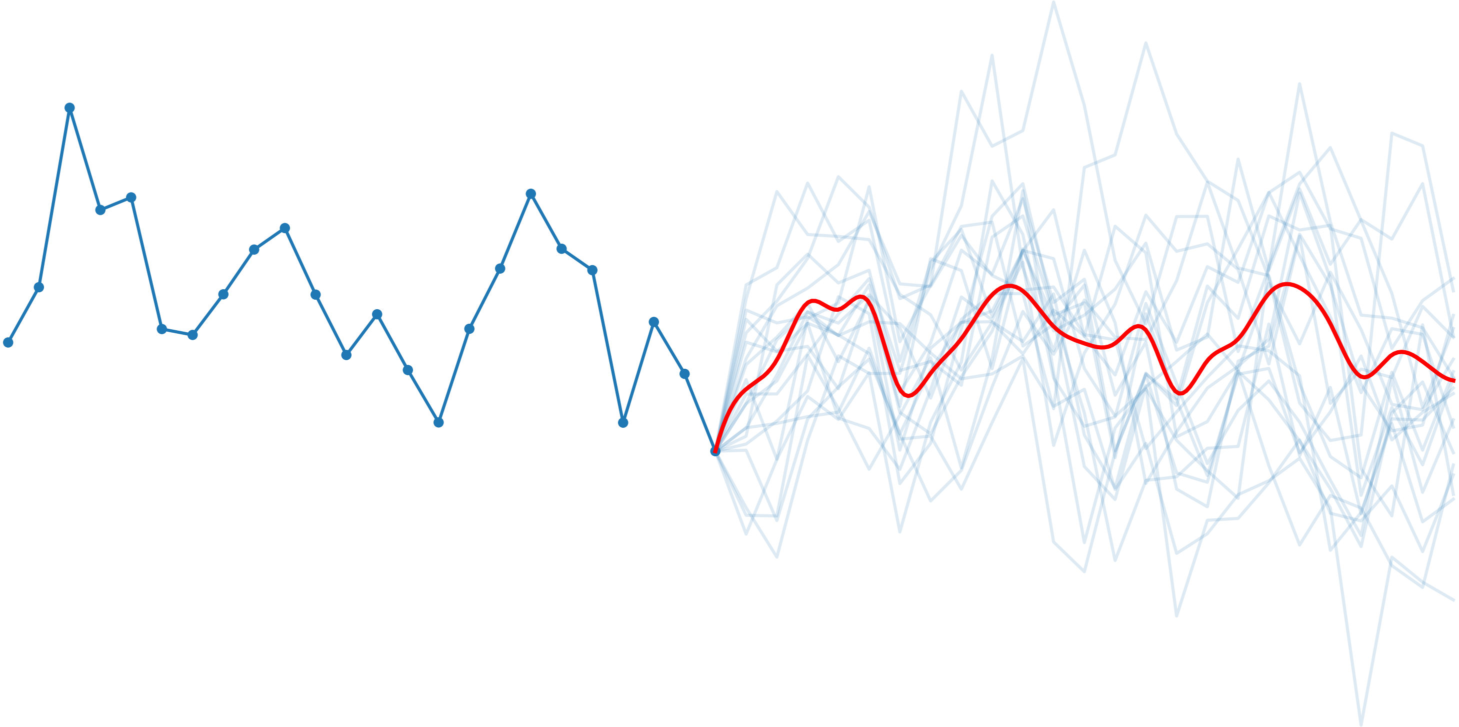


Predictive Modelling of Electrical Loads for Grid Infrastructure Planning

Segmentation and Stochastic Simulation of Residential and Commercial Energy Demand with Machine Learning

Master's Thesis in Complex Adaptive Systems

OSCAR LITORELL
EMRIK ÖSTLING

DEPARTMENT OF ELECTRICAL ENGINEERING
Chalmers University of Technology
Gothenburg, Sweden 2025
www.chalmers.se

MASTER'S THESIS 2025

Predictive Modelling of Electrical Loads for Grid Infrastructure Planning

Segmentation and Stochastic Simulation of Residential and Commercial Energy Demand with Machine Learning

OSCAR LITORELL
EMRIK ÖSTLING

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025

Predictive Modelling of Electrical Loads for Grid Infrastructure Planning
Segmentation and Stochastic Simulation of Residential and Commercial Energy Demand with Machine Learning
OSCAR LITORELL
EMRIK ÖSTLING

Supervisor: Johan Lamm, Sebastian Löf, Endre Technologies AB
Examiner: Anh Tuan Le, Department of Electrical Engineering

Master's Thesis 2025
Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone +46 31-772 10 00

Cover: Stochastic residual simulation (see Figure 4.14).

Typeset in Typst
Gothenburg, Sweden 2025

Predictive Modelling of Electrical Loads for Grid Infrastructure Planning
Segmentation and Stochastic Simulation of Residential and Commercial Energy Demand with Machine Learning
OSCAR LITORELL
EMRIK ÖSTLING
Department of Electrical Engineering
Chalmers University of Technology

Abstract

The transition towards a more sustainable energy system has increased the need for more precise and interpretable models for predicting future electricity demand. This thesis proposes a machine-learning based framework for predictive modelling of electrical loads, with the goal of enhancing grid infrastructure planning, particularly at the local level. The framework encompasses three components: (1) a load decomposition model that separates historical load into base load, temperature-dependent load and a solar generation component using only basic information such as temperature and solar radiation, (2) a building-informed segmentation model that estimates the coefficients for these load categories based on aggregated local building data, and (3) a residual process model that is used to model the stochastic variations that are not accounted for in the previous models. The models implement statistical methods, including machine learning with neural networks, modelling of stochastic processes and Monte Carlo simulations.

The proposed framework can be used to provide several insights, such as peak demand, load duration curves and how these are affected by changing climate patterns or construction of new buildings. In addition to estimating extreme values, the framework can be used to model daily demand patterns, valuable when evaluating adoption of sustainable energy sources which are often less controllable. By explicitly linking electricity usage to observable variables such as temperature and solar radiation, the models allow energy and utility companies to make data-driven decisions for grid planning without sacrificing interpretability or operational transparency.

Keywords: Machine Learning, Electric Load Segmentation, Grid Infrastructure Planning, Neural Networks, Solar PV Generation, Stochastic Modelling, Monte Carlo Simulation

Acknowledgements

We would like to express our sincere gratitude to several individuals and organizations who have supported us throughout our master's thesis work. First and foremost, we thank our supervisors at Endre Technologies AB, Johan Lamm and Sebastian Löf, whose support, guidance and valuable feedback have been instrumental to this project.

We are also deeply grateful to David Steen and Elias Hartvigsson, co-founders of Endre, who went far beyond what was expected in their roles.

Lastly, we would like to thank our partners, families and friends for their unwavering support and encouragement.

Oscar Litorell and Emrik Östling, Gothenburg, 2025-08-27

Contents

List of Figures

List of Abbreviations

AR(1)	autoregressive model of order one	55, 56
ARMA	autoregressive moving-average	30, 56
EPBD	energy performance of buildings directive	9
GNLL	gaussian negative log likelihood	3, 13
MA	moving average	55
MAE	mean absolute error	12
MAPE	mean absolute percentage error	42
MSE	mean square error	12, 13, 25, 33, 34
PV	photovoltaic	5, 24, 37, 38, 42, 65, 66
ReLU	rectified linear unit	10, 23
RMS	root mean square	26
SGD	stochastic gradient descent	11

1

Introduction

1.1 Background

Modern society is increasingly dependent on electricity, not only for essential services like heating, lighting, and communication, but also for the transition to a more sustainable future through electrification of transport, industry and heating. This growing demand, alongside the decentralization of production (e.g. solar panels and wind power), places new pressures on the electricity grid. European climate policy goals require a reduction of at least 80 % of all CO_2 emissions by 2050 [1].

To accommodate these changes, the electrical grid must be continuously expanded and reinforced. An important aspect of this planning involves determining the future electricity demand in specific areas. Estimating this demand accurately is necessary to avoid both under-dimensioning (leading to overloads or outages) and over-dimensioning (leading to unnecessary costs and unused infrastructure).

One of the key challenges is estimating the maximum power needed in a local area, which directly affects the sizing of components like substation transformers. Traditional estimation methods often rely on historical averages or general assumptions, which may not reflect the true variation and growth in usage patterns.

This thesis explores modern methods for estimating local electricity demand using neural networks. The work was carried out in collaboration with Endre Technologies AB, a company developing software tools that leverage societal data to improve grid planning decisions. The focus of the study has mainly been on electrical grids located in Sweden.

1.2 Aim

The aim of this study is to develop and evaluate a machine learning-based framework for predictive modelling of electrical load profiles to assist grid infrastructure planning. The framework seeks to decompose load time series into interpretable components, estimate these segmentation parameters from building data, and simulate stochastic demand variations, thereby enhancing the accuracy and interpretability of local power demand modelling.

1.3 Goals

The overarching goal of this thesis is to contribute to the field of electrical grid planning by developing and evaluating methods for predicting power usage characteristics. This is motivated by the increasing need for accurate, data-driven dimensioning of grid infrastructure in response to electrification and distributed energy generation.

To this end, we formulate two central research questions:

1. For a given period, such as the expected lifetime of the transformer in the substation, what is the expected maximum power it will have to deliver?
2. How does the distribution of electricity demand depend on external variables such as temperature, solar radiation, time of day, and building characteristics?

Question 1 is of main interest to electric grid companies, while answering question 2 hopes to provide a more specific and general description of the usage patterns that can be used to answer question 1.

1.4 Limitations and Demarcations

Power usage is heavily linked to temperature in Sweden [2]. Therefore, to properly model the future power usage there is a need to also simulate the temperature process. This is something this thesis does not cover and would be an opportunity for further studies.

The model will only be evaluated on Sweden and is focused on Sweden, challenges with other geographies will only be discussed briefly.

2

Theory

This chapter presents the theoretical foundations upon which the project is built. It begins with an overview of traditional methods for grid dimensioning, such as the Velander model and typical load profiles, highlighting their limitations in modern, dynamic demand environments. This is followed with a discussion of different types of electrical load and how they impact local electricity usage.

The second part of the chapter introduces artificial neural networks and other machine learning concepts relevant to this work, including autoencoders, loss functions, and uncertainty estimation via gaussian negative log likelihood (GNLL). These techniques form the basis for the predictive models developed later in the thesis.

2.1 Velander Model

The de facto standard for estimating the power demand is the Velander model [3] which describes how the maximum power usage scales with the number of buildings. It is a relatively simple model that assumes the usage of different buildings follows a normal distribution and that these usages are independent of each other, which is not necessarily the case. Furthermore, it also does not describe the effects of other parameters on the usage, such as temperature and time of day, or when the peak power occurs. These electricity consumers may all align their usage, (and in that case it underestimates the power usage) or they may often not align their usage, and the energy company will have built an unnecessarily powerful transformer.

For peak load of the Xth percentile Velander model is in practice given as

$$P_i^{(X)} = W_i + \beta\sqrt{W_i}. \tag{2.1}$$

Where $\beta = K^{(X)}\theta$. θ is variance-to-mean ratio, W_i is the estimated annual average power and $K^{(X)}$ is the probability density of peak power $P^{(X)}$ [4], [5].

2.2 Typical Load Profile

Another common method for peak power calculation is the typical load profile method [6]. This is a way to address the limitations of the Velander method, but it also introduces some limitations of its own. See the visual comparison in Figure 2.1 and Figure 2.2.

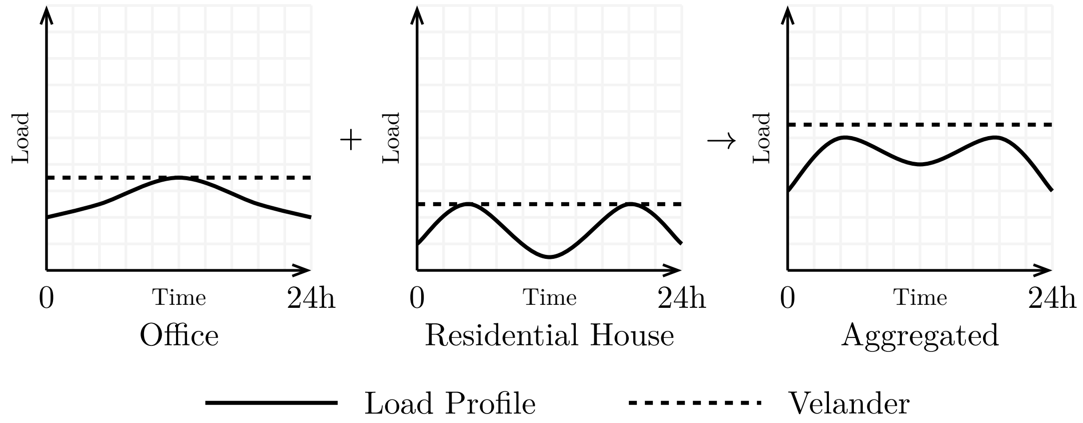


Figure 2.1: Comparison between the Velander model and typical load profile method. Here, the Velander model overestimates the actual load since peak load occurs at different points for the different buildings.

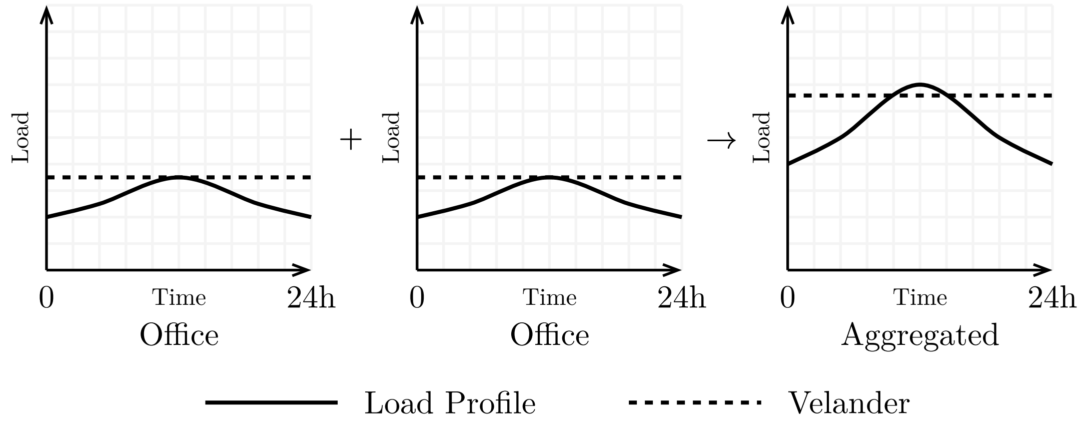


Figure 2.2: Comparison between the Velander model and typical load profile method. Here, the Velander model underestimates the actual load since the peak load of the buildings align.

The problem with the typical load profile method is mainly the lack of flexibility; it relies on having a typical load curve for every distinct building type and every distinct geographical location. These curves also need to be updated with changed power consumption patterns [2]. A benefit of this model is that it is possible to create typical load profiles for anything that has a regular power usage. This can, for example, be used to combine 100 houses with 20 curves for charging of electrical vehicles if the prediction is that 20% of houses will have electrical vehicles [7].

The optimal model would have the modularity and consistency of typical load profiles and the stochastic modelling and flexibility of the Velander model.

2.3 Types of Loads

This project focused on three different types of loads and power generation:

1. Solar generation.

2. Temperature dependent load. This includes everything used to either heat buildings during winter or to cool them during the summer.
3. Base load, for example usage of refrigerators, stoves, computers and indoor lighting.

2.3.1 Solar PV Generation

Solar photovoltaic (PV) panels are used to convert the solar radiation into electricity through the photovoltaic effect. As the use of solar panels continues to rise in Sweden, their contribution to the electric power balance must be considered when modelling electricity demand. Instead of being a traditional load, solar PV generation offsets usage, therefore effectively reducing the net load seen by the grid during periods of sunlight (if the load can be consumed by the producer).

The power output of solar panels depends on several factors, including:
- Solar irradiance
- Panel orientation and tilt
- Ambient temperature
- Panel technology and material

Figures 2.3 to 2.6 illustrate the theoretical and simulated relationships between solar production and environmental conditions.

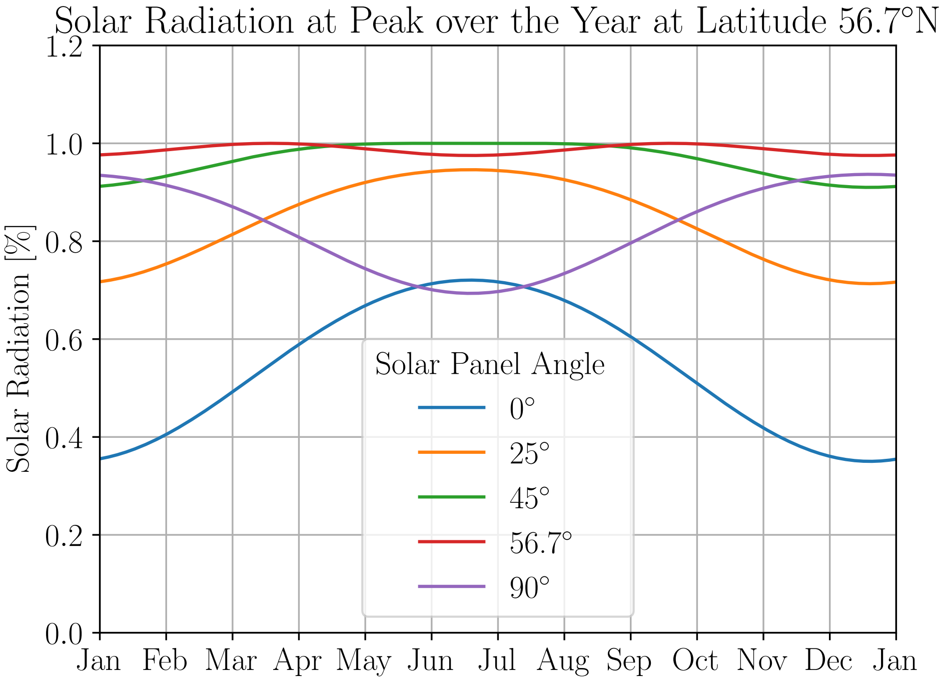


Figure 2.3: Captured solar radiation as a function of the angle of the solar panel, during midday. In this figure, the captured radiation is calculated at 56.7°N and is purely geometric without any atmospheric effects. A solar panel angle of 0° means flat on the ground. The y-axis expresses the captured solar radiation as a percentage to what it would be if the solar panel was perpendicular to the solar radiation.

The effect demonstrated in Figure 2.3 is purely geometrical. In reality, it would also be impacted by atmospheric factors since more solar radiation is scattered at shallow solar angles. The solar radiation data that is provided for this project is measured as global radiation, meaning that it is the radiation power experienced by a flat horizontal disc. Figure 2.3.

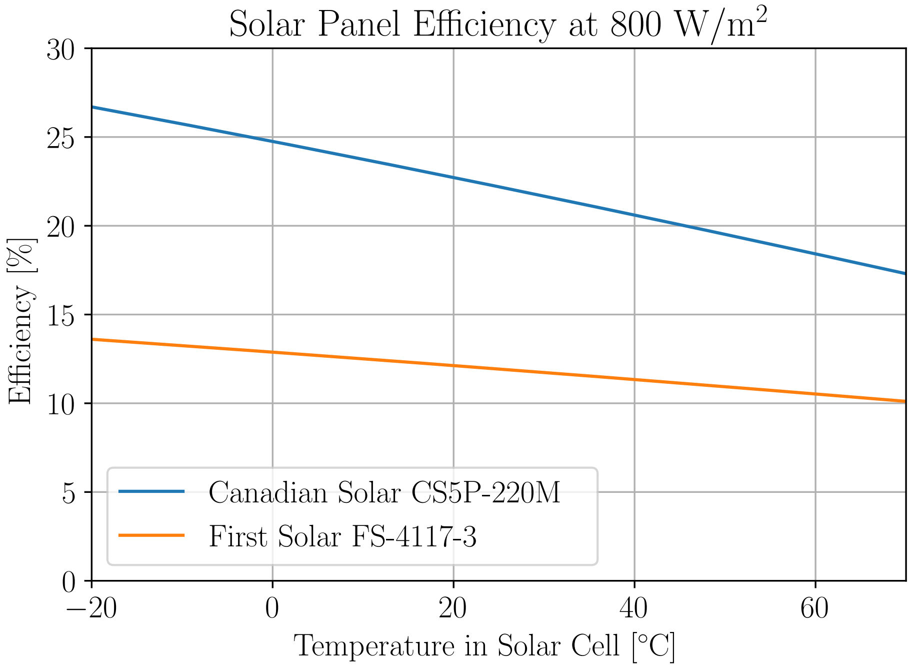


Figure 2.4: The dependence of temperature on solar panel efficiency. The power output is dependent on what solar cell is installed and its material, this figure shows a Canadian Solar CS5P-220M made with monocrystalline silicon and a thin film First Solar FS-4117-3. The rest of this chapter will only focus on monocrystalline silicon solar panels.

Figure 2.4 shows that for typical temperatures (−20 °C to 70 °C), the dependence on temperature is almost linear. Since the sun also heats up the panels, the solar panels often have lower efficiency at times of high power output. During the day, solar cells can reach a peak surface temperature 35 °C above the ambient temperature according to M. Koehl, M. Heck, S. Wiesmeier, and J. Wirth [8]. For air temperatures of 30 °C, this would translate to peak cell temperatures of up to 65 °C. Theoretical calculations for 30 °C air temperatures in Sweden give solar cell temperatures of 58 °C [9] which align with M. Koehl, M. Heck, S. Wiesmeier, and J. Wirth [8].

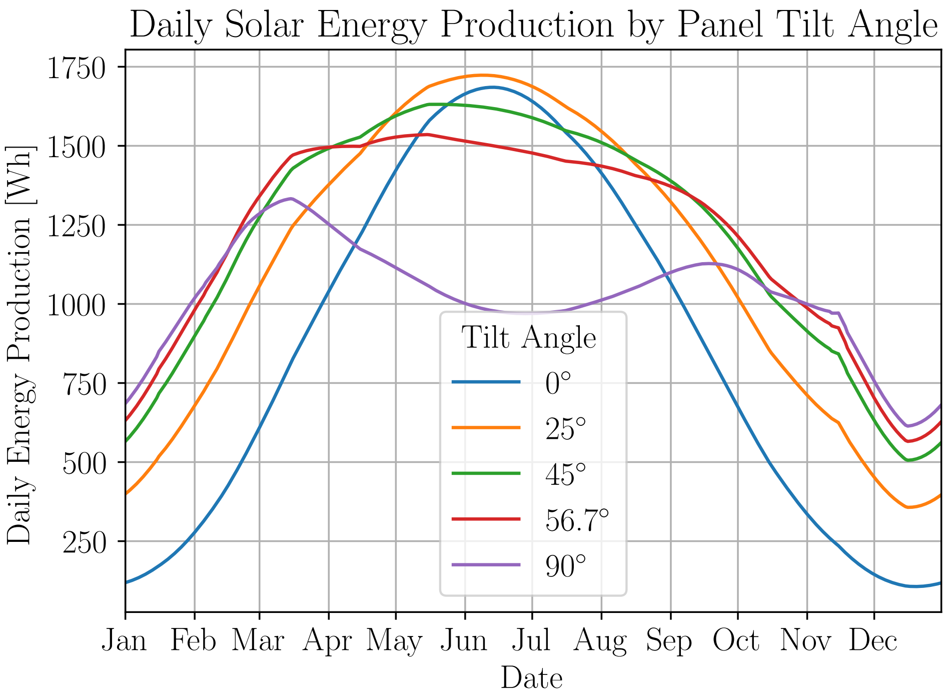


Figure 2.5: Figure 2.3 with simulated atmospheric effects for 56.7°N 12.8°W on a Canadian Solar CS5P-220M with a nominal power of 220W [10].

