The requirements for the model are based on solving the problem of policy communication around electrical grids. The requirements are:

- The model must be able to reproduce the behavior of the Dutch grid on an average day to a reasonable accuracy.
- The parameters of the model must be based on official data from the Dutch grid whenever possible.
- The model must include the flow of electrical power.
- The model must include carbon emissions caused by electricity generation.
- The model must include a way to determine the reliance on foreign electricity imports.
- The model must account for grid congestion.
- It must be possible to simulate the model using Illuminator software.

2.2 Simulation Requirements

The simulation requirements are governed by two factors: the requirements for simulating the Dutch grid, and the requirements for providing the data necessary for visualization.

In order to simulate the Dutch grid, the simulator should be able to calculate the power flows through a non-centralized energy network. This means a network in which the power does not flow through one central control node. It must be able to calculate and present emission, congestion, and independence, while simultaneously optimizing for those things.

In order to provide the necessary data for visualization, the simulator should be able to output the transmission line powers, so that they can be visualized by group A2 [4], and some of the models should be able to output the relevant quantities for the visualizations done by group A1 [3].

2.3 Physical Prototype

For the purpose of delivering a physical demonstration of our work to the jury, a prototype needs to be made using the resources available to us. We were informed that the university could provide us with a maximum of 15 Raspberry Pi's. This adds the following requirement:

- A (possibly simplified) version of the model must be able to run using 15 Raspberry Pi's or less.

This requirement is not fundamental to this thesis, so it is not treated as a priority.

Chapter 3

DESIGNING THE MODEL

Central to the this project is the creation of an accurate model of the Dutch electrical grid. The different aspects of the model can be broadly divided into three categories. The first category is the network topology. This refers to the layout and location of the transmission lines, power nodes, generators and loads. The second category is generation. This involves modeling the behavior of the various types of electricity sources. The last category are the loads. This is the way that the grid gets loaded by electricity consumers. Because of the goal to aid with policy communication, the model needs to be based on official government data. In this chapter, public information on these aspects are studied and combined into a detailed model of the grid. In addition, a simplified model is made for the physical prototype. The resuling diagrams of both the full and the simplified models are shown in Figure 3.1.

3.1 Topology

The purpose of the model is to describe how electric power flows through the Dutch grid. Thus, a graph representation of the grid needs to be made. This graph needs to have enough information to be able to perform a DC power flow analysis. That technique will be discussed in detail in chapter 4. What is relevant for now is the required information for the graph. The vertices of the graph represent points in the power system. They are referred to as power nodes or busses. These power nodes can be connected to generators and loads, which are not power nodes themselves but properties of power nodes. The power nodes are connected by edges that represent transmission lines. To perform a DC power-flow analysis, it is necessary to know the electrical reactances of all the transmission lines. Resistance is ignored. Nodes that are connected by relatively low reactances may be aggregated together into one power node. This is because the voltage angle between such nodes is small. Additionally, it is also necessary to know the maximum current rating of the transmission lines. This is needed for modeling congestion. The research and design of this graph is discussed in this section.

3.1.1 Transmission Network

The Dutch transmission network refers to the system of high-voltage transmission lines that facilitate the transportation of electricity on the national scale. These are the transmission lines that run at a voltage of 110 kV or higher. The Dutch transmission network is operated by the transmission network operator TenneT[13]. While all of the transmission network is managed by TenneT, only the lines at or above 220 kV are centrally planned[14]. This is because these higher voltage lines are the primary way by which power is transmitted on the national scale[15]. Data on transmission lines below 220 kV tends to be harder to find. Since the model aims to mimic the grid on a national level, the decision was mode to only model transmission lines at or above 220 kV.

Substations are places where voltages levels are transformed between transmission lines. This is necessary at places where a generator or a distribution network with lower voltage is connected. This makes substations a very natural location for the power nodes, as they also represent possible locations of generators and loads. Furthermore, technical data is usually listed by substation. Therefore, the graph should generally have power nodes corresponding to substations, and edges corresponding to transmission lines between substations.

A number of websites were used to model the transmission network. The official national grid

map is published by TenneT[16], along with the official grid diagram[17]. Historical grid maps are archived by HoogspanningsNet[16]. They sometimes contain more information. An official map of the geographical location of the transmission lines (without labels) can be found on ArcGIS[18]. ENTSO-E[1] publishes a map of al the international connections [19]. The maximum current, reactance and other electrical parameters of different connections was published by TenneT in 2023[20]. Some of the locations of the Generators can be found on an ENTSO-E database[21]. Various news articles, official websites and Google Maps are also occasionally used.

The final grid diagram is shown in Figure 3.1a. The impedances and current ratings of the transmission lines are listed, along with the names of the power nodes. Broadly speaking, each power node corresponds to a substation. There are some exceptions made for regions with many generators. One such region is Eemshaven, as depicted in Figure 3.1b. Regions with a lot of generators tend to have a lot of substations. The impedance between these substations is low. In the case of Eemshaven, they are about 2 orders of magnitude lower than what is typical. In such cases, multiple substations are aggregated into one power node.

3.1.2 Province Locations

Certain components in the model, such as loads, are aggregated by province. Such components require a power node that can represent the aggregated data of an entire province. This power node is chosen by looking at the population centers of the province. Most provinces have a single region that is much more populous than the rest of the province. The power node that corresponds to a substation in that region is chosen to represent the province. For example, the metropolitan area around Groningen (City) contains 61% of the population in the province. It is served by the Vierverlaten substation, so the Groningen province is aggregated to the Vierverlaten power node. Detailed justifications for each province can be found in Appendix A.

Flevoland

Flevoland is a special case because it is the only province whose grid is separated by a voltage transformer. Power from the northern part of the province (known as Noordoostpolder) needs to pass through the transformer in Ens to get to the southern part of the province (known as Flevopolder). This transformer has a large reactance of almost 13 Ω, which will result in a large angular difference despite their geographic proximity. This is the reason why the province is split in two. For simplicity, both Flevopolder and Noordoostpolder will be referred to as 'provinces'. Data for these two provinces is calculated manually by summing the data from their municipalities.

3.2 Generation

The Netherlands currently relies on six major types of electricity production. These include biomass, coal, natural gas, nuclear fission, solar and wind. Wind energy can be split into offshore, nearshore and onshore wind. There also exists a small amount of production sources categorized as 'other'. These will be treated as a separate category.

3.2.1 General considerations

There are certain considerations that apply to multiple types of electricity sources. These will be discussed here.

[1] ENTSO-E is the association of all European transmission system operators.

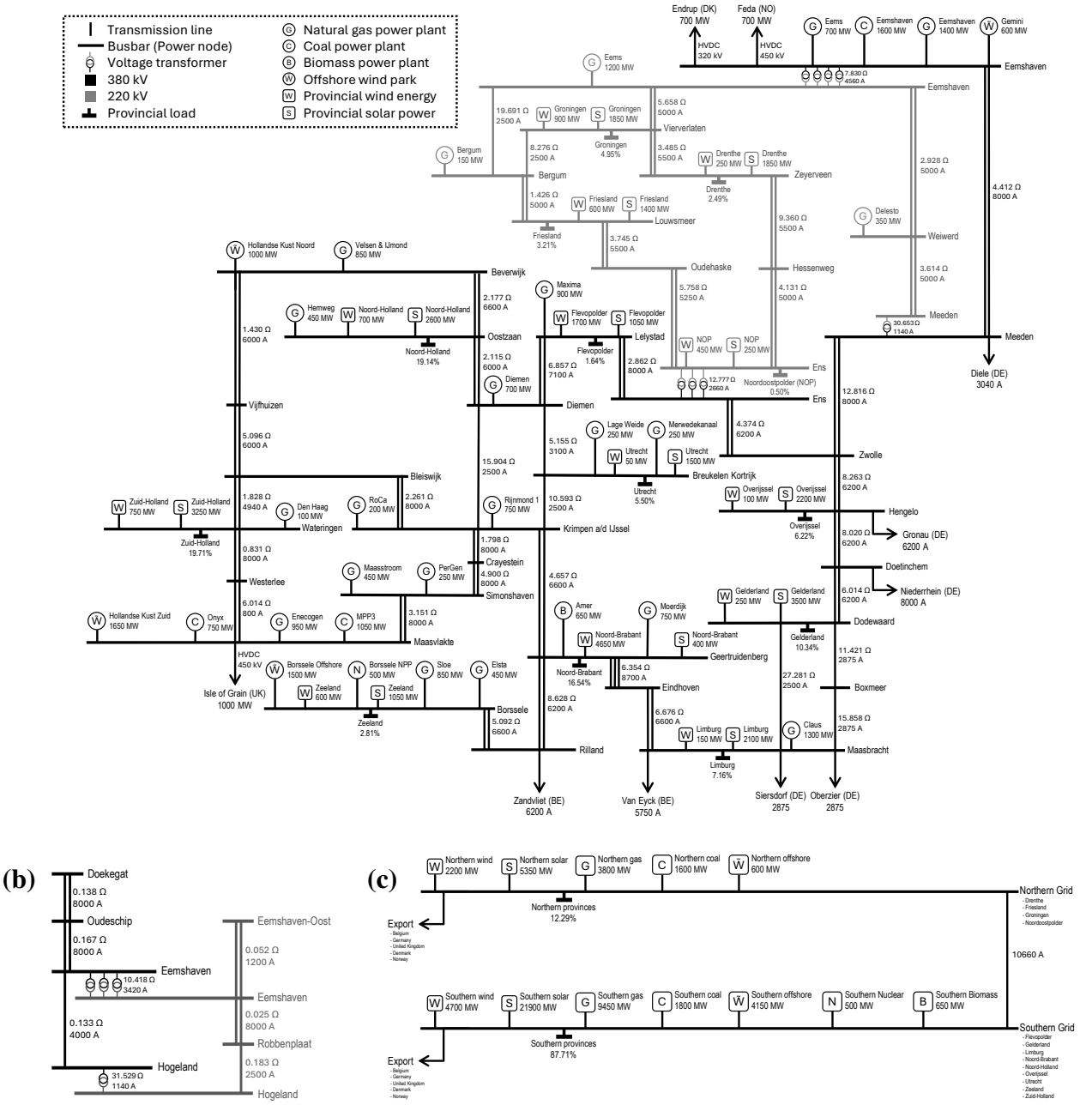


Figure 3.1: Grid diagrams of the Dutch grid models developed based on publicly available data. The legend in the top left shows the meaning of the symbols. The names, power capacity, reactance and maximum current of the nodes, generators and transmission lines are shown to the right. Some generators are aggregated by province or region, indicated by a square label. The percentage underneath the loads indicate their average share of the total national load. The layouts are somewhat geographically inspired. **(a)** The full model that aims to accurately mimic the Dutch transmission network. Aside from the loads, only solar and onshore wind are aggregated by province. The power nodes roughly correspond to all the existing high-voltage substations. **(b)** Grid diagram of the substations around the Eemshaven area. These substations are aggregated in the full model. **(c)** A simplified model that aims to demonstrate the functionality of the Illuminator system. All generators and loads are aggregated into two power nodes.

Aggregation

Electricity generation can be broadly divided into large scale production, and distributed production. Large scale installations have a capacity above 500 MW, and are directly managed on the national level[14]. These include natural gas, coal, offshore wind, nuclear fission and biomass power plants. Distributed sources are installed with capacities below 100 MW, which include onshore wind turbines and solar panels. Electricity from such installations are largely consumed locally, often without providing anything for the high-voltage grid[22]. Their development is decentralized through targets assigned to the provinces[23]. Since the model is supposed to give a national overview of the grid, distributed sources are aggregated by province. This also helped with data collection, as the public data about these sources is also aggregated by province. On the other hand, large scale installations are considered separately.

That still leaves the installations between 100 MW and 500 MW. Even though they are managed locally, these generators are deemed to be of 'national importance' because of their contribution to the energy supply[24]. The vast majority of installations in this range are natural gas plants. Because of their importance as well as the availability of data, these plants are treated as large scale sources. This means that they are not aggregated and can be found in Figure 3.1a. The other installations which are in this range are the three nearshore wind farms. 'Princess Ariane Wind Farm' has a theoretical maximum capacity of 184 MW[25], but due to the variability of wind its average contribution is closer to 90 MW. 'Windpark Fryslân' is a 400 MW wind farm [26], which is over 66% of the total provincial capacity. 'Windpark Noordoostpolder'[27] (450 MW) is the only wind farm in the Noordoostpolder province. For all these three wind farms, aggregation would have little effect. To preserve consistency with how these wind farms are classified by the government, these wind farms will be treated the same as onshore wind. They will be aggregated by province.

Finally there are the 'other' electricity sources. Because of the lack of information on the location of these sources and their small contribution to the grid, they are aggregated on a national level.

Capacity

The capacity of a production source refers to the theoretical maximum power output of that source. A database of the capacity of all the coal, biomass and nuclear power plants can be found on ENTSO-E[21]. Data on the capacity of offshore wind farms and be found on a government website[28]. For onshore wind and solar, the installed capacity per province can be found on another government website[29]. There is little information on the 'other' sources, except that they contribute to about 3.84% of total production[30]. How this category is handled will be discussed below. ENTSO-E lists all gas power plants above 100 MW. Unfortunately, this only accounts for about 70% of total production capacity. The other 30% tends to be privately owned by industrial companies for immediate local use. Their impact on the overall grid negligible, even more so considering only a portion of the total gas capacity is used at once. For that reason, these small plants are ignored entirely. The capacities of the different sources are shown in Figure 3.1a. To account for slightly different figures between websites, all production values are rounded to 50 MW.

Emissions

Emissions are quantified using carbon intensity (CI), which refers to the amount of GHG emitted for the production of one unit of electricity. The mixture of GHG emissions are measured in carbon dioxide equivalent units (CO_2-eq), which is the amount of CO_2 that needs to be emitted to cause an

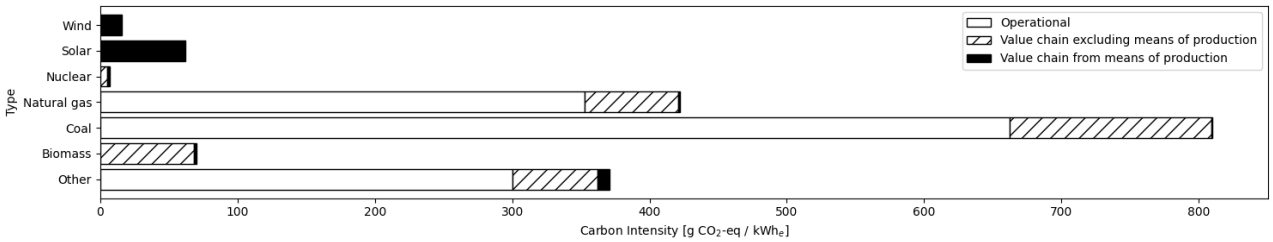


Figure 3.2: Bar chart depicting the carbon intensity of electricity sources used in the Netherlands.

equivalent amount of global warming over a 100 year period[2]. The exact calculation of this can be found in the IPCC report[31, Sec.7.6, Tab.7.15].

Emissions can be split into operational emissions[3] and value chain emissions[4]. Operational emissions are the emissions that are a direct result of electricity production, such as the burning of fossil fuels. The value chain emissions encompass all of the additional indirect emissions from the production chain. Value chain emissions include the emissions from fuel extraction, fuel processing, fuel transportation, electricity transmission losses, and others. A special component of value chain emissions is the value chain emissions from the means of production (MoP)[5]. These are the emissions caused by the construction and demolition of the power plant itself.

The value chain emissions from MoP can be somewhat misleading. MoP emissions are calculated from the average carbon footprint over the entire life-cycle of a power source. This may give the impression that solar panels that produce more electricity also have higher emissions. But all solar panels undergo the same production, assembly and disposal process, thus the total MoP emissions are identical. For this reason, the value chain emissions from MoP will be excluded from the emissions in our model.

The value chain CI figures used by the Dutch state are the ones calculated by Bruinsma and Nauta[32]. The official operational CI values can be found on NED[30]. The CI emissions of the 'other' category is discussed in subsection 3.2.6. There is some controversy regarding biomass[33], which the government deems to be operational emissions free. Because of our requirement to design based on official figures, this controversy will be ignored. The CI for different sources is shown in Figure 3.2. The sum of the operational emissions (white) and the value chain emissions excluding MoP (hatched) are used as the emissions in the model. Exact values can be found in Table C.1.

Variable renewable energy

Wind and solar are forms of variable renewable energy (VRE). The production of solar and wind energy fluctuate throughout the day due to changes in weather and sunlight. Importantly, these factors are outside of human control. Therefore the fluctuating output of VRE's needs to be part of the model. The fraction of electric power generated compared to the maximum capacity is known as the capacity utilization factor, denoted F, as shown in Equation 3.1

$$F(t) = \frac{P(t)}{P_{max}} \tag{3.1}$$

[2]Also known as Global Warming Potential 100.

[3]Also known as Scope 1 and Scope 2 emissions.

[4]Also known as Scope 3 emissions.

[5]Also known as life-cycle emissions.

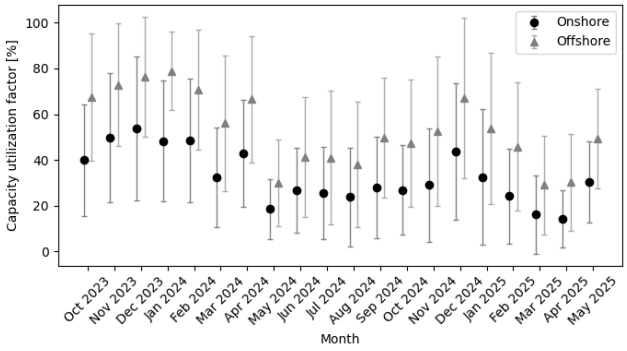


Figure 3.3: Monthly average capacity utilization factors for onshore and offshore wind. The black circles show the mean capacity utilization factor for onshore and the gray triangles for offshore wind generation. Vertical bars indicate the central 68% intervals of the instantaneous values. Uncertainties of the mean are not depicted, but they are roughly equal to the size of the markers.

where $P(t)$ denotes the instantaneous power output and P_{max} is the installed capacity. This quantity is useful as its behavior is independent from the amount of capacity that is installed. To accurately model VRE's, the fluctuations of F needs to be studied across different locations and times of year. This will be discussed in more detail in the separate sections for wind and solar.