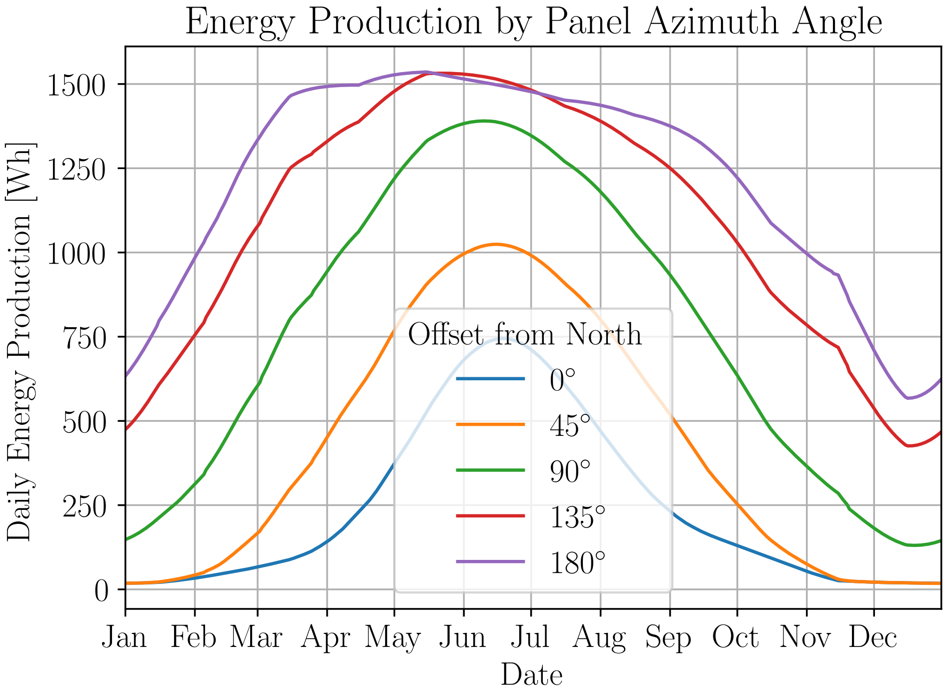


Figure 2.6: Daily solar-cell production for a Canadian Solar CS5P-220M on 56.7°N 12.8°W depending on its azimuth where 180° is straight south. The tilt is 56.7°.

The effect of solar panel azimuth is symmetrical if the temperature differences are ignored.

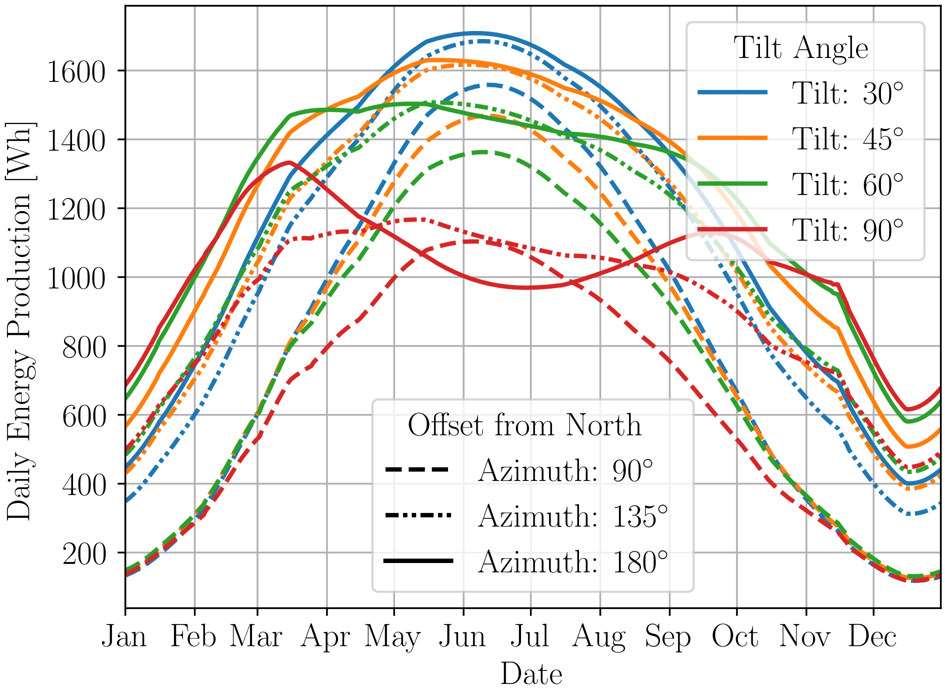


Figure 2.7: Daily solar-cell production for a Canadian Solar CS5P-220M on 56.7°N 12.8°W depending on its azimuth where 180° is straight south and the panel tilt where 0° is flat on the ground. Here the takeaway is that solar cells facing west or east will be more likely to have a lower tilt, since it is more effective.

These simulations were done using pvlib [11] without any temperature variations except for Figure 2.4.

2.3.2 Temperature Load

Temperature load refers to the power used for heating or cooling a building. In Sweden, district heating is the most common type with approximately 50% of the total heating measured in output power (2015) [12]. Around 15% is resistive heating, 20% is heat pumps, and 15% is burning of oil and biofuel. In this project, only the sources using electricity are of interest.

Residential air conditioning is relatively uncommon in Sweden. In residential buildings, 40% of all heat pumps are air to air heat-pumps, and can therefore be used to cool the air [13]. In offices and industrial buildings, there is a high adaption of district cooling [14].

Thermal transmittance is often expressed as a U-value. U-values are measured in $\mathrm{W\,m^{-2}\,K^{-1}}$, and represent how much thermal energy escapes from buildings through surfaces exposed to the outside air per unit of area and time, for a given temperature difference. [15].

The buildings that exist today have been built over the course of hundreds of years and are therefore built in different ways. For an outside temperature of $-20\,°\mathrm{C}$, it takes less than a day for an old Swedish house built around 1880 to cool down from room temperature to $5\,°\mathrm{C}$. [16] Meanwhile, a larger low-power villa built in 2007 takes

4 days in the same conditions [16]. One major factor is that older buildings have one or two pane windows with U-values of 2.8, while modern buildings have three panes with U-values of 0.84, which means they lose a third of the power per m^2. Walls of modern houses have U-values of 0.09, their ceilings 0.08, their floors 0.09 and doors have U-values of 1.4. This means that modern houses with many windows are relatively bad at containing thermal energy. In older houses, windows can be replaced and insulation can be added which improves their energy efficiency. Ventilation systems are another factor for heat loss. In modern houses, ventilation systems that use heat recovery are common, which can recycle 70 % of the heat energy in the air leaving the building [17]. This can also be installed in older buildings. With the increased power usage of older houses, it can even be profitable to renovate [18], but it requires an upfront investment. The number of different variables related to heating introduces an uncertainty when trying to predict the actual power usage for buildings.

Since Sweden is a part of EU, the energy performance of buildings directive (EPBD) applies. EPBD is a building energy classification system that is mandatory for all buildings built since 2006. For small residential houses built before 2006, classification must also be done when sold. The goal of this is to promote renovation and make buildings more efficient for the 2050 goal of a fully decarbonized building stock. The energy classification is an estimate of a building's power usage compared to newly constructed buildings. Inclusion of this data could be useful for creating a better temperature load estimation. However, this thesis did not use any building energy classification data.

2.3.3 Base Load

The base load should represent the load that is not directly related to heating or energy production. This includes activities such as boiling water, charging of electric vehicles, or use of other household appliances. In Velander's model, it is calculated using the mean of the load and adding a variance to it depending on the chosen β. In the typical load profile model, it is instead handled by having a curve for the base load and adjusting it depending on the season.

For the seasonality effect, it has been shown that the average household in the UK uses 40% more electricity during the winter for cooking, 200% more for lighting, and 70% more for washing or drying [19]. The average household also uses 35% less energy during the winter for cold appliances such as refrigerators [19]. Lighting is the clearest example that it is not directly linked to temperature and more to behaviour due to the time of the year. This load should be included in the base load and not in the temperature since it is not temperature dependent.

2.4 Artificial Neural Networks (ANN)

The term "machine learning" includes several different methods, and one of the most common is using artificial neural networks. As the name suggests, the method is inspired by the neural networks that are found in biology. They can be used to solve several different categories of problems, including classification and regression, with the latter being of most importance in this project.

In contrast to the biological counterpart, artificial neural networks often consist of several sequential layers, with each layer consisting of several artificial neurons and the outputs of each layer feeding into the inputs of the next. To calculate the output, or *activation*, of a neuron, a weighted sum of the inputs is first taken, in addition to adding a bias (offset). This result is then fed into a non-linear function, often called an *activation function*, to get the final neuron output. The reason for using non-linear activation functions is that if only linear or affine activation functions were used, the resulting composition of these could only ever be linear or affine. A few commonly used functions include rectified linear unit (ReLU) [20] and σ [21]:

$$\mathrm{ReLU}(x) = \max(x, 0), \tag{2.2}$$

$$\text{Leaky ReLU}(x) = \begin{cases} x \text{ if } x \geq 0 \\ \alpha x \text{ if } x < 0 \end{cases} \tag{2.3}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{2.4}$$

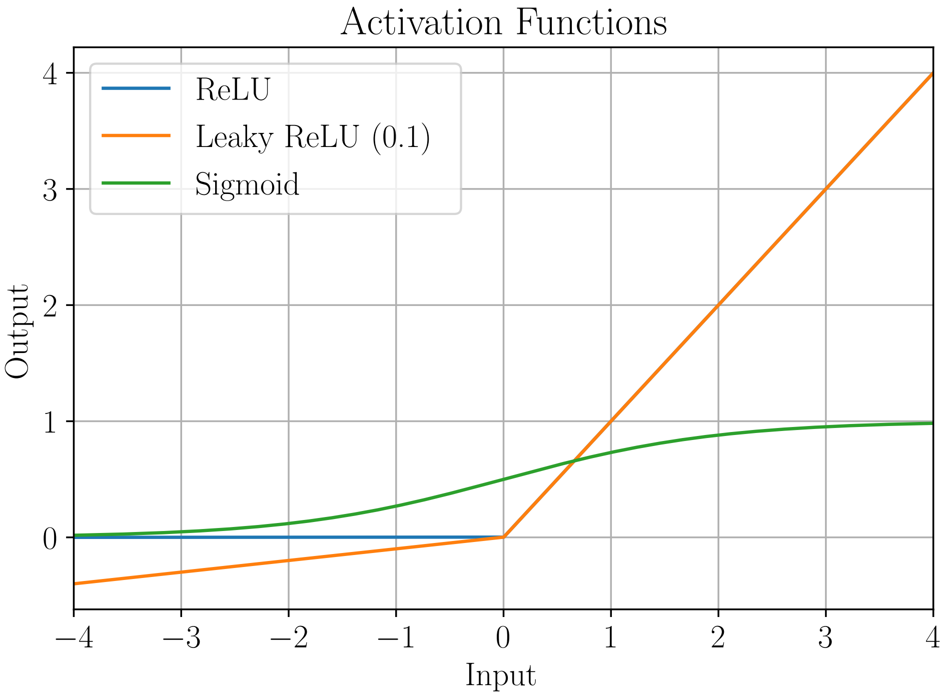


Figure 2.8: ReLU, leaky ReLU (with negative slope 0.1) and sigmoid activation functions. ReLU and leaky ReLU is the same when input > 0.

In a fully connected (sometimes referred to as "dense") network, the outputs of all neurons in one layer are fed into the inputs of all neurons in the next layer. The layers between the input and output layers are called the hidden layers. If the inputs to the neurons are expressed as a vector, then all weights for a layer can be combined into a weight matrix, and the bias into a bias matrix, which allows for efficient computations.

The method used to fit the weights and biases during training is usually based on gradient descent, which involves finding how the prediction error depends on the different weights and biases, and then changing them slightly. The method used to calculate gradients is known as back propagation and is essentially another name for the chain rule for matrices. This is commonly handled by machine learning libraries such as PyTorch [22] or TensorFlow [23].

To avoid moving into local minima, a modified version of gradient descent called stochastic gradient descent (SGD) can be used. This method involves performing gradient descent on random subsets (batches) of the dataset.

In gradient descent, the step size is dependent on the learning rate η. There are so called *optimizers* that change the step size dynamically to speed up training. Examples include Adam [24], AdamW [25], AdaDelta [26], RMSProp [27] and AdaGrad [28].

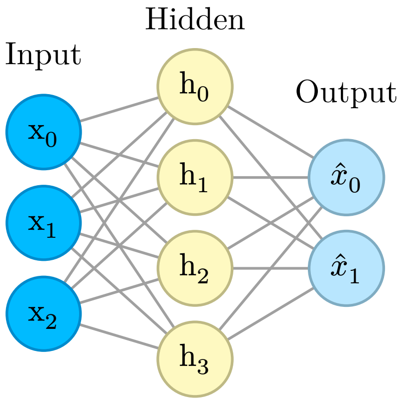


Figure 2.9: Example of a classic neural network. All the networks in this project used this layout but with different sizes and number of hidden layers.

2.4.1 Autoencoders

In some cases, it is useful to reduce the dimensionality of data to decrease the number of computations or storage needed. One way to achieve this is with the use of autoencoders [29]. The autoencoder architecture is relatively simple, and it works by having a few sequential neural layers that reduce the dimensionality, and then a few more that increase the dimensionality to the original size again [29].

In the special case of linear/affine autoencoders, only one layer is needed for the encoder and decoder respectively, which can be represented by a single matrix multiplication.

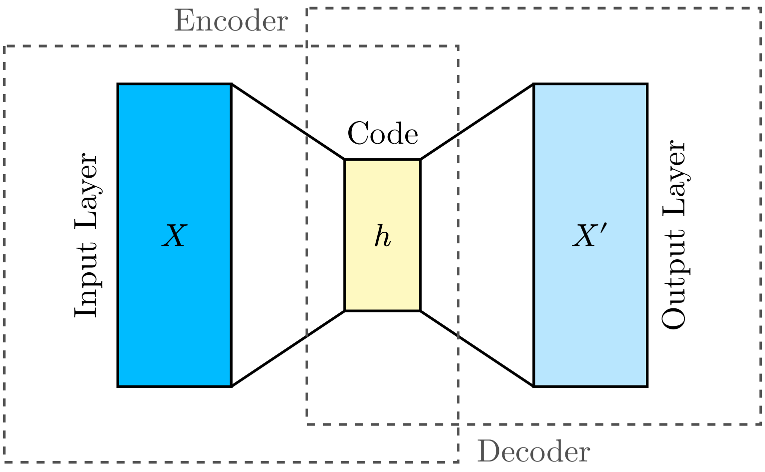


Figure 2.10: Overview of the autoencoder. It consists of two main parts, an encoder that maps the input to a code and a decoder that maps the code to the output. Both the encoder and decoder are then trained simultaneously.

2.5 Loss Functions

When training artificial neural networks, there needs to be a score to optimize for, and how that score is calculated is called the loss function. This score is often dependent on the output of the neural network, and the parameters of the network are changed during training to minimize this score.

Common loss functions for regression include mean square error (MSE) and mean absolute error (MAE), defined as follows:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2, \tag{2.5}$$

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \hat{Y}_i\right|, \tag{2.6}$$

where Y_i is the true value and $\hat{Y}_i$ is the predicted value.

Mean Absolute Percentage Error (MAPE) is a metric that is related to loss functions, but which is not commonly used as a loss function itself. It is rather used as a measurement for evaluating the prediction performance of a model:

$$\mathrm{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right|. \tag{2.7}$$

As the name suggests, it is defined as a mean of all absolute percentage errors.

2.6 Gaussian Negative Log Likelihood

In cases where the data is heteroskedastic, meaning that the variance is dependent on the input parameters, it is often desirable to know how it depends on the input parameters. Gaussian negative log likelihood (GNLL) is a method that can be used in combination with neural networks to solve this. More specifically, it estimates the parameters of the distribution of the output for a given set of inputs, which is assumed to follow a normal distribution.

The method works by expanding the output from the network from one to two outputs, representing both the estimated mean as well as variance, as opposed to only the mean, which is normally the case. In the prior case with only the mean as the output, the loss function is often relatively straight-forward, and often simply consists of a function of the difference between the prediction output and the target output, e.g. MSE. In the case of 2 outputs representing the mean and variance, there is initially not an obvious loss function to use since there is no “target variance”. It is, however, possible to construct a loss function that solves this, by trying to maximize the likelihood. The likelihood $\mathcal{L}$ is defined as the probability (or probability density) of getting the observed data x, given some distribution parameters θ:

$$\mathcal{L}(\theta|x) = p(x|\theta). \tag{2.8}$$

In this case, the notation $\mathcal{L}(\theta|x)$ should be interpreted as the likelihood that the true distribution has parameters θ, given observed data x. In the case of multiple data x_i which are assumed to be independent, we can write it in the following way:

$$\mathcal{L}(\theta|x_1, x_2, ..., x_n) = p(x_1, x_2, ..., x_n|\theta) = p(x_1|\theta) \cdot p(x_2|\theta) \cdot ... \cdot p(x_n|\theta). \tag{2.9}$$

By taking the logarithm of this, we can break up the long multiplication into a sum instead. Since the logarithm function is monotonic, maximizing $\log(\mathcal{L})$ is equivalent to maximizing $\mathcal{L}$. In practice, the negative log likelihood is often used to get a minimization problem instead of a maximization problem. This gives us:

$$\begin{aligned} -\log \mathcal{L}(\theta|x_1, x_2, ..., x_n) &= -\log(p(x_1|\theta)) - \log(p(x_2|\theta)) - ... - \log(p(x_n|\theta)) \\ &= \sum_{i=1}^{n} -\log p(x_i|\theta). \end{aligned} \tag{2.10}$$

In this case, the likelihood is defined for a given value of n and is not comparable with likelihoods for other sample sizes. This can easily be solved by taking the mean instead of the sum, which practically means multiplying with a factor of $\frac{1}{n}$. This is equivalent to taking a geometric mean of the factors in the non-logarithm likelihood.

$$-\frac{1}{n} \log \mathcal{L}(\theta|x_1, x_2, ..., x_n) = \frac{1}{n} \sum_{i=1}^{n} -\log p(x_i|\theta). \tag{2.11}$$

In other words, this means taking the mean of $-\log p(x)$ for all x in a dataset.

For a normal distribution, our θ represents our mean μ and variance σ^2. We can write the probability density function as follows:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right). \tag{2.12}$$

Taking the negative logarithm gives us:

$$-\log p(x) = \frac{1}{2}\log(2\pi) + \frac{1}{2}\log(\sigma^2) + \frac{(x-\mu)^2}{2\sigma^2}. \tag{2.13}$$

In practice, the first constant term is often omitted. This leaves us with the following loss function:

$$\text{GNLL Loss} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{2}\log(\sigma^2) + \frac{(x-\mu)^2}{2\sigma^2}\right). \tag{2.14}$$

2.6.1 Generalized Gaussian Negative Log Likelihood

The method above can be generalized for multivariate normal distributions. In this case, the scalar x is substituted with the vector $\boldsymbol{x}$, and the variance σ^2 is substituted with the covariance matrix $\boldsymbol{\Sigma}$. The covariance matrix is defined as

$$(\boldsymbol{\Sigma})_{ij} = \mathbb{E}[X_i - \mu_i] \cdot \mathbb{E}\big[X_j - \mu_j\big] = \begin{cases} \mathrm{Var}(X_i) \text{ if } i = j \\ \mathrm{Cov}\big(X_i, X_j\big) \text{ if } i \neq j. \end{cases} \tag{2.15}$$

This allows us to generalize the probability density function as follows: [30]

$$p(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^D\ |\boldsymbol{\Sigma}|}}\exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right), \tag{2.16}$$

where D is the number of dimensions and $|\boldsymbol{\Sigma}|$ is the determinant of $\boldsymbol{\Sigma}$. Taking the negative logarithm gives us:

$$-\log p(\boldsymbol{x}) = \frac{D}{2}\log(2\pi) + \frac{1}{2}\log(|\boldsymbol{\Sigma}|) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) \tag{2.17}$$

2.6.2 Cholesky Decomposition

For $\boldsymbol{\Sigma}$ to be a valid covariance matrix, it needs to be positive definite. A symmetric real matrix M is defined as positive definite when [31]:

$$\boldsymbol{x}^T M \boldsymbol{x} > 0 \text{ for all } \boldsymbol{x} \in \mathbb{R}^n \setminus \{\boldsymbol{0}\}. \tag{2.18}$$

This causes a problem when assigning the outputs of, for instance, a neural network to the elements of the matrix, as there is nothing that enforces this condition. Fortunately, this can be solved by letting the neural network output the elements to a matrix that factorizes $\boldsymbol{\Sigma}$ in a specific way:

$$\boldsymbol{\Sigma} = \boldsymbol{B}\boldsymbol{B}^T, \tag{2.19}$$

where $\boldsymbol{B}$ is any real matrix. Inserting this into Equation (2.18) shows that this factorization will satisfy the positive definiteness condition:

$$\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{B}^T\boldsymbol{x} = \left(\boldsymbol{B}^T\boldsymbol{x}\right)^T\left(\boldsymbol{B}^T\boldsymbol{x}\right) = \|\boldsymbol{B}^T\boldsymbol{x}\|^2 > 0 \text{ for all } \boldsymbol{x} \in \mathbb{R}^n \setminus \{\boldsymbol{0}\}. \tag{2.20}$$

We can impose an additional condition on $\boldsymbol{B}$, which is that it should be lower triangular. This comes with a few advantages that will be shown later. We call this matrix $\boldsymbol{L}$. In this case, we have:

$$\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^T, \tag{2.21}$$

which is called the *Cholesky decomposition*. We have not yet proved that any positive definite matrix can be factorized this way. This turns out to be true and is proven by T. A. Manteuffel [32]. In other words, every covariance matrix $\boldsymbol{\Sigma}$ can be factored in this way, and every lower triangular matrix $\boldsymbol{L}$ gives a valid covariance matrix. This allows us to parameterize $\boldsymbol{\Sigma}$ in a way where we can continuously move in the space of all covariance matrices, without concern for leaving that space.

Substituting this Cholesky decomposition into Equation (2.17) gives us

$$-\log p(x) = \frac{D}{2}\log(2\pi) + \frac{1}{2}\log(|\boldsymbol{L}\boldsymbol{L}^T|) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\left(\boldsymbol{L}\boldsymbol{L}^T\right)^{-1}(\boldsymbol{x}-\boldsymbol{\mu}). \tag{2.22}$$

Using that $|\boldsymbol{L}\boldsymbol{L}^T| = |\boldsymbol{L}| \cdot |\boldsymbol{L}^T| = |\boldsymbol{L}|^2$ and that $\left(\boldsymbol{L}\boldsymbol{L}^T\right)^{-1} = \left(\boldsymbol{L}^{-1}\right)^T\left(\boldsymbol{L}^{-1}\right)$, we can simplify further:

$$\begin{aligned}
-\log p(x) &= \frac{D}{2}\log(2\pi) + \log(|\boldsymbol{L}|) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\left(\boldsymbol{L}^{-1}\right)^T\left(\boldsymbol{L}^{-1}\right)(\boldsymbol{x}-\boldsymbol{\mu}) \\
&= \frac{D}{2}\log(2\pi) + \log(|\boldsymbol{L}|) + \frac{1}{2}\left(\boldsymbol{L}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)^T\left(\boldsymbol{L}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right) \\
&= \frac{D}{2}\log(2\pi) + \log(|\boldsymbol{L}|) + \frac{1}{2}\|\boldsymbol{L}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\|^2.
\end{aligned} \tag{2.23}$$

To avoid calculating the inverse of $\boldsymbol{L}$, it is possible to instead let the neural network parameterize $\boldsymbol{L}^{-1}$ directly:

$$-\log p(x) = \frac{D}{2}\log(2\pi) - \log(|\boldsymbol{L}^{-1}|) + \frac{1}{2}\|\boldsymbol{L}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\|^2. \tag{2.24}$$

A result of using $\boldsymbol{L}$ or $\boldsymbol{L}^{-1}$ instead of $\boldsymbol{\Sigma}$ or any other factorization is that it simplifies calculating the determinants. Since both $\boldsymbol{L}$ and $\boldsymbol{L}^{-1}$ are lower triangular matrices, their determinant can be calculated by simply multiplying their diagonal elements. Furthermore, the logarithm of the determinants can be calculated by summing the logarithms of the diagonal elements.

3

Methods

This chapter aims to present the methods used in the project to achieve the stated goals. It includes sections on data preparation, as well as the different models used. Figure 3.1 describes an overview of the proposed framework.

The proposed framework consists of the three following models:

1. The **Load Decomposition Model** is used to segment the measured load for a substation service area into a few interpretable categories, including a temperature based component, a solar radiation based component and a few components based on time of day and time of year. In other words, it learns how each of these components depend on their respective input values, and expresses the total load profiles for the different substation service areas as weighted sums of these components, scaled by what we refer to as *load segmentation coefficients*.
2. The **Building-Informed Segmentation Model** is used to estimate load segmentation coefficients for substation service areas where historical load data is not available. These are estimated from aggregated building data in the substation service area.
3. The **Residual Process Model** is used to model the stochastic variations that the previous models do not account for. These stochastic variations have a large effect on the extreme values of the load profiles, and are thus of high importance for the main research questions of this thesis.