3.2.2 Wind

Wind is a VRE so the behavior of the capacity utilization factor F_{wind} needs to be modeled. This involves studying the behavior of F_{wind} on both the national and provincial level. A model is made that reflects the behavior of F_{wind} on an average day.

National Level Analysis

An official government API [30] can be used to access figures on the power output of different energy sources per province. This can be used to find the momentary value of F_{wind}. Values are given at 10 minute intervals starting from October 2023 for a total of around 80,000 entries. This large dataset enables statistical observations with a high degree of confidence.

The first analysis is on the way F_{wind} fluctuates throughout the year. Figure 3.3 depicts the average national value of F_{wind} per month. The graph clearly shows statistically significant differences between months. These differences do not appear to have an obvious period. For example, November 2023 is very different from November 2024 (for contrast see Figure 3.8). There does appear to be a general tendency for winter months to have a higher F_{wind} than the summer, but there are exceptions such as February 2024.

Fluctuations throughout the day are analyzed per month. Figure 6.1b shows the extreme cases of January and July 2024: two months with a large difference in average in average F_{wind}. It is noteworthy that the moving average value is relatively flat and stays close to the overall average. Looking at the individual days, the lines appear to be mostly smooth which is a result of the inertia from the blades. A few jumps do exist, likely corresponding to curtailment events. The general shape of the curves does not appear to change throughout the days or even between months. For example Figure 3.4a includes a day where F_{wind} starts out around 40%, gradually rises to 100% before falling back down at the end of the day. A very similar curve can be found in Figure 3.4b, but such curves are less likely due to the lower monthly average. Additional graphs can be found in Figure C.2 and Figure C.3.

Some additional statistical analysis of F_{wind} are shown in Figure 3.5. Of particular note is

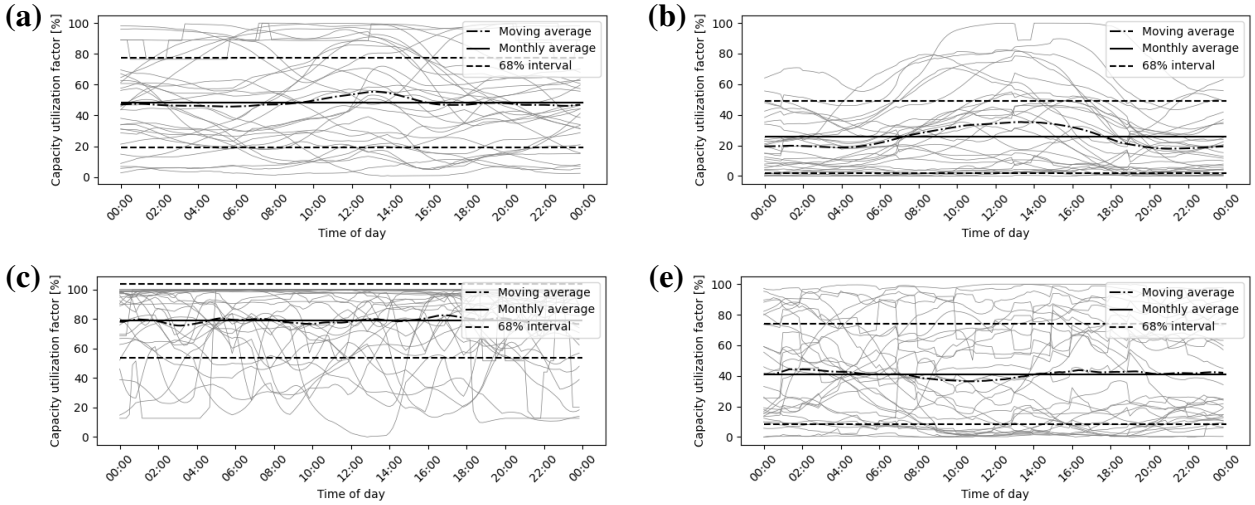


Figure 3.4: Graphs depicting the capacity utilization factors of wind throughout the day. The gray lines are the individual days. The dash dotted line is the moving average of the gray lines. The solid black line is the average of the entire month. The dashed lines are the 68% intervals of the instantaneous values compared to the monthly average. **(a)** Onshore wind in January 2024. **(b)** Onshore wind in July 2024. **(c)** Offshore wind in January 2024. **(d)** Offshore wind in July 2024.

Figure 3.5b which shows that the probability distribution of $\frac{\mathrm{d}^2F}{\mathrm{d}t^2}$ in the interval $[1/3, 1/2]$ is Gaussian. That interval is the equiprobable region, which is the region where the derivative of the probability density function is close to 0, as seen in Figure 3.5c. These graphs will be discussed in more detail in the following section, where thay are used to model the behavior of F_{wind}.

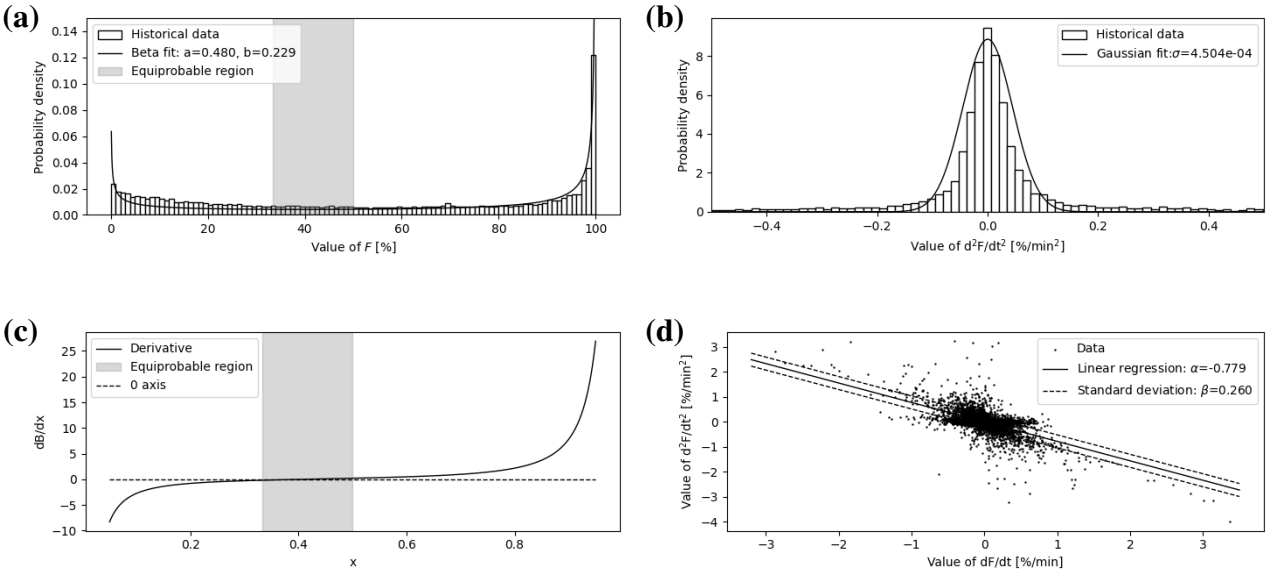


Figure 3.5: Graphs depicting various statistical analyses of the capacity utilization factor F for offshore wind in the Netherlands. **(a)** Histogram of F along with a least-squares fit beta probability density function. The equiprobable region is highlighted. **(b)** Histogram of the second derivative of F within the equiprobable region. A Gaussian fit is also shown. **(c)** The derivative of the Beta distribution. The equiprobable region is highlighted. **(d)** Linear regression of the first and second derivatives of F within the equiprobable region.

Stochastic Modeling of Capacity Utilization Factor

From Figure 6.1b, it is clear that there is no such thing as an average wind day. The behavior must be modeled stochasticly. Because of the random behavior of the monthly fluctuations and the similarities between them, the decision was made to have just one model for the entire year. Let $X(t)$ denote a stochastic function that models $F_{\text{wind}}(t)$. This function must be defined in such a way that it statistically resembles the historical wind data. Let $x(t)$ denote a realization of $X(t)$. From the observations discussed above, the following properties of $x(t)$ are inferred:

- **Smoothness:** $\frac{\mathrm{d}^2x}{\mathrm{d}t^2}$ is well defined.
- **Steady state:** The behavior of x does not depend on t. To be precise: it is strictly stationary.
- **Gaussian acceleration in equiprobable region:** $p(\frac{\mathrm{d}^2x}{\mathrm{d}t^2})$ is Gaussian for $x \in [1/3, 1/2]$.

There are occasional exceptions to these properties, such as the discontinuities caused by curtailment. But these cases are infrequent enough to be negligible. A general recipe was developed to create a stochastic function X with the properties outlined above. The steps are as follows:

1. Find the probability density function of $F(t)$, denoted $p(x)$.
2. If $p(x)$ has a restricted domain on the interval $\mathcal{I}$, find a sigmoid function $\Lambda : \mathbb{R} \to \mathcal{I}$.
3. Find an equiprobable region $\mathcal{E}$ where $\frac{\mathrm{d}p}{\mathrm{d}x}$ is relatively close to 0. It is important that Λ is approximately linear for the range $\mathcal{E}$.
4. Within $\mathcal{E}$, perform a linear regression to find α in Equation 3.2. Let β denote the standard deviation from that linear regression.

$$\frac{\mathrm{d}^2F}{\mathrm{d}t^2} \approx \alpha \frac{\mathrm{d}F}{\mathrm{d}t} \tag{3.2}$$

5. The random variable X is defined using the stochastic differential equation in Equation 3.3:

$$\begin{cases} \frac{\mathrm{d}^2S}{\mathrm{d}t^2} = \alpha \frac{\mathrm{d}S}{\mathrm{d}t} + \beta\sqrt{\mathrm{d}t}\eta(t) + \frac{\beta^2}{2\alpha}\frac{\mathrm{d}}{\mathrm{d}S}\log\left[p(\Lambda(S))\left|\frac{\mathrm{d}\Lambda}{\mathrm{d}S}\right|\right] \\ X = \Lambda(S) \end{cases} \tag{3.3}$$

   where $\eta(t)$ are independent normalized Gaussian distributed random variables.

This method was found by realizing that that F_{wind} resembles the behavior of a physical particle in a thermal reservoir with an energy potential. A more detailed description of how this recipe was derived can be found in Appendix B.

In theory this recipe should work on any function that satisfies the assumptions made above. For our purposes it was only applied to find a stochastic model for F_{wind}. Offshore wind will be used as an example to demonstrate how the recipe works:

1. Figure 3.5a shows the the probability density function of $F_{\text{wind}}(t)$. It can be described by a beta distribution with the probability density function given by Equation 3.4.

$$B(x) = \frac{x^{a-1}(1-x)^{b-1}}{B(a,b)}, \qquad B(a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \tag{3.4}$$

   where Γ denotes the gamma function. The parameters a and b were determined using a least-squares fit resulting in $a = 0.480, b = 0.229$.

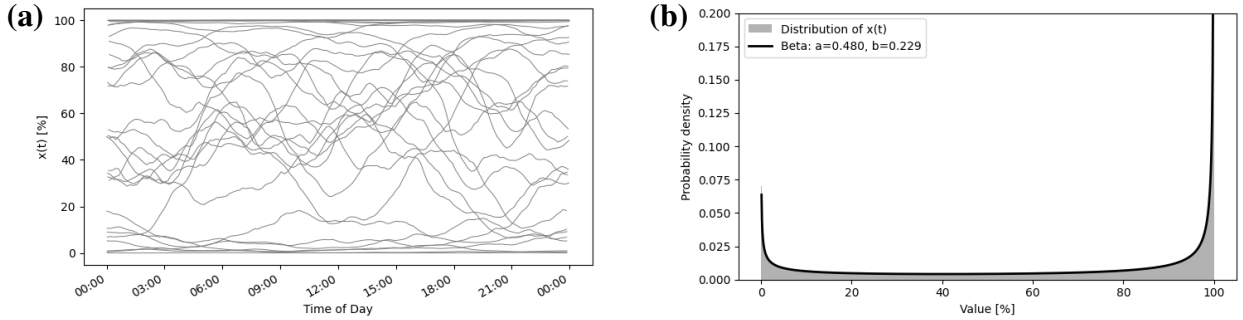


Figure 3.6: Graphs depicting the results of generating realizations $x(t)$ from a stochastic differential equation. The stochastic function aims to mimic the capacity utilization factor of offshore wind energy. **(a)** Graph showing a realization $x(t)$ over the period of 31 days. The gray lines are the 31 individual days. **(b)** Graph showing the distribution of $x(t)$ over 100,000 days. The gray area is a high resolution histogram of the values. The black line is the expected probability density function that was measured from the offshore wind data.

2. The domain of $B(x)$ is restricted to $\langle 0, 1\rangle$, so the sigmoid function $\Lambda(s)$ is defined as in Equation 3.5.

$$x = \Lambda(s) = \frac{1}{1 + e^{-4s}} \tag{3.5}$$

3. The derivative of $B(x)$ is shown in Figure 3.5c. The interval $[1/3, 1/2]$ is chosen as the equiprobable region. Note that the purpose of the multiplication by 4 in Equation 3.5 is to make sure that $\frac{\mathrm{d}x(1/3)}{\mathrm{d}\Lambda} \approx \frac{\mathrm{d}x(1/2)}{\mathrm{d}\Lambda} = 1$.

4. The linear regression of $\frac{\mathrm{d}^2F}{\mathrm{d}t^2}$ and $\frac{\mathrm{d}F}{\mathrm{d}t}$ is shown in Figure 3.5d, resulting in $\alpha = 0.779\ \mathrm{min}^{-1}$ and $\beta = 0.260\ \mathrm{min}^{-2}$.

5. The final stochastic differential equation becomes:

$$\begin{cases} \frac{\mathrm{d}^2S}{\mathrm{d}t^2} = \alpha\frac{\mathrm{d}S}{\mathrm{d}t} + \beta\sqrt{\mathrm{d}t}\eta(t) + \frac{\beta^2}{2\alpha}\frac{a-be^{4S}}{(e^{4S}+1)/4} \\ X = \Lambda(S) \end{cases} \tag{3.6}$$

A sample of the functions generated by Equation 3.6 is shown in Figure 3.6a. These generated functions do indeed to similar to the ones from Figure 3.4c and Figure 3.4d. A histogram of 100,000 days is shown in Figure 3.6b. It is clear that X has the same distribution as F_{wind}. This same process was repeated for onshore wind, resulting in Figure C.1.

Local Variation

Equation 3.3 is only used to generate a national average value for F_{wind}. In the case of onshore wind, the different provinces can have values that differ from the national average. An example for Noord-Holland is shown in Figure 3.7a. From this figure it appears that the ratio has the properties necessary such that Equation 3.3 can be applied. However, when looking at the probability distribution of different provinces, such as shown in Figure 3.7b and Figure 3.7c, that distribution can have substantial differences. Thus, Equation 3.3 would need to be solved 12 times for all the provinces, which was too much work.

Due to time constraints, the provincial variation ratio is modeled as independent and identically distributed over time with a Laplace distribution that is fitted for every province. This is the same

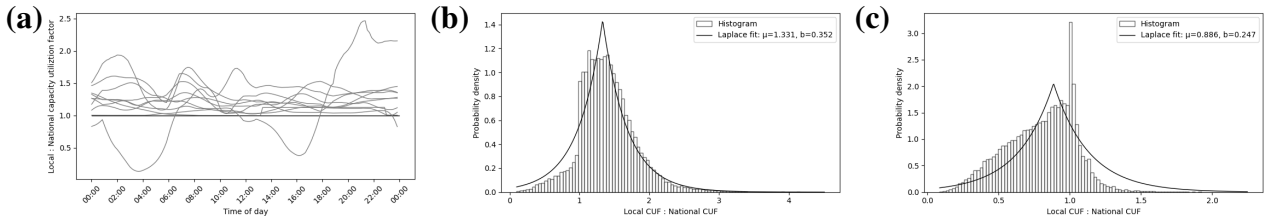


Figure 3.7: Graphs of the ratio between the local and national capacity utilization factor for onshore wind. **(a)** Depicts the ratio for the 10 days from 1 November 2023 to 10 November 2023 in the Noord-Holland province. **(b)** Shows a histogram based on the data from October 2023 to May 2025 for Noord-Holland. The black line is a least-squares fit of a rescaled Laplace distribution. The Laplace distribution is rescaled such that the area under the positive side is unitary. **(c)** Idem but for Flevoland.

technique as used for solar, where it is discussed in more detail in section 3.2.3. Additional graphs can be found in Figure C.6. Local variations in offshore wind are ignored due to a lack of data.

3.2.3 Solar

Solar is a VRE so its output behavior needs to be modeled. Some of the techniques used here are similar to the ones used for modeling wind (subsection 3.2.2).

National Average

The monthly variations of F_{solar} are shown in Figure 3.8. The graph clearly shows predictable yearly periodic variations in the average value of F_{solar}. For that reason, each month in the year will be modeled separately.

The daily progression of F_{solar} is shown in Figure 3.9. Although the curve can vary from day to day, this variation consists mostly of momentary fluctuations and changes in the height of the peak. The overall shape of the curves stays mostly consistent between days. Therefore, it makes sense to use the moving average curve to represent an average day. This moving average curve is calculated for every month on the year. These curves can be found in Figure C.4.

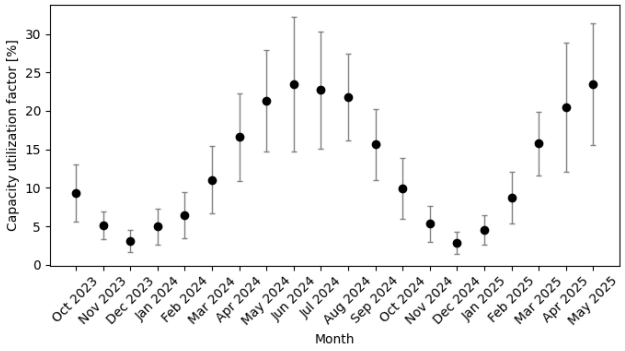


Figure 3.8: Monthly average capacity utilization factors for solar. The black circles show the mean capacity utilization factor. Vertical bars indicate the central 68% intervals of the daily mean value. Uncertainties of the mean are not depicted, but they are roughly equal to the size of the markers.

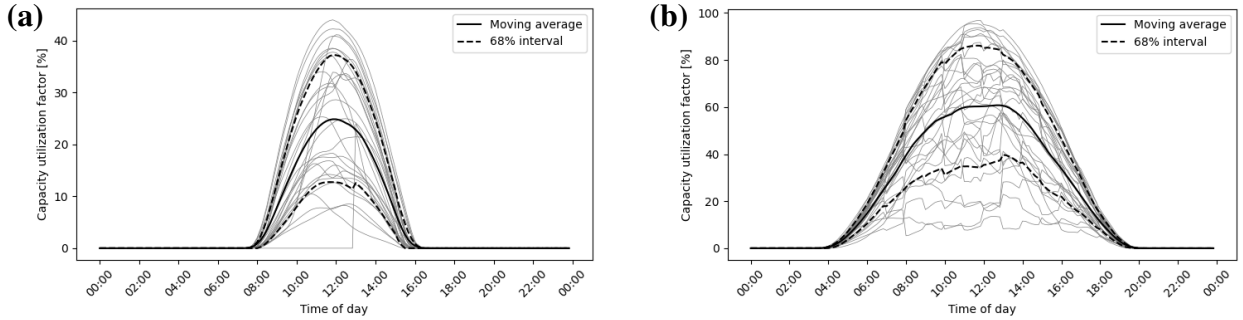


Figure 3.9: Graphs depicting the capacity utilization factors of solar throughout the day. The gray lines are the individual days. The dash dotted line is the moving average of the gray lines. The solid black line is the average of the entire month. The dashed lines the are the 68% intervals of the instantaneous values compared to the monthly average. **(a)** January 2024. **(b)** July 2024.

Local Variations

Local variations in F_{solar} are studied and shown in Figure 3.10. Looking at the daily curves in Figure 3.10a, the curves do not appear to be smooth. This is especially true in Figure 3.10b, which only plots the data for every 30 minute interval. The points in this graph appear to be independent and identically distributed. This is how the provincial variations will be modeled.

Looking at the probability distributions in Figure 3.10c and Figure 3.10d, all the provincial variations have Laplace distributions. Therefore, in the model the provincial variation ratio is randomly

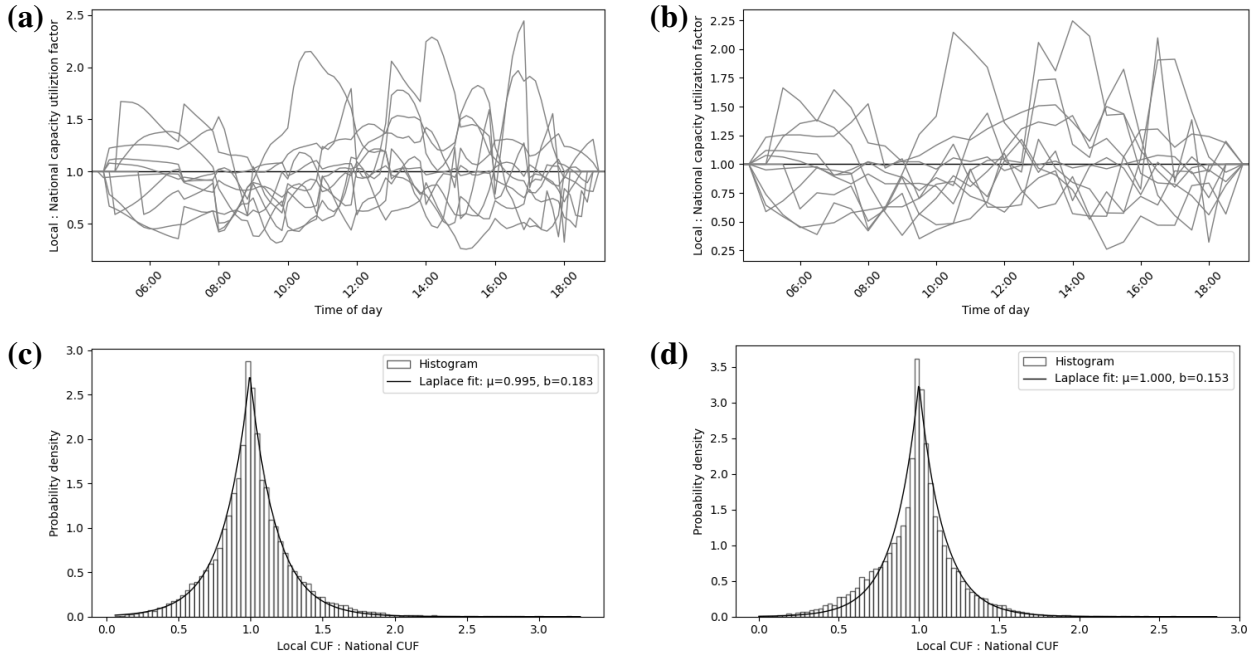


Figure 3.10: Graphs of the ratio between the local and national capacity utilization factor for solar. **(a)** Depicts the ratio for the 10 days from 1 November 2023 to 10 November 2023 in the Noord-Holland province. The view is limited to times where the power output is non-zero. **(b)** Depicts the same but only using entries for every 30 minutes. **(c)** Shows a histogram based on the data from October 2023 to May 2025 for Noord-Holland. The black line is a least-squares fit of a rescaled Laplace distribution. The Laplace distribution is rescaled such that the area under the positive side is unitary. **(d)** Idem but for Flevoland.

chosen from a Laplace distribution that was fitted to the data. It should be noted that the Laplace distribution is normally defined over $\mathbb{R}$, so the distribution needs to be rescaled such that the positive curve is unitary. The resulting probability curves are also shown in the figure. Results for other provinces can be found in Figure C.7. The random ratio is independently regenerated for every time step.

3.2.4 Nuclear Waste

The only operational nuclear power plant (NPP) in the Netherlands is the Borssele NPP operated by EPZ. It has an electrical output of 500 MW. The reactor does not have any operation GHG emissions, but it does produce radioactive waste. Low-level waste (LLW) and intermediate level waste (ILW) needs to be shielded for 30 years before it can be safely disposed. High-level Waste can stay radioactive for thousands of years, so its management is an environmental issue of international concern[34]. The Borssele NPP produces 33 m^3 of LLW-ILW and 4 m^3 of HLW per year for a total of 3771 GWh of electricity. This results in 8.751 mm^3/kWh for LLW-ILW and 1.061 mm^3/kWh for HLW [35]. These two types of waste are both be part of the model.

3.2.5 Coal and Biomass

The coal power plants in the Netherlands run on mix of coal with some small fraction of biomass. The only power plant that has published its biomass fraction is Eemshaven at 15% [36]. This value will also be used for the other coal power plants.

3.2.6 Other sources

There is little information on the 'other' category. From NED data it is possible to deduce that around 59% of the other sources are waste-to-energy plants. This same data also shows that the 'other' sources provide around 3.84% of the total electricity. In the model the 'other' category will be modeled as a constant 3.84% reduction in the total load. The emissions are equal to the official production emissions of waste-to-energy plus the value chain emissions of the 'other' category[6]. The carbon intensity can be seen in Figure 3.2.

3.3 Load

A log of the instantaneous load for every 15 minutes on the Dutch grid is provided by ENTSO-E[21]. This data was analyzed in a similar way as VRE sources. The monthly fluctuations are shown in Figure 3.11. Like with solar, periodic annual fluctuations are clearly visible. For that reason the load is modeled for every month.

The daily load curves are shown in Figure 3.12. Again, like with solar, the overall shape of the curves stays mostly consistent between days. However, Figure 3.12b shows that there is a clear distinction between weekdays and weekends. This can also be seen in other months, which are shown in Figure C.5. Due to time constraints, this distinction is not made in our model. The national load is simply the average day for every month.

There is no data on provincial load available. For an educated guess, the provincial loads are set to the fraction of that provinces annual electricity usage compared to the national total. This is shown as a percentage in Figure 3.1a. Fluctuations are introduced with the exact same parameters as solar. This is because solar power, like electricity consumers, are often just individual households.

[6]The official government value chain carbon intensity of 'other' is just a weighted average of all the CI values.

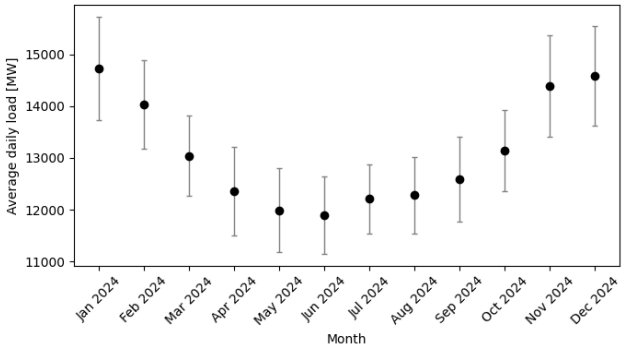


Figure 3.11: Monthly average load on the Dutch national grid. The black circles shows the mean load per month. Vertical bars indicate the central 68% intervals of the daily mean value. Uncertainties of the mean are not depicted, but they are roughly equal to the size of the markers.

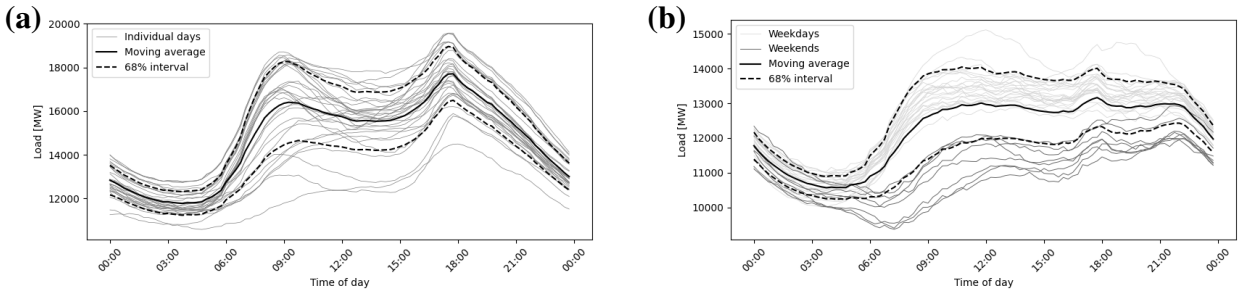


Figure 3.12: Graphs depicting the load on the Dutch grid throughout the day. The gray lines are the individual days. The dash dotted line is the moving average of the gray lines. The solid black line is the average of the entire month. The dashed lines the are the 68% intervals of the instantaneous values compared to the monthly average. **(a)** January 2024. **(b)** July 2024 with the weekends specially highlighted.

3.4 Final considerations

Some final remarks about the model are given below.

3.4.1 Time Resolution

The time resolution of the model is set to 30 minutes per time step. This is because the distribution of local fluctuations of solar energy becomes independent at this time scale (as discussed in section 3.2.3). An additional benefit is that data for the VRE sources and the load have resolutions of 10 and 15 minutes respectively. This makes 30 minutes time steps easier to implement.

3.4.2 Battery Storage

The total capacity of battery storage is 229 MW which is a negligible amount [37]. Any effect of battery storage is deemed to be adequately approximated by the 'other' sources category.

3.5 Simplified model

The model developed thus far was designed for accuracy, but it cannot be used for the physical prototype. The physical prototype is limited to 15 Raspberry Pi's, meaning that only 15 components can be visualized. Therefore the detailed model needs to be aggregated. To his end, the detailed model was first converted to a graph. The vertices correspond to power nodes and the edges correspond

to transmission lines, with the weight set to their reactance. This graph was visualized using the ForceAtlas2 graph spatialisation algorithm by Jacomy et al. [38]. In essence, this algorithm attempts to relate the weights of the edges between nodes to their spacial distance. Nodes connected by low weights tend to cluster together, and vice versa how high weights. Since nodes which are close will tend to have a lower impedances between them, compact clusters of nodes will also tend to have smaller angular differences. These clusters can therefore be approximated as one single power node. This algorithm is imperfect as it can find different local minima depending on initial conditions. For this reason, the algorithm was run 5 times using different starting conditions for different spatialisations. The result of one such spatialisation is shown in Figure 3.13.

Looking at the resulting spatialisation, it became clear that the graph can be roughly split into a northern cluster and a southern cluster. This split is also shown in the figure. By aggregating these clusters into two power nodes, a simplified model is created. The final result of the simplified model is shown in Figure 3.1c. It is can fully run on 15 Raspberry Pi's: 2 for the power nodes, 1 export node (which is also a power node) and 12 for the generators. For the physical prototype, a stylized version of the model was also created. It can be found in Figure 3.14.

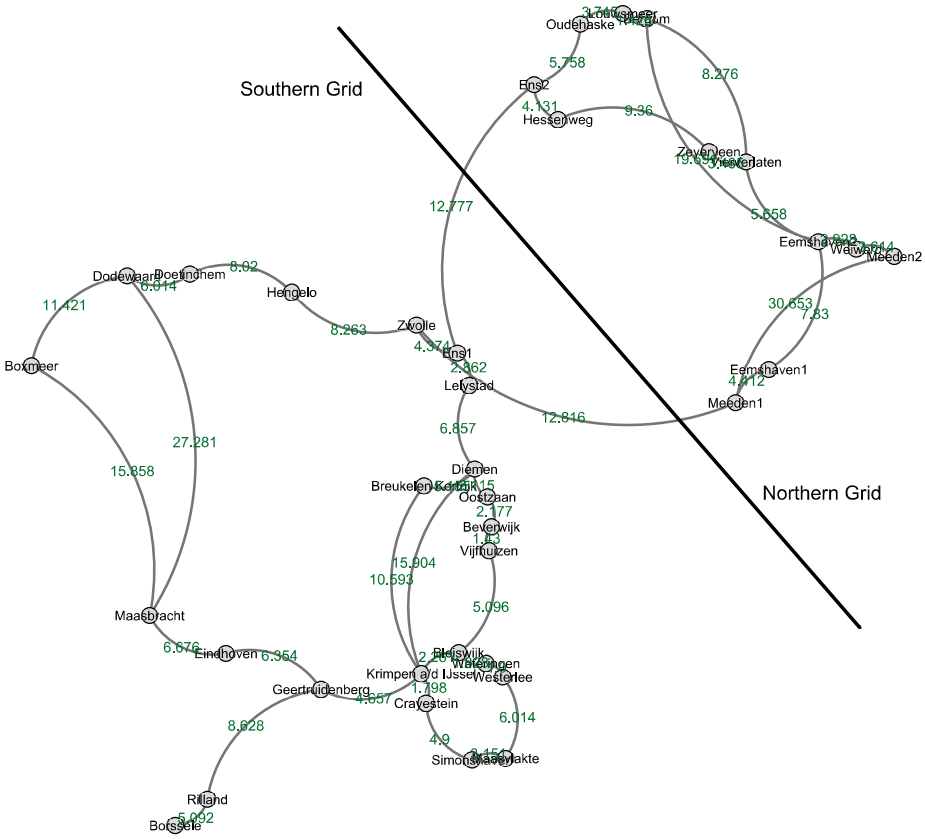


Figure 3.13: Graph spatialisation generated by the ForceAtlas2 algorithm. The nodes correspond to power nodes in the Dutch grid model. The edges are the transmission lines, with the weight equal to the reactances. The edges are curved to improve readability. This is unrelated to the spatialisation algorithm which uses straight line euclidean distances. The diagonal black line depicts the proposed split between the northern and souther aggregated power node.

Figure 3.14: A stylized version of the simplified grid model. This figure is intended to be printed on 9 A3 papers. Raspberry Pi's with the Illuminator software can be placed on the circles to run the model.

Chapter 4

DESIGN BACKGROUND

4.1 Control Schemes

The grid to be simulated is the one in figure Figure 3.1a. Recall that the goal of the simulation is to gain insight into the theoretical ability of the current Dutch grid to minimize emissions and foreign dependence, while staying within the grid capacity limits. From figure Figure 3.1a, it can be deduced that the Dutch grid is a decentralized power network. There isn't a single bus where all the power flows through. Instead it consists of multiple interconnected buses which each have their own producers and consumers. The important thing to know about about decentralized power flow is that the buses, by definition, have no routing agency. They are essentially merely conductors. Instead, the power over the transmission lines is a function of the grid parameters and the powers at the buses, both incoming and outgoing [39]. This has important consequences for congestion management, which concerns routing the power flow in a network such that the power limits of the transmission lines are not exceeded [39]. Because of this phenomenon, congestion management is done by managing the power output, or input, of the various controllable components in the network, such as generators, batteries, and external grid connections. In actuality, congestion is the secondary problem that arises in the process of solving the primary grid problem: power balancing, which concerns matching power supply to power demand in order to maintain the nominal grid frequency [39]. There are a few different control schemes which can be used for power balancing and congestion management. Below the options considered for the simulation are presented.

4.1.1 Markets

In practice, the Dutch grid is controlled via markets [40]. There are three relevant time scales in the European energy market. The longest time scale is the forward and futures market. In it, energy buyers and sellers secure contracts which guarantee that they will be able to buy or sell a certain amount of energy for a pre-agreed price at some point in the future. The time scale for this market is between four years and one month [41].