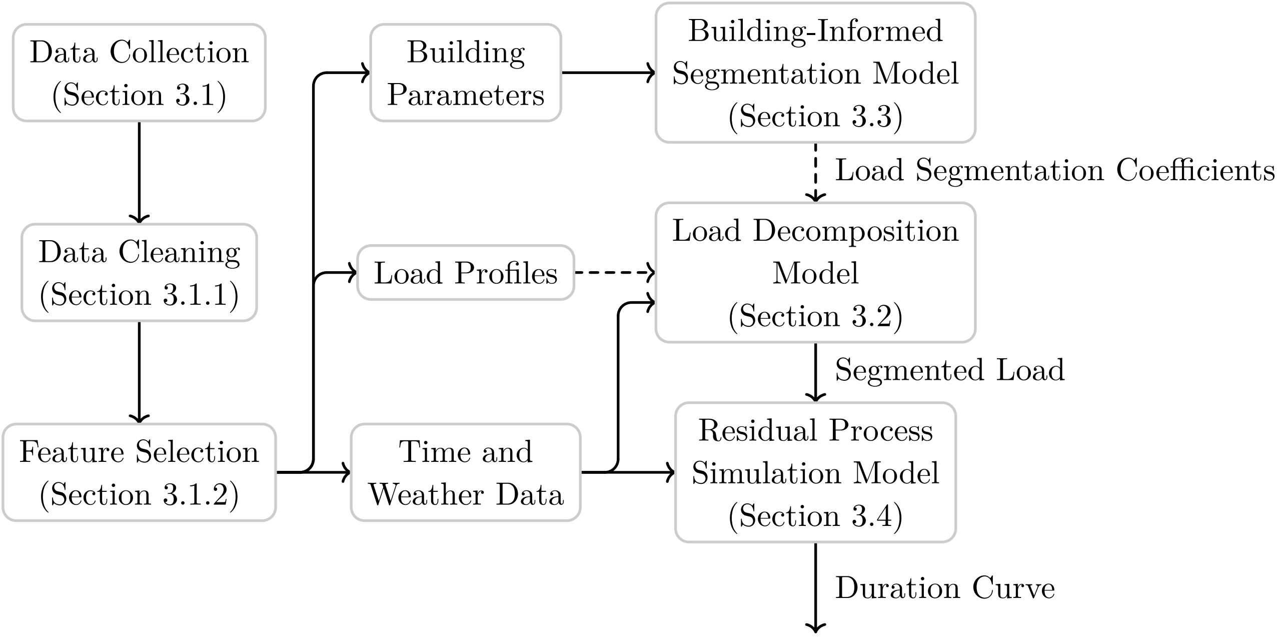

Data Collection (Section 3.1)
Data Cleaning (Section 3.1.1)
Feature Selection (Section 3.1.2)
Building Parameters
Load Profiles
Time and Weather Data
Building-Informed Segmentation Model (Section 3.3)
Load Segmentation Coefficients
Load Decomposition Model (Section 3.2)
Segmented Load
Residual Process Simulation Model (Section 3.4)
Duration Curve

Figure 3.1: Overview of how all parts of the model are connected. This represents the model in its inference step, the setup is different when training. The lines represent the data flow or dependency, meaning that, for instance, the data collection step has to be performed before the data cleaning step. The dashed arrows represent alternative data dependencies, in this case that when there is no historical load data available, the output from the building-informed segmentation model can be used instead.

3.1 Data Preparation

Three primary datasets were used in this study: load data, building data, and weather data. The building data included information on building outlines, heights, and classifications. The weather dataset comprised of hourly measurements of solar radiation and temperature.

To ensure the privacy of residents, the load data was aggregated over larger geographic areas, which will be called substation service areas for the rest of this report. These areas varied in size, including anywhere from a few buildings to several hundred. The load data was available at an hourly resolution but limited to a single year. Corresponding weather data was available for the same time duration as the load data.

The building data was augmented by including more specific categorical and building usage data from other sources.

The data processing was done using GeoPandas [33] which is an extension to the python data manipulation library Pandas [34] for geospatial data.

3.1.1 Data Cleanup

Initially, substations with a large amount of missing data for the load profiles were removed. For some substation load profiles, there were a few individual missing data points; these were substituted by neighbouring load data points. After this, outlier substations were identified by comparing the total substation energy usage and the total building volume for the respective service areas. These substations were often relatively small in scale, and were connected to electricity consuming buildings or infrastructure with abnormal or extreme usage patterns that were considered outside the scope of this study.

3.1.2 Feature Selection

The framework consisted of three models that used different features as inputs.

Two of the features for the load decomposition model (Section 3.2) are the time of day and time of year, both encoded as 2D vectors with the sine and cosine of the time such that the vectors rotate one revolution along the unit circle every day or year respectively. This is done to both make the end and start of a day or year continuous, while providing a unique value for every hour. The model also has binary holiday data [35] since holidays cause a change in behaviour and therefore power usage. All holidays and non-workdays, Saturday and Sunday, are considered equal. These values are used to create base load curves, which can be combined to create typical base load patterns.

The temperature dependent power usage is predicted using temperature averages for the last 1, 2, 4, 8, 16 and 32 hours. Additional historic values were tested without visible improvement.

The load decomposition model also included a solar dependent power prediction, which used time of year, encoded as a 2D-vector as before, and global solar radiation data as input features. The time of year was used as an input to allow the model to compensate for the difference between the solar panel output and the global radiation, caused by the effects detailed in Section 2.3.1.

For the building-informed segmentation model, (Section 3.3), 15 different building types were used, as described in Table 3.1.

Table 3.1: Comparison between the different building categories by count and volume.

Building Category	**Percent of Total Building Count**	**Percent of Total Building Volume**
Complement	52.94 %	9.59 %
Villas	26.78 %	20.50 %
Chain House	4.73 %	2.83 %
Row House	3.88 %	2.41 %
Apartments	3.78 %	24.17 %
Allotments	1.77 %	0.22 %
Commercial	1.31 %	13.81 %
Public	1.19 %	4.37 %
Multifamily Residential	1.15 %	2.08 %
Industry	0.84 %	13.09 %
Other	0.63 %	0.36 %
Farm Building	0.44 %	0.88 %
Education	0.26 %	2.71 %
Unspecified	0.17 %	0.08 %
Offices	0.11 %	2.90 %

This set of building types was chosen as it was deemed to include most types of typical usage patterns, without having too many individual types which would increase the risk of overfitting. The building volumes were calculated by multiplying the projected ground area with the height estimated.

For each substation, the service area consisted of a number of grid connection points. The dataset only contained the geographical coordinates of these points, but not the buildings they were connected to, which meant that buildings would have to be matched with their grid connection points. The buildings were assigned to their closest grid connection point if the point was within 20 m from the outline of the building. For some building types commonly found in sparsely populated areas, the limit was increased to 100 m. The reason for having a distance limit was that some buildings were not connected to any substation. Some buildings had several grid connection points in their interior. For these cases, the building was divided up amongst these points based on closest distance. The aggregated volumes for each building type and substation service area were then used as input features for the building-informed segmentation model.

The residual process simulation model (Section 3.4) used input features already defined for the load decomposition and building-informed segmentation models, meaning that no extra feature selection was necessary.

3.1.3 Data Normalization

When the data was used as inputs to neural networks, it was normalized so that all input values were mapped to between -1 and 1. This was done to allow for better convergence during training. The load dataset was normalized by dividing it with its mean.

3.2 Load Decomposition Model

The goal of the load decomposition model was to decompose the load profiles for the substation service areas into the following categories:

- Time-dependent (base) load
- Temperature-dependent load
- Solar radiation-dependent load or production.

This was done by letting the model simultaneously learn how each type of load is a function of its corresponding variable, e.g. how the temperature-dependent load varies with temperature, and the size of that load category in relation to the other load categories for a specific substation service area.

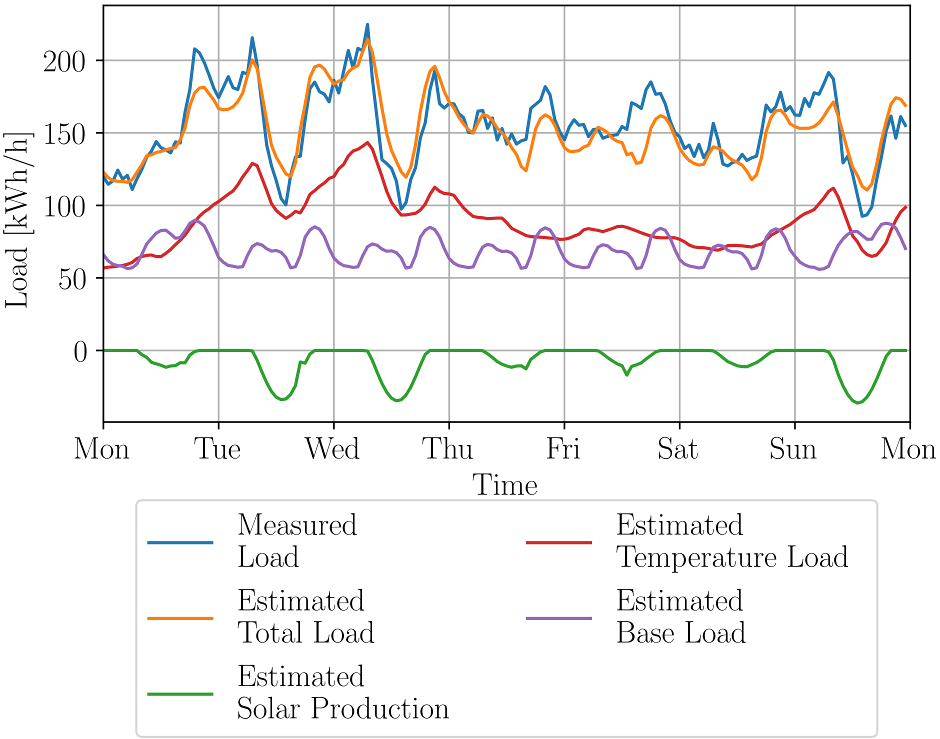


Figure 3.2: An example week of the different categorical load profiles in spring. “Estimated total load” refers to the sum of the estimated load profiles for the other categories.

Summing the load profiles for the different categories with their fitted weights for a given substation service area will result in an approximation of the true load profile for that substation service area. This allows the true load profile to be reduced to a low-dimensional vector consisting of the coefficients related to each categorical load profile. This approximation is considered to be deterministic, since it is only a function of other known variables. The difference between this approximation and the true load profile, the *residual*, will likely depend on other factors beyond the scope of the data at hand, and will thus be considered a stochastic process. A model of this stochastic process is presented in Section 3.4.

3.2.1 Architecture

This subsection describes the decomposition model architecture in more detail. Figure 3.3 illustrates a diagram of the model.

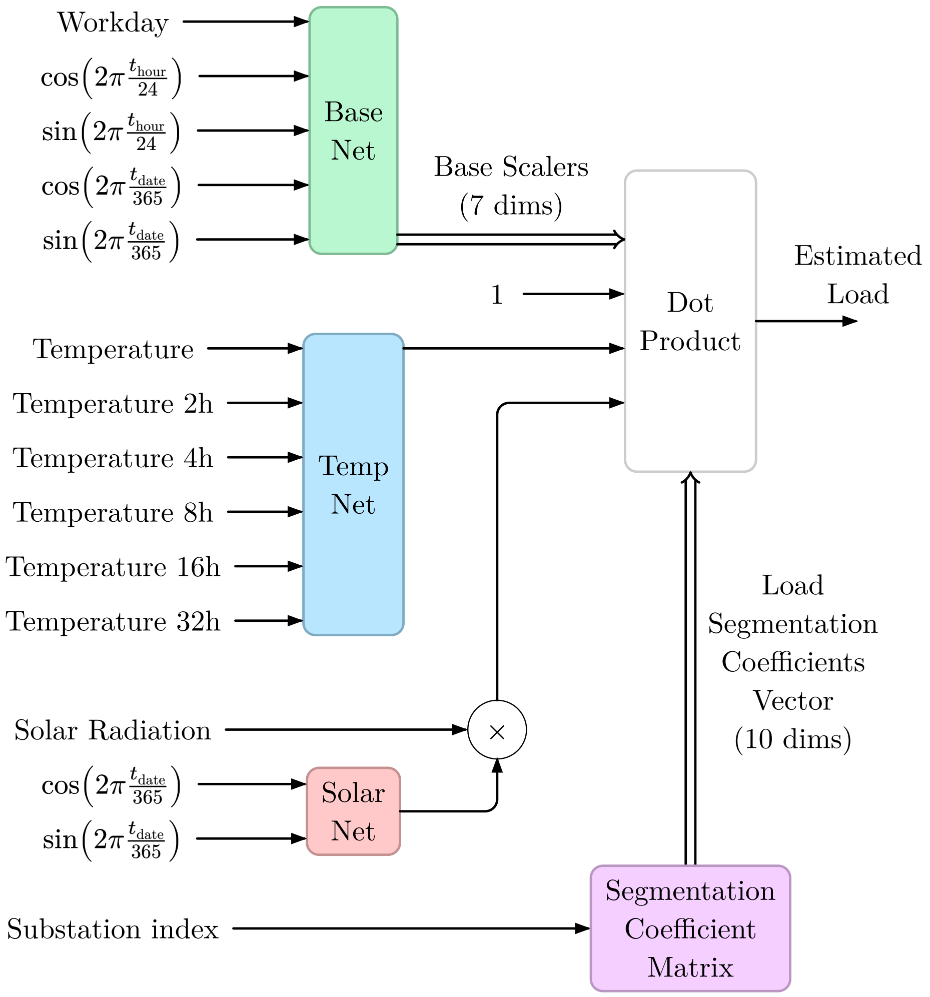

Workday
$\cos\left(2\pi\frac{t_{\text{hour}}}{24}\right)$
$\sin\left(2\pi\frac{t_{\text{hour}}}{24}\right)$
$\cos\left(2\pi\frac{t_{\text{date}}}{365}\right)$
$\sin\left(2\pi\frac{t_{\text{date}}}{365}\right)$
Base Net
Base Scalers (7 dims)
1
Dot Product
Estimated Load
Temperature
Temperature 2h
Temperature 4h
Temperature 8h
Temperature 16h
Temperature 32h
Temp Net
Load Segmentation Coefficients Vector (10 dims)
Solar Radiation
×
$\cos\left(2\pi\frac{t_{\text{date}}}{365}\right)$
$\sin\left(2\pi\frac{t_{\text{date}}}{365}\right)$
Solar Net
Substation index
Segmentation Coefficient Matrix

Figure 3.3: Overview of the model architecture used for load decomposition.

The model consists of the following three neural networks:

- Base Network (green)
- Temperature Network (blue)
- Solar Radiation Network (red).

The base network is a fully connected neural network that has an input vector consisting of 5 arguments: a binary parameter for whether it is a workday at the given point in time, along with the cosine and sine of the time of year along with time of day. The reason for encoding time as sine and cosine was to avoid discontinuities at midnight and at the beginning of the new year. In addition, the base network has two hidden layers with 16 neurons each, and seven output neurons. Between the layers there is a ReLU activation function (Section 2.4).

The aim is that the final output of the base network should represent the deviations from the mean of the base load. Thus, it is calculated by, for a given output, subtracting the outputs from a baseline input vector with zeros for the cosine and sine inputs, and one for the workday input as shown in Figure 3.4.

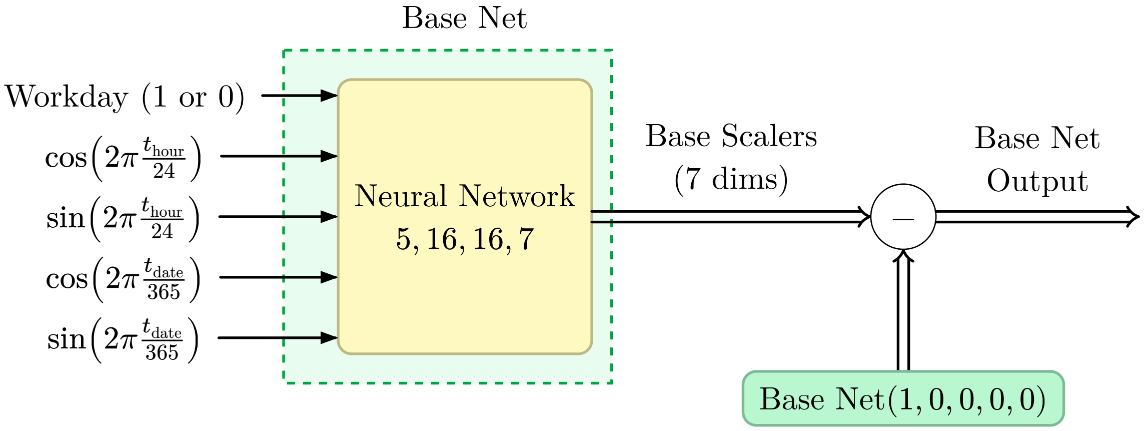

Base Net
Workday (1 or 0)
$\cos\left(2\pi\frac{t_{\text{hour}}}{24}\right)$
$\sin\left(2\pi\frac{t_{\text{hour}}}{24}\right)$
$\cos\left(2\pi\frac{t_{\text{date}}}{365}\right)$
$\sin\left(2\pi\frac{t_{\text{date}}}{365}\right)$
Neural Network
$5, 16, 16, 7$
Base Scalers
(7 dims)
$-$
Base Net
Output
Base Net$(1, 0, 0, 0, 0)$

Figure 3.4: Overview of the model architecture used for the Base Net as seen in green in Figure 3.3. $16, 16$ is the size of the hidden layers with the same setup as Figure 2.9. Five is the size of the input and seven is the size of the output.

The temperature network is a fully connected neural network that has an input vector consisting of 6 arguments. They are all temperature means but over different time frames. 20 °C was chosen as the reference point for the model since it is a common indoor temperature [36].

The aim is that the output of the temperature network should represent the power usage for a certain temperature history. The idea is that an average thermal inertia of buildings will be encoded in the weights for the different time spans.

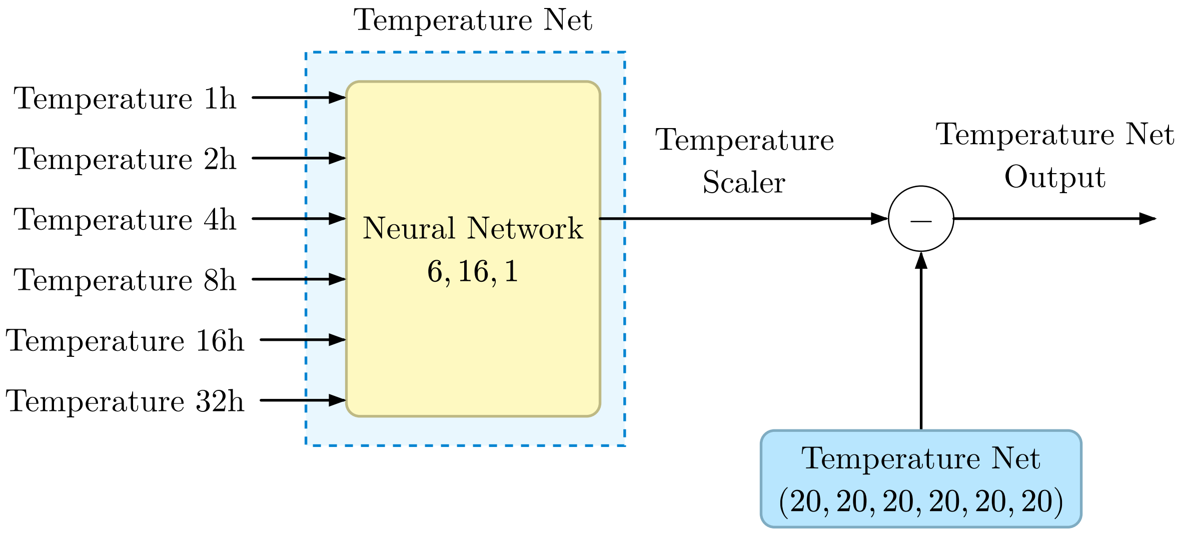


Figure 3.5: Overview of the model architecture used for the "Temperature Net" as seen in blue in Figure 3.3. The 20 °C values shown as inputs for the reference signal at the bottom are in reality normalized such that they fall between -1 and 1 (Section 3.1.3).

In this project, the solar PV model learns a seasonally adjusted solar power output scaling curve that captures the average effects of atmospheric scattering, orientation, tilt and temperature. This allows the neural network to approximate the real-world impact of PV systems on electricity usage over the year, without requiring detailed installation data for each panel. To do this, the solar network uses the current time of year as input, encoded as a 2D vector using sine and cosine as previously described. The output of this network is then fed though a sigmoid (σ) activation function to keep it between 0 and 1. This output is then multiplied with the current solar radiation for the area to provide the estimated current solar PV generation.

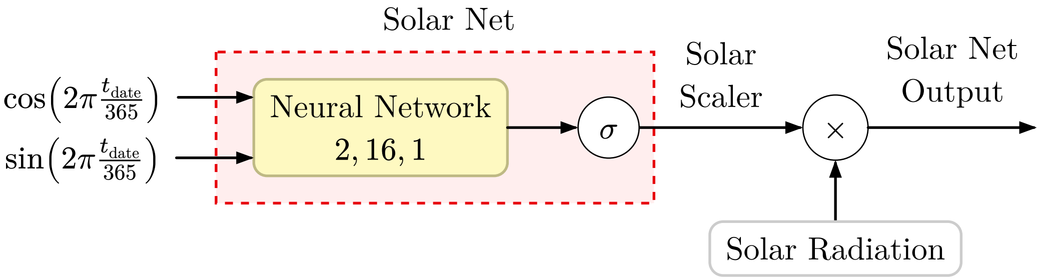

Solar Net
$\cos\left(2\pi \frac{t_{\text{date}}}{365}\right)$
$\sin\left(2\pi \frac{t_{\text{date}}}{365}\right)$
Neural Network
$2, 16, 1$
σ
Solar Scaler
Solar Net Output
$\times$
Solar Radiation

Figure 3.6: Overview of the model architecture used for the "Solar Net" as seen in red in Figure 3.3. σ is the sigmoid activation function.

To take the effects from Figure 2.3, Figure 2.4, Figure 2.5 and Figure 2.6 (Section 2.3.1) into account, the neural network outputs a solar generation scaling curve for the year to compensate for differences between the measured global solar radiation and the actual power outputs of the solar panels.

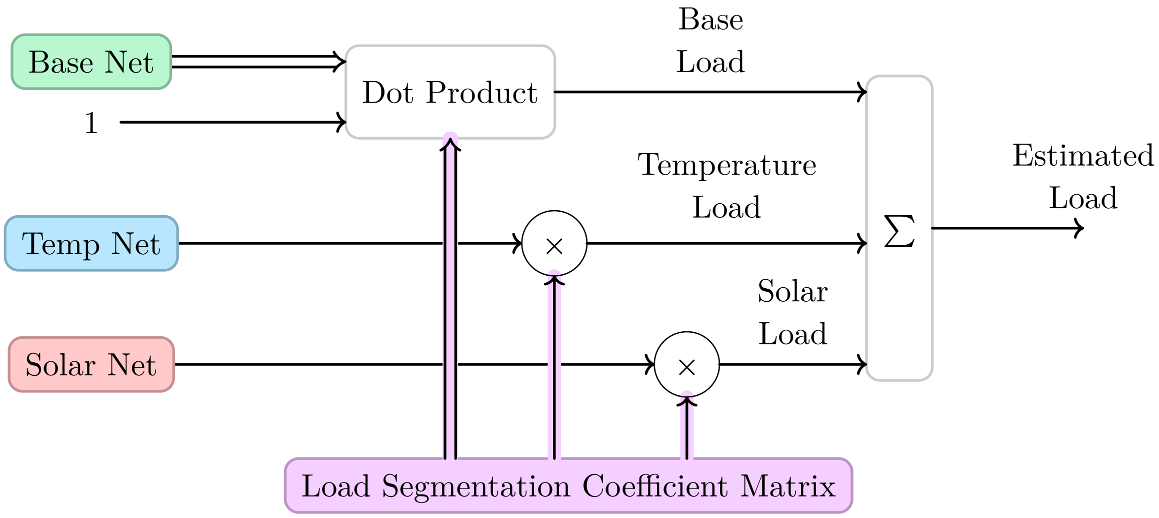


Figure 3.7: Diagram showing the dot product block from Figure 3.3 in more detail. The load segmentation coefficient matrix has values for each substation service area, which scale the load curves from the networks on the left side. The “1” below the Base Net is concatenated to the output from the Base Net.

3.2.2 Loss Function

For the load decomposition model, a simple MSE loss function was used (Section 2.5). A modified version was also tested, where the errors of cold days carried higher weight, but this was not used in the final version of the model. The reason for this was that in case of noisy (load and temperature) signals, the noise would have a disproportionate effect since the model would place too much weight on very few individual days.

3.2.3 Training

The model was implemented in PyTorch, which allowed the model to be trained using CPUs or GPUs. Section 2.4 describes the general theory for training neural networks.