The mid-level time scale is the day-ahead market [41]. This market is an auction that closes at noon on the day before the consumption day. In it, energy suppliers (mostly owners of power plants) submit for every one of their assets the maximum available energy and the cost of that energy for every hour of the following day. The energy buyer then organizes these bids in ascending price order (again, for every hour). The buyer has forecasted the total energy demand for every hour. Based on this, all assets, from cheapest to more expensive, needed to fulfill that demand are committed to supply energy for that hour. This is called unit commitment. The buyer pays every asset the energy price of the most expensive committed asset. This price is called the clearing price.

The shortest time scale is the intra-day market [41]. In it, buyers and sellers make deals without being mediated by an auction. Deals need to be closed at least five minutes before delivery. The TSO uses this market to allow the order volumes to be adjusted from the day-ahead auction based on updated forecasts. Congestion management necessitates not only considering which asset is the cheapest, but also where the asset is in relation to the source of the demand, in order to prevent transmission line

current capacities from being exceeded.

4.1.2 Optimal Power Flow

Optimal Power Flow (OPF) is an instantaneous cost optimization algorithm that outputs generator dispatch commands [39]. It can be considered to have collapsed all the price negotiations of the previously mentioned markets into one, as it takes the energy price of every asset as its input. However, it completely disregards the quantity negotiations, as it decides how much power every generator is going to produce in that very moment. Congestion management functionality is implemented by specifying the optimization constraints. Optimal Power Flow by itself is an instantaneous calculation and therefore doesn't consider generator ramp-up time (ramp-up time one of the main reasons that planning ahead is needed [42]), so in practice it can not be used in isolation. Rather it can be used by the energy buyer during market negotiations, in conjunction with forecasting, to make energy purchasing decisions ahead of time. In a simulation that doesn't consider generator ramp-up time, OPF can be used in isolation.

For this simulation, OPF in isolation is chosen. This is done for the following reasons. Firstly, the generator model used in this simulation, which will be discussed later, doesn't have a ramp-up time. Secondly, the time step size of this simulation is 30 minutes, virtually eliminating the need to consider generator ramp-up time. Lastly, the goal of this simulation requires the controller to optimize for minimal emissions and dependence. With OPF, this is easily done by choosing the CO2 emission per unit of energy as the cost factor for the generators, and setting the supply cost of the external grid connections very high to discourage its use. More on this in the PandaPower section. Incorporating markets, which don't necessarily optimize for minimal emissions and dependence, would defeat the purpose of this investigation.

Using OPF in isolation requires the following mental interpretation: all the decisions made by the OPF calculation are actually made in advance by the TSO, in order to prepare the generators to deliver the power. In the simulation, the control is instantaneous.

4.2 Power Flow Analysis Methods

OPF uses power flow analysis to incorporate the transmission line capacity constraints in the optimization process. As mentioned in section 4.1, the flow of power depends on the grid parameters and the powers at the buses. Power flow analysis models this dependency to calculate the power flow. There are two types of power flow analyses: AC power flow and DC power flow [39]. AC power flow is considered the physically comprehensive method, while DC power flow is a simplification of AC power flow. AC power flow gives a complete power profile of the grid, including active and reactive power, power angle, voltage magnitude, and voltage angle. DC power flow only models active power and voltage angle. This has consequences for calculating transmission line current, necessary for OPF. Transmission line current is calculated using equation 4.1

$$I = \frac{S}{V} \tag{4.1}$$

Where S is the apparent power and V is the actual voltage. With DC power, only the active power and the nominal voltage are known, resulting in the approximation of I in equation 4.2.

$$I = \frac{F}{V_n} \tag{4.2}$$

Where F is the active power and V_n is the nominal voltage.

All this makes AC power flow the seemingly obvious choice to incorporate into the simulation. However, for several reasons, DC power flow is chosen anyway. Firstly, AC power flow requires more grid parameters to be known than are available from the grid research. DC power flow in its simplest form only needs the transmission line reactances. Secondly, AC power flow is more computationally intensive than DC power flow. Since the goal is to simulate an entire day consisting of many time steps for instantaneous results, this is not desirable. Lastly, the power factor (amount of active power per unit of apparent power) of modern grids doesn't often venture far below 0.95 [43], making active power a relatively good approximation of apparent power, for the purposes of calculating transmission line currents.

4.3 Analysis Tool

The DC OPF functionality will be implemented using PandaPower [44]. This is a power analysis Python library. It adds Pandas dataframe functionality to another power analysis Python library called PYPOWER. PYPOWER in turn is a Python wrapper for MATPOWER, the power analysis tool based on MATLAB. PandaPower is chosen because it can be conveniently integrated in Illuminator, for both the initialization stage, in which the grid topology is built, and the simulation stage. Its integration of Pandas dataframes makes it very flexible in handling data, of which the importance will become clear in section 5.3. It can perform both AC and DC power flow analysis, as well as AC and DC OPF.

4.4 Transformer Parameters

In order to bridge a gap between the grid research results and the implementation in PandaPower, a transformer parameter calculation needs to be done. This calculation is based on a simple transformer circuit seen in figure 4.1.

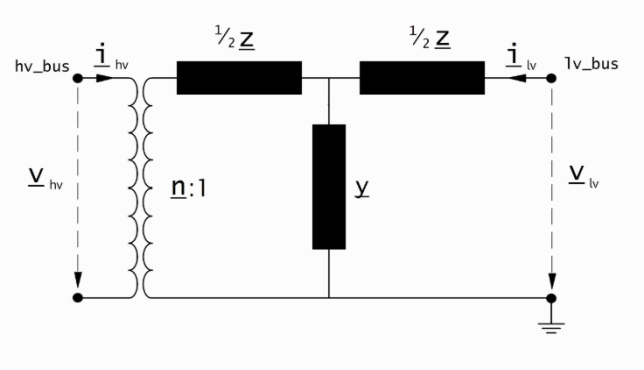


Figure 4.1: Transformer model used in PandaPower [44]

The known quantities are the equivalent reactance X_{eq}, the rated primary side voltage $V_{1,n}$, and the rated primary side current $I_{1,n}$ (these are found through the Dutch grid research). The desired quantity is the relative short-circuit voltage $V_{SC,r}$ (this is needed for the transformer declaration in Pandapower). The relative short-circuit voltage is the fraction of the rated primary side voltage $V_{1,n}$ required to send the rated primary side current through the transformer, when the secondary side is shorted. It is related to the transformer impedance magnitude by equation 4.3.

$$|\mathbf{Z_{eq}}| = \frac{V_{SC,r} \cdot V_{1,n}}{I_{1,n}} \tag{4.3}$$

The impedance is in turn related to the reactance according to equation 4.4

$$X_{eq} = \sqrt{|\mathbf{Z_{eq}}|^2 - R_{eq}^2} \tag{4.4}$$

Because R_{eq} is typically two orders of magnitude lower than X_{eq} [20], it is ignored, leading to the simplified equation 4.5

$$X_{eq} = \frac{V_{SC,r} \cdot V_{1,n}}{I_{1,n}} \tag{4.5}$$

Chapter 5

IMPLEMENTATION: THE NEW ILLUMINATOR

5.1 What It Consists of

Figure 5.1 shows what a new Illuminator topology looks like. The topology is now clearly decentralized. Conceptually, it has 4 types of models: Non-Controllable Peripherals (NCP), Controllable Peripherals (CP), Stations (S), and the controller. The NCP's are all the models that force a certain power at their output. This could be incoming or outgoing. Examples are Load, PV, Wind, Nuclear. The CP's are all the models of which the power is controlled by the controller. Examples are Fossil, GridConnection, Battery. The stations are buses. They don't produce or consume power. They also don't have the agency to route power, as the power distribution over the transmission lines (marked with X because of their reactance) is purely determined by the net power values at the stations. The controller has no power connection to any of the other models. This is because the controller doesn't route the power physically by switching, but by determining the output of the CP's. Note that the transmission lines are drawn as directed edges (arrows). This is because each transmission line 'belongs' to a station. This is particularly relevant for visualization, which will be discussed in section 5.2. The code can be found on Github [45]. The code of the controller model is highlighted in Appendix E.

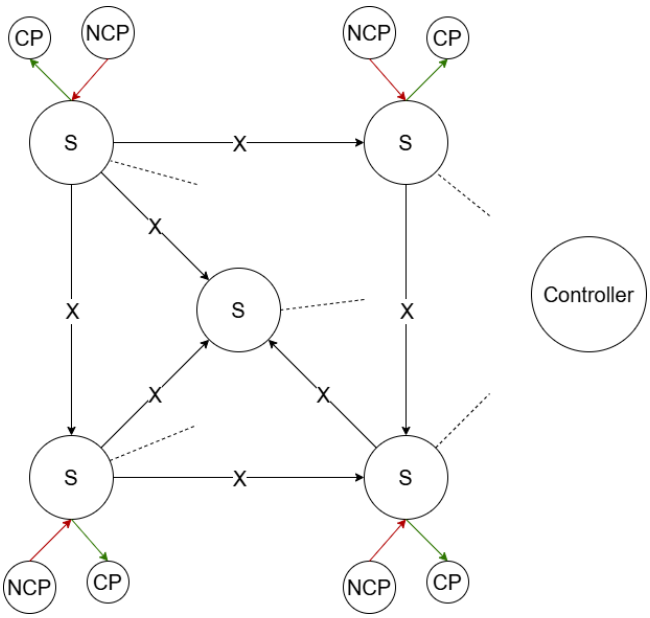


Figure 5.1: New Illuminator topology

5.2 How It Simulates

During simulation, the actual Illuminator connections look a lot different than in figure 5.1. This is because, during simulation, there is one thing more important than where the power flows to. That is where the information of the power flows to. See fig 5.2 for the information flow. At the beginning of the simulation, the entire topology is communicated to the controller. This will be discussed in section 5.3. Then, the first thing that happens every time step is that the NCP's send their power data to the controller. The NCP's bypass the stations, because the stations will be receiving data from

the controller later. If the stations were to send data to the controller as well, it would create a loop. Illuminator has solutions to work with such loops [2], but for the sake of synchronicity they are best avoided, as these loops necessarily delay certain communications by one time step.

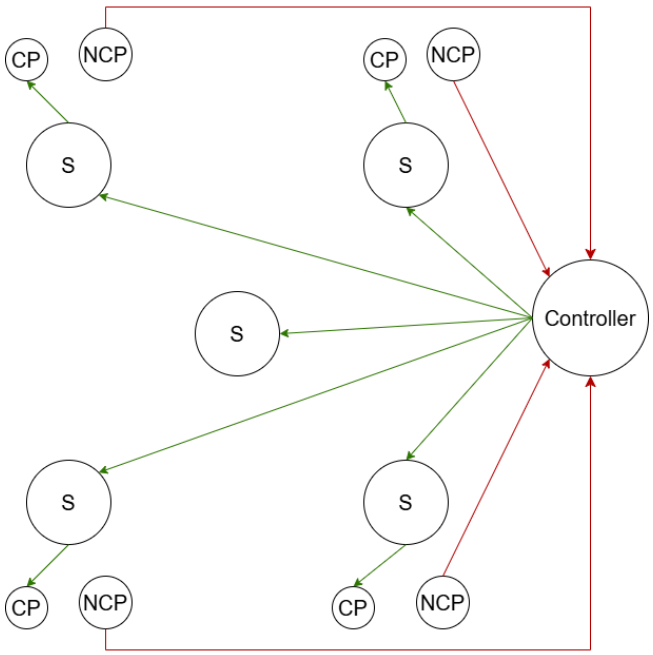


Figure 5.2: Information flow

The NCP's send their data in a single-entry dictionary with their name as the key, so that the controller knows where the power came from without having to have a separate input field for every NCP. This way, the controller can connect to virtually infinitely many models and command an infinitely large network. The controller uses this information alongside its knowledge of the topology to calculate the optimal power flow. This gives the controller the results for the required CP outputs and with that the transmission line loadings. The CP outputs are put into a nested Python dictionary with the highest keys layer being the stations. Each station key contains a dictionary with the CP's belonging to that station as keys, and the required powers as values. A similar thing happens for the transmission lines. Both of these nested dictionaries get sent to all stations. The stations pick out the inner dictionaries corresponding to their own name. They then pass the CP dictionary to all of the CP's connected to them. The CP's in turn unpack the dictionary further, using their own name as a reference. This cascaded approach of first sending the data to the stations, upon which they pass it on, is chosen to reduce the amount of redirections that need to be done in the initialization stage discussed in section 5.3.

In a digital simulation, nothing happens to the transmission lines occupancy data after it enters the station. When a physical simulation is executed however, the stations send the transmission line data to the line LED control models, which do the same thing with the data as the CP's. This is also where the importance of a line 'belonging' to a station comes in. The LED control model receives the LED data from the station to which the line belongs.

5.3 Initialization

For this decentralized model, the initialization stage is of utmost importance. Figure 5.2 showed that the actual data connections do not correspond at all with the network topology. The initialization stage bridges this gap. The simulation configuration is declared in the YAML config file according to figure 5.1. This is done so that the user need only concern themselves with describing the network topology, and not with the data connections, which will be automatically generated. When the simulation run command is called, engine.py, which handles among other things the initialization, does the following things, always with the YAML file as the source of its information. It appends the name of each model to its own parameters, so that every model knows its own name, without the user having to specify the name twice. It puts all the model names and their parameters (with values) in the controller parameter 'peripherals', so that the controller has all the relevant data about the CP's (including the station that the CP is connected to), which it uses in the optimal power flow calculation. It does the same with the stations, so that the controller knows their voltages. For this reason, all relevant parameters should be explicitly declared in the YAML file. With the connections it does three things. Firstly it provides the controller with the information about which stations are connected to each other and with which transmission line (specified in a new connection key: 'line_id'). Secondly it re-routes the connections of the NCP's from the stations to the controller. Lastly it blocks the connections between stations. In terms of data flow this connection is irrelevant. Stations only take input from the controller. This connection, like the NCP-station connection only exists in the YAML file to provide topological information.

Finally, the controller takes in all the information from the YAML file, and the transmission line specifications from a separate CSV file (because the transmission lines are not models in Illuminator). It uses this information to construct a network in Pandapower, with line reactances and capacities, the maximum output capacities of generators and their emission cost, and a very high cost for tapping from an external connection to discourage that type of behavior.

5.4 Peripheral Models

Peripheral models refers to the models that represent active grid components, such as generators and loads. Their job is to take in information from either the controller model or external sources (such as CSV files or physical peripherals), perform calculations, and provide outputs, either to the controller, or to the monitor, or both. They do this for every time step of the simulation. For this project, the purpose of the peripheral models is two-fold:

- Contribute to the simulation of the Dutch grid by calculating the relevant quantities and interacting with the controller model.
- Provide input and output fields as part of the interface between the Illuminator software and the physical visualizers developed by sub-group A1 [3].

In order to fulfill the first requirement, all the peripheral models relevant to the purely digital simulation of the Dutch grid (PV, Wind, Load, Nuclear, Fossil generator, External grid connection) are equipped with the following features. All of those models have a 'name' parameter (not for the type, but for the specific instance), of which the value should be identical to the model name in the YAML file. This way the model is aware of which instance it is.

In the Dutch grid simulation, all model instances of PV and Load read from one CSV file per

type (so one for PV and one for Load). The PV CSV file contains one fraction for every time step. The fraction is of the rated capacities of the PV instances. All instances receive the same fraction, but they all have their own unique rated capacity. The PV models themselves multiply the fraction with their rated capacity (a parameter) and introduce random variation from a Laplace distribution with the input from the CSV files as the mean. This becomes the power information output to the controller. The Load models work almost exactly the same, except they get the total national load from the CSV file, and multiply that with their own fraction, representing how much of the total national load they consume. For wind, two separate models are created that implement the Brownian motion calculation explained in section 3.2.2. One for onshore wind, and one for offshore. These models provide the same service to the Wind models as the PV CSV file provides to the PV models.

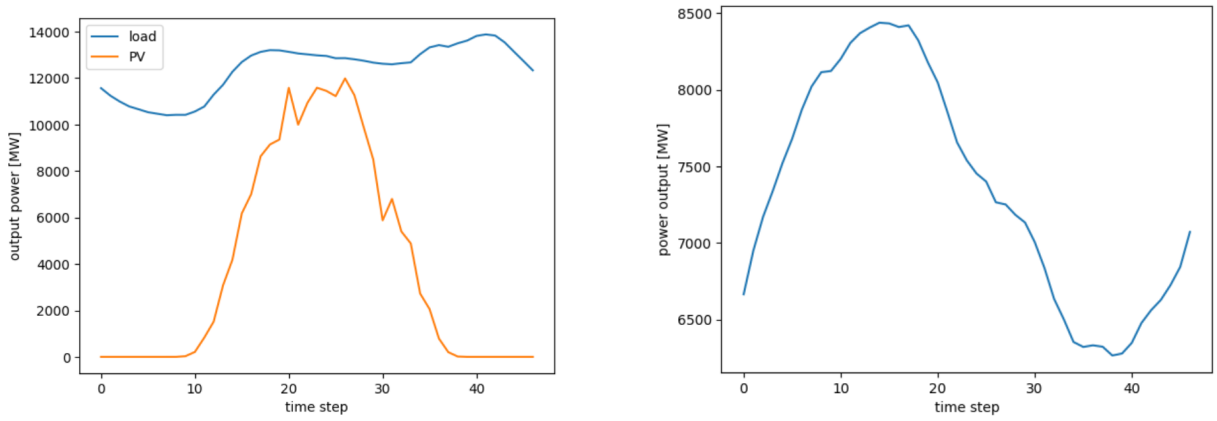
In order to model the Dutch grid, two peripheral models are added to Illuminator: Fossil generator and Nuclear generator. The fossil generator model is a controllable model that takes the drawn power as input and calculates its CO2 emissions based on that. Besides the 'name' parameter, it has two additional parameters: 'emission rate' and 'maximum power'. Both of these parameters are communicated to the controller during initialization. The controller uses these parameters in its Optimal Power Flow (OPF) calculation. The nuclear generator produces a constant power and calculates the waste produced by that power, low-level, intermediate-level, and high-level.

In order to fulfill the visualization requirement, every model that has a physical visualizer has either an input field or output field to connect to that visualizer [models that have input; models that have output] This means that some models need additional output fields that are not relevant to the simulation. Specifically, the fossil generator is a controllable peripheral, which does not need a power information output, because the controller calculates how much power the generator produces and sends that to the generator, and all derived quantities to other models that need the information (such as transmission line power). Still, in order for the fossil generator model to provide information to the mist generator (a physical component visualizing emissions) [3], it has a power output field.

Chapter 6

SIMULATION RESULTS

The system was tested on the model in figure 3.1a for one average day in April, with a time interval of 30 minutes. Simulating this day yielded the following results. Figure 6.1 shows the load, solar and wind data.

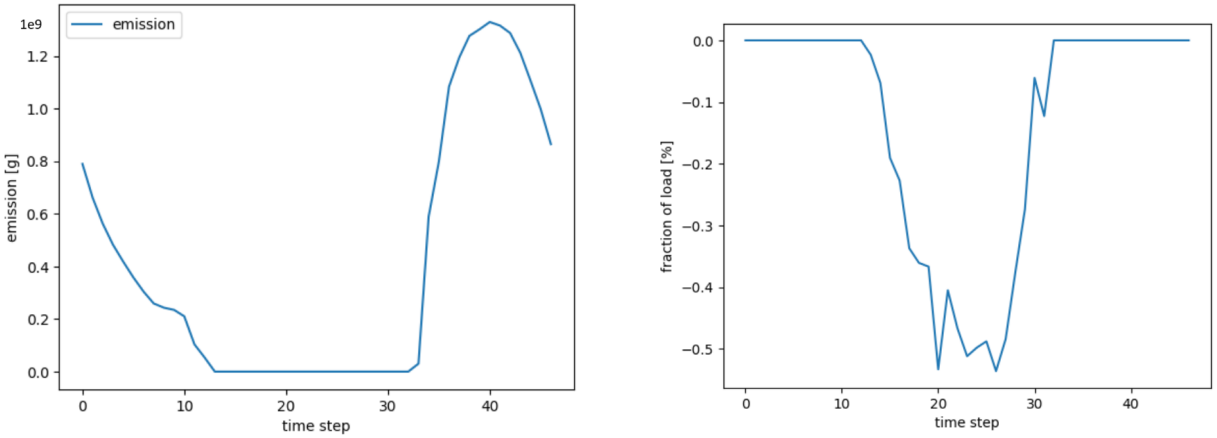


(a) Load and solar data

(b) Wind data

Figure 6.1: Load, solar, and wind data

Figures 6.2a and 6.2b show the emission data and grid dependence data respectively. Negative dependence means that the Dutch grid is supplying power to the external grids. The fraction is that of the total demand in that time step.



(a) Emission data

(b) Dependence data; perunage

Figure 6.2: Emission and dependence data

Figure 6.3 shows the maximal transmission line current occupation for every time step. This is not always the same transmission line. Transformers are also included in this analysis. The highest point,

at time step 23, is reached by the transformer at the Ens station in the Noord-Oostpolder (see figure Figure 3.1a).

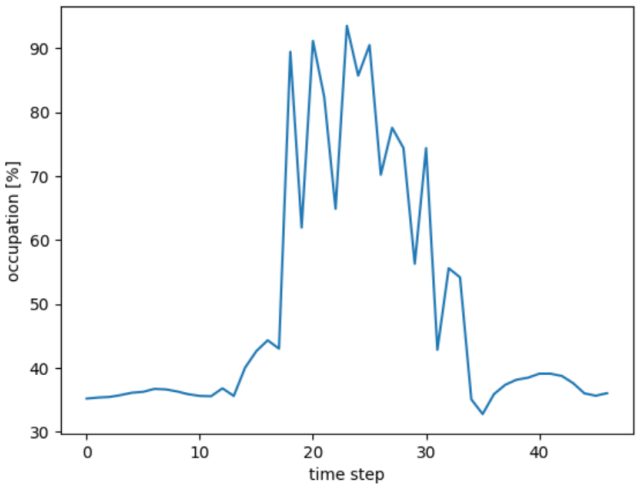


Figure 6.3: Transmission line occupation data

Chapter 7

DISCUSSION OF RESULTS

The first thing to note is the high total wind power output at the wind peak. While this is still within the bounds of the maximum wind capacity of The Netherlands, which around 10 MW [46], it is quite high considering that the occupied capacity usually doesn't exceed 40% in practice [47], due to realistic wind speeds. This would amount to 4 MW. This is significantly lower than the almost 8,5 MW seen in figure 6.1b. This inconsistency with reality will somewhat affect the validity of the remaining results.

In the emissions graph, the effect of the solar power can clearly be seen. During the middle of the day, all of the generators are switched off, because all of the demand is satisfied by renewable sources. In the evening, when the sun has set and wind power is *coincidentally* low (compared to the rest of the day), the generators have to pick up the slack. At the emission peak, over 1,2 million kilos of CO2 are emitted per 30 minutes.

In the dependence graph, it can be seen that at no point does the Dutch grid receive power from external grid connections, at least in this simulation. This results from two things. Firstly, in the simulation, the cost of external energy is set significantly higher than the cost of energy from any Dutch generator, which means that using external energy is always a last resort. Secondly, in this scenario the total demand never exceeds the total domestic capacity. Moreover, during the day, renewable sources produce so much power that a significant excess can actually be supplied to the external grid connections. Around the solar peak, an amount of power equivalent to 50% of the total demand at that time is supplied to the external connections, meaning that the renewable sources are producing power equivalent to 1,5 times the total demand at that time.

In the transmission line occupation graph it can be seen that the high solar penetration does come at a cost. During the night hours, when a significant fraction of the power is supplied by the generators, and is thus controllable, the power can be routed such that no transmission line is ever occupied beyond 50% of its maximum current capacity. However, during the day, when an enormous amount of power is generated by non-controllable solar panels, the occupation of some transmission lines or transformers exceeds 90%. This exposes a bottle neck which will become relatively tighter as more renewables are integrated in the Dutch grid.

From these findings, some conclusions can be drawn about the green transition of the Dutch grid. Firstly, the fact that there is sometimes an excess of renewable power and sometimes a shortage suggests that there isn't necessarily too little renewable generation capacity, but rather that the timing of the generation isn't optimal. This *could* be solved by making more use of consistent sources of green energy, such as nuclear or geothermal, perhaps importing this from neighboring countries. However, this would lead to unused capacity anytime solar and wind generate a lot of power, let alone less independence. An arguably better way to solve this problem would be to introduce local short term storage methods to neighborhoods, that store the excess solar energy and discharge when there is a generation shortage in the evening. Moreover, further integration of renewable energy necessitates an increase in the power transmission capacity in some areas of the grid.

Chapter 8

CONCLUSION

This thesis discussed the development of a model for the Dutch electric grid. In addition, this model was simulated to gain insight into its performance. Separate conclusions for these two parts are given.

8.1 Model

A comprehensive model of the Dutch national grid was developed using official public data. The model includes all of the transmission lines at or above 220 kV. The model aggregates certain data by province and ignores some smaller electricity sources. Due to a lack of data, a lot of assumptions had to be made. Some of the stochastic behavior of the grid was modeled with varying degrees of accuracy. The national average offshore and onshore wind power was modeled accurately with a newly developed stochastic differential equation. The provincial fluctuations of certain sources was not modeled accurately due to time restrictions. The load variations between different provinces was not modeled accurately due to a lack of data. Overall, the model is as comprehensive as was possible considering the time frame and the available data. A simplified model was also made that will be demonstrated during the thesis defense.

When comparing the final model to the requirements as outlined in chapter 2, it becomes clear that all of the initial requirements were successfully met. It is true that the attempts at simulating the model on Illuminator were unsuccessful, but that had nothing to do with the model itself. However, this does not mean that the model is free from shortcomings.

8.1.1 Shortcomings and Future Recommendations

Regarding the grid model, there are several shortcomings to point out. Firstly, voltages below 220 kV are not modeled at all. One could get the length of all these lines from the grid map, and use that to estimate the reactances of the lines. This is quite some work, but Zomerdijk et al. have shown that this method is possible[8]. Second, as shown in Figure 3.12b, the electrical loads during weekends is significantly different than that of weekdays. This is not part of our model. Third, currently only Flevoland is split into smaller sections. But other provinces also have significant reactances, such as Noord-Brabant. Reducing aggregation would increase model accuracy. This would add a lot of work due to the need to sum up values from the municipalities, but it would make the behavior of distributed sources and loads more accurate.

Perhaps the most significant shortcoming of the model regards local fluctuations. The provincial variations of both onshore wind and solar can be better stochastically modeled using Equation 3.3. As stated in section 3.2.2, this requires finding separate probability distribution functions for each province. We believe that this is possible by fitting polynomials to the histograms. A more accurate model of the provincial variations would also allow for much greater time resolutions. This is because the fluctuations no longer need to be independent and identically distributed. There is also the added value from knowledge gained by solving these statistical problems.

Future efforts should consider spending time researching and modeling lower voltage transmission lines, differentiating between weekends and weekdays, reducing aggregation, and accurately modeling local fluctuations. The model accuracy can additionally be improved if more data can be found, such as the electricity load per province.

8.2 Simulation

In order to be able to simulate the model of the Dutch grid, the capabilities of Illuminator were expanded, mainly to enable it to simulate a decentralized power network. Pandapower was used to calculate DC Optimal Power Flow, with an emphasis on optimizing for emissions and independence, while respecting the transmission line power constraints. This capability was implemented in such a way that setting up and running the simulation remains intuitive. The user only has to specify the grid topology. The data routing between the models is handled automatically. Simulation showed that the Dutch grid could score perfectly on independence. However, it is not yet capable of satisfying the demand with renewables alone, due to the time discrepancy between peak supply and peak demand. In some areas, the transmission components are pushed close to their power limit around the solar peak, exposing a bottleneck for further renewables integration.

In the future, some changes might be made to the simulation. Firstly, incorporating markets will enable this simulator to evaluate the performance of grids in the context of free energy markets, as opposed to what is essentially full governmental control. Secondly, if more grid parameters can be found, it might be worth incorporating AC Power Flow analysis into the simulation, in order to model the grid more accurately. Ramp rates might be added to the generator model to incorporate that information into the power routing. Logic for storage elements are yet to be incorporated in the controller. On the usability side, it might be worth incorporating the transmission line specification in the YAML file, instead of a separate CSV file. This way the user need only make changes to one file before simulating. It might also be interesting to modify Illuminator such that it can connect to the Dutch national load API [21]. Lastly, this simulator is yet to be tested in a physical simulation, together with the software and visualization components of groups A1 [3] and A2 [4].

BIBLIOGRAPHY

[1] IPCC, *Climate Change 2022: Mitigation of Climate Change. Contribution of Working Group III to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*, P. Shukla, J. Skea, R. Slade, *et al.*, Eds. Cambridge, UK and New York, NY, USA: Cambridge University Press, 2022. [Online]. Available: http://dx.doi.org/10.1017/9781009157926.

[2] J. Groen, D. Georgiadi, and M. Cvetkovic, "The illuminator: An open source energy system integration development kit," Delft University of Technology, Intelligent Electrical Power Grids (IEPG) Group, Technical Report / Project Page, 2025. [Online]. Available: https://www.tudelft.nl/ewi/over-de-faculteit/afdelingen/electrical-sustainable-energy/intelligent-electrical-power-grids-iepg-group/projects/current-projects/illuminator.

[3] J. I. Libier and R. Bonouvrie, "Visualization of energy system modules," Delft University of Technology, Tech. Rep., 2025, Bachelor's Thesis.

[4] B. Dorland and A. Vermeer, "Embedded system communication for the hardware integration of the illuminator simulation tool," Delft University of Technology, Tech. Rep., 2025, Bachelor's Thesis.

[5] kabinet-Schoof, "Energietransitie, leveringszekerheid en klimaatadaptatie," in *Regeerprogramma kabinet-Schoof*, Rijksoverheid, 2024, pp. 55–65. [Online]. Available: https://www.rijksoverheid.nl/documenten/publicaties/2024/09/13/regeerprogramma-kabinet-schoof.

[6] TenneT and Gasunie, "Infrastructure outlook 2050," TenneT and Gasunie, Tech. Rep., 2019, Accessed: 4 June 2025. [Online]. Available: https://www.tennet.eu/nl-en/infrastructure-outlook-2050.

[7] B. Koirala, S. Hers, G. Morales-España, Ö. Özdemir, J. Sijm, and M. Weeda, "Integrated electricity, hydrogen and methane system modelling framework: Application to the dutch infrastructure outlook 2050," *Applied Energy*, vol. 289, p. 116 713, 2021, ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2021.116713. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261921002336.

[8] W. Zomerdijk, D. Gusain, P. Palensky, and M. Cvetkovic, "Open data based model of the dutch high-voltage power system," in *Proceedings of the 2022 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, vol. 2022-October, IEEE, 2022, pp. 1–6. [Online]. Available: http://dx.doi.org/10.1109/ISGT-Europe54678.2022.9960703.

[9] DIgSILENT. "Powerfactory." (2025), [Online]. Available: https://www.digsilent.de/en/powerfactory.html.

[10] etap. "Etap power simulator." (2025), [Online]. Available: https://etap.com/power-simulator.

[11] Electricity Maps, *Live 24/7 co2 emissions of electricity consumption*, Accessed: 3 June 2025, 2025. [Online]. Available: https://app.electricitymaps.com/map/72h/hourly.

[12] OFFIS, *Mosaik: A flexible smart grid co-simulation framework*, Accessed: 2 June 2025, 2025. [Online]. Available: https://mosaik.offis.de/.

[13] TenneT, *The grid*, Accessed: 3 Jun 2025, 2025. [Online]. Available: https://www.tennet.eu/grid#crosslink-operations.

[14] Rijksoverheid, *Artikel 5.156 besluit kwaliteit leefomgeving*, Accessed: 13 June 2025, 2021. [Online]. Available: https://wetten.overheid.nl/jci1.3:c:BWBR0041313&hoofdstuk=5&afdeling=5.1&paragraaf=5.1.7&sub-paragraaf=5.1.7.3&artikel=5.156&z=2024-12-21&g=2024-12-21.

[15] P. H. Schavemaker and L. van der Sluis, *Electrical Power System Essentials*, Second. Hoboken, NJ: John Wiley & Sons, 2008, ISBN: 9780470987681.

[16] TenneT, *Tennet netkaart onshore nederland*, Accessed: 3 Jun 2025, 2024. [Online]. Available: https://www.tennet.eu/nl/de-elektriciteitsmarkt/netkaarten.

[17] TenneT, *Tennet netschema van het nederlandse hoogspanningsnet*, Accessed: 3 Jun 2025, 2019. [Online]. Available: https://www.tennet.eu/nl/de-elektriciteitsmarkt/transparantie-nederland/netschema.

[18] TenneT, *Tennet assets hoogspanning (arcgis online web map)*, Accessed: 14 June 2025, 2025. [Online]. Available: https://www.arcgis.com/home/item.html?id=02d4cfaf2a9241b68f9520748081d7e9.

[19] ENTSO-E, *Grid map*, Accessed: 15 June 2025. [Online]. Available: https://www.entsoe.eu/data/map/.

[20] TenneT, *Overview of 380kv and 220kv grid components*, Jan. 2023. [Online]. Available: https://www.tennet.eu/node/585.

[21] European Network of Transmission System Operators for Electricity, *Entso-e transparency platform*, Accessed: 3 Jun 2025, 2025. [Online]. Available: https://transparency.entsoe.eu/dashboard/show.

[22] CBS, *Elektriciteit en warmte; productie en inzet naar energiedrager*, Accessed: 13 June 2025, 2024. [Online]. Available: https://opendata.cbs.nl/statline/#/CBS/nl/dataset/80030ned/table?dl=A76D0.

[23] Min. van Klimaat en Groene Groei, R.d. voor Ondernemend Nederland, *Nationaal plan energiesysteem*, Accessed: 26 Apr 2025, 2024. [Online]. Available: https://www.rvo.nl/onderwerpen/energiesysteem/nationaal-plan-energiesysteem.

[24] Min. van Economische Zaken en Klimaat, *Integrale effectenanalyse Programma Energiehoofdstructuur 2023 - Ontwikkeling energiehoofdinfrastructuur 2030-2050*, J. Vendrik, R. van Ooij, M. Deen, and M. de Sain, Eds. Rijksoverheid, 2023. [Online]. Available: https://www.rijksoverheid.nl/documenten/rapporten/2023/06/02/pondera-en-ce-delft-integrale-effectenanalyse-programma-energiehoofdstructuur-2023-rapport-2-6-2023.

[25] Vattenfall, *Prinses ariane*, Accessed: 2025-06-13. [Online]. Available: https://powerplants.vattenfall.com/prinses-ariane/.

[26] Windpark Fryslân, *Windpark fryslân*, Accessed: 2025-06-13. [Online]. Available: https://www.windparkfryslan.nl/.

[27] Windpark Noordoostpolder, *Windpark noordoostpolder*, Accessed: 2025-06-13. [Online]. Available: https://www.windparknoordoostpolder.nl/.

[28] Rijksoverheid, *Windenergie op zee*, Accessed: 14 June 2025, 2025. [Online]. Available: https://www.rijksoverheid.nl/onderwerpen/duurzame-energie/windenergie-op-zee.

[29] Rijksoverheid, *Regionale klimaatmonitor*, Accessed: 15 June 2025, 2025. [Online]. Available: https://klimaatmonitor.databank.nl/home.

[30] N. E. Dashboard, *Nationaal energie dashboard*, Accessed: 26 April 2025, 2024. [Online]. Available: https://ned.nl/.

[31] IPCC, *Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*, V. Masson-Delmotte, P. Zhai, A. Pirani, *et al.*, Eds. Cambridge, UK and New York, NY, USA: Cambridge University Press, 2021, vol. In Press. [Online]. Available: http://dx.doi.org/10.1017/9781009157896.