For hyper-parameters, the model included learning rate, weight decay, epoch count and the sizes of the hidden neurons in the various networks. Since the network has few parameters, it has a low risk of overfitting the data, and therefore more epochs generally give better results but with diminishing returns.

A hyper-parameter sweep was also carried out, which consisted of automatically testing a number of configurations to find the best combination of hyper-parameters. The results are shown in Figure 3.8.

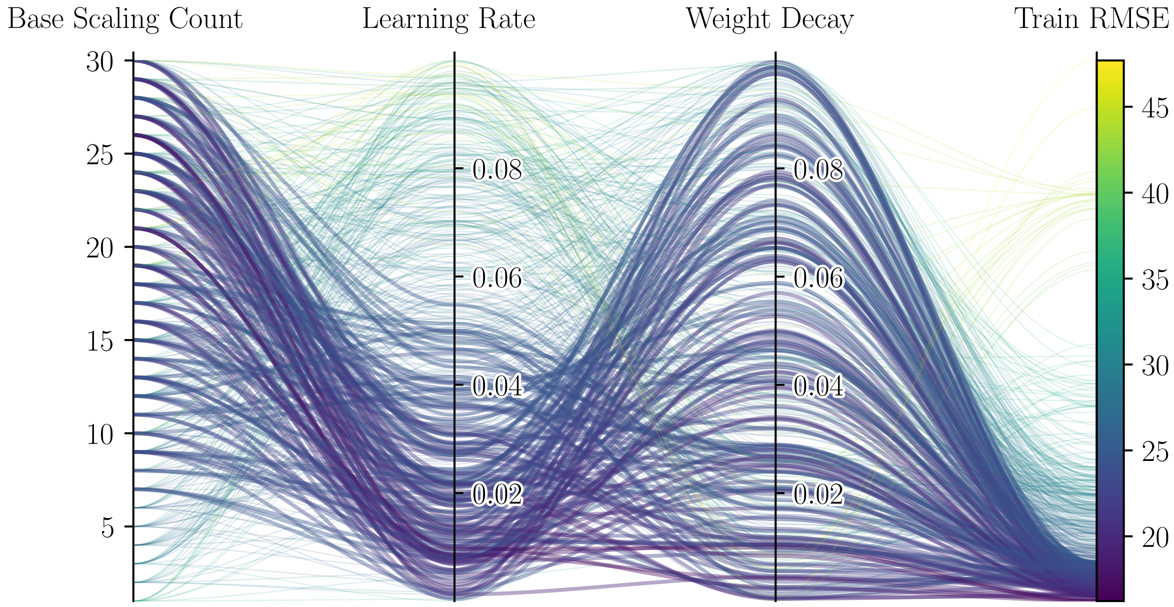


Figure 3.8: Diagram showing one of the sweeps that was carried out. The diagram displays how the number of base scalers, learning rate and weight decay affect the train loss. From these results, it was decided that seven was a sufficient compromise between simplicity and performance. For each run, the parameters were sampled from a uniform distribution. Train root mean square (RMS) error was chosen as the metric to optimize for since the model was considered to not be complex enough to overfit considerably.

3.3 Building-Informed Segmentation Model

In some cases, there is limited historical load data that can be used for segmentation, which means that other data sources will have to be used to estimate future load. This includes both completely new substation service areas, and modifications to old substation service areas such as new building projects. This is the goal of the Building-Informed Segmentation Model.

To summarize, the Building-Informed Segmentation Model is essentially a regression model that estimates the different load segmentation coefficients for a substation service area based on aggregated building information for that substation service area. In addition, it also estimates the size of the errors of its prediction.

The building information used as inputs in this regression is described in section Section 3.1.2. To summarize, it consists of features representing the quantities of each building category for the substation service areas. These quantities were represented by two features per category, one for total building volume and one for building count. The case with the building counts omitted was also tested. The target outputs for the regression model during training were the learned load segmentation coefficients as described in Figure 3.7.

3.3.1 Architecture

The building-informed segmentation model uses linear regression to estimate the load segmentation coefficients from the building data for that substation service area. Mean Square Error was used as the loss function. The reason for choosing linear regression was that the total load of the substation service area is a sum of the loads for each building, which increases linearly with more buildings. In addition, building volume was also chosen as a feature since larger buildings can house more electricity-consuming activities. Figure 3.9 displays the linear regression model in more detail.

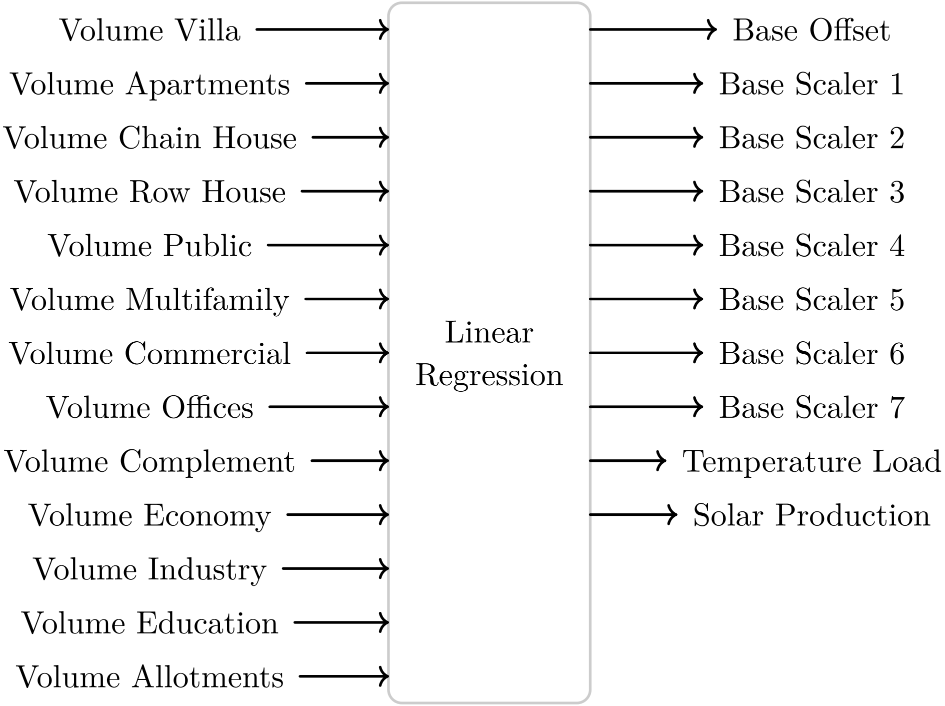


Figure 3.9: Overview of the building-informed segmentation model architecture.

This linear regression can be represented with a single matrix that transforms the input vector to the output vector, and can also be thought of as separate linear regressions for each output variable. To stop the regression from predicting something unphysical, sign constraints were placed on some of the outputs. This includes the base offset and temperature-based load, which were forced to be positive, as well as the solar based contribution which was forced to be negative.

3.3.2 Training

When training this model, two different models were used to do the linear regression depending on whether there was a non-negativity constraint on the coefficients. For the variables without any constraints, SciKit Learn's [37] `LinearRegression` method was used. For the non-negative values, the `nnls` method from the SciPy [38] Optimize library was used.

For the case of the non-negative regression, it was desirable for the model to use a larger share of all input features rather than relying on a single feature and setting the weights for the others to 0. For instance, in areas where the relative distributions of building categories are very similar, it can be hard for the system to find how the output is dependent on each specific building category, so it can be desirable to give the model some incentive to utilize more of the inputs for better robustness. To achieve this, L_2 regularization (a special case of ridge regression [39], also known as Tikhonov regularization [40]) was used. This can be thought of as a type of weight decay, since it penalizes the size of the weights.

Since both libraries do not support L_2 regularization for the respective linear regression functions, it was added manually by extending the data matrix with an identity matrix scaled with the square root of the desired regularization amount. Thus, the minimization problem is as follows:

$$\left\| \begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|^2 = \|Ax - b\|^2 + \left\| \sqrt{\lambda} I x - 0 \right\|^2 = \|Ax - b\|^2 + \lambda \|x\|^2. \tag{3.1}$$

The matrix $\sqrt{\lambda}I$ is a special case of the Tikhonov matrix Γ. The difference is that the Tikhonov matrix is a diagonal matrix where the values along the diagonal are not necessarily equal, whereas all diagonal values are equal in our case. To use a single regularization parameter λ, the input and output data had to be normalized. If it were not normalized, the amount of regularization for each feature would be dependent on the unit of that feature, which would be both arbitrary and unphysical. The normalization was done by dividing both input features and output features with the respective root mean square of all data for each feature. No offset was used, as is often the case with normalization. Since the relationship between the input and output quantities was believed to be linear, it was desirable to use a linear model as opposed to an affine one.

3.3.3 Error Estimation

The result of the linear regression presented earlier can be used to get an estimation of the likely load segmentation coefficients for a given substation service area. However, there is often a difference between the estimated parameters and the actual parameters, and it is useful to have an estimate of how large this difference is for a prediction. In this section, the method used to estimate the accuracy of the predictions is presented.

The first step of training this model was to create a list of residual vectors by subtracting the estimated load segmentation coefficient vectors from the actual.

The next step involved calculating the distributions of these residual vectors. These residual vectors were assumed to be multivariate normally distributed, which means that their distributions can be parameterized by a mean vector as well as a covariance matrix. In this case, the mean vector was equal to the zero vector because the resulting average difference from the output of the linear regression and the true values are zero by the definition of linear regression.

Since a vector of segmentation coefficients is estimated, it is not necessarily enough to only analyse the variances in each dimension. One might ask what the reason is for trying to estimate the covariance matrix instead of only the variances of each dimension. The reason for this is that there might often be some correlation between different dimensions of the error vector. For instance, two dimensions might correspond to two very similar load decomposition profiles that are almost linearly dependent, meaning that you can achieve close to the same result in multiple ways. In this case, if the model underestimates one and overestimates the other, the result when summing them might be small, while the error for each dimension is large. By using a covariance matrix to express the uncertainty, it is possible to say "coefficient A and coefficient B have high variance, but their sum is always close to 0" for instance. Thus, the goal of the residual prediction part of this model is to estimate the covariance matrix that represents the uncertainty of the prediction.

The covariance matrix was estimated by creating covariance matrices for each input feature, and then creating the resulting covariance matrix by summing the product of each feature times its covariance matrix. The covariance matrices for each feature (Σ_i) are constructed from their Cholesky decompositions (L_i), which are lower triangular matrices as described in Section 2.6.2. In practice, these are constructed from flattened learnable parameter vectors $\left(L_i^{-1}\right)_{\text{flat}}$. The specific steps to create each individual covariance matrix corresponding to input i are as follows:

1. Reshape $\left(L_i^{-1}\right)_{\text{flat}}$ into a triangular matrix L_i^{-1}.
2. Apply $\text{Softplus}(x) = \log(1 + e^x)$ to diagonal entries to ensure positive definiteness.
3. Invert L_i^{-1} to create L_i.
4. Construct Σ_i using $\Sigma_i = L_i L_i^T$.

The reason for doing the inverting step was for numerical stability during training, since estimating L_i directly would often lead to the loss randomly spiking up.

The final covariance matrix is then calculated by summing the products of each covariance matrix multiplied by the corresponding input value v_i:

$$\Sigma = \sum_i v_i \Sigma_i \tag{3.2}$$

The rationale behind this method is rooted in the central limit theorem, and assuming that each substation service area is made up of multiple smaller elements. By assuming that each individual element has a certain mean μ_k and covariance matrix Σ_k, then the central limit theorem states that the sum of these elements (assuming that they are independent) is normally distributed, with the mean equal to the sum of the individual means and the covariance equal to the sum of the individual covariances.

When training, the Generalized Gaussian Negative Log Likelihood was used, as described in Section 2.6.1. No weight decay was used for the matrix elements. Adam was used as the optimizer, and the model was trained for 200 epochs with a batch size of 256. A manual learning rate scheduler was used, with a learning rate of 1 initially, which was increased to 100 after 20 epochs, and then decreased to 10 after

100 epochs. The learning rate was determined empirically to get the fastest convergence speed. The β values of Adam were set to $\beta_1 = 0.9$ and $\beta_2 = 0.99$.

For the model to scale properly for substation service areas with higher load, multiple substation service areas were combined to create larger substation service areas. This dataset was created by selecting a random integer between 1 and 5, and then selecting and combining that many substation service areas from the same city. This was repeated 1000 times. The resulting dataset was split into a training dataset and a test dataset, with 80% of the data used for training and 20% for testing. The training dataset was then used to train the model, while the test dataset was used to ensure the model was not overfitting.

3.4 Residual Process Simulation Model

The third model in this project is named the “Residual Process Simulation Model”. As the name suggests, it is used to simulate the stochastic part of the load profile. It is used in cases where there already exists historical load data available for a substation service area, and modelling its future behaviour is desired. This is useful in cases where the load duration curve is of interest, since simply using the load profile from the load decomposition model will often lead to a flatter curve than in reality, which underestimates maxima and overestimates minima.

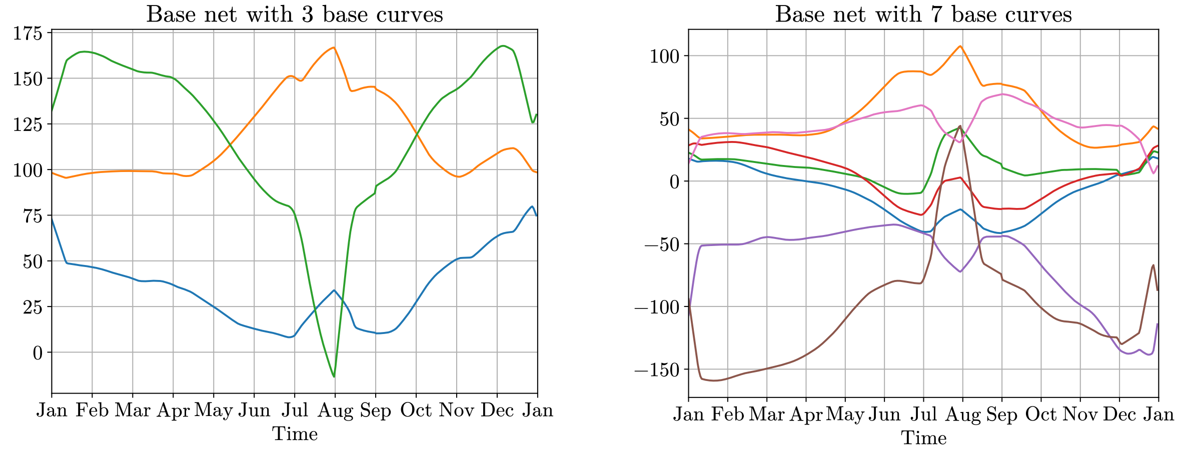
3.4.1 Background and Motivation

We define the residual as the difference between the output from the load decomposition model and the actual load. Since the output of the load decomposition model has previously been assumed to cover the deterministic part of the load (see Section 3.2), the residual can thus be considered the stochastic part. Worth noting is that we call these deviations stochastic, even though they could be considered deterministic if other data sources were used. For instance, there was a substation connected to a football pitch, which would have increased load during games.

The goal is to model this stochasticity in a manner that is not only possible but also realistic. To define what constitutes a “realistic” residual process, we can begin with a few examples that would be considered too unrealistic. A common method to model stationary stochastic processes is with linear autoregressive models such as autoregressive moving-average (ARMA). There are a few limitations of using this method, one of them being that it does not capture any exogeneity (dependence on other variables such as temperature, time of day or solar radiation), which can be assumed to influence the behaviour of the process. For instance, there is higher uncertainty in the contribution from the solar radiation during the day compared to the night. Furthermore, it does not capture any periodic behaviour of the stochastic process.

A challenge related to modelling the stochasticity is to define what should be considered stochastic deviations, and what is unique deterministic or recurring deviations for a given substation service area. For instance, there might be a decrease in load during a few weeks in the summer for a substation service area. This could be caused by some of the residents leaving their homes for their yearly summer holidays, which one could reasonably expect to be a recurring event. On the other hand, it could also be caused by one of the facilities in the area being closed due to unexpected maintenance. Since load data was only available for a single year, one could assume that having data for more years could help with this. However, there are also large changes in the substation service areas due to new buildings being built, and more solar panels being installed, which would mean that the additional years of data would already be out-of-date.

A result of this is that the choice of what is considered stochastic as opposed to deterministic is somewhat arbitrary and dependent on the chosen model architecture. A simple assumption is that short-term deviations are more likely to be stochastic, whereas long-term deviations are deterministic. In our model, determining what constitutes long- and short-term deviations is done by the load decomposition model. More specifically, increasing the number of base scalers (which are different for all substation service areas) will allow the model to fit the profile arbitrarily well. In the end, a base scaler count of 7 was chosen, which ended up having comparable losses to models with up to 30 base scalers. It is not immediately evident what equivalent “time constant” this translates to, but plotting the different profiles hints at that they are likely on the order of months. Furthermore, the base scalers also need to represent differences in intraday and holiday patterns, which results in less representational capacity allocated to the yearly patterns. This supports the idea of the timescale being on the order of months, since if the 12 months in a year are divided with a bit less than 7 parameters, it results in a few months per parameter.



(a): Base net with 3 base curves.

(b): Base net with 7 base curves.

Figure 3.10: Diagrams showing the yearly base load component curves for 3 and 7 base scaler coefficients respectively. The diagrams show the daily averages to improve legibility. A higher number of base scaler curves provides the model with more opportunities to fit specific short-term deviations that occur at different times of year.

3.4.2 High Level Description of Solution

The proposed solution for this problem involves dividing the hourly residuals into daily chunks of length 24, predicting the distribution for the next chunk autoregressively, and then sampling from that distribution. A Monte Carlo simulation can be run by repeating this process several times to get insights about the stochastic behaviour of the load, including metrics such as the expected maximum load in a year or week. Furthermore, in each simulation run, simulated or historical weather data could also be used. In cases where the load segmentation coefficient distributions have been estimated from the building data, one could also sample these in every run to include the uncertainty from that estimation.

3.4.3 Architecture

The architecture can be divided into two models. The purpose of the first model is to estimate the means for the next 24 hours of load, and the purpose of the second model is to estimate the covariance matrix for these values. Both models have the same inputs values.

The first model, which estimates the mean, consists of a linear transform of the input variables without any offset.

The second model, which estimates the covariance matrix, consists of a neural network with a hidden layer of size 16 and an output layer of size 300. The output size of 300 is the 24th triangle number, which is the number of parameters in the 24×24 lower triangular matrix L that parametrizes the covariance matrix as described in Section 2.6.2. In other words, the model outputs L^{-1}, by which the covariance matrix Σ is calculated in the following way:

$$\Sigma = \left((L^{-1})^T L^{-1} \right)^{-1} \tag{3.3}$$

The chosen input features were the following:

- The 24 previous hourly load values.
- The solar radiation of the current day. This is projected down from 24 hourly dimensions down to 3 using a linear autoencoder to reduce computational complexity and risk of overfitting. The curves representing each of the three dimensions can be found in Figure 3.14.
- The temperature of the current day. This is projected down to 3 dimensions similarly to the solar radiation (Figure 3.13).
- A workday parameter that is 1 for workdays and 0 during holidays and weekends.
- The sine and cosine of the time of year. More information about this signal is presented in Section 3.1.2.
- The load segmentation coefficients for that substation service area.

All features have been normalized appropriately as described in Section 3.1.3.

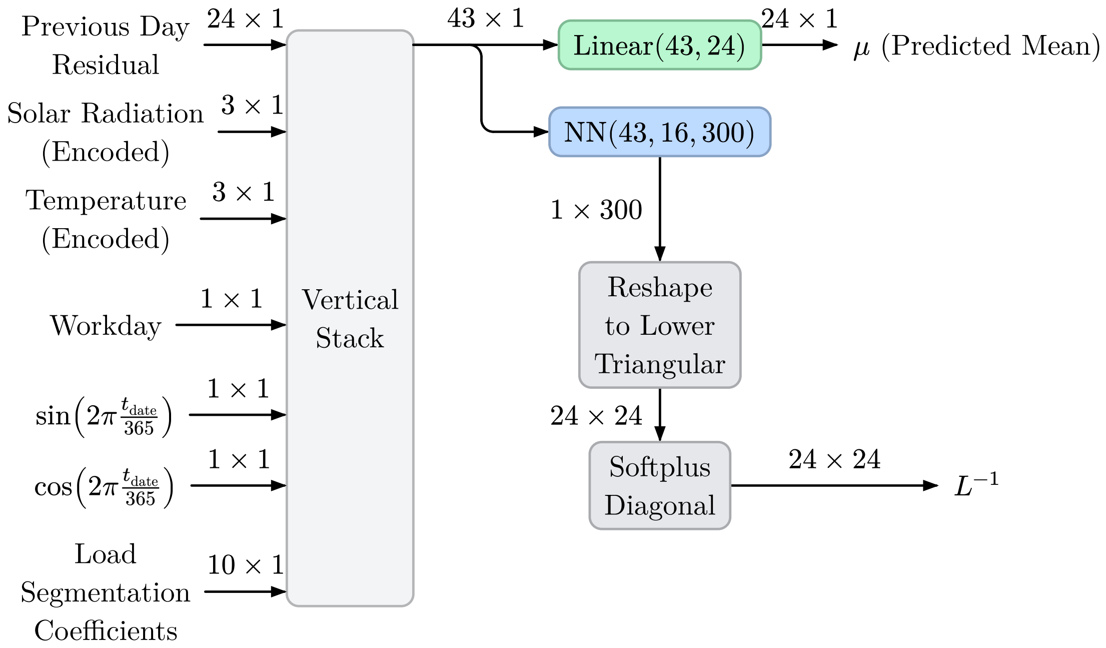


Figure 3.11: The residual prediction system. It predicts the mean of the next day's load residual for each hour (μ), as well as the inverse of the Cholesky decomposition of the covariance matrix for these 24 residuals. "Linear" is a linear transform with no bias or activation function. "NN" is a neural network with ReLU activation functions. The number 300 is the shape of a flattened 24 × 24 lower triangular matrix ($\frac{24(24+1)}{2}$).

3.4.4 Training

The training for this system consisted of two parts. The first part involves training of the autoencoders that project down the solar and temperature signal to lower dimensions. When training these autoencoders, MSE loss was used. There was no mini-batching done during training; all data was concatenated into a single input vector for each epoch. A total epoch count of 500 was used. The optimizer was Adam and the learning rate was 0.1, along with no weight decay being used. Since there were so few tuneable parameters, there was no need to split the dataset into separate training and test sets.

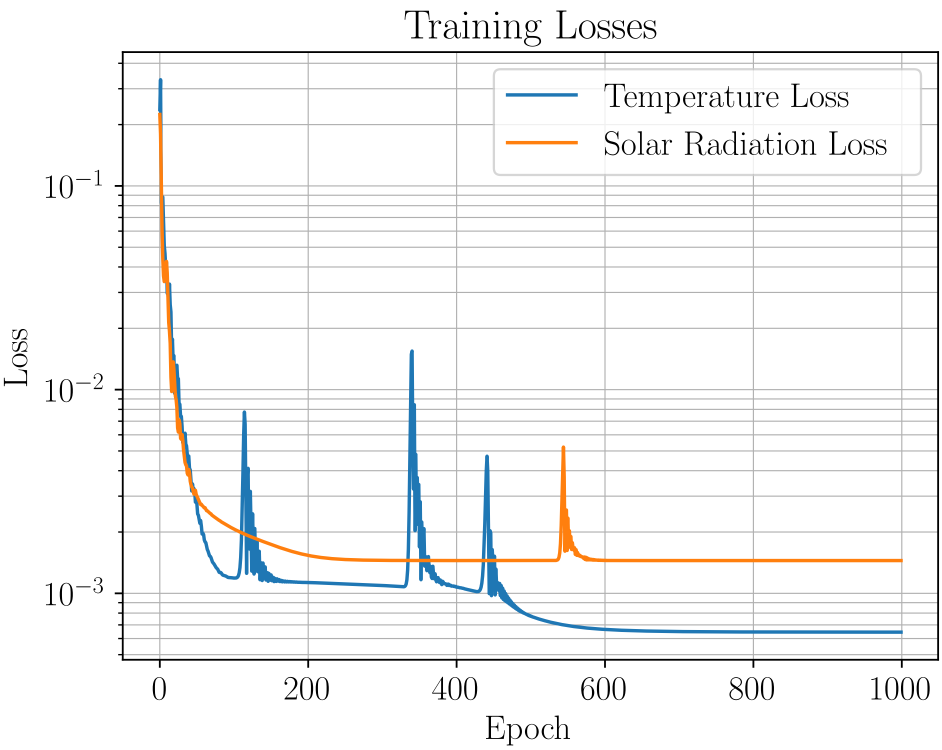


Figure 3.12: The training loss for the autoencoders that project down the temperature and solar radiation signals. The loss is the MSE between the original signal and the reconstructed signal.

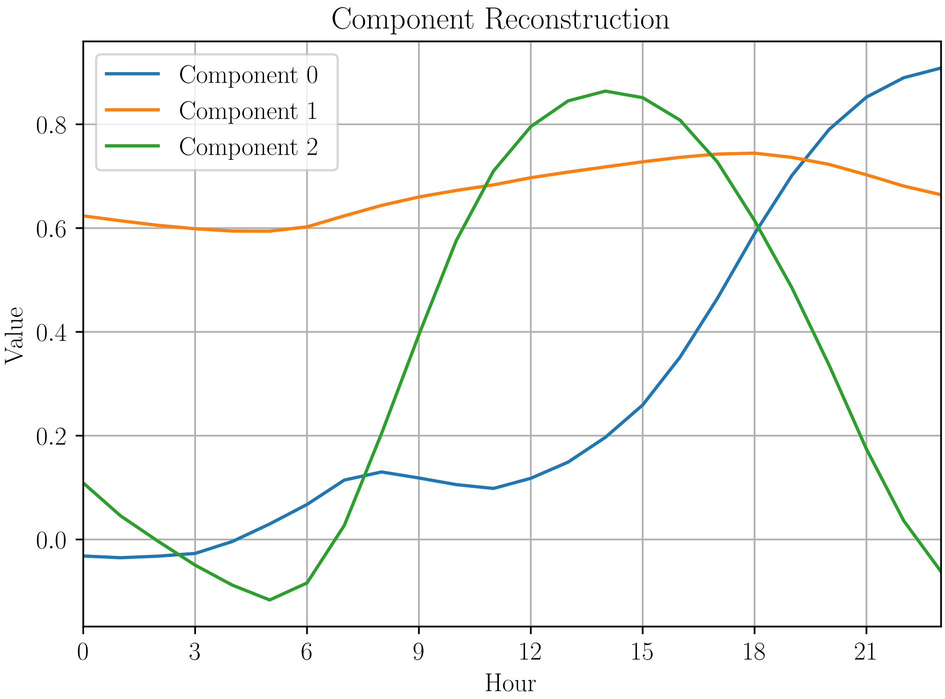


Figure 3.13: The three basis functions used to reconstruct the daily temperature signal, as learned by the autoencoder.

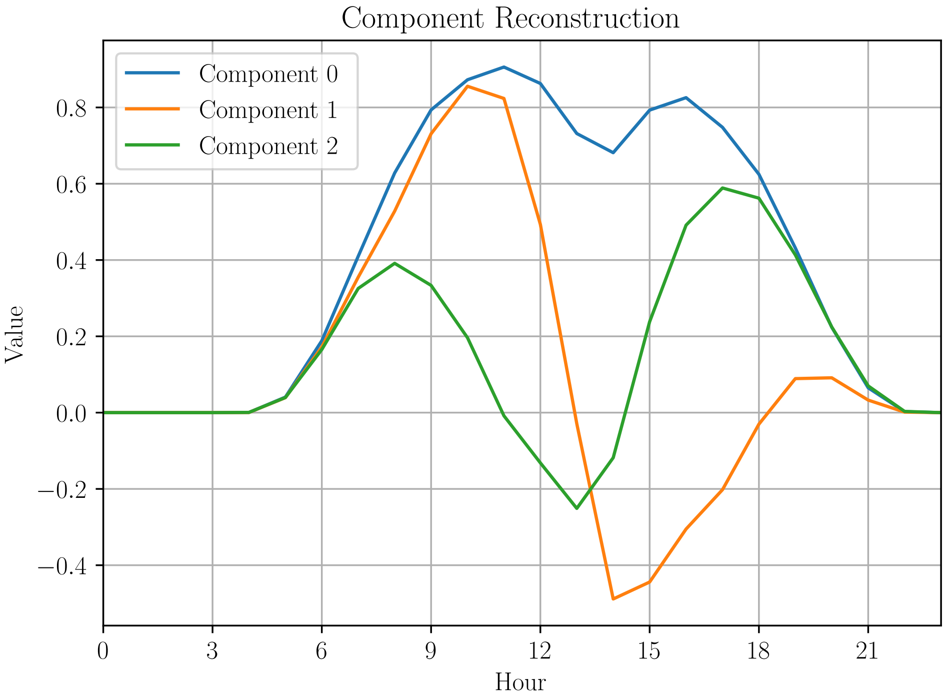


Figure 3.14: The three basis functions used to reconstruct the daily solar radiation signal, as learned by the autoencoder.

The second part of training this system involves the autoregressive residual predictor. Since the model estimates both the mean and the covariance, Generalized Gaussian Negative Log Likelihood was used as a loss function. The dataset was split into a training set and test set, with both sets containing equal proportions of the data. Adam was used as the optimizer, with a learning rate of 0.02, and $\beta_1 = 0.9$ and $\beta_2 = 0.95$. A batch size of 512 was used, as well as 50 epochs in training.

4

Results

This chapter presents the results obtained from the developed models, accompanied by a discussion of their implications and limitations. The results are structured according to the three primary components of the framework, beginning with the load decomposition model.

4.1 Load Decomposition Model

The load decomposition model aims to segment the total electrical load into interpretable components. Results are presented for each of the three modelled load types, Solar PV generation, temperature-dependent load, and base load, as well as an evaluation of the combined model performance.

4.1.1 Solar PV Generation

The Solar Network lowers the peak estimated solar generation during the summer. This aligns with the theory presented in Section 2.3.1.

Since only the aggregate load data is available in this project, no true reference values exist for direct comparison. However, the results align with the theory presented in Section 2.3.1. Figure 4.1 shows a diagram of how the solar network modifies the solar radiation signal during the year.

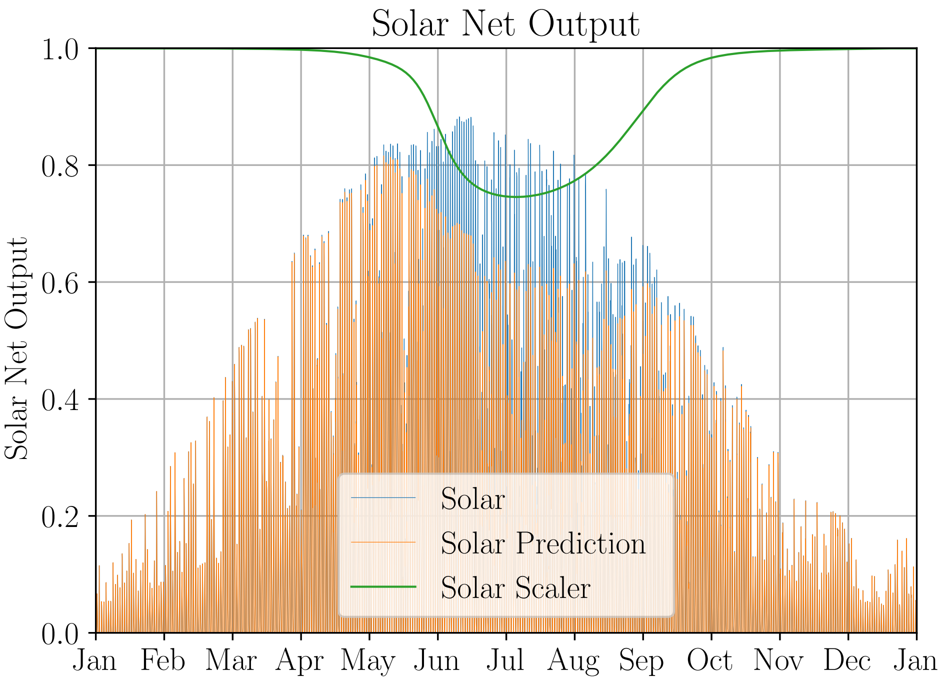


Figure 4.1: The predicted solar cell output from solar data. “Solar” refers to the actual solar radiation measured as global radiation; the radiation experienced for 1 m^2 laid flat on the ground. “Solar prediction” is the “Solar” curve multiplied with the “Solar Scaler” curve.

4.1.2 Temperature Load

Figure 4.2 shows how the output of the temperature network depends on the temperature. The figure shows a simplification of when all the past temperatures are constant, which is rarely the case in reality, but gives a sufficient estimate of the model output for different temperatures.

One observation for the temperature-based load was that there is a slight increase in estimated load as the temperature increases over 20 °C. This was theorized to be caused by electric-powered cooling systems, such as air conditioning. This effect is applied similarly to all buildings, but should ideally be decoupled from the heating-based load category. A possible issue with this is that the cooling based load likely has some correlation with the solar PV radiation, which could make them difficult to separate. This approach was attempted, but was intentionally omitted due to the relatively small total effect it had, as well as the desire to reduce parameter count to avoid overfitting.

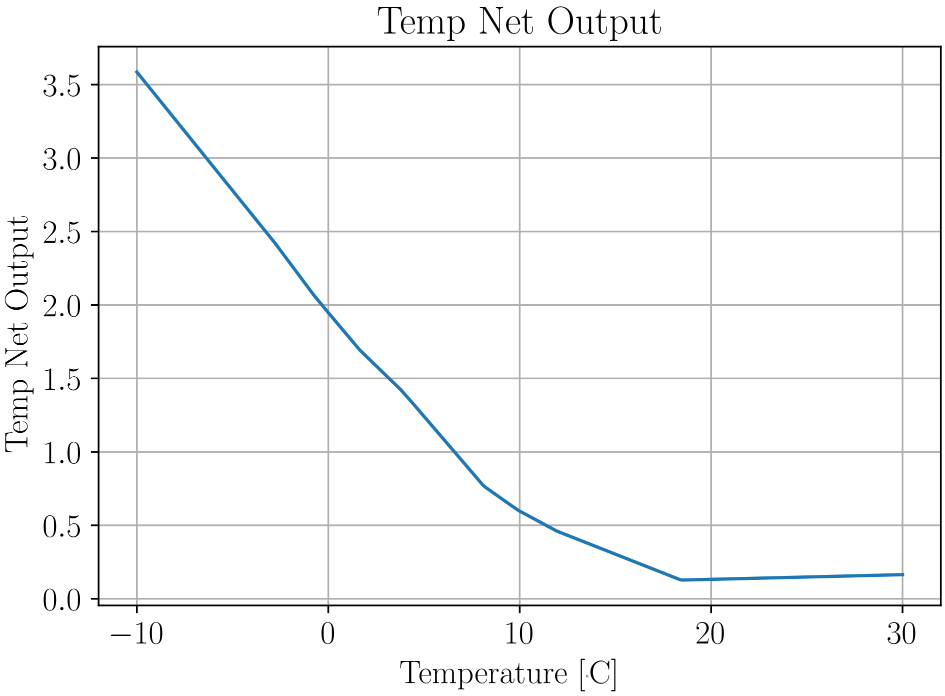


Figure 4.2: Diagram show the predicted temperature net output as a function of the outdoor temperature (with the assumption that it is constant in time).

4.1.3 Base Load

Figure 4.3 shows an example output for the base load, calculated by performing a weighted sum of the outputs from the load decomposition model. It can be seen that it captures a daily variation as well as some peaks during weekends. It also has some yearly variations. Closer inspection shows that there is some difference between the output for the first and last days of the year, which is somewhat unexpected since the sine and cosine encodings of the time of year are expected to lead to periodic and continuous patterns, which is not seen here. A theory of what causes this could be that the assumption that electricity demand patterns are constant from year to year is not necessarily true. In reality, there could be differences from construction of new buildings, changing consumer behaviour or a tendency towards electrification of consumer products such as cars, heating, and other appliances.

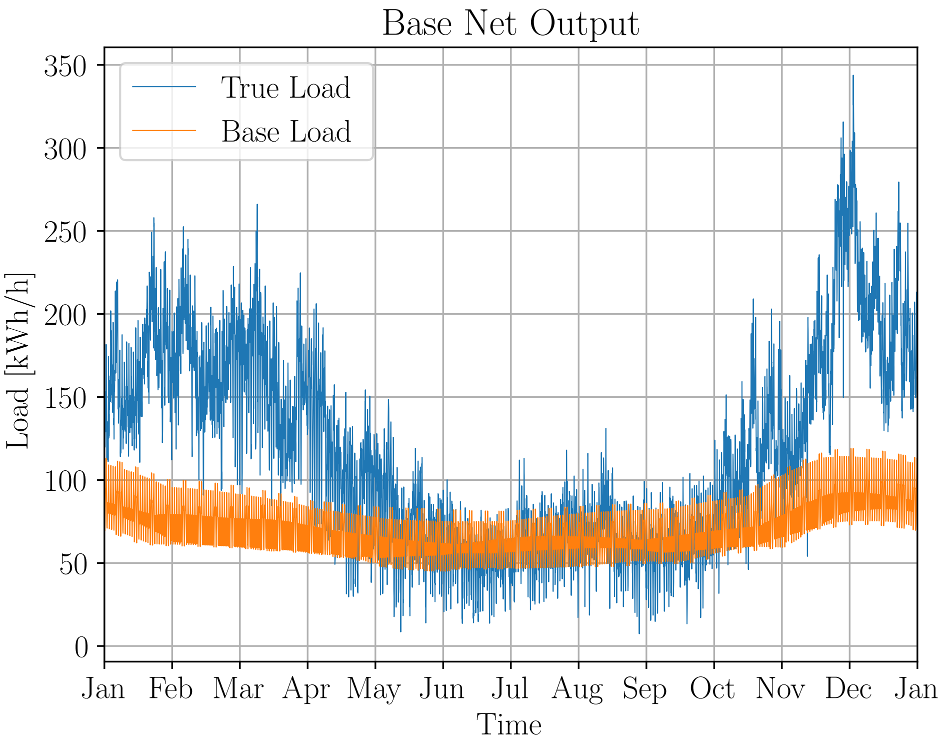


Figure 4.3: The predicted base load compared to the true load for a typical residential area.

4.1.4 All Loads

Figure 4.4 displays the different load profiles. The loads for each category are added together to form the total load prediction, which is compared with the measured load.

Figure 4.5 shows the final prediction for the total load. It can be seen that the model does not always follow the measured load profile for the extreme values due to their stochastic nature. To take this into account, the residual process simulation model can be used.

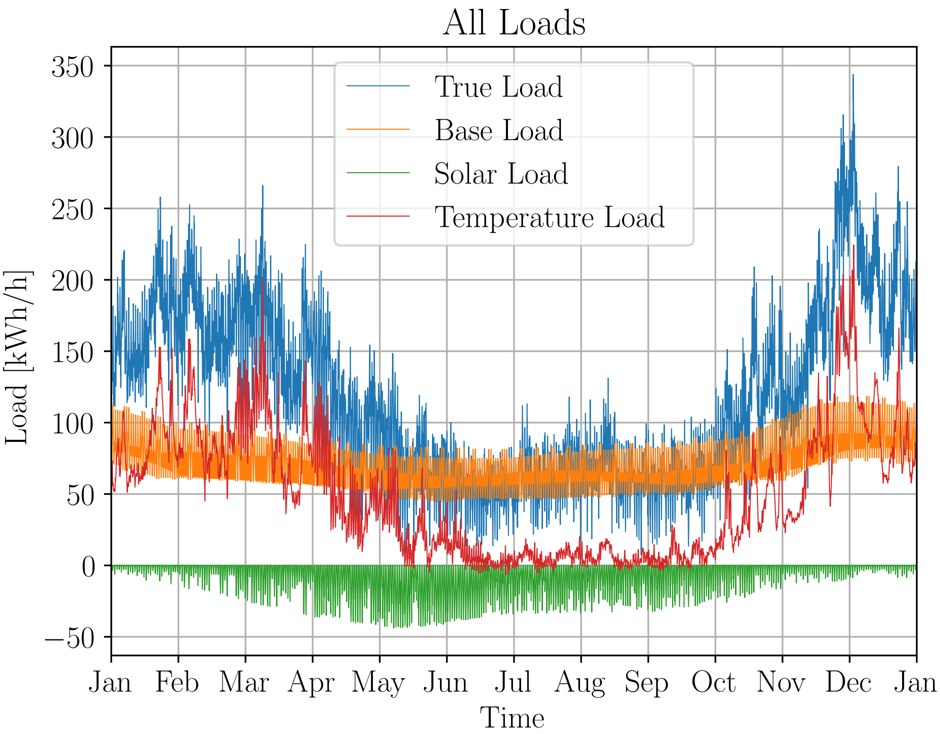


Figure 4.4: All three predicted loads compared to the true load.

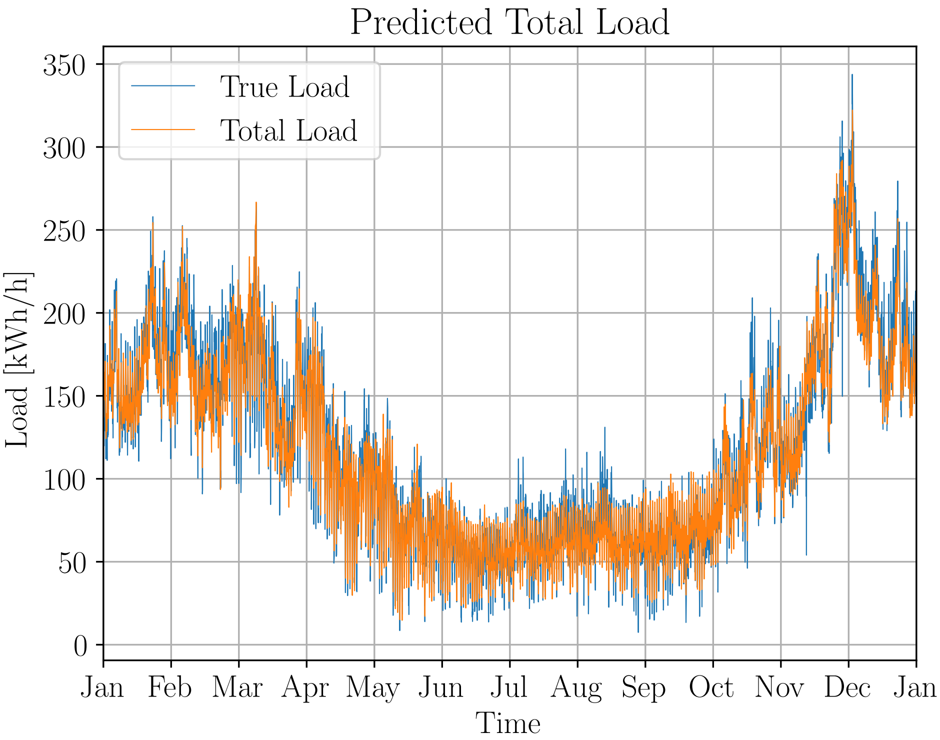


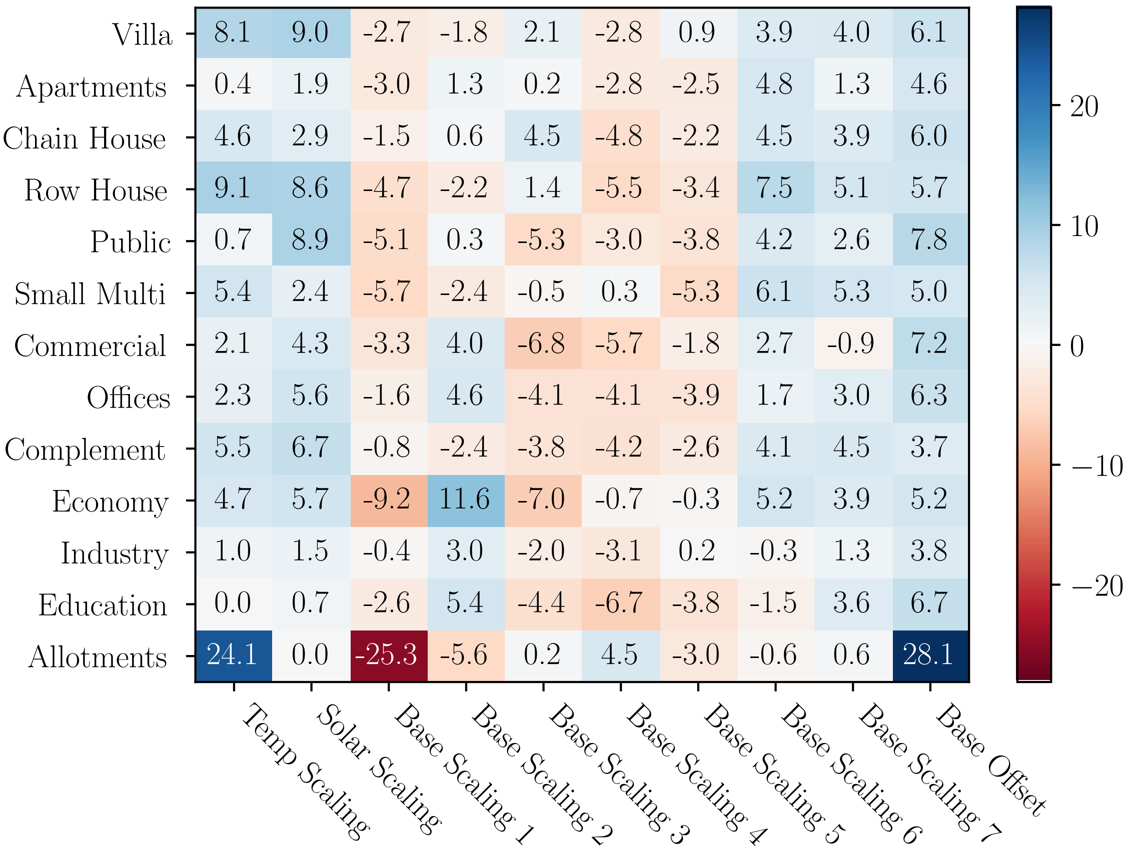
Figure 4.5: The sum of all separate loads in Figure 4.4 compared to the true load.

A metric was created to allow for comparison of the results for different substation service areas. This involved splitting each measured and predicted load profile into 12 months, and calculating the monthly maximum loads (which might occur at different times in the month for the estimated and true load profiles). The mean absolute percentage error (MAPE) scores were then calculated for each station by comparing the monthly predicted max loads to the measured ones. This resulted in a mean MAPE of 14.5 % and median MAPE of 12.0 %. This includes substation service areas of all sizes, from one to two houses up to thousands of houses with equal weight for each substation service area. The model performed worst on smaller substation service areas since a larger portion of their load is stochastic.

4.2 Building-Informed Segmentation Model

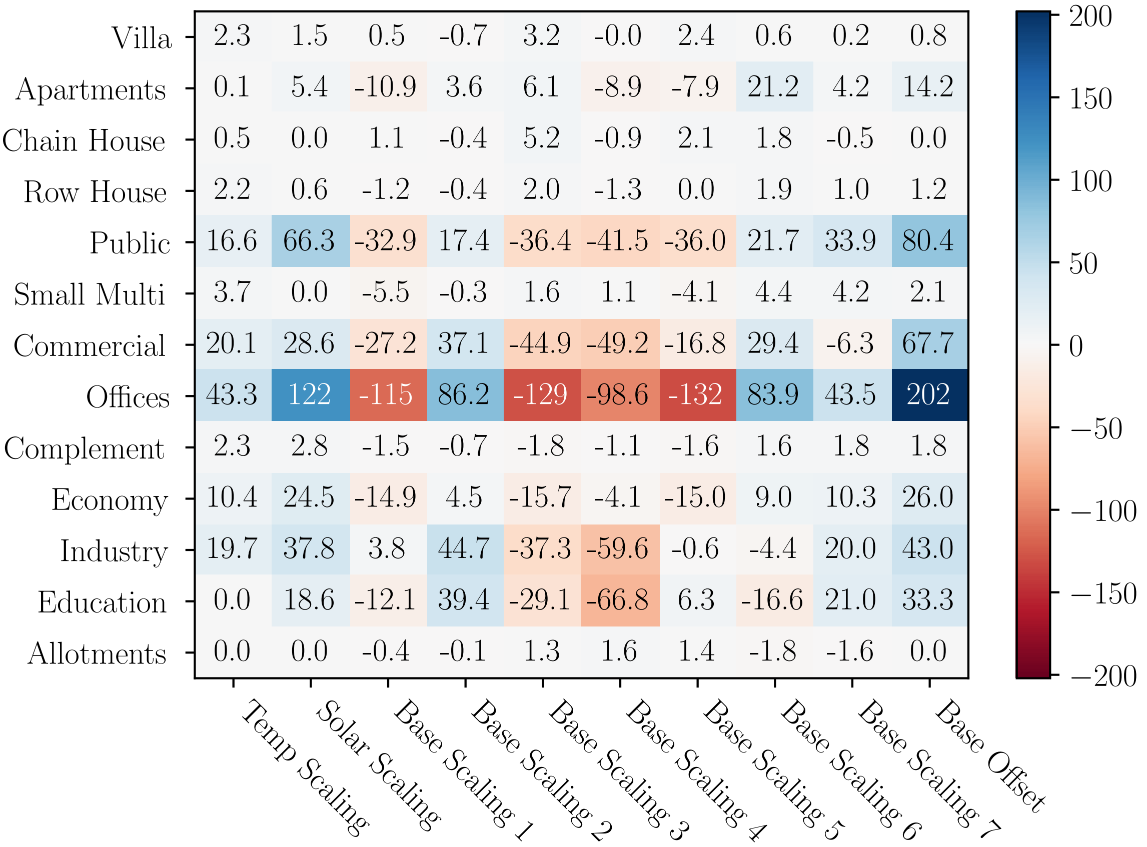
On average, predicting the load segmentation coefficients from only the building parameters resulted in rather high variance. For some segmentation coefficients, such as the solar radiation coefficient, and in part the temperature coefficient, this is relatively expected. The reason for this is that there is often a high variance in how much solar generation is installed per building and per building volume, which leads to high uncertainty. Furthermore, since no data on the type of heating in the buildings was used (e.g. district heating, local combustion, local electricity-driven heat pumps, resistive heating), there is a high uncertainty for the temperature component as well. Data for installed solar PV capacity or satellite images from which installed PV capacity could be calculated would be useful for improving the solar PV generation estimate. Similarly, the source of heating and the building energy classification would be useful for improving the temperature dependent power estimate.