[32] M. Bruinsma and M. Nauta, "Ketenemissies elektriciteit. actualisatie elektriciteitsmix 2021," *CE Delft*, 2023. [Online]. Available: https://ce.nl/publicaties/ketenemissies-elektriciteit-actualisatie-elektriciteitsmix-2021/.

[33] J. D. Sterman, L. Siegel, and J. N. Rooney-Varga, "Does replacing coal with wood lower co2 emissions? dynamic lifecycle analysis of wood bioenergy.," *Environmental Research Letters*, vol. 13, no. 1, p. 015 007, Jan. 2018. [Online]. Available: https://doi.org/10.1088/1748-9326/aaa512.

[34] IAEA, *Radioactive waste and spent fuel management*, Accessed: 3 Jun 2025, 2025. [Online]. Available: https://www.iaea.org/topics/radioactive-waste-and-spent-fuel-management.

[35] EPZ, *Jaarverslag 2023: 50 jaar positieve energie uit borssel*, Accessed: 3 Jun 2025, 2025. [Online]. Available: https://www.epz.nl/wie-wij-zijn/publicaties/#:~:text=jaarverslag/.

[36] RWE, *Eemshavencentrale*, Accessed: June 15 2025, 2025. [Online]. Available: https://benelux.rwe.com/locaties-en-projecten/kolencentrale-eemshaven/.

[37] CBS, *Grote batterijen voor opslag van elektriciteit*, Accessed: June 15 2025. [Online]. Available: https://opendata.cbs.nl/statline/#/CBS/nl/dataset/85929NED/table?ts=1750036357330.

[38] M. Jacomy, T. Venturini, S. Heymann, and M. Bastian, "Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software," *PLOS ONE*, vol. 9, no. 6, pp. 1–12, Jun. 2014. DOI: 10.1371/journal.pone.0098679. [Online]. Available: https://doi.org/10.1371/journal.pone.0098679.

[39] P. Schavemaker and L. Van der Sluis, *Electrical power system essentials*. John Wiley & Sons, 2025.

[40] Tennet, *Overview of the electricity markets*, Accessed: 30 June 2025, 2025. [Online]. Available: https://www.tennet.eu/overview-electricity-market.

[41] Tennet, *Market types*, Accessed: 30 June 2025, 2025. [Online]. Available: https://www.tennet.eu/market-types.

[42] N. Energies, *Dispatching power plants*, Accessed: 30 June 2025, 2025. [Online]. Available: https://nanoenergies.eu/knowledge-base/dispatching-power-plants.

[43] T. J. Overbye, X. Cheng, and Y. Sun, "A comparison of the ac and dc power flow models for lmp calculations," in *37th Annual Hawaii international conference on system sciences, 2004. Proceedings of the*, IEEE, 2004, 9–pp.

[44] L. Thurner, A. Scheidler, F. Schäfer, *et al.*, "Pandapower—an open-source python tool for convenient modeling, analysis, and optimization of electric power systems," *IEEE Transactions on Power Systems*, vol. 33, no. 6, pp. 6510–6521, 2018. DOI: 10.1109/TPWRS.2018.2829021.

[45] A. Zhang and K. Rahimatulla, *Illuminator with pandapower*, Accessed: 30 June 2025, 2025. [Online]. Available: https://github.com/Kyr-updog/Illuminator-BAP-EE-2025/tree/Kyr.

[46] CBS, *Hernieuwbare energie in nederland 2023*, Accessed: 30 June 2025, 2024. [Online]. Available: https://www.cbs.nl/nl-nl/longread/rapportages/2024/hernieuwbare-energie-in-nederland-2023/4-windenergie#:~:text=Ontwikkelingen,was%20voor%203%20978%20megawatt..

[47] CBS, *Windenergie op land; productie en capaciteit per provincie*, Accessed: 30 June 2025, 2025. [Online]. Available: https://opendata.cbs.nl/statline/#/CBS/nl/dataset/70960NED/table.

[48] A. Einstein, "Über die von der molekularkinetischen theorie der wärme geforderte bewegung von in ruhenden flüssigkeiten suspendierten teilchen," *Annalen der Physik*, vol. 322, no. 8, pp. 549–560, 1905, For English translation see: https://archive.org/details/investigationont0000albe/page/6/mode/2up. [Online]. Available: https://doi.org/10.1002/andp.19053220806.

[49] A. Einstein, "Zur theorie der brownschen bewegung," *Annalen der Physik*, vol. 324, no. 2, pp. 371–381, 1906, For English translation see: https://archive.org/details/investigationont0000albe/page/6/mode/2up. [Online]. Available: https://doi.org/10.1002/andp.19063240208.

[50] W. Coffey and Y. Kalmykov, *The Langevin Equation: With Applications to Stochastic Problems in Physics, Chemistry and Electrical Engineering, 3rd Edition*. Sep. 2012, ISBN: 978-981-4355-66-7. DOI: 10.1142/8195. [Online]. Available: http://dx.doi.org/10.1142/8195.

Appendix A

JUSTIFICATIONS FOR PROVINCIAL NODE CHOICES

Trivial provinces:
The following provinces have just one substation in the them: Drenthe, Noordoostpolder, Flevopolder, Limburg, Utrecht and Zeeland. In such cases the load is simply connected to the corresponding power node.

Friesland:
Louwsmeer is chosen because it is the closest substation to Leeuwarden, which has a population of 96000. All other settlements have a population of less than 45000. Additionally, Louwsmeer is located in the center of the province, while the other substations are near the provincial border.

Groningen:
The Groningen (city) metropolitan area contains 61% of the provinces population in a small region. The only substation in this region is Vierverlaten, so it was chosen.

Gelderland:
The two substations in Gelderland are Dodewaard and Doetinchem. Dodewaard is close to the major cities of Nijmegen, Ede, Tiel and Wageningen with a total population of 330000. Doetinchem has just one close city: Doetinchem with a population of 46000. Other large settlements mostly lie in the middle of the two. So Dodewaard was chosen.

Noord-Brabant:
The two substations in this province are Eindhoven and Geertruidenberg. They both have about the same population near them. However, geertruidenberg is located more centrally in the province. So Geertruidenberg was chosen.

Noord-Holland:
The Amsterdam metropolitan region contains 84% of the population of the province. There are two substations in this region: Oostzaan and Diemen. Diemen is located near the border with Utrecht, while Oostzaan is close to the center of the province. Therefore, Oostzaan was chosen.

Overijssel:
This province has two substations: Zwolle and Hengelo. Hengelo, is close to 3 major cities: Enschede, Hengelo and Almelo, with a total population of 314000. The Zwolle is only close to Zwolle which has a population of 130000. Other settlements either lie in the middle or have much lower populations. Therefore, Hengelo was chosen.

Zuid-Holland:
This is a highly urbanized province so no real obvious substation can represent the province. Therefore Wateringen was chosen simply because it is the most central of all the substations. It also sits between The Hague and Rotterdam, which are two dense cities.

Appendix B

DERIVATION OF THE STOCHASTIC FUNCTION RECIPE

This chapter discusses the way the recipe for Equation 3.3 was derived. This is done from an engineering perspective, which is somewhat reliant on intuition. This should not be seen as a formal mathematical proof (though it does try to be mathematically precise when possible). In our view, Figure 3.6a and Figure 3.6b are the real proof that this works.

Just like in that section, $F(t)$ denotes the variable that has to be modeled. $X(t)$ denotes the stochastic function that models F. $x(t)$ is a realization of F. The same assumptions are made about $F(t)$: it is smooth, its behavior is strictly stationary, and in regions of equal probability its second derivative has a Gaussian distribution. To simplify notation, dots will be used to denote derivatives, so $\ddot{x} \equiv \frac{\mathrm{d}^2 x}{\mathrm{d}t^2}$.

Start by considering F in the equiprobable region. It is known that the system is in steady state and that the acceleration $\frac{\mathrm{d}^2 F}{\mathrm{d}t^2}$ has a Gaussian distribution centered around 0. A stochastic differential equation modeling this behavior must be of the form Equation B.1.

$$\ddot{X} = \alpha \dot{X} + \beta \sqrt{\mathrm{d}t}\eta(t) \tag{B.1}$$

where α, β are constants and $\eta(t)$ are independent gaussian distributed stochastic variables. Note that $\sqrt{\mathrm{d}t}$ is to ensure that the variance of $\eta(t)$ has units of $\mathrm{d}t$. Since the relationship between $\ddot{X}$ and $\alpha\dot{X}$ is linear, the constants α and β can be easily computed using a linear regression.

Now, Equation B.1 needs to be extended such that it correctly models F over the entire range of F. Assume that the equation is of the form Equation B.2:

$$\ddot{X} = \alpha \dot{X} + \beta \sqrt{\mathrm{d}t}\eta(t) + f(X) \tag{B.2}$$

where f is some function that cannot be dependent on t, as it has to be strictly stationary. Now the important realization is that Equation B.2 is identical to the Langevin equation used to model physical particles undergoing Brownian motion. This should not be surprising. Brownian motion occurs when heavy particles are placed in a thermal reservoir at constant temperature. Such a reservoir must be at thermal equilibrium, just like F is in steady state. Now that the equivalence between Equation B.2 and the Langevin equation has been established, known theorems for the Langevin equation can be applied to Equation B.2.

Imagine a very large ensemble of realizations of X (perhaps these represent different days in a year). This ensemble will always have a constant number of realizations N, a constant range $V = [0, 1]$ and a constant temperature T. That last statement basically says that the random fluctuations of x are at steady state. Since the system has constant NVT, this system is completely analogues to an canonical ensemble.[1]

It is a well known fact that the probability distribution of a canonical ensemble is given by the Boltzmann distribution given by Equation B.3:

$$p(x) \propto e^{-U(x)/T} \tag{B.3}$$

[1] If it suits the reader, they may imagine the atmosphere as a thermal reservoir, which constantly exchanges energy with the specific bit of air around the wind turbine.

where U is a potential energy function, and T is some temperature. Using Equation B.3 to solve for $U(x)$ results in Equation B.4:

$$U = -T \log p(x) + C' \tag{B.4}$$

where C' is some constant. Getting back to Equation B.2, f can be viewed as an external *force* that *pushes* X away from certain values and toward others. In other words, the problem comes down to finding the correct $f(x)$ such that it pushes the probability density function of X to match that of F. In the physical case, f actually corresponds to force, so it is given by Equation B.5:

$$f = -\frac{\mathrm{d}U}{\mathrm{d}x} = T\frac{\mathrm{d}}{\mathrm{d}x}\log[p(x)] \tag{B.5}$$

The temperature can be found using the fluctuation-dispersion relation in Equation B.6:

$$T = \frac{\beta^2}{2\alpha} \tag{B.6}$$

Filling Equation B.5 and Equation B.6 into Equation B.2 results in Equation B.7.

$$\ddot{X} = \alpha\dot{X} + \beta\sqrt{\mathrm{d}t}\eta(t) + \frac{\beta^2}{2\alpha}\frac{\mathrm{d}}{\mathrm{d}X}\log[p(X)] \tag{B.7}$$

To reiterate: a realization of Equation B.7 will be a continuous function $x(t)$, that as $t \to \infty$, must approach the $p(x)$ distribution. A more rigorous proof of this was first given by Einstein[48], [49], also see [50] for information.

The problem with Equation B.7 is that certain distributions of $p(x)$ only have a finite domain, while the Langevin can result in any real number. This is the case for F as its range is restricted to $[0, 1]$. The solution is to map $\mathbb{R}$ to the interval using a sigmoid function $x = \Lambda(s)$, where s is the result of the Langevin process. The exact form of this sigmoid function is not very important. The crucial part is that $\frac{\mathrm{d}^2\Lambda}{\mathrm{d}x^2} \approx 0$ in the equiprobable region. This is to ensure that the linear regression for α and β is still correct since:

$$\ddot{s} = \alpha\dot{s} \ \wedge \ x = cs \quad \longrightarrow \quad \ddot{s} = \alpha\dot{s} \tag{B.8}$$

when c is constant. A change of variables from x to s will need to be performed on the Langevin equation. This finally results in the equation that was presented in Equation 3.3.

Appendix C

ADDITIONAL TABLES AND FIGURES

Table C.1: Carbon intensity per source type.

Type	Carbon intensity [g CO_2-eq/kWH_e]			
	Operational	Value chain excl. MoP	Value chain from MoP	Total excl. MoP
Biomass	0	68	2	68
Coal	662	147	1	809
Natural gas	353	68	1	421
Nuclear	0	5	2	5
Solar	0	0	62	0
Other	300	62	9	362
Wind	0	0	16	0

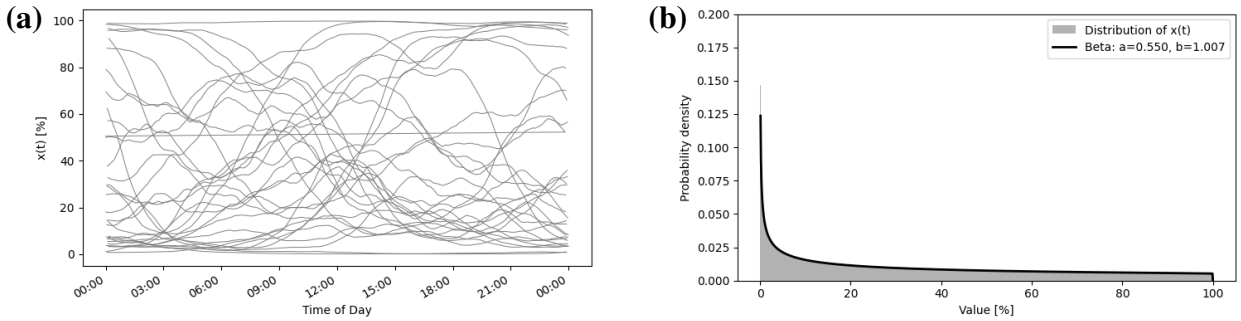


Figure C.1: Graphs depicting the results of generating realizations $x(t)$ from a stochastic differential equation. The stochastic function aims to mimic the capacity utilization factor of onshore wind energy. **(a)** Graph showing a realization $x(t)$ over the period of 31 days. The gray lines are the 31 individual days. **(b)** Graph showing the distribution of $x(t)$ over 100,000 days. The gray area is a high resolution histogram of the values. The black line is the expected probability density function that was measured from the onshore wind data. The parameters are: $a = 0.550, b = 1.007, \alpha = 0.407\ \text{min}^{-1}, \beta = 0.090\ \text{min}^{-2}$.

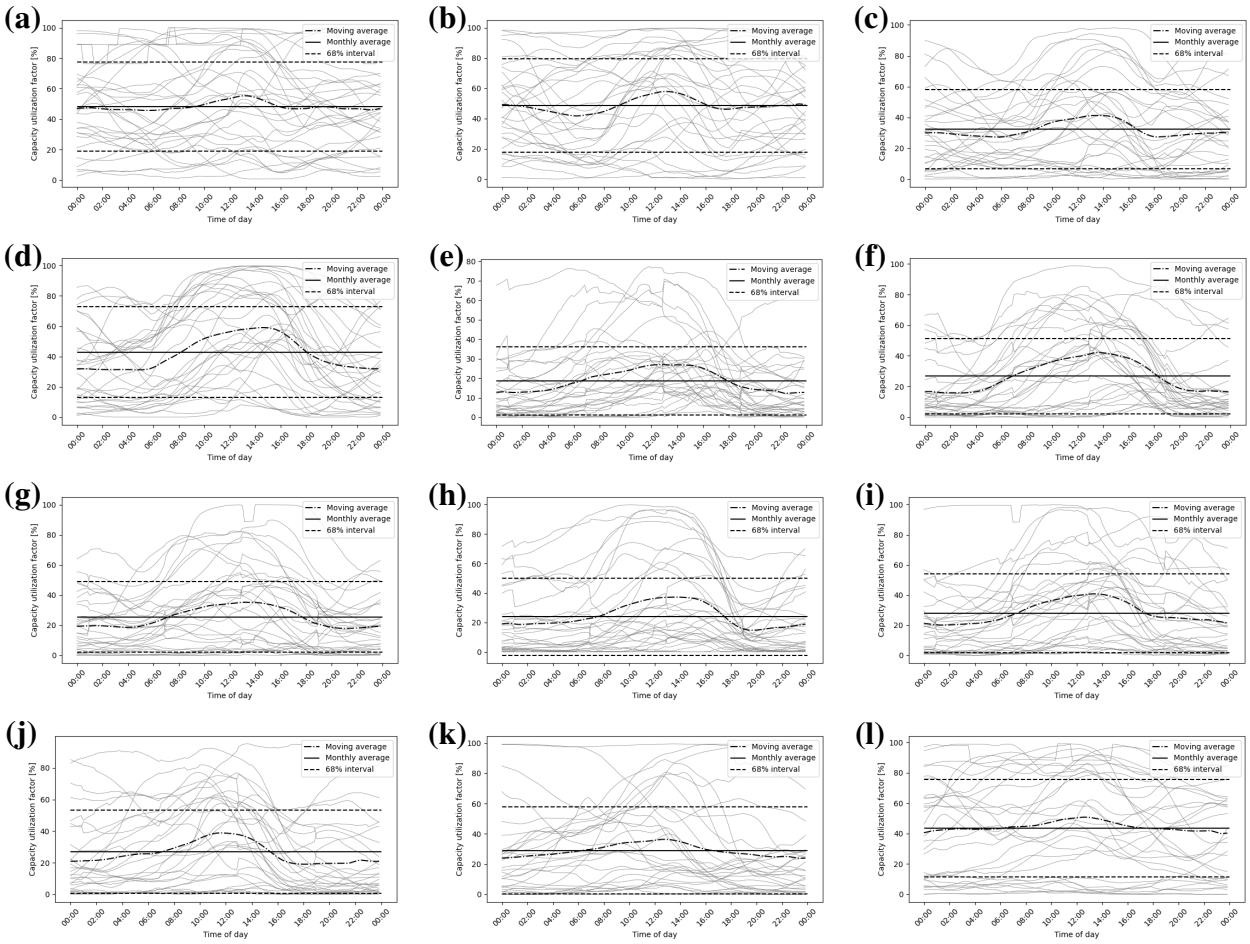


Figure C.2: Graphs depicting the capacity utilization factors of onshore wind throughout the day. The gray lines the individual days. The dash dotted line is the moving average of the gray lines. The solid black line is the average of the entire month. The dashed lines the are the 68% intervals of the instantaneous values compared to the monthly average. Each graph indicates corresponds with a month in 2024. **(a)** January. **(b)** February. **(c)** March. **(d)** April. **(e)** May. **(f)** June. **(g)** July. **(h)** August. **(i)** September. **(j)** October. **(k)** November. **(l)** December.

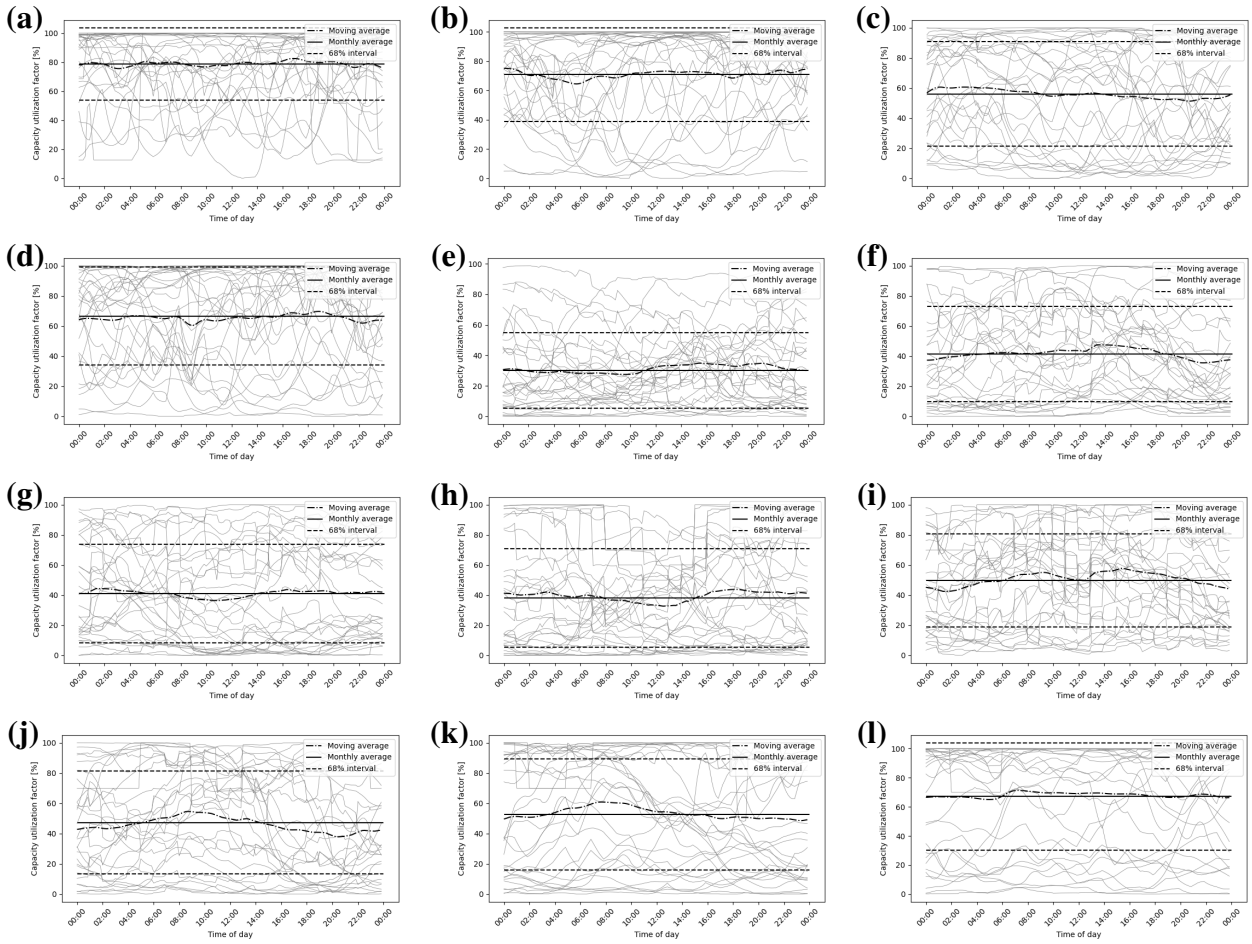


Figure C.3: Graphs depicting the capacity utilization factors of offshore wind throughout the day. The gray lines are the individual days. The dash dotted line is the moving average of the gray lines. The solid black line is the average of the entire month. The dashed lines the are the 68% intervals of the instantaneous values compared to the monthly average. Each graph corresponds with a month in 2024. **(a)** January. **(b)** February. **(c)** March. **(d)** April. **(e)** May. **(f)** June. **(g)** July. **(h)** August. **(i)** September. **(j)** October. **(k)** November. **(l)** December.

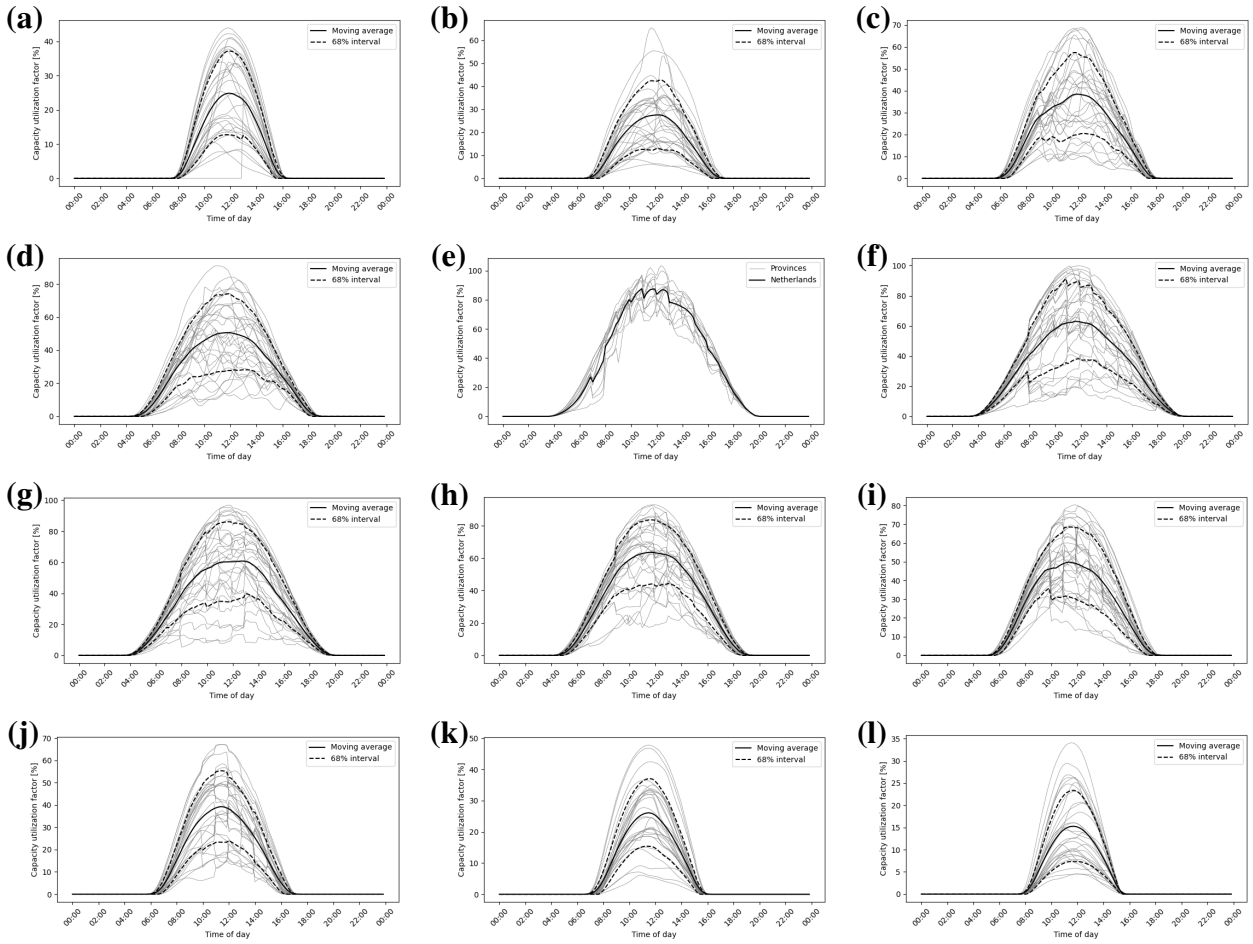


Figure C.4: Graphs depicting the capacity utilization factors of solar throughout the day. The gray lines are the individual days. The dash dotted line is the moving average of the gray lines. The solid black line is the average of the entire month. The dashed lines the are the 68% intervals of the instantaneous values compared to the monthly average. Each graph corresponds with a month in 2024. **(a)** January. **(b)** February. **(c)** March. **(d)** April. **(e)** May. **(f)** June. **(g)** July. **(h)** August. **(i)** September. **(j)** October. **(k)** November. **(l)** December.

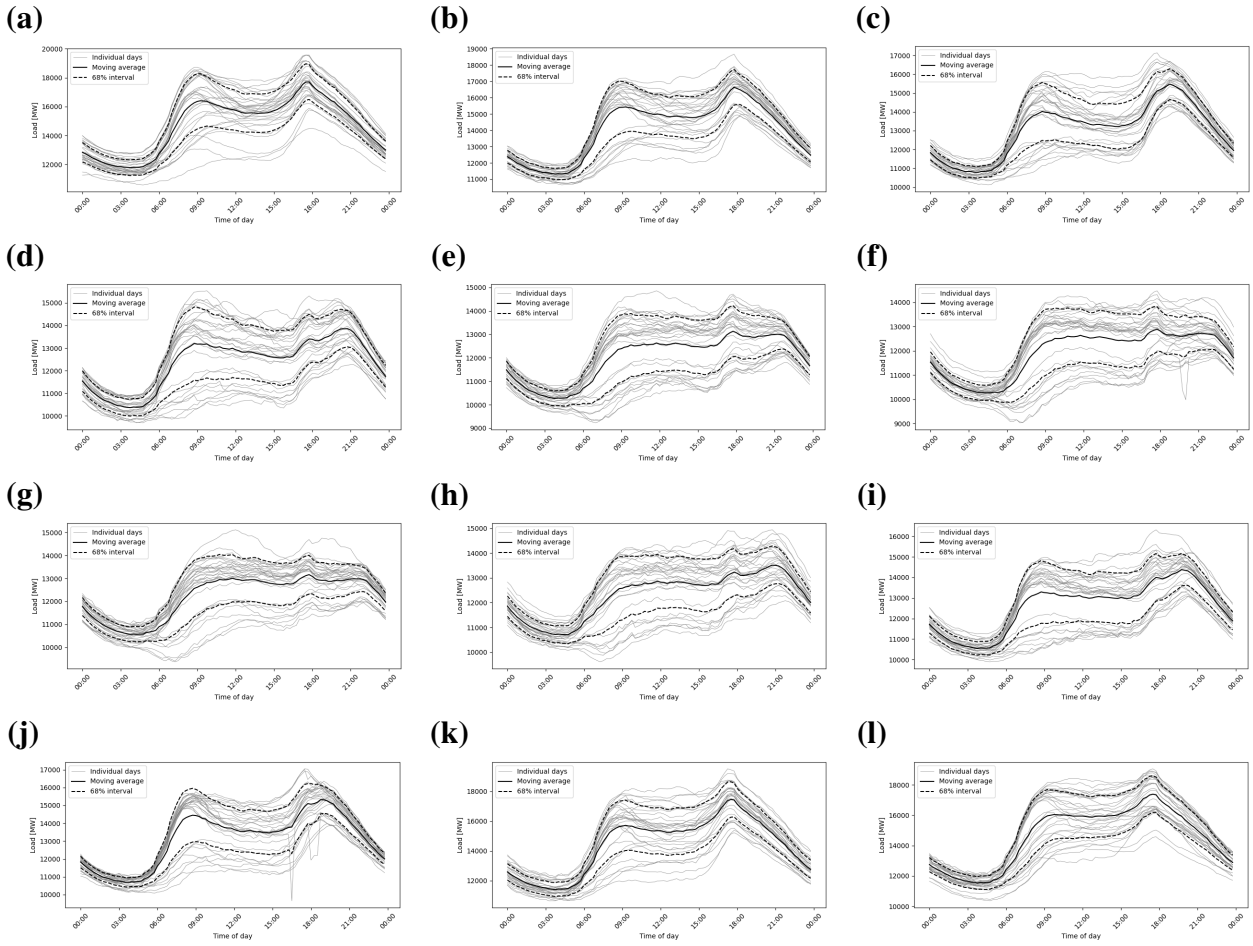


Figure C.5: Graphs depicting the load on the Dutch grid throughout the day. The gray lines are the individual days. The dash dotted line is the moving average of the gray lines. The solid black line is the average of the entire month. The dashed lines the are the 68% intervals of the instantaneous values compared to the monthly average. Each graph corresponds with a month in 2024. **(a)** January. **(b)** February. **(c)** March. **(d)** April. **(e)** May. **(f)** June. **(g)** July. **(h)** August. **(i)** September. **(j)** October. **(k)** November. **(l)** December.

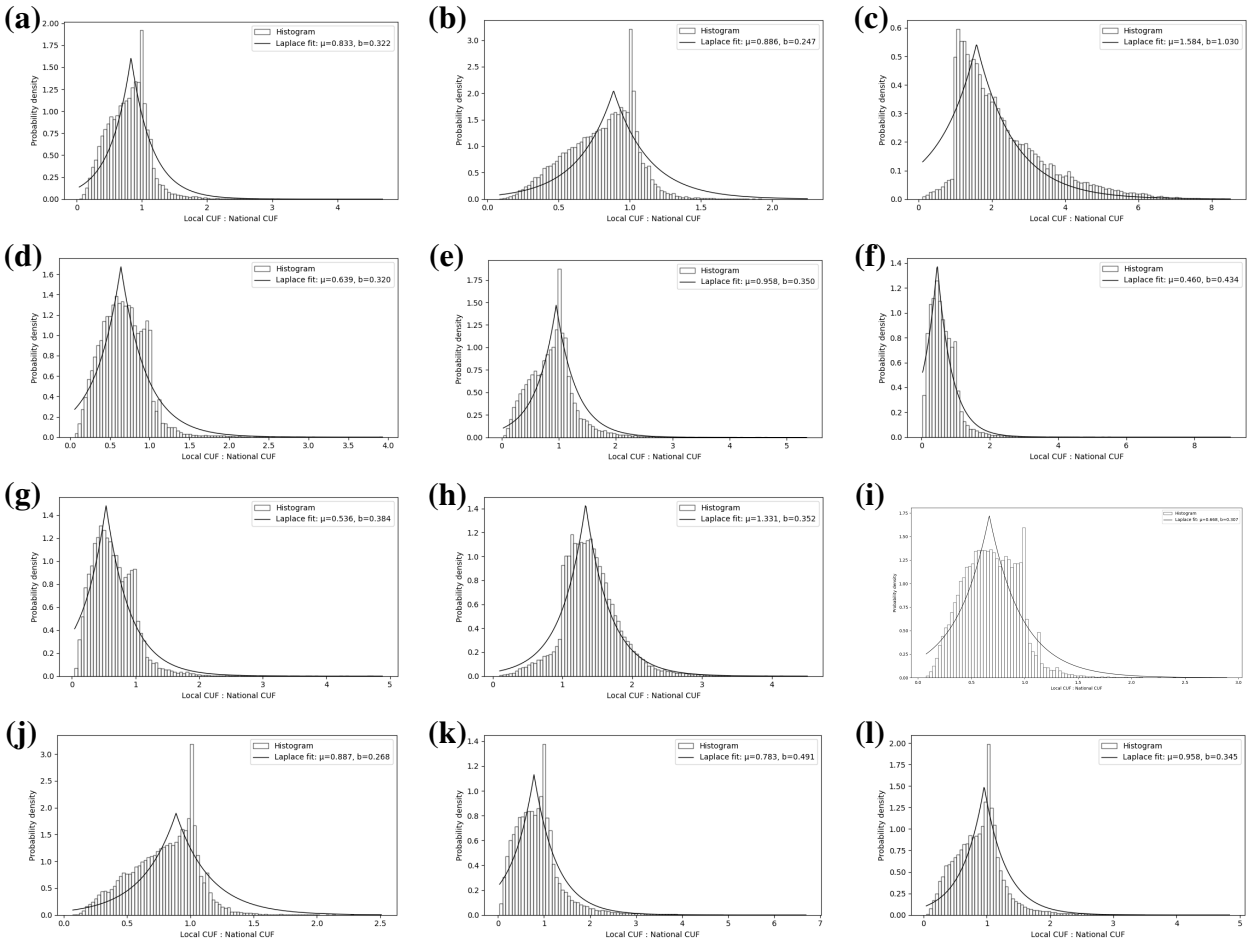


Figure C.6: Histograms depicting the ratio between the local and national capacity utilization factor for onshore wind. The histograms are based on the data from October 2023 to May 2025. The black line is a least-squares fit of a rescaled Laplace distribution. The Laplace distribution is rescaled such that the area under the positive side is unitary. Each graph corresponds with a different province. **(a)** Drenthe. **(b)** Flevoland. **(c)** Friesland. **(d)** Gelderland. **(e)** Groningen. **(f)** Limburg. **(g)** Noord-Brabant. **(h)** Noord-Holland. **(i)** Overijssel. **(j)** Utrecht. **(k)** Zeeland. **(l)** Zuid-Holland.

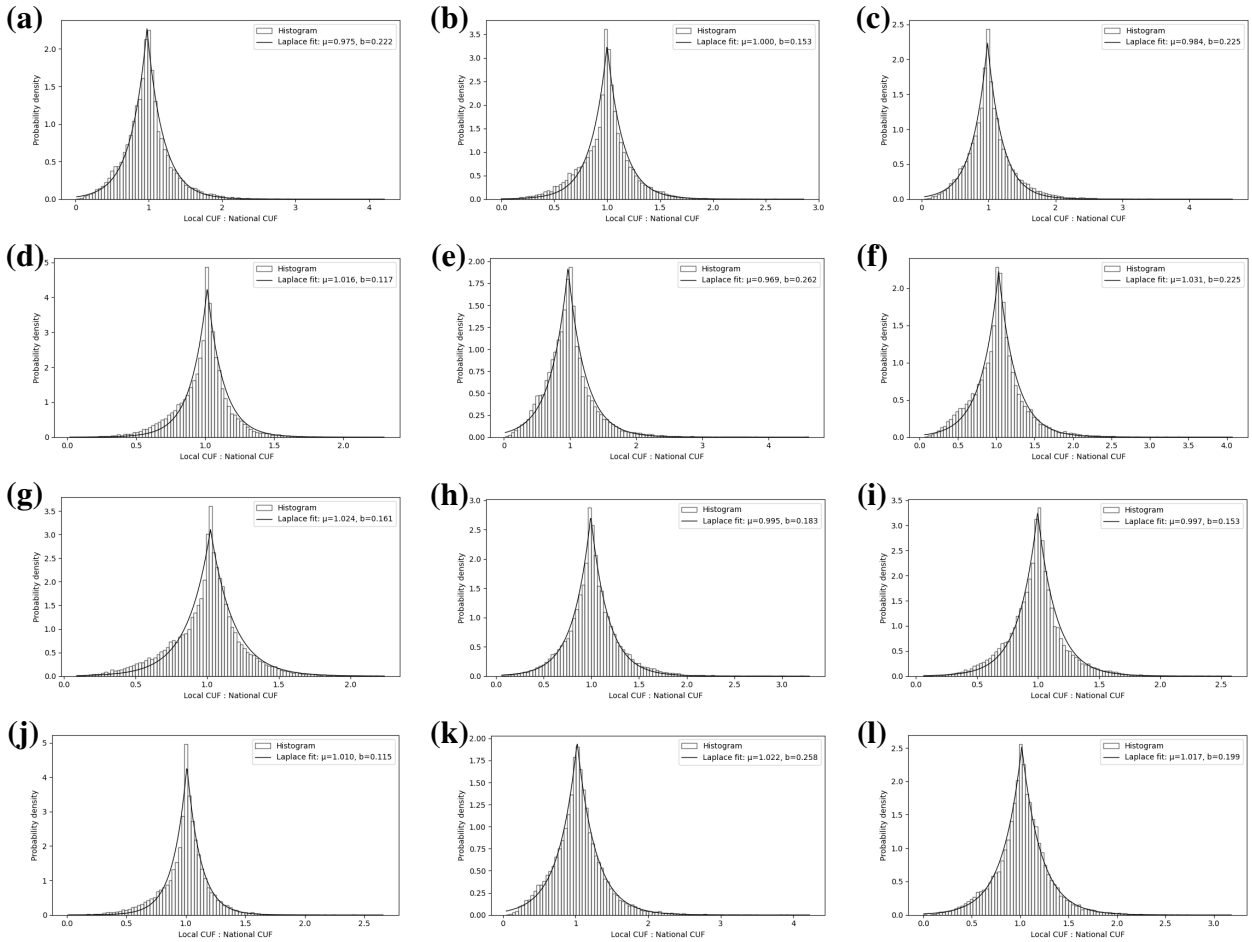


Figure C.7: Histograms depicting the ratio between the local and national capacity utilization factor for solar. The histograms are based on the data from October 2023 to May 2025. The black line is a least-squares fit of a rescaled Laplace distribution. The Laplace distribution is rescaled such that the area under the positive side is unitary. Each graph corresponds with a different province. **(a)** Drenthe. **(b)** Flevoland. **(c)** Friesland. **(d)** Gelderland. **(e)** Groningen. **(f)** Limburg. **(g)** Noord-Brabant. **(h)** Noord-Holland. **(i)** Overijssel. **(j)** Utrecht. **(k)** Zeeland. **(l)** Zuid-Holland.

Appendix D

DIVISION OF LABOR

The division of labor during this project was the following:

- **Andy Zhang** was responsible for the model.
  - Chapter 3, Appendix A, Appendix B, Appendix C.
- **Kyrian Rahimatulla** was responsible for the simulation.
  - Chapter 4, Chapter 5, Chapter 6.

Appendix E

CODE: PANDAPOWER CONTROLLER MODEL

```
from illuminator.builder import ModelConstructor
import pandapower as pp
import pandas as pd
import time as tee
import numpy as np
import pandapower as pp
import pandapower.networks as nw
import pandapower.plotting as plot
import matplotlib.pyplot as plt
colors = ["b", "g", "r", "c", "y"]

# construct the model
class PandaController(ModelConstructor):
    # Define the model parameters, inputs, outputs, and states
    parameters={'peripherals': {},
                'stations': {},
                'ps_connections': {},
                'ss_connections': {},
                'lines_file_path': 'line_specs.csv' # File contains line ID's with
                ↪  their reactances and capacities
                }
    inputs={'ncp_powers': {}
            }
    outputs={} # No outputs
    states={'cp_powers': {},
            'tl_powers': {},
            'one_l_congestion': 0,
            'max_congested': None,
            'max_congested_all': 0,
            'independence': 0,
            'execution_time': 0,
            'incoming': 0,
            'outgoing': 0
            }

    # define other attributes
    time_step_size=1
    time=None

    def __init__(self, **kwargs) -> None:
        """
```

```
        Initialize the analyzer/controller model with the provided parameters.

        Parameters
        ----------
        kwargs : dict
            Additional keyword arguments to initialize the...
        """
        super().__init__(**kwargs)
        self.peripherals = self._model.parameters.get('peripherals')
        self.stations = self.parameters['stations']
        self.ps_connections = self.parameters['ps_connections']
        self.ss_connections = self.parameters['ss_connections']
        self.lines_file_path = self.parameters['lines_file_path']

        # Build graph here !!!
        self.net = pp.create_empty_network()

        # Buses
        for station, kv in self.stations.items():
            pp.create_bus(self.net, vn_kv=kv, name=station)

        # Lines and Transformers
        self.lines_df = pd.read_csv(self.lines_file_path, decimal='.')

        self.transformers_from_stations = {}

        for line_id, connection in self.ss_connections.items():
            text, id = line_id.split('_')
            id = int(id)
            line = self.lines_df[self.lines_df['line_id'] == id]
            max_i_ka = line['capacity'] # Capacity
            if line['tf'].iloc[0] == 0:
                from_bus = pp.get_element_index(self.net, 'bus', connection[0])
                to_bus = pp.get_element_index(self.net, 'bus', connection[1])
                #pp.create_line(self.net, from_bus, to_bus,
                ↪  length_km=line['length_km'], std_type='149-AL1/24-ST1A 110.0',
                ↪  max_loading_percent=50, name=line_id)
                pp.create_line_from_parameters(self.net, from_bus, to_bus,
                ↪  length_km=line['length_km'], r_ohm_per_km=0,
                ↪  x_ohm_per_km=line['X_per_km'], c_nf_per_km=0,
                                               r0_ohm_per_km=0, x0_ohm_per_km=0,
                                               ↪  c0_nf_per_km=0, max_i_ka=max_i_ka,
                                               ↪  name=line_id,
                                               ↪  max_loading_percent=100)
            else:
                kvs = {}
                kvs[connection[0]] = self.stations[connection[0]]
                kvs[connection[1]] = self.stations[connection[1]]
```

```
                high_station = max(kvs, key=kvs.get)
                if high_station == connection[0]:
                    self.transformers_from_stations[line_id] = 'high'
                else:
                    self.transformers_from_stations[line_id] = 'low'
                low_station = min(kvs, key=kvs.get)
                hv_bus = pp.get_element_index(self.net, 'bus', high_station)
                lv_bus = pp.get_element_index(self.net, 'bus', low_station)
                X_ohm = line['length_km'] * line['X_per_km']
                vk = X_ohm * 1000*max_i_ka
                vk_percent = 100 * vk/(1000*line['prim_kv_rating'])
                #pp.create_transformer(self.net, hv_bus, lv_bus, std_type="100 MVA 220/110 kV", max_loading_percent=50, name=line_id)
                pp.create_transformer_from_parameters(self.net, hv_bus, lv_bus, sn_mva=max_i_ka*kvs[high_station], vn_hv_kv=kvs[high_station], vn_lv_kv=kvs[low_station], vkr_percent=0, vk_percent=vk_percent, pfe_kw=0,
                                                      i0_percent=0, vector_group='Dyn', vk0_percent=0, vkr0_percent=0, mag0_percent=0, mag0_rx=0, si0_hv_partial=0, name=line_id, max_loading_percent=100)
        #self.net.line.to_csv('lines.csv')
        # Peripherals
        ncps = ['PV', 'Wind', 'Load'] # Excluding nuclear, because that one gets special treatment
        for name, specs in self.peripherals.items():
            bus_index = pp.get_element_index(self.net, 'bus', specs['station'])
            if specs['type'] in ncps:
                pp.create_load(self.net, bus_index, p_mw=10, controllable=False, name=name) # Negative load is generation
            elif specs['type'] == 'Nuclear':
                pp.create_load(self.net, bus_index, p_mw=specs['rated_pow'], controllable=False, name=name)
            elif specs['type'] == 'Fossil':
                fossil = pp.create_gen(self.net, bus_index, p_mw=10, min_p_mw=0, max_p_mw=specs['rated_pow'], vm_pu=1.0, controllable=True, name=name)
                if specs['fos_type'] == 'coal':
                    pp.create_poly_cost(self.net, fossil, 'gen', cp1_eur_per_mw=specs['coal_emission_rate'])
                elif specs['fos_type'] == 'gas':
                    pp.create_poly_cost(self.net, fossil, 'gen', cp1_eur_per_mw=specs['gas_emission_rate'])
                else:
```

```
                    pp.create_poly_cost(self.net, fossil, 'gen',
                    ↪  cp1_eur_per_mw=specs['bio_emission_rate'])
            elif specs['type'] == 'GridConnection':
                power_limit = specs['connection_capacity']

                connection = pp.create_ext_grid(self.net, bus_index,
                ↪  min_p_mw=-power_limit, max_p_mw=power_limit, name=name)
                #pp.create_pwl_cost(self.net, connection, 'ext_grid',
                ↪  [[-100000,0,0],[0,100000,10000]])
                pp.create_poly_cost(self.net, connection, 'ext_grid',
                ↪  cp1_eur_per_mw=10)

            elif specs['type'] == 'Battery':
                power_limit = specs['max_p']
                battery = pp.create_storage(self.net, bus_index, p_mw=10,
                ↪  max_e_mwh=specs['max_energy'], soc_percent=specs['init_soc'],
                ↪  min_p_mw=-power_limit, max_p_mw=power_limit, controllable=True,
                ↪  in_service=True, name=name) # Update
                ↪  Battery_v3.py!!!!!!!!!!!!!!!!
                pp.create_poly_cost(self.net, battery, 'storage', cp1_eur_per_mw=0)
            else:
                pass
        #self.net.gen.to_csv('gens.csv')
        self.max_congested = ['None', -1]
        """
        plot.create_generic_coordinates(self.net, respect_switches=False)
        sizes = plot.get_collection_sizes(self.net)
        bc = plot.create_bus_collection(self.net, self.net.bus.index,
        ↪  size=sizes['bus'], color='b', zorder=10)
        tlc, tpc = plot.create_trafo_collection(self.net, self.net.trafo.index,
        ↪  color="g", size=sizes['trafo'])
        lcd = plot.create_line_collection(self.net, self.net.line.index,
        ↪  color="grey", linewidths=0.5, use_bus_geodata=True)
        sc = plot.create_bus_collection(self.net, self.net.ext_grid.bus.values,
        ↪  patch_type="rect", size=sizes['ext_grid'], color="y", zorder=11)
        plot.draw_collections([lcd, bc, tlc, tpc, sc], figsize=(8,6))
        plt.savefig('network.png')
        """

    # define step function
    def step(self, time: int, inputs: dict=None, max_advance: int=1) -> None:  #
    ↪  step function always needs arguments self, time, inputs and max_advance.
    ↪  Max_advance needs an initial value.
        """
        Advances the simulation one time step.
        Args:
            time (float): Current simulation time in seconds
            inputs (dict): Dictionary containing input values:
```

```
            -
        max_advance (int, optional): Maximum time to advance in seconds.
        ↪  Defaults to 1.
    Returns:
        float: Next simulation time in seconds
    """
    start = tee.time()

    input_data = self.unpack_inputs(inputs)  # make input data easily
    ↪  accessible
    self.time = time

    ncp_powers = input_data['ncp_powers']
    results = self.control_and_analyze(ncp_powers)

    end = tee.time()

    execution_time = end - start

    results['execution_time'] = execution_time

    self.set_states(results)

    # return the time of the next step (time until current information is
    ↪  valid)
    return time + self._model.time_step_size



def control_and_analyze(self, ncp_powers) -> dict:
    total_power = 0
    total_load = 0
    total_supply_from_grid = 0

    for element in ncp_powers:
        name = list(element.keys())[0]
        if self.peripherals[name]['type'] != 'Nuclear':
            ncp_index = pp.get_element_index(self.net, 'load', name)
            self.net.load.at[ncp_index, 'p_mw'] = -element[name]
            if name == 'PV1':
                incoming = self.net.load.at[ncp_index, 'p_mw']
            if self.peripherals[name]['type'] == 'Load':
                total_load += -element[name]
            total_power += element[name]
        else:
            pass

    pp.rundcopp(self.net, delta=1e-16)
```

```
191
192         cp_powers = {}
193         tl_powers = {}
194         SoCs = {}
195
196         generator_outputs = []
197         for name, specs in self.peripherals.items():
198             if specs['type'] == 'Fossil':
199                 fos_index = pp.get_element_index(self.net, 'gen', name)
200                 fos_results = self.net.res_gen.at[fos_index, 'p_mw']
201                 if name == 'FossilCoal1':
202                     outgoing = fos_results
203                 cp_powers.setdefault(specs['station'], {})[name] = fos_results
204                 generator_outputs.append(fos_results)
205                 total_power += fos_results
206             elif specs['type'] == 'GridConnection':
207
208                 grid_index = pp.get_element_index(self.net, 'ext_grid', name)
209                 grid_results = self.net.res_ext_grid.at[grid_index, 'p_mw']
210                 cp_powers.setdefault(specs['station'], {})[name] = grid_results
211                 total_supply_from_grid += grid_results
212                 total_power += grid_results
213
214             elif specs['type'] == 'Battery': # Manually adjust SoC
215                 bat_index = pp.get_element_index(self.net, 'storage', name)
216                 bat_results = self.net.res_storage.iloc[bat_index]
217                 cp_powers.setdefault(specs['station'], {})[name] =
                    ↪ bat_results['p_mw']
218             else:
219                 pass
220
221
222
223         for station in self.stations.keys():
224             cp_powers.setdefault(station, {})['key'] = 'value'
225
226         independence = total_supply_from_grid/total_load
227         int_max_congested = ['None', -1]
228
229         for line_id, connection in self.ss_connections.items():
230             text, id = line_id.split('_')
231             id = int(id)
232             line = self.lines_df[self.lines_df['line_id'] == id]
233             if line['tf'].iloc[0] == 0:
234                 line_index = pp.get_element_index(self.net, 'line', line_id)
235                 line_results = self.net.res_line.iloc[line_index]
236                 tl_powers.setdefault(connection[0], {})[line_id] =
                    ↪ line_results['p_from_mw']
```

```
237                 congestion = abs(line_results['p_from_mw']) /
                    ↪  (self.net.line.at[line_index, 'max_i_ka'] *
                    ↪  self.stations[connection[0]])
238             else:
239                 trafo_index = pp.get_element_index(self.net, 'trafo', line_id)
240                 trafo_results = self.net.res_trafo.iloc[trafo_index]
241                 if self.transformers_from_stations[line_id] == 'high':
242                     tl_powers.setdefault(connection[0], {})[line_id] =
                        ↪  trafo_results['p_hv_mw']
243                     congestion = abs(trafo_results['p_hv_mw']) /
                        ↪  self.net.trafo.at[trafo_index, 'sn_mva']
244                 else:
245                     tl_powers.setdefault(connection[0], {})[line_id] =
                        ↪  trafo_results['p_lv_mw']
246                     congestion = abs(trafo_results['p_lv_mw']) /
                        ↪  self.net.trafo.at[trafo_index, 'sn_mva']
247             if line_id == 'line_1':
248                 one_l_congestion = congestion
249             if congestion > int_max_congested[1]:
250                 int_max_congested = [line_id, congestion]
251
252         for station in self.stations.keys():
253             tl_powers.setdefault(station, {})['key'] = 'value'
254
255         max_all = int_max_congested[1]
256         if max_all > self.max_congested[1]:
257             self.max_congested = int_max_congested
258
259         self.net.load.to_csv('res_load.csv')
260         self.net.res_line.to_csv('res_line.csv')
261         self.net.res_bus.to_csv('res_bus.csv')
262         self.net.res_trafo.to_csv('res_trafo.csv')
263         self.net.res_gen.to_csv('res_gen.csv')
264         with open("res_cost.txt", "w") as text_file:
265             text_file.write("Cost: %s" % int(self.net.res_cost))
266
267         re_params = {'cp_powers': cp_powers, 'tl_powers': tl_powers,
        ↪  'one_l_congestion': one_l_congestion, 'max_congested':
        ↪  self.max_congested[0], 'max_congested_all': max_all, 'independence':
        ↪  independence, 'incoming': 0, 'outgoing': outgoing} # Also battery SoC
268
269         return re_params
```