The learned coefficients from the linear regression can be shown in a heat map, see Figure 4.6 and Figure 4.7.



	Temp Scaling	Solar Scaling	Base Scaling 1	Base Scaling 2	Base Scaling 3	Base Scaling 4	Base Scaling 5	Base Scaling 6	Base Scaling 7	Base Offset
Villa	8.1	9.0	-2.7	-1.8	2.1	-2.8	0.9	3.9	4.0	6.1
Apartments	0.4	1.9	-3.0	1.3	0.2	-2.8	-2.5	4.8	1.3	4.6
Chain House	4.6	2.9	-1.5	0.6	4.5	-4.8	-2.2	4.5	3.9	6.0
Row House	9.1	8.6	-4.7	-2.2	1.4	-5.5	-3.4	7.5	5.1	5.7
Public	0.7	8.9	-5.1	0.3	-5.3	-3.0	-3.8	4.2	2.6	7.8
Small Multi	5.4	2.4	-5.7	-2.4	-0.5	0.3	-5.3	6.1	5.3	5.0
Commercial	2.1	4.3	-3.3	4.0	-6.8	-5.7	-1.8	2.7	-0.9	7.2
Offices	2.3	5.6	-1.6	4.6	-4.1	-4.1	-3.9	1.7	3.0	6.3
Complement	5.5	6.7	-0.8	-2.4	-3.8	-4.2	-2.6	4.1	4.5	3.7
Economy	4.7	5.7	-9.2	11.6	-7.0	-0.7	-0.3	5.2	3.9	5.2
Industry	1.0	1.5	-0.4	3.0	-2.0	-3.1	0.2	-0.3	1.3	3.8
Education	0.0	0.7	-2.6	5.4	-4.4	-6.7	-3.8	-1.5	3.6	6.7
Allotments	24.1	0.0	-25.3	-5.6	0.2	4.5	-3.0	-0.6	0.6	28.1

Figure 4.6: A heat map of the resulting matrix from the linear regression. The colours show how each estimated output segmentation coefficient depends on the volume of each building type. The values are horizontally comparable, but not vertically, since they scale different quantities. The unit is how much each segmentation coefficient increases per $1\,000\,\mathrm{m}^3$ of building volume, equivalent to a $10\,\mathrm{m} \times 10\,\mathrm{m} \times 10\,\mathrm{m}$ building.



	Temp Scaling	Solar Scaling	Base Scaling 1	Base Scaling 2	Base Scaling 3	Base Scaling 4	Base Scaling 5	Base Scaling 6	Base Scaling 7	Base Offset
Villa	2.3	1.5	0.5	-0.7	3.2	-0.0	2.4	0.6	0.2	0.8
Apartments	0.1	5.4	-10.9	3.6	6.1	-8.9	-7.9	21.2	4.2	14.2
Chain House	0.5	0.0	1.1	-0.4	5.2	-0.9	2.1	1.8	-0.5	0.0
Row House	2.2	0.6	-1.2	-0.4	2.0	-1.3	0.0	1.9	1.0	1.2
Public	16.6	66.3	-32.9	17.4	-36.4	-41.5	-36.0	21.7	33.9	80.4
Small Multi	3.7	0.0	-5.5	-0.3	1.6	1.1	-4.1	4.4	4.2	2.1
Commercial	20.1	28.6	-27.2	37.1	-44.9	-49.2	-16.8	29.4	-6.3	67.7
Offices	43.3	122	-115	86.2	-129	-98.6	-132	83.9	43.5	202
Complement	2.3	2.8	-1.5	-0.7	-1.8	-1.1	-1.6	1.6	1.8	1.8
Economy	10.4	24.5	-14.9	4.5	-15.7	-4.1	-15.0	9.0	10.3	26.0
Industry	19.7	37.8	3.8	44.7	-37.3	-59.6	-0.6	-4.4	20.0	43.0
Education	0.0	18.6	-12.1	39.4	-29.1	-66.8	6.3	-16.6	21.0	33.3
Allotments	0.0	0.0	-0.4	-0.1	1.3	1.6	1.4	-1.8	-1.6	0.0

Figure 4.7: A heat map of the resulting matrix from the linear regression. The colours show how each estimated output segmentation coefficient depends on the number of buildings for each building type. The values are horizontally comparable, but not vertically, since they scale different quantities.

This model can be used to show how well the linear regression model estimates the different segmentation coefficients. Figures 4.8, 4.9 and 4.10 plot the actual and estimated coefficients, along with their correlation coefficient.

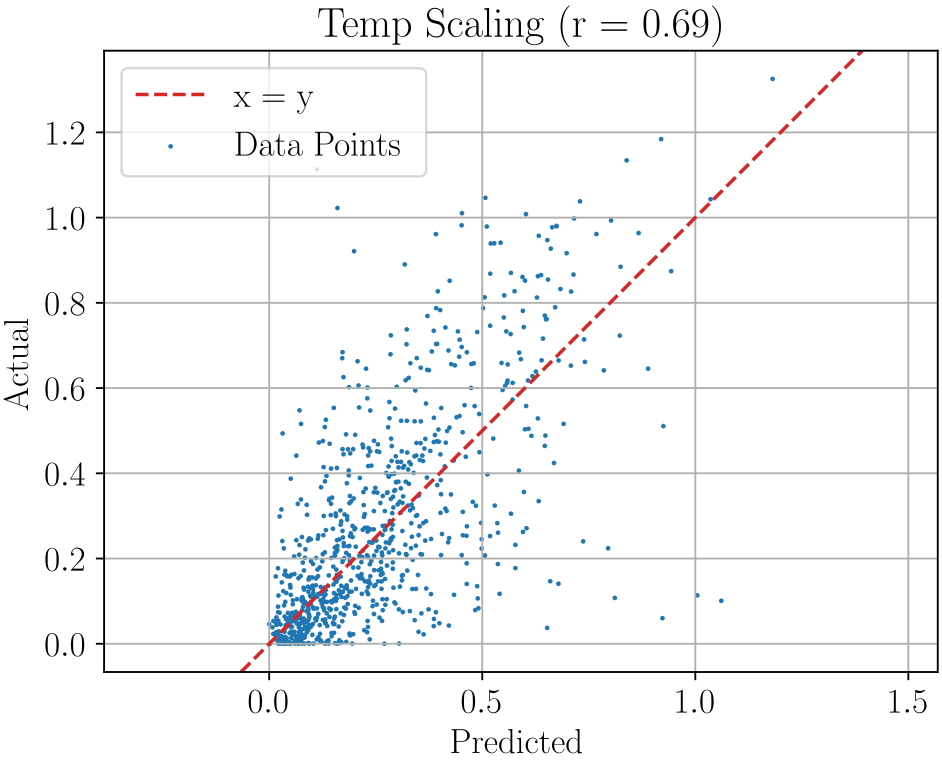


Figure 4.8: Predicted and actual values for the `temp_scaling` coefficient for all substation service areas. The actual values are calculated using the load decomposition model. The Pearson correlation coefficient in this case is 0.69.

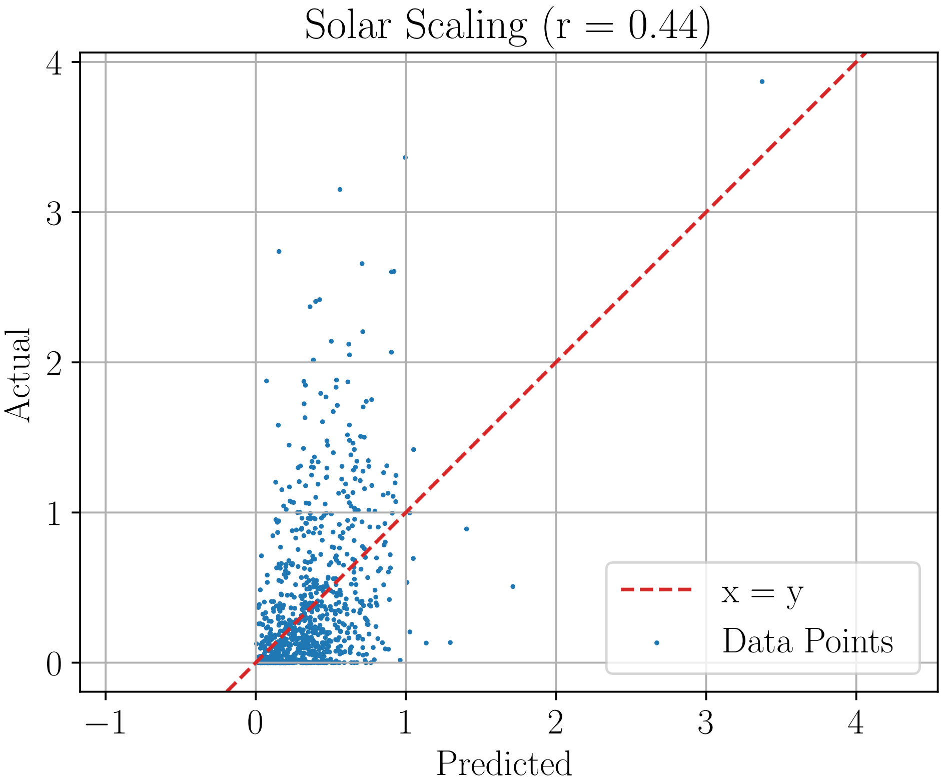


Figure 4.9: Predicted and actual values for the `solar_scaling` coefficient for all substation service areas. The actual values are calculated using the load decomposition model. The Pearson correlation coefficient in this case is 0.44. This shows that the amount of solar energy does not correlate too linearly with the aggregated building data, at least for substation service areas with sizes of those in the dataset. For larger substation service areas, the effect is expected to be closer to linear.

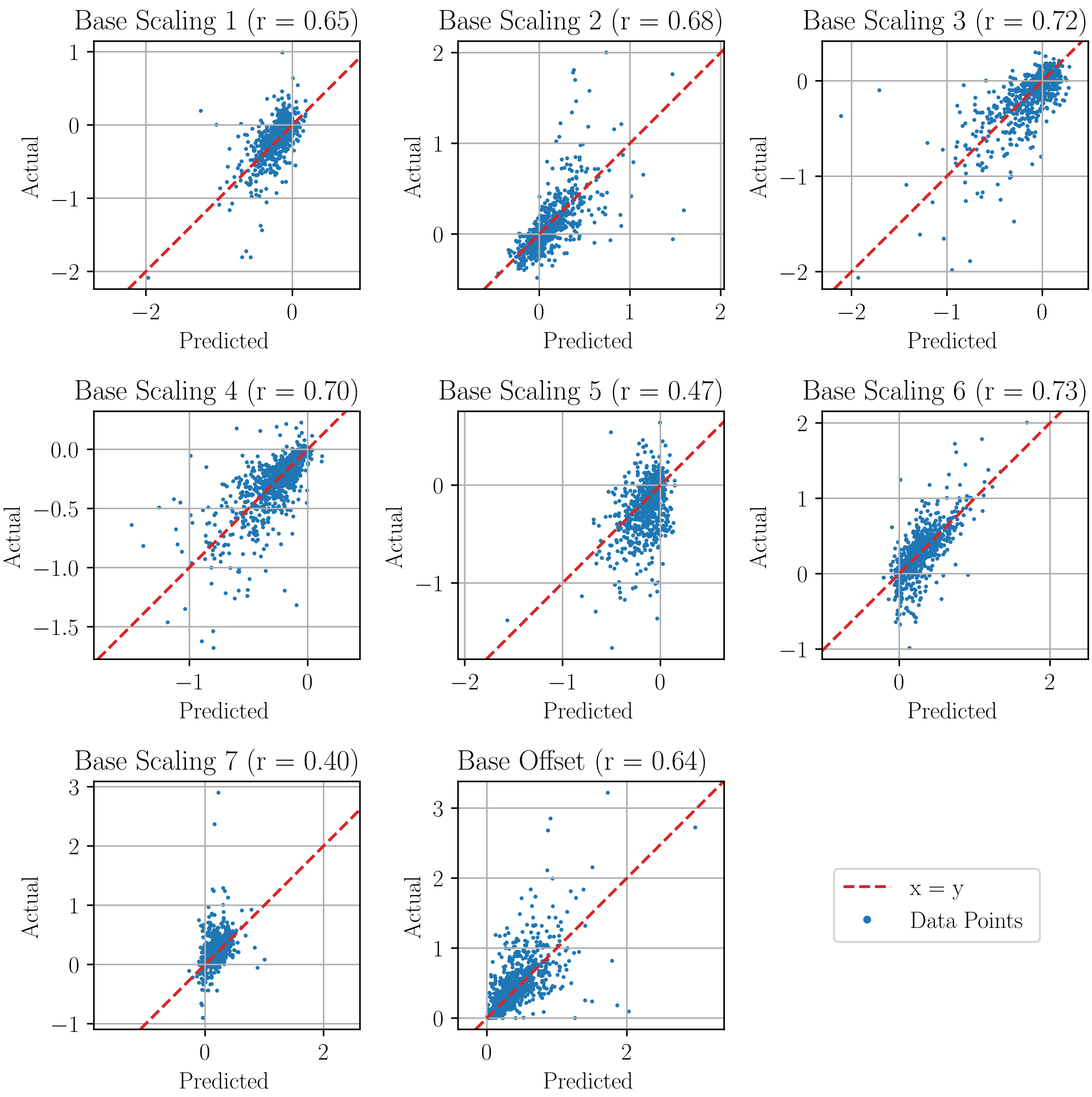


Figure 4.10: Predicted and actual values for the `base_scaling` coefficient for all substation service areas. The actual values are calculated using the load decomposition model. The base scaler coefficients have no non-negativity constraint, except for the base offset coefficient.

With the relatively high errors from the linear regression model, evaluation of the error prediction part of the model becomes of high interest. The error prediction model has been optimized to minimize the negative log likelihood, which can be useful when comparing models to each other, but it can be hard to infer whether the model actually performs well or not. One possible way to measure this is by studying the average errors compared to the average predicted standard deviations, and there are multiple ways to approach this.

For the case of a univariate normal-distributed variable, one can express the error between the estimation and actual value in terms of standard deviations, in which case it is referred to as the z-score. Here, the average of the z-score squared is equal to 1. This can be generalized to multivariate distributions, where it is instead named the Mahalanobis distance [41]. The Mahalanobis distance is, similar to the z-score, a measure of how far away a point is from a distribution. The expected square of the Mahalanobis distance generalizes to the number of dimensions, instead of 1. The Mahalanobis distance can be calculated by applying a whitening transformation to the error vectors, which essentially means transforming them to vectors that have the identity matrix as their estimated covariance matrix. After applying this transform, the Mahalanobis distance is simply the Euclidean distance in the transformed space. This can be done by transforming the error vectors with the inverse of the Cholesky decomposition of the covariance matrix (L^{-1}). [42]

It is possible to analyse the performance of the error prediction model by graphing the distributions of the z-scores for the errors of each segmentation coefficient. This is shown in Figure 4.11, along with standard normal distributions.

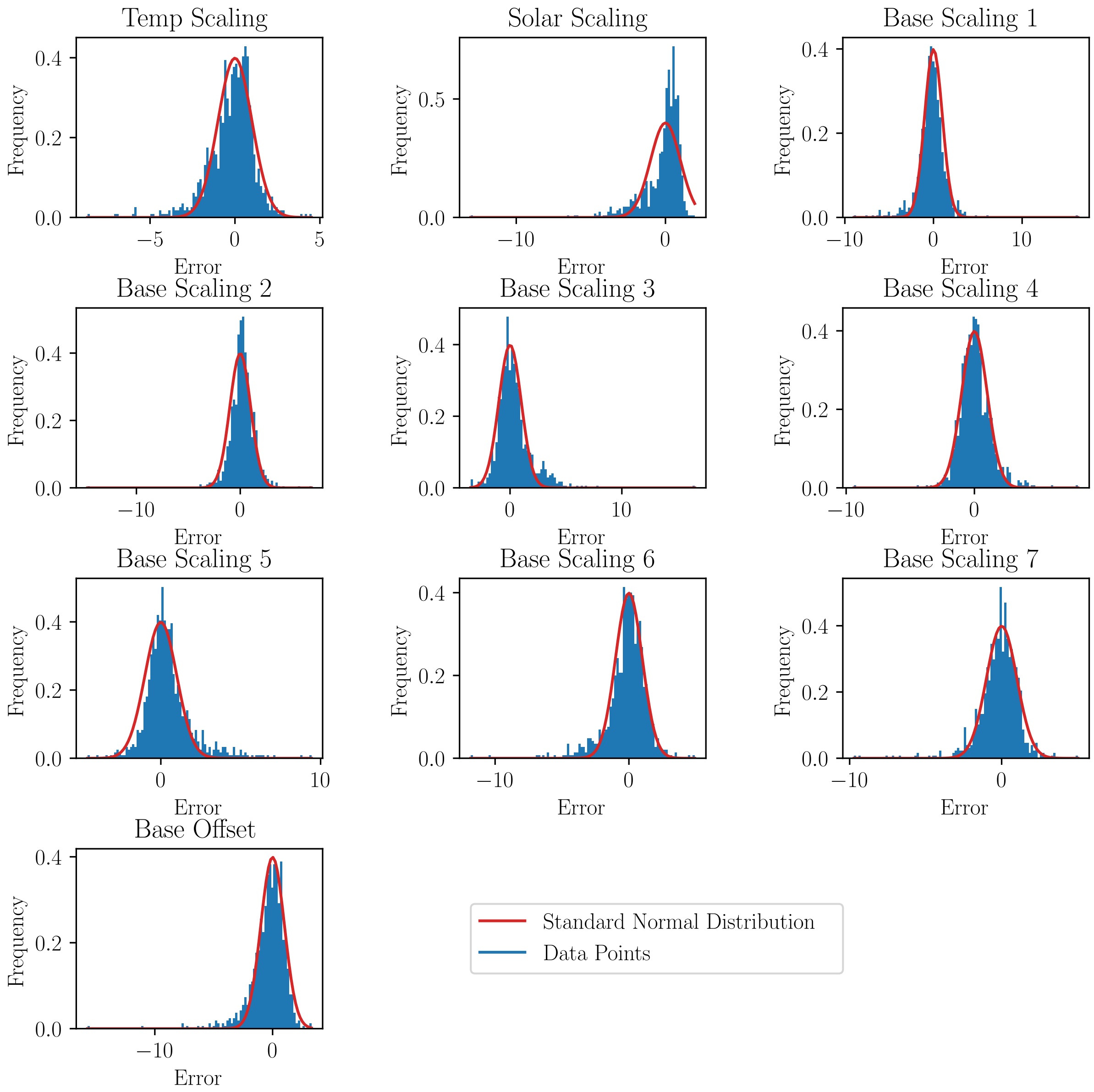


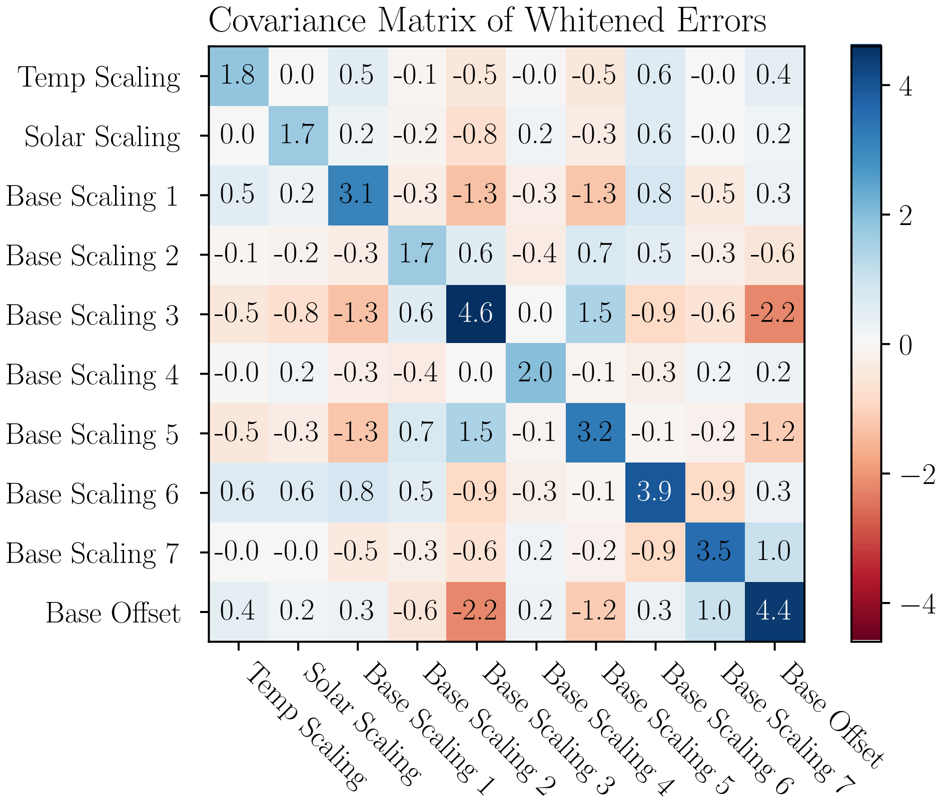
Figure 4.11: Histograms showing the errors from the model, measured in term of the expected standard deviation from the error estimation model. The histograms align with the standard normal distributions rather well, indicating that the error estimation model estimates the sizes of the errors relatively well.

In addition, other metrics such as the average Mahalanobis distance and average Mahalanobis distance squared can be calculated. In a standard multivariate normal distribution with n variables, the square of the Mahalanobis distance follows a χ_n^2 distribution, which has expected value $n = 10$, while the Mahalanobis distance follows a χ_n distribution. The χ_n distribution has the following expected value [43]:

$$\sqrt{2}\frac{\Gamma\left(\frac{n+1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)} = \{n = 10\} = \sqrt{2}\frac{\Gamma(5.5)}{\Gamma(5)} \approx 3.084. \tag{4.1}$$

The actual average Mahalanobis distance was 4.46, while the average Mahalanobis distance squared was 38.0. One likely reason for the high average squared

Mahalanobis distance is that there were some outliers with very high errors, as seen in Figure 4.11. The fact that the average and expected Mahalanobis distance are relatively close compared to the average and expected squared Mahalanobis distances further supports this conclusion. Furthermore, the model was trained on combined substation service areas, which meant that it had a slight bias towards substation service areas with higher load.


Covariance Matrix of Whitened Errors

	Temp Scaling	Solar Scaling	Base Scaling 1	Base Scaling 2	Base Scaling 3	Base Scaling 4	Base Scaling 5	Base Scaling 6	Base Scaling 7	Base Offset
Temp Scaling	1.8	0.0	0.5	-0.1	-0.5	-0.0	-0.5	0.6	-0.0	0.4
Solar Scaling	0.0	1.7	0.2	-0.2	-0.8	0.2	-0.3	0.6	-0.0	0.2
Base Scaling 1	0.5	0.2	3.1	-0.3	-1.3	-0.3	-1.3	0.8	-0.5	0.3
Base Scaling 2	-0.1	-0.2	-0.3	1.7	0.6	-0.4	0.7	0.5	-0.3	-0.6
Base Scaling 3	-0.5	-0.8	-1.3	0.6	4.6	0.0	1.5	-0.9	-0.6	-2.2
Base Scaling 4	-0.0	0.2	-0.3	-0.4	0.0	2.0	-0.1	-0.3	0.2	0.2
Base Scaling 5	-0.5	-0.3	-1.3	0.7	1.5	-0.1	3.2	-0.1	-0.2	-1.2
Base Scaling 6	0.6	0.6	0.8	0.5	-0.9	-0.3	-0.1	3.9	-0.9	0.3
Base Scaling 7	-0.0	-0.0	-0.5	-0.3	-0.6	0.2	-0.2	-0.9	3.5	1.0
Base Offset	0.4	0.2	0.3	-0.6	-2.2	0.2	-1.2	0.3	1.0	4.4

Figure 4.12: The covariance matrix of the whitened (decorrelated) error vectors. These error vectors are calculated by transforming the original error vectors with the inverse of the Cholesky decomposition matrix (L^{-1}). For a true standard normal distribution, which would be expected after performing a whitening transformation, the covariance matrix would be the identity matrix.

It is possible to visualize the predicted segmentation coefficients and their estimated errors by plotting the resulting load profile for an example substation service area. This is shown in Figure 4.13, which includes a typical substation service area with mostly single-family houses.

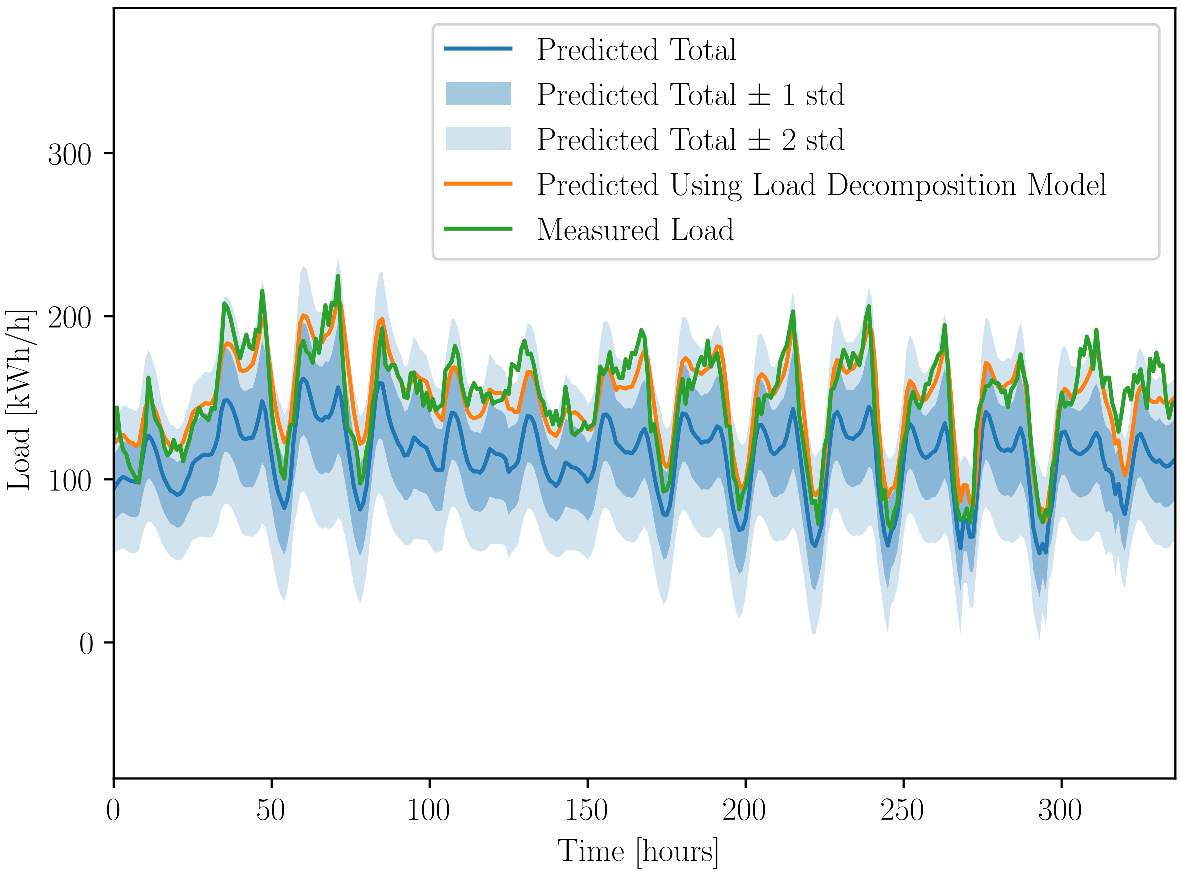


Figure 4.13: An example load profile for a substation service area with mostly single-family houses. The blue line (“Predicted Total”) shows the predicted mean load profile using the output weights from the building-informed segmentation model, while the shaded area shows the prediction uncertainty, which is calculated from the covariance matrix. The figure shows two arbitrarily chosen weeks in March.

4.2.1 Discussion

While the error estimation model can predict the size of the errors to some extent, there are cases where the actual error is much larger than the predicted error. There might be multiple reasons for this, one of these could be that these large errors correspond to substation service areas where there are large consumers of electricity that are not reflected in the building data, such as some industrial consumers, street lighting, or vehicle charging infrastructure.

Another assumption that the results show might not be completely valid is the assumption that the errors are normally distributed. This fact is especially apparent for the solar radiation coefficient, where the model often estimates the average amount of solar panels, which most of the time is slightly higher than the actual amount of solar panels installed (which is often very close to zero), while in a few cases, the estimation is a lot lower than the actual amount installed.

Furthermore, the normal distribution assumption is not always compatible with the positivity constraint for the segmentation coefficients. The purpose of the positivity constraint is to ensure that some of the segmentation coefficients are non-negative for physical reasons. However, by assuming a normal distribution, the model will predict a distribution with non-zero probability of negative values. This is especially apparent when the standard deviation is large in relation to the mean, which is common for substation service areas with lower load. One possible way to solve this could be to assume another multivariate distribution, such as a multivariate log-normal distribution. From this, a log-likelihood could be calculated from the expression for the joint probability distribution, which could be used as a loss function for the model. This would be closer to estimating the relative error instead of the absolute error.

One possible way to improve the model could be to change the way the loss is calculated. Instead of calculating the loss based on the difference between the predicted and estimated segmentation coefficients, the loss could be calculated based on the resulting load profiles for the corresponding segmentation coefficient vectors, and comparing these to the measured load profiles.

In cases where there is data available for specific segmentation coefficients, such as the amount of solar panels installed, it is possible to get a more accurate estimate of the other coefficients by using conditional probability. [44]

4.3 Residual Process Simulation Model

The residual process simulation model is used to simulate the difference between the estimated total load from the load decomposition model and the actual total load. The idea is that the load decomposition model covers the deterministic part of the load, while the residual process simulation model covers the stochastic part of the load in a realistic way.

The model works by estimating the mean and covariance matrix of the residuals for the next 24 hours based on the previous 24 hours as well as some other parameters which are described in Section 3.4.3. This is exemplified in Figure 4.14, which shows the simulated residual of the last day, and a number of residuals from the estimated distribution, along with the estimated mean of the distribution.

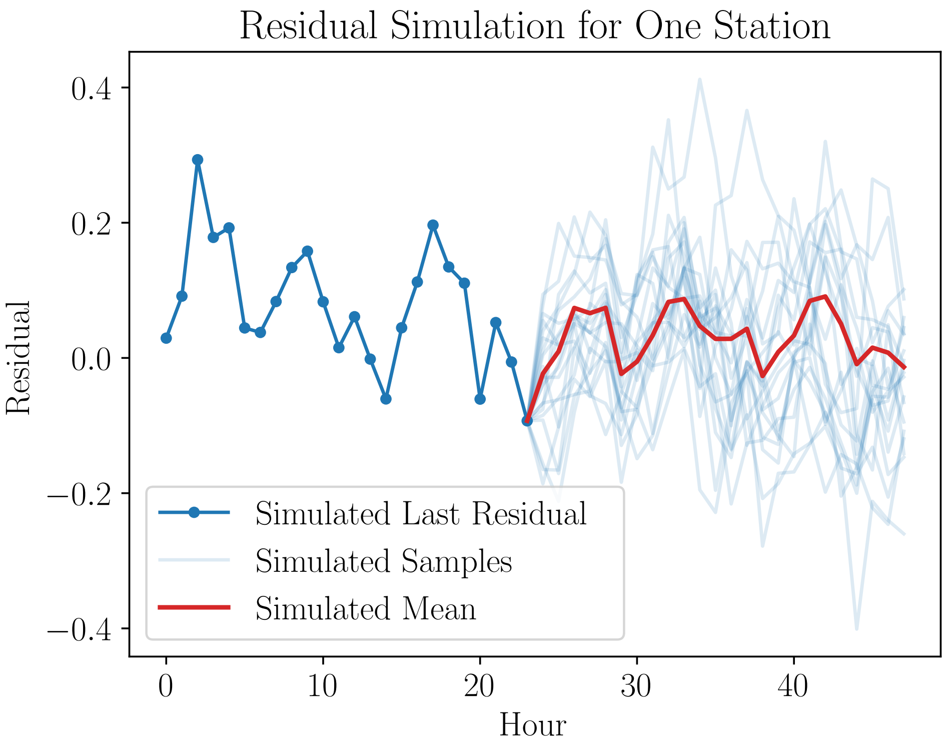


Figure 4.14: An example of possible residuals for a day (blue, right), as well as the mean of the distribution (red) of residuals along with the residuals for the previous day (blue, left). By comparing the mean and previous day, it becomes apparent that model has learned that residuals tend to repeat, but in lower magnitude than the day before (see dip at hour 14 and hour 38, 24 hours apart).

Since linear regression was used to predict the mean vector of the next day, this can be represented by a single matrix multiplication. A heat map and description of this matrix is shown in Figure 4.15. There are a number of things of interest in this matrix. First, there is a strong diagonal showing that the mean at one hour of the next day is very correlated with the mean for the same hour of the previous day. It can also be seen that around midday, the correlation is not as strong, and seems to be slightly more spread out over a few hours before and after. Furthermore, the horizontal gradient line at hour 23 shows that the first hours of the next day are very correlated with the last hours of the previous day, which is expected. However, the correlation falls off almost exponentially and is almost 0 after 24 hours.

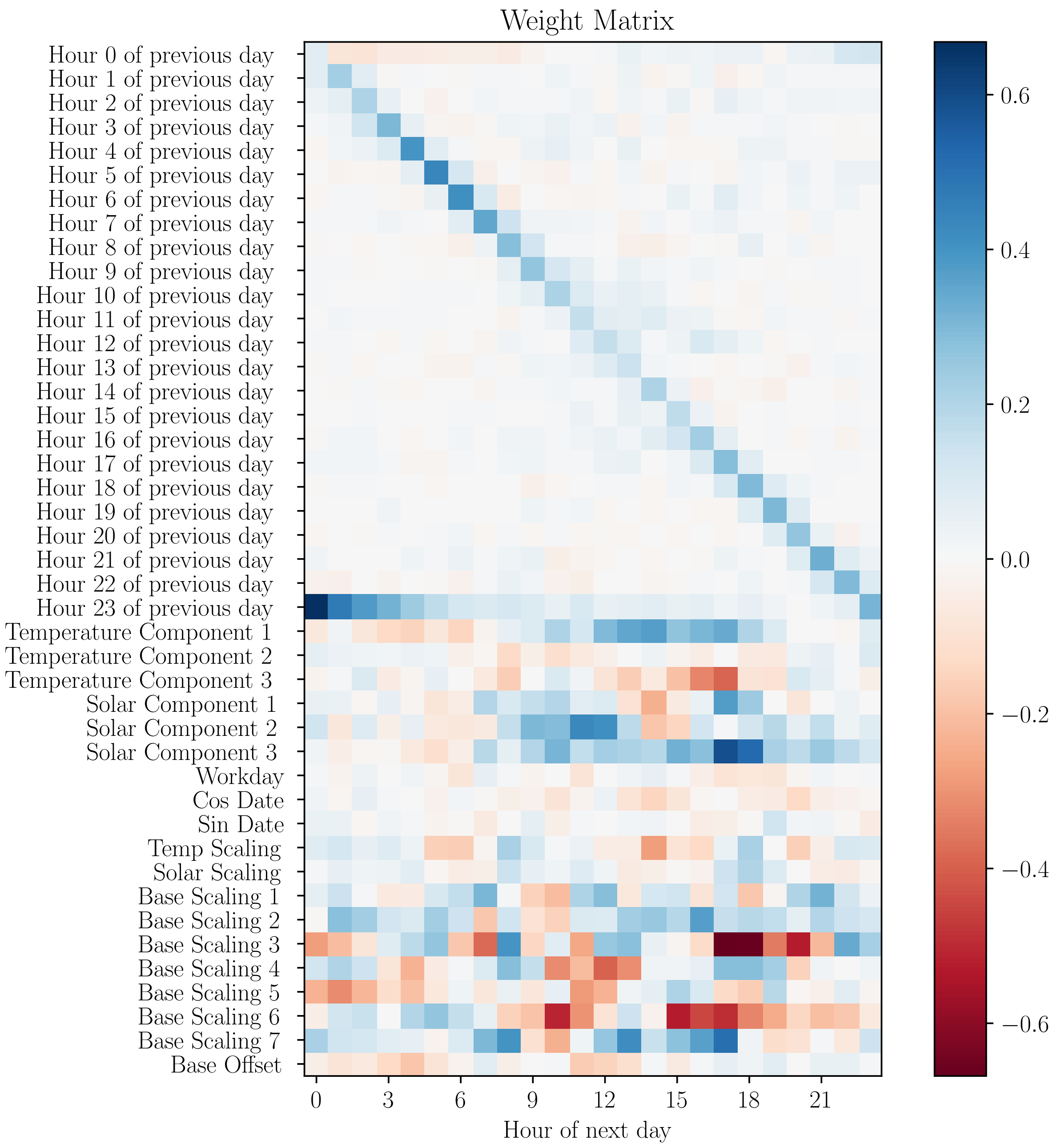


Figure 4.15: The weight matrix used when predicting the mean residual vector for the next day. The vertical labels show the input values, and the horizontal labels show the output values. The first 24 rows show the residual at given times in the previous day. The Temperature Component and Solar Component rows represent the autoencoder's encoding of the previous day's temperature and solar signals. The values for the upper part of the matrix including the previous day's residuals have the same unit (1) and can be comparable to each other, but the rows below that have arbitrary units and can not be comparable with other rows, only with other hours within the same row. The diagonal line shows there is a 24-hour periodicity in the residuals, and the horizontal line at hour 23 indicates continuity between the residuals of subsequent days.

The stochastic part of the process is represented by the covariance matrix, which states how much the different hours of the next day vary, and how much they correlate to each other. It can be represented as-is in its matrix form Σ, or it can be parametrized using its Cholesky decomposition L where $\Sigma = LL^T$. The inverses of these matrices are also of interest. The Cholesky decomposition matrix L is possibly one of the more intuitive matrices, as it essentially states how the stochastic parts of the residuals of the next day are sampled. If X is an n-dimensional standard multivariate normally distributed vector, meaning that each of its component are independently standard normally distributed, then $Y = LX$ is a vector sampled from the distribution represented by Σ, where $\Sigma = LL^T$. Figure 4.16 shows an L matrix that the model predicted for an example day (although most days look very similar and vary mostly in magnitude). The figure shows how the diagonal has the highest magnitude, with what looks like an exponential falloff to the left or downwards. This is essentially equivalent to a moving average (MA) model, where the stochastic part is the weighted sum of some previous noise signal values. The exponential falloff is a special case of the MA model and can simply be represented as an autoregressive model of order one (AR(1)), meaning that each value is defined as a constant (often slightly less than one) times its previous value, with some noise added. In the case of a random walk with no tendency to return to 0, the matrix would simply have its lower triangular part uniformly filled. By closer inspection, one can see that the lower triangular part of the matrix does not completely approach 0, meaning that some non-reverting random walk behaviour might still occur.

In this case, it is the tendency to return to the center that causes the falloff that is seen.

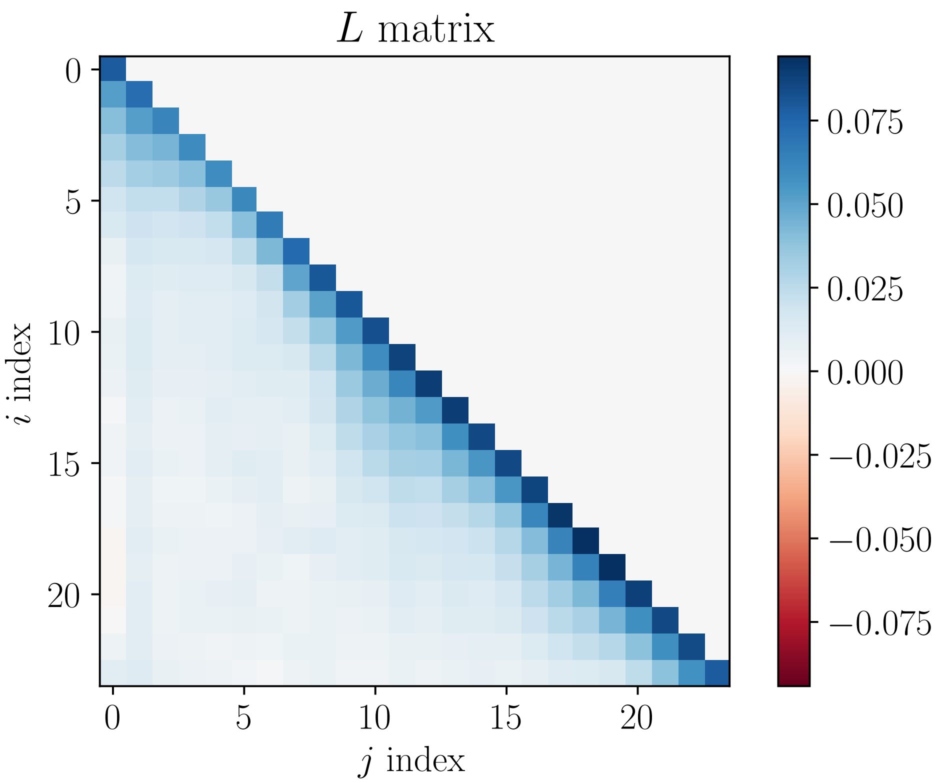


Figure 4.16: The predicted L matrix for an example substation service area and day. The matrix has been constructed by taking the inverse of the L^{-1} matrix that was the output from the model. It is lower triangular by definition; the model is only able to alter the values at or below the diagonal. The L matrix can be used to sample the stochastic parts of the residual. The exponential falloff of the values below the diagonal are similar to those of an AR(1) process.

The inverse of L, L^{-1} can also be shown (see Figure 4.17). This matrix is interesting because it is the actual matrix that the model predicts. The figure shows that this the predicted L^{-1} looks very sparse, with positive values along the diagonal, and negative values along the sub-diagonal. In the case of a theoretical MA-process with exponential falloff (equivalent to an AR(1) process), the diagonal and sub-diagonal would be uniform. In the special case of a random walk with no tendency to return to 0, the diagonal and sub-diagonal would be equal in absolute magnitude and only differ in sign. The fact that the model estimates a diagonal that is not uniform means that it has learned something that an ARMA-model would not be able to learn on its own.

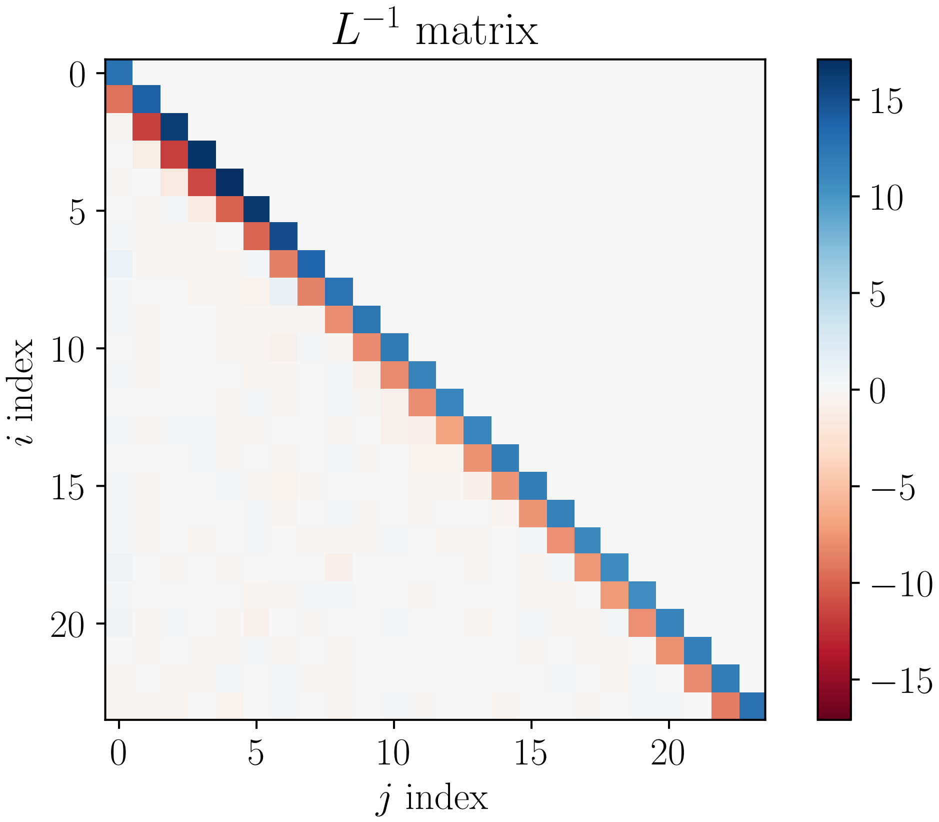


Figure 4.17: The predicted L^{-1} matrix for an example substation service area and day. The L^{-1} matrix is the direct output of the model. It can be seen that the model predicts a mostly sparse matrix, with only the diagonal and sub-diagonal populated.

The covariance matrix $\Sigma = LL^T$ and precision matrix Σ^{-1} are shown in Figure 4.18 and Figure 4.19 respectively. In the diagram with the covariance matrix, it can be seen that the residuals between the hours mostly correlate with other hours that are close in time. Furthermore, the total variance is generally higher toward the end of the day. This aligns with the fact that the falloff in the lower part of the L matrix did not completely approach 0, suggesting that it behaves like the sum of a random walk and mean-reverting random walk.

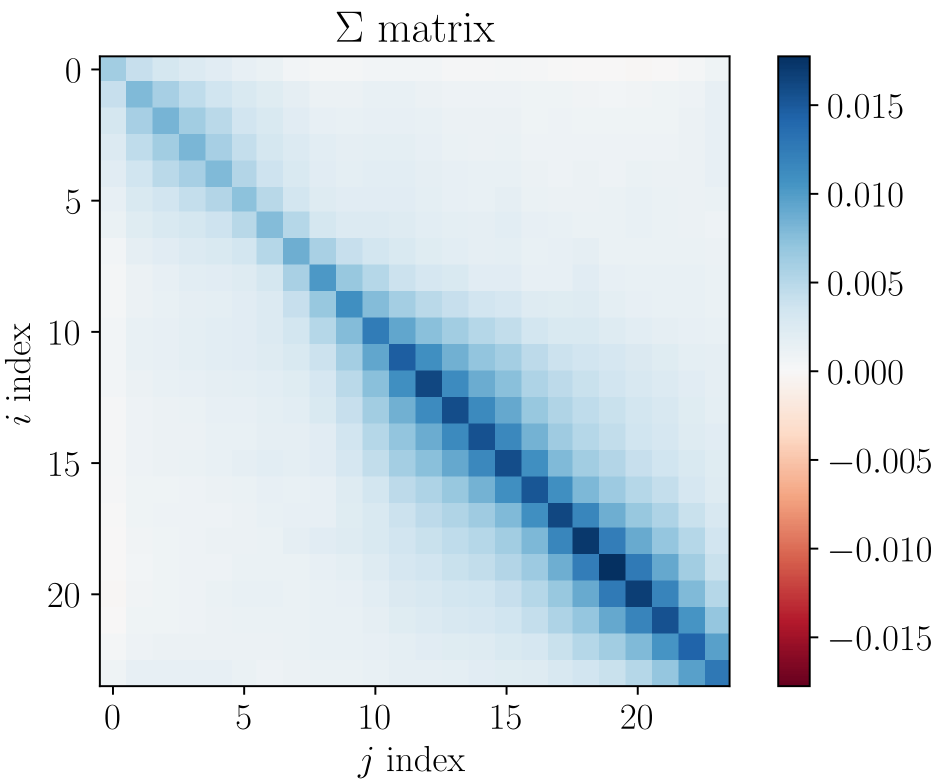


Figure 4.18: Heat map of the covariance matrix Σ for an example substation service area and day. It can be seen that the residuals correlate mostly with the values nearby in time, and that the covariance is slightly higher after more time has passed.

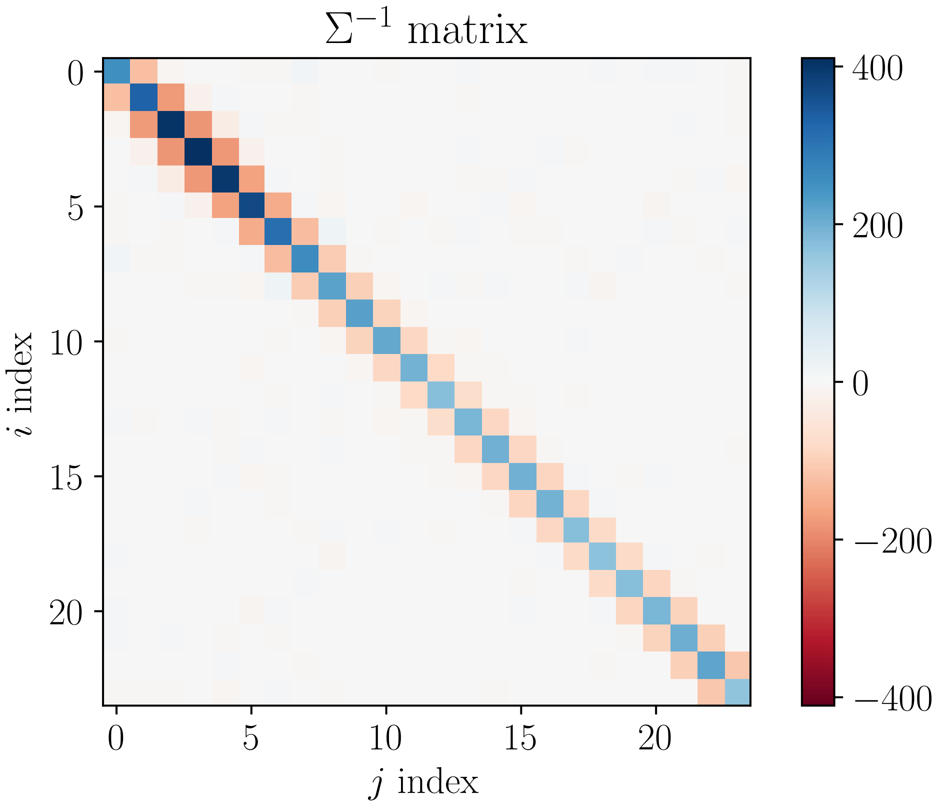


Figure 4.19: Heat map of the precision matrix (Σ^{-1} for an example substation service area and day. It can be seen that the model predicts that this is sparse, with the diagonal being positive and the sub- and super diagonals being negative.)

By sampling the residual vectors for the next day autoregressively, the model can sample possible residual time series. By adding these to the total output estimated by the load decomposition model, it is possible to generate load profiles that include this stochasticity. By sampling these load profiles multiple times in a Monte Carlo simulation, it is possible to get an estimate of a number of metrics, such as what the probability is to get a maximum load over a given limit during a certain period of time. If the values of this load profile are sorted, the result is a duration curve. An example duration curve for a station is shown in Figure 4.20. The top 1 % is shown in Figure 4.21.

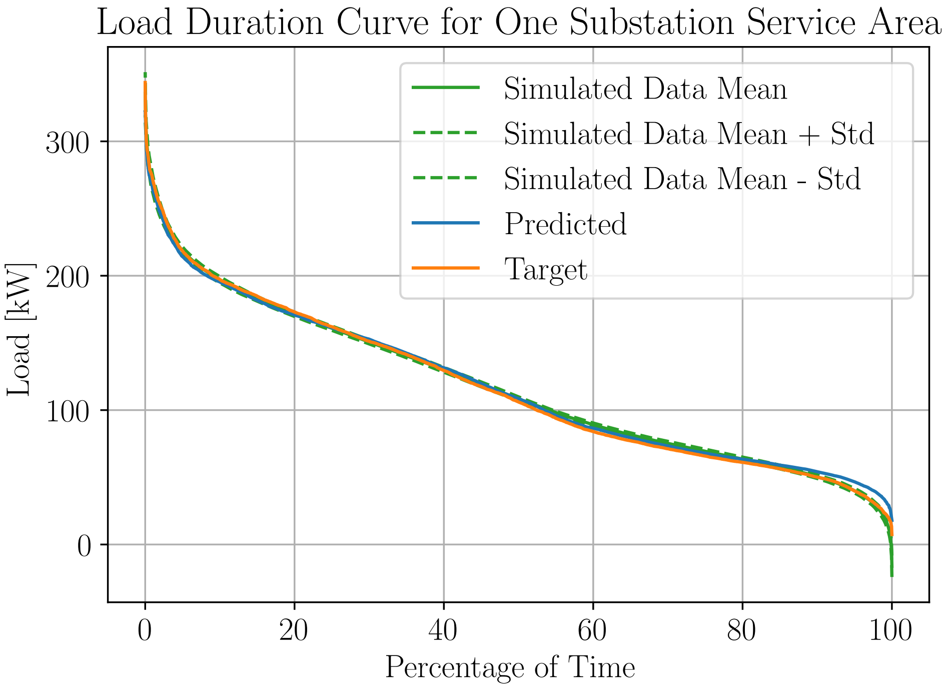


Figure 4.20: An example duration curve for a substation service area. The x-axis refers to the percentage of time that the load is above a given y-value. The substation service area is relatively typical, with mostly single-family residential buildings. The "Target" line represents the measured load for that year, "Predicted" represents the total output from the load decomposition model. "Simulated Data" is calculated by simulating 50 load profiles and creating duration curves from these. The mean and standard deviation are calculated at each percentile and shown as lines. It can be seen that the difference between the curves is rather small most of the time, except during the extreme values where the output of the load decomposition model is generally flatter. Figure 4.21 shows this diagram zoomed in on the top 1 % of the load.

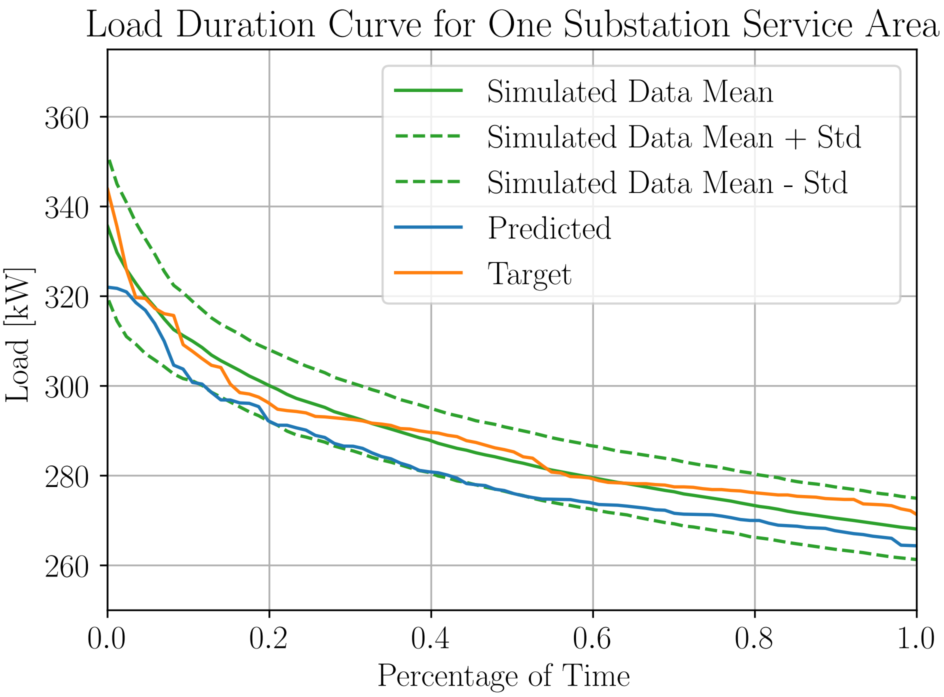


Figure 4.21: The same curve as Figure 4.20, but zoomed in on the top 1 %.

In order to evaluate how well the model works, the model was used to create 20 runs for each substation service area. The mean and standard deviation of the maximum values for each substation service area were then calculated, and compared to the maximum true measured value and maximum in the summed output from the load decomposition model. There are a few statistics that can be calculated from these runs. Firstly, the mean of the simulated peak standard deviations expressed as percentages of the simulated peak means was 5.5 %, and the root mean square of the same quantities was 6.4 %. This essentially means that from simulation to simulation, the peaks tended to vary within ±6%.

On average, the max values from the stochastic simulations were closer to the measured max values than the outputs from the load decomposition model were to the measured max values. We can define the z-scores for the outputs from the stochastic simulations and the outputs from the load decomposition model as follows (for a given substation service area):

$$z_{\mathrm{RPM}} = \frac{\mathrm{mean}_i\big(\max_t V_{\mathrm{RPM},i}(t)\big) - \max_t V_{\mathrm{true}}(t)}{\mathrm{std}_i\big(\max_t V_{\mathrm{RPM},i}(t)\big)}, \tag{4.2}$$

$$z_{\mathrm{LDM}} = \frac{\max_t V_{\mathrm{LDM}}(t) - \max_t V_{\mathrm{true}}(t)}{\mathrm{std}_i\big(\max_t V_{\mathrm{RPM},i}(t)\big)}, \tag{4.3}$$

where z_{RPM} refers to the z-score of the residual process model, z_{LDM} is the z-score for the load decomposition model, and the different $V(t)$ are the load profiles from the different models. The subscript i refers to the i:th simulation run. Essentially, this means that we compare the differences between the different models' max values to the true max values, and normalize them by dividing them with the standard deviations from the stochastic simulations. This can also be thought of as how close the left ends of the green and blue curves are to the orange curve in Figure 4.21.

For the load decomposition model, the average z-score was -2.53, and for the residual process model it was 0.09. This suggests that by adding the residual process model step, the yearly max values are more accurate. For the load decomposition model, the average absolute z-score was 2.63, while for the residual process model it was 1.46. The standard deviation of the z-score for the load decomposition model was 2.74, while it was 2.23 for the residual process model. For the load decomposition model, 27% of the z-scores were within the range $[-1, 1]$, whereas 45% of the average z-scores for the residual process model were within the same range.

These results show that by incorporating a stochastic part into the framework, as opposed to only relying on the output from the load decomposition model, the average maximum load values are much closer to the measured maximum load values. There is an inherent uncertainty in how the “actual” maximum load values for a station behave, since there is only ground-truth load data available for one year, which makes it difficult to deduce if the seen maximum load is representative for the typical maximum load. Due to this, some variability in the z-scores is expected, and could even be an indication that the model is not overfitting to the measured load profiles.

In practice, the residual process simulation model will have to be combined with another model that simulates the other input parameters, including the temperature and solar radiation. Since most of the measured load can already be explained by the deterministic relationship as presented in Section 3.2, the weather variations likely have a greater effect on the total load compared to the stochastic variation discussed in this subchapter.

Some feature engineering was also carried out, which included running the model with different combinations of input parameters, and comparing the test losses. These are presented in Table 4.1. Instead of using the losses from the Generalized Gaussian Negative Log Likelihood, these were converted into their likelihood and normalized by dividing them with the best score as the baseline (see Section 2.6). From this, it can be seen that the model performed best when using all input variables, which was expected. It can also be seen that the load segmentation coefficients are the most important feature, since these are the only ones that contain any information about each individual substation service area which is needed to know the scale of the variations.

Table 4.1: Results from a sweep over all possible combinations for features. Date and Temp are the least important features. Load segmentation coefficients and workdays are the most important features.

Date	Segmentation Coefficients	Workdays	Solar Radiation	Temp	Relative Likelihood
✓	✓	✓	✓	✓	100 %
×	✓	✓	✓	×	96.9 %
×	✓	✓	×	✓	95.7 %
×	✓	✓	✓	✓	93.5 %
✓	✓	✓	✓	×	93.1 %
×	✓	✓	×	×	92.2 %
×	✓	×	✓	×	89.7 %
✓	✓	✓	×	×	87.9 %
✓	×	✓	✓	×	83.6 %
×	✓	×	✓	✓	82.4 %
×	×	×	✓	×	81.3 %
×	✓	×	×	✓	81.2 %
✓	×	✓	×	×	81.1 %
✓	✓	×	✓	×	80.1 %
×	×	✓	✓	✓	78.2 %
✓	✓	×	×	✓	77.6 %
×	×	✓	×	×	77.3 %
✓	✓	×	✓	✓	75.4 %
✓	×	✓	✓	✓	74.4 %
×	×	×	✓	✓	74.4 %
✓	✓	✓	×	✓	72.8 %
✓	×	×	✓	✓	72.1 %
×	✓	×	×	×	71.4 %
✓	✓	×	×	×	67.9 %
×	×	×	×	✓	67.0 %
×	×	✓	×	✓	66.9 %
×	×	×	×	×	65.7 %
✓	×	×	✓	×	65.4 %
✓	×	✓	×	✓	65.4 %
✓	×	×	×	×	65.2 %
✓	×	×	×	✓	65.0 %
×	×	✓	✓	×	64.7 %

4.4 Sources of Error

One source of error in this study was the fact that only one year of load data was used. This made the error prediction part of the work very uncertain, since there was no possibility of comparing to other years. For example, training on one year and evaluating on another would be very valuable to verify the usability of this model. In practice, this was solved in a suboptimal way by diving the year into different training and test datasets by alternating weeks.

In addition, the building data was static, which made it impossible to infer construction of new buildings. These newly constructed buildings would still have contributions on the total load, which meant the model was forced to use other parameters to explain these changes. This, in turn, causes systematic errors in the model.

Another cause of lower precision in the model is the fact that the substation service areas for which load data is available are relatively large and few in number, and in the city these areas can be very heterogeneous. This causes potential problems for the building-informed segmentation model to accurately predict what building category is actually causing what type of electricity usage, since the contributions of all buildings are aggregated in the total load.

5

Conclusion

This chapter summarizes the key findings and contributions of each model, reflecting on their performance and limitations.

5.1 Load Decomposition Model

Using the load decomposition model, it was possible to recreate the original load curve with a high degree of accuracy using a linear combination of a few learned categorical load profiles. Evaluation of the accuracy of the actual segmentation proved to be challenging, since there was no ground truth data available for how large the different load categories were for each substation service area. The true solar PV generation could be estimated to some extent using manual inspection of satellite imagery of the substation service areas. In general, the model's estimation of the amount of solar PV generation seemed to correlate well with the manual satellite imagery estimation. This verification does however have a high margin of error and was not done systematically.

The thermal component of the estimated segmentation could not be verified with any ground truth data. However, due to the high correlation between the estimated total load profile and the measured, there is still reason to believe that the estimation is accurate. Since the model was trained on Swedish data where electric cooling is relatively uncommon, the performance of the model in situations with high cooling demand is still unknown. In order to be able to utilize the model in warmer climates, modifications would have to be done to allow for a cooling component in addition to the heating component.

In general, the relative difference between the estimated and true load profiles was lowest in large substation service areas with mostly residential buildings. This was theorized to be partly caused by residential buildings being more frequent in the training data, as well as residential buildings having more predictable load behaviour when aggregated. With larger substation service areas, the impact of an individual consumer's stochastic behavior becomes smaller in comparison.

5.2 Building-Informed Segmentation Model

By using aggregated building data, the aim of the building-informed segmentation model was to provide a means to infer the relative contributions of base load, temperature-dependent load, and solar generation. The empirical evaluation indicates that the variability in the compositions of the substation service areas leads to large errors when using linear regression, especially for smaller substation service areas.

However, the error estimation part of the model provides a way to estimate the sizes of the errors, allowing decision makers an insight into possible outcomes.

In order to improve the accuracy of the model and make it more reliable, other data sources could be used during training and inference, including information on solar PV generation installed or what heating sources are used in different buildings. This could include using satellite imagery and computer vision to obtain data on installed solar panels.

Another possible area for future work could be to analyze the load data for individual buildings using unsupervised learning in order to find typical load patterns relating to different sources of heating, as well as how frequent they are.

Whether the building-informed segmentation model is sufficient on its own when predicting electricity usage patterns depends on the type of application and the desired accuracy. When used to evaluate construction of new major substation service areas, the accuracy of the model can be improved by incorporating data on planned installation of solar PV generation and heating sources in buildings. In case it is used to estimate minor construction and other changes to existing substation service areas, the additional data might not be as necessary since the relative error will be smaller.

In conclusion, the Building-Informed Segmentation Model demonstrates that it can in some cases be feasible to generate segmentation coefficient estimates from static building data. Its integration into the modelling framework can enable long-term infrastructure planning in environments without historical load data, thereby contributing to the advancement of data-driven, interpretable tools for future grid design.

5.3 Residual Process Simulation Model

The Residual Process Simulation Model serves as a component in capturing the stochastic variability in electricity demand that is not explained by deterministic factors such as temperature or solar radiation. By modelling the residuals using stochastic processes and incorporating Monte Carlo simulations, the model enables the generation of realistic demand scenarios that reflect uncertainty and variability inherent in real-world usage patterns.

This capability is essential for robust grid planning and risk-aware decision-making, especially in the context of increasing integration of intermittent energy sources. The model's flexibility allows it to be calibrated to local conditions and extended with additional data sources, making it a valuable tool for enhancing the completeness and realism of the overall predictive framework.

Bibliography

[1] European Commission, “Energy 2020: A Strategy for Competitive, Sustainable and Secure Energy,” Nov. 2010. Accessed: May 21, 2025. [Online]. Available: https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=COM:2010:0639:FIN:En:PDF

[2] L. Larsson, S. Lindskoug, and M. Linden, “The characteristics of electric power demand in cold weather; Elfoerbrukningens karaktaer vid kall vaederlek,” Oct. 2006. [Online]. Available: https://www.osti.gov/etdeweb/servlets/purl/20819825

[3] S. Velander, “Fördelningen av ett elverks fasta kostnader på olika abonnenter eller abonnentgrupper,” *Teknisk Tidskrift*, vol. 65, no. 7, pp. 103–107, Jul. 1935.

[4] K. Fürst, P. Chen, I. Y.-H. Gu, and L. Tong, “Improved peak load estimation from single and multiple consumer categories,” *CIRED*, vol. 2020, no. 1, pp. 178–181, 2021, doi: 10.1049/oap-cired.2021.0300.

[5] E. Persson and P. Jonsson, “Utvärdering av Velanders formel för toppeffektberäkning i eldistrubutionsnät,” 2018.

[6] Svenska Elverksföreningen, *Belastningsberäkning med typkurvor*. 1991, p. 217.

[7] E. Tynngård, “Typkurvor för elbilsladdning i hushåll: Framtagning av typkurvor och simuleringar i delar av ett lågspänningsnät,” 2022.

[8] M. Koehl, M. Heck, S. Wiesmeier, and J. Wirth, “Modeling of the nominal operating cell temperature based on outdoor weathering,” *Solar Energy Materials and Solar Cells*, vol. 95, no. 7, pp. 1638–1646, 2011, doi: https://doi.org/10.1016/j.solmat.2011.01.020.

[9] E. R. Etu, “A Study of Residential Solar Energy in Three Swedish Cities: Stockholm, Malmö and Umeå,” 2025.

[10] pvxchange, “Canadian Solar CS5P-220M Solar Module.” Accessed: May 16, 2025. [Online]. Available: https://www.pvxchange.com/Solar-Modules/Canadian-Solar/CS5P-220M_1-2103721

[11] K. S. Anderson, C. W. Hansen, W. F. Holmgren, A. R. Jensen, M. A. Mikofski, and A. Driesse, “pvlib python: 2023 project update,” *Journal of Open Source Software*, vol. 8, no. 92, p. 5994, 2023, doi: 10.21105/joss.05994.

[12] Profu AB, "Analys av den ekonomiska potentialen för effektiv värme och kyla - Inför Art 14-rapporteringen," Mölndal, Oct. 2020.

[13] N. Fridlund, "Hur många hushåll har luftluftvärmepump?." Accessed: Apr. 09, 2025. [Online]. Available: https://varvilla.se/energivarme/varmepumpar/hur-manga-hushall-har-luftluftvarmepump/

[14] Statistiska centralbyrån (SCB), "Fjärrkyla. År 2018 - 2023." Accessed: Apr. 09, 2025. [Online]. Available: https://www.statistikdatabasen.scb.se/pxweb/sv/ssd/START___EN___EN0105___EN0105A/FjarrKyla/

[15] Wikipedia contributors, "Thermal transmittance — Wikipedia, The Free Encyclopedia." Accessed: May 23, 2025. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Thermal_transmittance&oldid=1221991211

[16] Energimyndigheten, "Så snabbt blir ditt hus kallt." Accessed: May 22, 2025. [Online]. Available: https://www.energimyndigheten.se/energiberedskap/hushall/elavbrott/5-enkla-rad-for-att-halla-varmen/sa-snabbt-blir-ditt-hus-kallt/

[17] A. n. Ferreras Pascual, "Simulation of FtX ventilation technique in a typical Swedish house." p. 81, 2010.

[18] A. S. Hassanzadeh and A. E. Alkhouli, "Energieffektivisering av småhus : En fallstudie på hur renovering påverkar energiförbrukning, ekonomi och klimat." p. 45, 2023.

[19] J.-P. Zimmermann *et al.*, "Household Electricity Survey: A Study of Domestic Electrical Product Usage," May 2012. Accessed: Apr. 09, 2025. [Online]. Available: https://assets.publishing.service.gov.uk/media/5a7c2fd940f0b67d0b11f6df/10043_R66141HouseholdElectricitySurveyFinalReportissue4.pdf

[20] A. F. Agarap, "Deep Learning using Rectified Linear Units (ReLU)." [Online]. Available: https://arxiv.org/abs/1803.08375

[21] S. R. Dubey, S. K. Singh, and B. B. Chaudhuri, "Activation Functions in Deep Learning: A Comprehensive Survey and Benchmark." [Online]. Available: https://arxiv.org/abs/2109.14545

[22] A. Paszke *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library." [Online]. Available: https://arxiv.org/abs/1912.01703

[23] Martín~Abadi *et al.*, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems." [Online]. Available: https://www.tensorflow.org/

[24] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization." [Online]. Available: https://arxiv.org/abs/1412.6980

[25] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization." [Online]. Available: https://arxiv.org/abs/1711.05101

[26] M. D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method." [Online]. Available: https://arxiv.org/abs/1212.5701

[27] G. Hinton, “rmsprop: Divide the gradient by a running average of its recent magnitude.” Accessed: May 22, 2025. [Online]. Available: https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf

[28] J. Duchi, E. Hazan, and Y. Singer, “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization,” *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[29] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning internal representations by error propagation,” in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*, Cambridge, MA, USA: MIT Press, 1986, pp. 318–362.

[30] K. P. Murphy, *Machine Learning: A Probabilistic Perspective.* Cambridge, MA: MIT Press, 2012.

[31] G. Strang, *Linear algebra and its applications.* 2000.

[32] T. A. Manteuffel, “An incomplete factorization technique for positive definite linear systems,” *Mathematics of computation*, vol. 34, no. 150, pp. 473–497, 1980.

[33] J. V. den Bossche *et al.*, “geopandas/geopandas: v1.0.1.” Accessed: May 27, 2025. [Online]. Available: https://doi.org/10.5281/zenodo.12625316

[34] The pandas development team, “pandas-dev/pandas: Pandas.” Accessed: May 27, 2025. [Online]. Available: https://doi.org/10.5281/zenodo.13819579

[35] A. Yakovets *et al.*, “vacanza/holidays: v0.71.” [Online]. Available: https://doi.org/10.5281/zenodo.15257984

[36] K. Tegmark Wisell and B. Bråstad, “Gemensamma författningssamlingen avseende hälso-och sjukvård, socialtjänst, läkemedel, folkhälsa mm,” *Folkhälsomyndigheten*, vol. 12, 2017.

[37] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

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

[39] A. E. Hoerl and R. W. Kennard, “Ridge regression: Biased estimation for nonorthogonal problems,” *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[40] A. N. Tikhonov and others, “On the stability of inverse problems,” in *Dokl. akad. nauk sssr*, 1943, pp. 195–198.

[41] P. C. Mahalanobis, “On the generalised distance in statistics,” *Proceedings of the National Institute of Sciences of India*, vol. 2, pp. 49–55, 1936.

[42] A. Kessy, A. Lewin, and K. S. and, “Optimal Whitening and Decorrelation,” *The American Statistician*, vol. 72, no. 4, pp. 309–314, 2018, doi: 10.1080/00031305.2016.1277159.

[43] T. W. Anderson, *An Introduction to Multivariate Statistical Analysis*, 3rd ed. Wiley, 2003.

[44] M. H. A. Davis, “Multivariate Distributions.” Accessed: May 27, 2025. [Online]. Available: https://www.columbia.edu/~mh2078/QRM/MultivariateDistributions.pdf

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY