UiT THE ARCTIC UNIVERSITY OF NORWAY

Faculty of Science and Technology, Department of Mathematics and Statistics

# Probabilistic Wind Power and Electricity Load Forecasting with Echo State Networks

—

**Petter Støtvig**

*STA-3900 Master's Thesis in Statistics 60 SP - May 2022*

# Abstract

With the introduction of distributed generation and the establishment of smart grids, several new challenges in energy analytics arose. These challenges can be solved with a specific type of recurrent neural networks called echo state networks, which can handle the combination of both weather and power consumption or production depending on the dataset to make predictions. Echo state networks are particularly suitable for time series forecasting tasks. Having accurate energy forecasts is paramount to assure grid operation and power provision remains reliable during peak hours when the consumption is high.

The majority of load forecasting algorithms do not produce prediction intervals with coverage guarantees but rather produce simple point estimates. Information about uncertainty and prediction intervals is rarely useless. It helps grid operators change strategies for configuring the grid from conservative to risk-based ones and assess the reliability of operations.

A popular way of producing prediction intervals in regression tasks is by applying Bayesian regression as the regression algorithm. As Bayesian regression is done by sampling, it naturally lends itself to generating intervals. However, Bayesian regression is not guaranteed to satisfy the designed coverage level for finite samples.

This thesis aims to modify the traditional echo state network model to produce marginally valid and calibrated prediction intervals. This is done by replacing the standard linear regression method with Bayesian linear regression while simultaneously reducing the dimensions to speed up the computation times. Afterward, a novel calibration technique for time series forecasting is applied in order to obtain said valid prediction intervals.

The experiments are conducted using three different time series, two of them being a time series of electricity load. One is univariate, and the other is bivariate. The third time series is a wind power production time series. The proposed method showed promising results for all three datasets while significantly reducing computation times in the sampling step.

# Acknowledgments

First of all, I would like to express my gratitude to my supervisor Filippo Maria Bianchi for the last ten months' guidance. It has been invaluable for the continuous progress of this thesis. I am also grateful for the dataset provided by Troms Kraft.

I would also like to thank my friends and family for their unconditional support through this journey.

Petter Støtvig,
Tromsø, May 2022.

# Table of Contents

## Part I / Introduction 1

## Part II / Technical Background 6

## Part III / Proposed Method 25

# List of Tables

# List of Figures

# Abbreviations

| | |
|---|---|
| ACF | Autocorrelation Function |
| AR | Autoregressive |
| ARIMAX | Autoregressive Integrated Moving Average Exogenous |
| CDF | Cumulative Distribution Function |
| ESN | Echo State Network |
| HPD | Highest Posterior Density |
| JAGS | Just Another Gibbs Sampler |
| MA | Moving Average |
| MCMC | Markov Chain Monte Carlo |
| MSE | Mean Squared Error |
| MSPE | Mean Squared Prediction Error |
| PACF | Partial Autocorrelation Function |
| PCA | Principal Component Analysis |
| PI | Prediction Interval |
| PICP | Prediction Interval Coverage Probability |
| PINAW | Prediction Interval Normalized Average Width |
| RNN | Recurrent Neural Network |
| SSVS | Stochastic Search Variable Selection |
| TS | Time Series |

# Part I / Introduction

## 1 Motivation

Electricity load forecasting with the use of historical observations has been of great interest ever since the start of the electric power industry (Hong & Fan, 2016), as electric power can only be converted to other forms of energy that can be stored and then later converted back to electric power (Jensen, 2021). In electric power systems planning and operations, having accurate forecasts for the electricity load is indispensable. Having accurate forecasts can increase the reliability of the power supply system as well as decrease the operating and maintenance costs (Almeshaiei & Soltan, 2011). A reasonable balance between electricity production and consumption must be achieved to guarantee that the operational limits of the electricity grid are not surpassed, in addition to minimizing the cost of over-and underproduction. However, a perfect equilibrium is not achievable in the real world (Infield & Freris, 2009). Electricity over-or underproduction can cause financial loss as electricity is sold via bidding in the electricity market, where the sellers and buyers are required to produce and buy the agreed-upon amount (Dalal, Mølnå, Herrem, Røen, & Gundersen, 2020). The information above makes it abundantly clear that forecasts are necessary, and the ability to quantify the uncertainty in the predictions is of paramount interest. Forecast models that can convey these uncertainties are referred to as probabilistic forecast models (Jensen, 2021).

In contrast to point forecasts which predict a single point, probabilistic load forecasts provide intervals, quantiles, or density functions for the predictions (Hong & Fan, 2016), which reflect the uncertainty of the predictions. The construction of prediction intervals is usually done with the available past observations in combination with explanatory variables to provide a range of values for subsequent observations for a given confidence level (Jensen, 2021). If a prediction interval achieves coverage for future observations equal to the designed confidence level, the prediction interval is termed valid (C. Xu & Xie, 2020). Valid prediction intervals are of utmost importance in high-risk situations, which enables risk-based specification and operation of the network (Jensen, 2021). A prediction interval with wider intervals indicates increased uncertainty as the width serves as a reliability measure of the forecast (Quan, Srinivasan, & Khosravi, 2013). Should an interval be overly wide or unreliable, it becomes ineffective. Therefore, the construction of narrow and valid or close to valid prediction intervals is key.

The task of forecasting electricity load is complex as the electricity load data display substantial temporal variations and regularly display non-linear dependencies (Yang, Wu, Chen, & Li, 2013). An electricity load forecaster must identify and learn the temporal dependencies exhibited by the data. In addition to this, assimilate the extent to which the load is affected by external factors should they be included in the model (Jensen, 2021). Classically, electricity load forecasters have been made using traditional time series models such as the autoregressive (AR), and the autoregressive integrated moving average (ARIMA) models along with their multivariate extensions (Dang-Ha, Bianchi, & Olsson, 2017). The advantage of such statistical methods is the ease of construction prediction interval, as they work under the assumption that the samples and errors of the time series are distributed according to the normal distribution (Jensen, 2021). Where these models falter is their inability to model non-linear dependencies in the data and the normal distribution assumption as the distribution of the underlying data generation process is usually unknown (Box, Jenkins, Reinsel, & Ljung, 2015).

Since these classical parametric time series models have their limitations, machine learning methods have been developed to solve the electricity load forecasting problem; neural networks, in particular, have been proposed (Jensen, 2021). The advantage of neural networks is the automatic non-linear relationship learning capabilities without requiring significant prior knowledge about the distribution of the data in addition to needing fewer data preprocessing steps in comparison to the statistical models (Gasthaus et al., 2019). However, neural networks, in general, do not produce probabilistic forecasts as their predictions come in the form of point estimates (Keren, Cummins, & Schuller, 2018). The qualities of neural networks are the core motivation for this thesis; to make a neural network model that produces valid prediction intervals suitable for use with electricity load and production datasets.

Echo state networks (ESN) are a type of recurrent neural networks (RNN), which is a class of neural networks. Recurrent neural networks are categorized by the internal recurrent connections, which are used for processing sequential data (Goodfellow, Bengio, & Courville, 2016). This recurrent connection allows the information to flow in loops within the network (Jensen, 2021). Where echo state networks differ from recurrent neural networks is how the network is trained. Whereas for RNNs, all the weights must be trained, only the output weights need to be trained for ESN models. The result of this is a neural-based model which can be efficiently trained through the use of linear regression algorithms such as Ridge regression. However, just like many other types of neural networks, echo state networks do not inherently produce probabilistic forecasts and produce point forecasts instead (Bianchi, Scardapane, Løkse, & Jenssen, 2020).

*Bayesian Echo State Network* (McDermott & Wikle, 2019) is a probabilistic forecasting technique that trains an echo state network model using Bayesian regression as opposed to Ridge regression to provide uncertainty estimates. The algorithm laid out in the paper by (McDermott & Wikle, 2019) uses a stochastic search variable selection prior, which combats over-parametrization. This, however, proves to be very computationally expensive as the dimensionality increases, thus deterring more widespread use. This increase in computation costs compared to Ridge regression is due to Bayesian inference using sampling as it is probabilistic rather than Ridge regression, which is deterministic.

With the advent of using computers to write algorithms for complex models and performing inference for large datasets, Bayesian inference saw a rise in popularity in the 1990s (Mushore, 2018). A way of doing this sampling is with the help of Markov chain Monte Carlo, and Gibbs sampling in particular (Gilks, Richardson, & Spiegelhalter, 1995). In 1999 launched one of the first freely available softwares for Bayesian inference; this software was called BUGS (Lunn, Thomas, Best, & Spiegelhalter, 2000).

However, Bayesian inference usually does not guarantee adequate coverage as the upper and lower bounds of a Bayesian prediction interval are obtained via quantile functions from the sampled predictions. A prediction interval that provides its designed coverage or is calibrated is highly desirable. A model that produces calibrated prediction intervals makes it possible to accurately quantify the uncertainty in the predictions, thus giving decision makers greater insight into how reliable the results are and which course of action to take.

# 2 Electricity Load and Production Forecasting

The electricity load forecast is of great importance to grid operators for planning and operation in the power industry as it predicts the expected electricity demand (Jensen, 2021). With the deployment of smart grid technology and the integration of electric power sources such as solar and wind power, the electric power industry has seen rapid changes (Hong & Fan, 2016). However, as these renewable energy sources intermittently produce power, several new difficulties have arisen for the operational reliability of the electric power grid on both sides of production and consumption (Taylor & McSharry, 2007). These difficulties stem from the varying electricity consumption combined with the highly irregular electricity production from renewable resources (Jensen, 2021). It is for this reason that accurate predictions are paramount.

What makes the task of electricity forecasting challenging are the factors that influence electricity consumption. These include but are not limited to climate conditions, temperature, and customer activity. This results in a complex and dynamic system (Jensen, 2021). There are also social and environmental factors that increase the unpredictability of electricity load data (Almeshaiei & Soltan, 2011), and how much the consumption is affected is highly variable (Jensen, 2021). For example, the temperature impacts electricity consumption to a more considerable degree in colder climates than in warmer climates due to heating demands. Electricity load also exhibits strong cyclical time dependencies (Yang et al., 2013), as the electricity load depends on the previous hours in addition to the same hours in the preceding weeks, which also have further dependencies (Jensen, 2021).

The electricity load forecasting time horizon for the forecasts is usually split into two different categories, short-term load forecasting and long-term load forecasting, where long-term forecasting is any time horizon longer than two weeks (Hong & Fan, 2016). Furthermore, electricity load forecasts are often aggregated together, consisting of individual consumers ranging from households to the industrial level (Jensen, 2021). The reason for this is forecasting of an individual household electricity load can be considerably more difficult than the aggregated electricity load due to large fluctuations in electricity consumption that a single household presents (Gasparin, Lukovic, & Alippi, 2019). With aggregation, the variation within the signal typically sees a decrease and exhibits less erratic behavior. Considering all these facts, it becomes evident that a single model is not a sufficient generalization of all electricity load use cases. Thus, models specific to the forecasting task at hand should be constructed based on the characteristics of the consumption data.

The existing electricity load forecasting methods mainly consist of models producing point estimates. This holds for both machine learning methods as well as statistical methods.

These point estimates are frequently the expected value of the conditional mean of future load (Chen, Kang, Chen, & Wang, 2020). However, point estimates are not that desirable as the uncertainty in the predictions remains unknown. On the other hand, probabilistic forecasts inherently provide these uncertainty estimates and are, therefore, more desirable in electricity load and production forecasting tasks. Inaccurate forecasts have the ability to cause companies operating in competitive power markets financial implications (Taylor & McSharry, 2007); therefore, the risk associated is of great interest.

Statistical-based methods and machine learning-based methods are two of the most commonly used techniques within the field of electricity load forecasting (Hong & Fan, 2016). In the field of time series forecasting, statistical models such as the autoregressive models have been at the forefront (Dang-Ha et al., 2017). However, in recent times neural networks have increased in popularity. Specifically, recurrent neural networks are one of the most frequently used neural networks for time series forecasting (Chen et al., 2020). This is due to the ability to effectively process and extract nonlinear information from a large historical input of data and the recurrent connection within the neural network that enables the network to retain information from the previous input. This lends itself to any field containing temporal dependencies (Choi, Bianchi, Kampffmeyer, & Jenssen, 2020).

As electricity load is heavily time-dependent, it naturally lends itself to recurrent neural networks. Echo state networks, a class of recurrent neural networks, are well suited for these tasks as training these models can be done quickly, whereas a more standard recurrent neural network might take a lot longer to train (Variengien & Hinaut, 2020). However, as stated earlier neural networks typically do not provide probabilistic forecasts. A solution to this is replacing the underlying training algorithm with one that provides probabilistic forecasts for the readout layer in the echo state network model. One such way is with Bayesian regression as opposed to Ridge regression. As Bayesian regression typically uses sampling, it naturally lends itself to providing probabilistic forecasts where each sample can be used to make a prediction, which can, in turn, be used to make prediction intervals, thus making probabilistic forecasts.

Bayesian regression, however, does not guarantee adequate coverage of the prediction intervals as Bayesian uncertainty estimates are often inaccurate (Kuleshov, Fenner, & Ermon, 2018). This can be due to a multitude of factors, such as model misspecification and the use of approximate inference. Having uncertainty estimates that grossly misrepresent the actual uncertainty can be harmful as it can portray a forecast with a high degree of certainty while, in reality, being very uncertain, leading to the wrong decision being taken. In electricity production, this can manifest itself as operators believing the production to be higher than it will be, leading to more electricity being sold than is produced.

# 3 Research Questions, Proposed Approach, and Contributions

Based on the motivational factors laid out in Section 1, this thesis will try to answer the following research questions:

- *Is it possible to reduce the computation time with a Bayesian echo state network to a degree where it is usable even with a large number of connections without reducing the performance?*
- *Can the prediction intervals offered by the Bayesian echo state networks be recalibrated to obtain approximately valid prediction intervals?*

This thesis proposes a probabilistic electricity load and production forecasting method based on echo state networks. The proposed method specifically reduces the overall dimensionality of the model to a more manageable size while using Bayesian regression to train the model rather than traditional regression methods to construct prediction intervals in the form of quantiles for the upper and lower bound. The prediction intervals are then recalibrated to produce better intervals that attain the designed coverage level. The experiments are conducted using univariate and multivariate electricity load data as well as multivariate electricity production data. The experimental results show promising results for the proposed method with the given data.

The key contributions of this thesis can be summarized as follows:

- Proposing a dimensionality reduction technique to be used in tandem with the Bayesian ESN model in an effort to reduce the computation time to train a Bayesian regression model as the ESN readout.
- As the Bayesian prediction intervals might not be approximately valid, a calibration technique is used to recalibrate the model to produce approximately valid prediction intervals.
- Applying the proposed method to both univariate and multivariate electricity load datasets in addition to an electrical power production dataset. Where the proposed method is pitted against the common statistical-based methods as well as the traditional ESN model.

# 4 Thesis Outline

This thesis is split into five distinct main parts. Part II presents the relevant background theory used in this thesis. These are related to the proposed method or used in the comparisons made against the proposed method. Each section in part II tackles the technical knowledge needed in part III, with the exception of section 5 and section 6. These two sections present the time series forecasting background as well as the statistical-based methods.

Part III revolves around the proposed method, presenting it and closely related works.

In the penultimate part, part IV, the experiments are conducted to investigate whether the proposed method has any validity in addition to examining the datasets.

Finally, part V is the last part focused on concluding remarks and where additional research is suggested for future work.

# Part II / Technical Background

## 5 Time Series Forecasting

A time series is simply a sequence of data points ordered after the time it was obtained (Shumway & Stoffer, 2017). By ordering them this way, more information can be acquired (Jensen, 2021). This time element makes it possible to study the trend and seasonality of whatever sequence in question, called time series analysis. Time series forecasting tries to predict future values of this time series. In some cases, additional time series are used as explanatory variables; these are called exogenous variables and are separate time series from the original. This is useful as variations in one time series may depend on the variation within another and thus may improve the accuracy of the predictions.

There are two general approaches to point time series forecasting, single-step and multi-step (Taieb, Bontempi, Atiya, & Sorjamaa, 2012). Single-step forecasting is predicting the subsequent time step for a time series. This amounts to this equation:

$$y_{(t+1)} = f(y_{1:t}, X^{(i)}_{1:t}, X^{(i)}_{(t+1)}), \quad i = 1, .., N \tag{1}$$

here y is the time series we want to predict, $X^{(i)}$ is the exogenous variables, t is the number of observations.

On the other hand, multi-step forecasting predicts several time steps instead of one. This can be done in a multitude of ways, and the two most common are direct and recursive methods (Jensen, 2021). The recursive one is an iterative method. It predicts the single-step prediction and then uses this as input in the model to make subsequent predictions. By using this approach, the uncertainty naturally increases as more predictions are made since the prediction errors add up. The direct approach creates a different model at each time step of the forecasting horizon. This avoids the recursive approach's error accumulation; however, it does not guarantee statistical dependence between the forecasts (Jensen, 2021). The equation for multi-step forecasting then becomes:

$$y_{(t+1):(t+T)} = f\left(y_{1:t}, X^{(i)}_{1:t}, X^{(i)}_{(t+1):(t+T)}\right), \quad i = 1, .., N \tag{2}$$

every variable here is represented by the same thing as in equation 1, with the addition of T, which is the length of the forecasting horizon.

In contrast to point forecasting is probabilistic forecasting, which instead of making a single prediction at each time step of the forecasting horizon, assumes there is a probability distribution for every time step. Probabilistic forecasting thus gives us an insight into the uncertainty of each prediction, and this can be useful in situations where a particular level of certainty is expected. While there are frequentist approaches that also produce probabilistic forecasts, such as quantile regression, the advantage of the Bayesian approach is in the use of a prior. This prior can act as surrogate data, providing additional information when the datasets are small. In this thesis, probabilistic forecasting will be done via the Bayesian approach.

## 5.1 Prediction Intervals

With probabilistic forecasting, the goal is usually to express this as prediction intervals as a way of conveying the uncertainties in the predictions where wider intervals indicate increased uncertainty. Prediction intervals are an estimate of the true interval that contains future observations with a certain probability (Eikeland, Hovem, Olsen, Chiesa, & Bianchi, 2022). This means there should be a 90% probability that the next value is within the prediction intervals' upper and lower bounds for a 90% prediction interval.

A generic way of expressing this probability at time step n is:

$$P(Y_n \in C(X_n)) \geq 1 - \alpha \tag{3}$$

here $\alpha$ is the significance level, C($X_n$) is the confidence interval centered around the covariate $X_n$, and $Y_n$ is the response variable. This means, for a 90% prediction interval, $\alpha$ must be 0.1.

The quality of a prediction interval can differ significantly from each other, and there might be undercoverage or overcoverage, meaning that there are too few or too many observations within the prediction intervals. The sharpness of a prediction interval is also crucial as an excessively wide prediction interval tends to be less informative and convey high uncertainty in the prediction model (Jensen, 2021). Sharpness refers to how tightly the prediction interval covers the actual distribution. The ideal prediction interval must therefore maximize the sharpness while simultaneously providing the correct amount of coverage (Gneiting & Katzfuss, 2014). In addition to this, an ideal procedure for constructing a prediction interval should involve no strong assumptions about the underlying data distribution (Romano, Patterson, & Candès, 2019).

A marginal coverage guarantee is a prediction interval coverage guarantee that can be defined on average over a set of test points for any fixed value $X_{n+1} = x$ (Barber, Candes, Ramdas, & Tibshirani, 2019). To appease the marginal coverage guarantee, the true test value $Y_{n+1}$ must be covered by the prediction interval of at least 1 - *alpha* on average over a random draw of the training and test data from any underlying distribution such that equation 3 is fulfilled. If a model satisfies this marginal coverage guarantee it is said to be calibrated.

# 6 Autoregressive Integrated Moving Average Model

## 6.1 Introduction

Autoregressive Integrated Moving Average is a statistical time series model. The model is a combination of two models, the autoregressive model and the moving average model (MA). These two models and their extensions have dominated the field of time series forecasting (Adhikari & Agrawal, 2013). The reason for their dominance can be chalked up to a sound theoretical background; this, in turn, makes their behavior and properties easy to comprehend (Jensen, 2021). These models do, however, have their limitations; as a time series gets more complex, the predictions often worsen due to the fact that they frequently rely on assumptions of linear relationships and temporal dependencies (Brownlee, 2018). Machine learning methods have been applied to overcome these limitations, such as echo state networks.

ARIMA models assume that future values of a given time series are distributed according to some known distribution, the normal distribution, for example, and are linearly dependent on its past historical observations (Adhikari & Agrawal, 2013). The assumptions made are critical to the ease of understanding, interpreting, and developing the models and are a large contributing factor as to why they are routinely used in time series forecasting (Jensen, 2021). However, while these assumptions increase the ease of use, linear models approximating non-linear responses generally do not provide adequate accuracy for real-world forecasting tasks (Zhang, 2001).

In order to properly implement these models, the model orders must be chosen, and this has been proven to require both skill and expertise (Box et al., 2015), as the optimal model orders greatly increase the predictive power of the model. In its infancy, the approach to choosing the model and its order was largely subjective and based on the experience of the user (Jensen, 2021). However, functions such as the autocorrelation and the partial autocorrelation can also inform the user of suitable model orders. Later on, several metrics to identify the optimal model and orders were developed, these include the likes of Akaike's information criterion (AIC), Akaike's final prediction error (FPE), and the Bayes information criterion (BIC) (De Gooijer & Hyndman, 2006). These all aim to find the optimal model orders which minimize the single-step forecast.

## 6.2 Autoregressive Model

An autoregressive model is a model that linearly depends on a specified number $p$ of past values at time $t$ denoted as $\mathbf{z}_t$. A univariate autoregressive model is termed AR($p$) while its multivariate counterpart is called a vector autoregressive model VAR($p$). A VAR($p$) model can be expressed mathematically as (Tsay, 2014):

$$\boldsymbol{z}_t = \boldsymbol{c} + \boldsymbol{a}_t + \sum_{j=1}^{p} \boldsymbol{\phi}_j \boldsymbol{z}_{t-j} \tag{4}$$

here $\mathbf{c}$ is a constant vector, $\boldsymbol{\phi}_j$ are the parameters for the model and $\boldsymbol{a}_t$ representing a stochastic vector with zero mean vector and positive-definite covariance matrix whose realizations are i.i.d.

## 6.3 Moving Average Model

A moving average model, in contrast to an autoregressive model, uses the errors, $\boldsymbol{\alpha}_t$, from past forecasts in its prediction at time step $t$. The model is termed MA($q$), where $q$ determines the number of past errors that affect the current value. As with the autoregressive model, extension into multivariate makes it a vector moving average model of order $q$, VMA($q$). This can be expressed according to this (Tsay, 2014):

$$\boldsymbol{z}_t = \boldsymbol{\mu} + \sum_{i=0}^{q} \boldsymbol{\theta}_i \boldsymbol{a}_{t-i} \tag{5}$$

here the $\boldsymbol{\theta}_i$ represent the model parameters, while $\boldsymbol{\mu}$ is the expectation $\boldsymbol{z}_t$ and the past error vector terms are $\boldsymbol{a}_{t-i}$. The weighted moving average can be thought of as values of $\mathbf{z}_t$ as implied by equation 4.

## 6.4 Autoregressive Integrated Moving Average Model Defined

As stated earlier, autoregressive moving average models are the combination of AR($p$) and MA($q$), denoted ARMA($p$,$q$); these models are often used for time series forecasting tasks. Forecasts are made from this model by a linear combination of past errors and past values of the time series. An assumption for this kind of model is stationarity, meaning that all statistical properties of the time series are constant (Jensen, 2021). Examples of non-stationary time series are time series that exhibit trends[1] or seasonalities[2].

To circumvent this issue, extensions of the ARMA process are applied; these models are termed autoregressive integrated moving average and seasonal autoregressive integrated moving average. These models are denoted as ARIMA($p$,$d$,$q$) and SARIMA$(p, d, q) \times (P, D, Q, m)$, respectively. The difference between ARMA and ARIMA models is the inherent differencing in the ARIMA model, where $d$ indicates the degree of differencing and thereby corresponds to the integrated part of the ARIMA model. Differencing is done to remove trends. An example of differentiation is shown below with three levels of differencing:

$$\begin{aligned}
z_t' &= z_t, & d &= 0\\
z_t' &= z_t - z_{t-1}, & d &= 1\\
z_t' &= z_t - 2z_t{-}1 + z_{t-2}, & d &= 2
\end{aligned}$$

The SARIMA model removes the trend just like the ARIMA model but with the distinction of taking the seasonality of the time series into account. This is done with the $(P, D, Q, m)$ orders of the model, where each letter represents the same as in the ARIMA model with the exception of $m$ that influenced the other seasonal elements. This means, for example with an $m = 12$ suggesting a cyclical pattern lasting a year and with a $P = 1$, the seasonal offset for this model then would be $t - (m \times 1) \ = \ t - 12$. A model parameter can be set to zero (Brownlee, 2018), for example an ARIMA(0,0,1) is reduced to an MA(1) model. The ARIMA model can be further expanded upon to create the ARIMAX model, and this model includes the use of exogenous variables $(X)$ using a linear combination.

In order to make the task of choosing optimal model orders easier, the autocorrelation function (ACF) and the partial autocorrelation function (PACF) are plotted. These can

[1] A change in behavior over time

[2] Cyclical pattern over time

be used to identify if the time series exhibits trends or seasonalities. Another way to check if the time series is stationary is through statistical tests, such as the Dickey-Fuller test (Dickey & Fuller, 1979). If the plots and the test show clear non-stationarity, then the time series needs to be differenced to possible trends, at least for the models that require a stationary time series. Finally, when the time series is deemed stationary, the AR and MA orders can be chosen using ACF and PACF once again, where these orders are chosen to be as small as possible within an acceptable error level (Jensen, 2021).

## 6.5 Prediction Interval in the ARIMA model

Prediction intervals from ARIMA models are calculated using the model residuals; the intervals are written in general from as (Hyndman & Athanasopoulos, 2018):

$$\hat{y}_{t+h} \pm c_\alpha \hat{\sigma}_h,$$

here $\hat{y}_{t+h}$ is the point prediction at time step $h$ while $c_\alpha$ is a constant that is chosen to receive the desired degree of confidence (Shumway & Stoffer, 2017), and $\hat{\sigma}_h$ is the standard deviation estimate of the forecast distribution at time $h$. These estimates are assumed to be uncorrelated and normally distributed and are calculated from the standard deviation for the model's residuals.

These prediction intervals usually increase in width as the forecasting horizon increases, as was stated in section 5; this is due to the associated rise in uncertainty (Jensen, 2021). Stationary models with d = 0 generally have prediction interval widths that converge for longer horizons, while models with $d \geq 1$ will have widths that continually increase (Hyndman & Athanasopoulos, 2018).

With one-step-ahead predictions, the standard deviation is approximately the standard deviation of the distribution therefore the prediction interval simply becomes $\hat{y}_{t+1} \pm c_\alpha \hat{\sigma}$ irrespective of the model orders for all ARIMA models (Hyndman & Athanasopoulos, 2018). The confidence level $c_\alpha$ is easily found from the quantile function for the standard normal distribution. While this simplification is possible for the one-step-ahead prediction, the same can not be said for multi-step ahead predictions. With multi-step ahead, predictions $\hat{\sigma}_h$ usually increases with $h$ and so does the complexity of the calculations. For more details on this, the reader is referred to the book written by (Brockwell, Brockwell, Davis, & Davis, 2016).

# 7 Echo State Network

## 7.1 Introduction

Echo state network provides an architecture to the class of computational dynamical systems, implemented following the principles of reservoir computing (Gallicchio & Micheli, 2011). This is done by way of feeding an input signal to a large, recurrent, and a randomly connected dynamic layer called the reservoir. Then, in combination with the output from a memory-less layer called the readout layer, the output is used to solve a specified task. In contrast to most other hard computing approaches, which rely on lengthy training procedures to acquire optimal model parameters through an algorithm, ESN has proved to be a speedy approach, usually by solving a convex optimization problem (Bianchi, Scardapane, Uncini, Rizzi, & Sadeghian, 2015; Bianchi & Suganthan, 2020).

The ESN approach has been used in a wide variety of contexts, some of them being static time series classification (Bianchi et al., 2020; Bianchi, Scardapane, Løkse, & Jenssen, 2017), time-series detrending (Maiorino, Bianchi, Livi, Rizzi, & Sadeghian, 2017), speech recognition (Skowronski & Harris, 2007), graph and trees classification (Gallicchio & Micheli, 2010; Bianchi, Gallicchio, & Micheli, 2022; Gallicchio & Micheli, 2013), condition monitoring systems (Noori, Waag, & Bianchi, 2020), intrusion detection (Tchakoucht & Ezziyyani, 2018), adaptive control (D. Xu, Lan, & Principe, 2005), harmonic distortion measurements (Deihimi & Rahmani, 2017) and various kinds of non-linear dynamical systems in general (Bianchi, Livi, & Alippi, 2016; Bianchi, Livi, Alippi, & Jenssen, 2017; Livi, Bianchi, & Alippi, 2017; Bianchi, Livi, & Alippi, 2018; Bianchi, Livi, Jenssen, & Alippi, 2017). Where ESN has seen its most use is in the field of predicting real-valued time-series relative, an example of this is electricity load which is used extensively in this thesis. The forecasts are typically performed 1-hour and 24-hours ahead (Bianchi, De Santis, Rizzi, & Sadeghian, 2015). Another area where ESN has achieved good results is in the prediction of chaotic time series, which is a testament to the capability of these neural networks to make accurate forecasts of a chaotic process from almost noise-free training data (Racca & Magri, 2021).

Even though a larger reservoir has the potential to capture the dynamic of the underlying system more accurately, it can result in a more complex model with a more considerable risk of lowered generalization capabilities due to overfitting (Gallicchio, Micheli, & Pedrelli, 2017, 2018). Here, several different regression methods have been adopted to train the readout layer. These could be affected by too many dimensions, which increased the demand for both software and hardware. However, it is possible to maintain meaningful distance relationships between original data and deal with the curse of dimensionality by using dimension reducing methods such as Principal Component Analysis. Reducing dimensions removes redundant features and algorithms previously unfit with large amounts of dimensions open up. In the literature, different methods have been proposed to increase the generalization ability of the network and regularize the output. This can be something in the vein of shrinking the weights of the connections from the reservoir to the readout layer or pruning some connections from the reservoir to the readout layer.

## 7.2 Echo State Network defined

Echo state networks contain a large, untrained recurrent layer of non-linear units and a linear, memory-less readout layer, and this readout layer is usually trained with linear regression. The equations that define ESN are as follows:

$$\mathbf{h}_k = \phi\left(\mathbf{W}_r^r \mathbf{h}_{k-1} + \mathbf{W}_i^r \mathbf{x}_k + \mathbf{W}_o^r \mathbf{y}_{k-1} + \xi\right) \tag{6}$$

$$\mathbf{y}_k = \left(\mathbf{W}_i^o \mathbf{x}_k + \mathbf{W}_r^o \mathbf{h}_k\right) \tag{7}$$

Where equation 6 describe the state-update and equation 7 describe the output and $\xi$ is a small i.i.d noise term. Figure 1 explain how the equations interact.

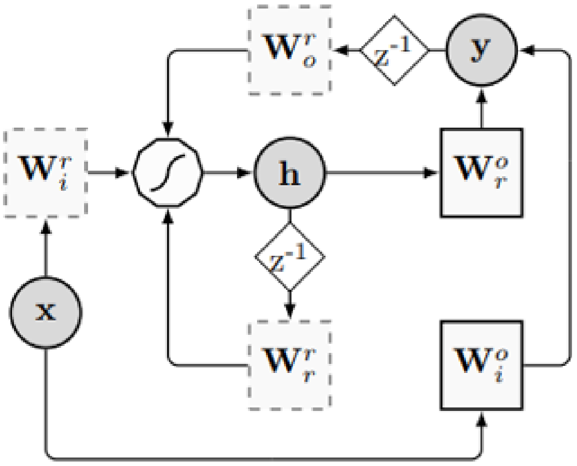


Figure 1: Showing a schematic depiction of the ESN architecture. Here the circles are the input $\mathbf{x}$, state $\mathbf{h}$ and output $\mathbf{y}$. The polygon is the activation function performed by the neurons and $z^{-1}$ is the unit delay operator. The two solid squared are the trainable matrices while the dashed squares are the randomly initialized matrices. Source: (Løkse et al., 2017).

Inside the reservoir there are $N_r$ neurons characterized by an activation function $\phi(\cdot)$, this is normally implemented as a *tanh* function. The network at time-step $k$ is driven by the input signal $\mathbf{x}_k \in \mathbb{R}^{N_i}$ generating the output $\mathbf{y}_k \in \mathbb{R}^{N_o}$, where $N_i$ and $N_o$ are the dimensions of the input and output. $\mathbf{h}_k \in \mathbb{R}^{N_r}$ being the vector that describes the instantaneous ESN states. There are several weight matrices, them being $\mathbf{W}_r^r \in \mathbb{R}^{N_r \times N_r}$, $\mathbf{W}_i^r \in \mathbb{R}^{N_r \times N_i}$ and $\mathbf{W}_o^r \in \mathbb{R}^{N_r \times N_o}$, where the first is the reservoir connections, the second being the input to reservoir and the last is the output to reservoir feedback. The values inside these matrices are sampled from the uniform distribution with -1 and 1 as the minimums and maximums, respectively.

In accordance with the ESN theory, the reservoir $\mathbf{W}_r^r$ must satisfy the "echo state property" (ESP) (Bianchi et al., 2020). Satisfying this property guarantees that a given input on the state of the reservoir will vanish in a finite number of time steps. There are several ways to re-scale the matrix $\mathbf{W}_r^r$, a widely used rule of thumb is to re-scale so that the spectral radius is less than one, i.e., $\rho(\mathbf{W}_r^r) < 1$, where $\rho(\cdot)$ denotes the spectral radius. However, there are also several theoretically founded approaches in the literature to tune $\rho$ properly. The other hyperparameters that require tuning are the density of the connections, the input scaling, and the number of connections in the reservoir. These hyperparameters, in addition to the spectral radius, should be tuned carefully using cross-validation techniques such as random or grid search as properly tuned hyperparameters for the ESN model are critical for the performance of the model (Thiede & Parlitz, 2019).

On the other hand, the weight matrices $\mathbf{W}_i^o$ and $\mathbf{W}_r^o$ are instead optimized for the specific task. To determine them, let us look at the training sequence of $T_{tr}$ desired input-output pairs given by

$$(\mathbf{x}_1, y_1^*)..., (\mathbf{x}_{T_{tr}}, y_{T_{tr}}). \tag{8}$$

The initial phase of the training is called *state harvesting*. Here the inputs are fed to the reservoir in compliance with Equation 3. This then produces a sequence of internal states $\mathbf{h}_1, ..., \mathbf{h}_{T_{tr}}$. As per the definition, the outputs of the ESN are not available for feedback. Instead, the desired output is used in Equation 4. The states are then filled into the matrix $\mathbf{S} \in \mathbb{R}^{T_{tr} \times N_i + N_r}$ and the desired output in a vector $\mathbf{y}^* \in \mathbb{R}^{T_{tr}}$ such that:

$$\mathbf{S} = \begin{bmatrix} \mathbf{x}_1^T, \mathbf{h}_1^T \\ . \\ . \\ . \\ \mathbf{x}_{T_{tr}}^T, \mathbf{h}_{T_{tr}}^T \end{bmatrix}, \mathbf{y}^* \equiv \begin{bmatrix} \mathbf{y}_1^* \\ . \\ . \\ . \\ \mathbf{y}_{T_{tr}}^* \end{bmatrix}. \tag{9}$$

The initial $D$ rows $\mathbf{S}$ and $\mathbf{y}^*$ should be discarded as they are washout elements that refer to a transient phase in the ESN's behavior.

As stated previously, the readout of the model is trained; this is done by solving a convex optimization problem, of which several closed-form solutions have been proposed in the ESN literature. The go-to standard here is regularized least-square regression, typically Ridge regression, which can be quite easily computed through the Moore-Penrose pseudo-inverse (Bianchi et al., 2020).

Ridge regression is performed by solving the following equation

$$\mathbf{W}_{ls}^* = \underset{\mathbf{W}_o^r \in \mathbb{R}^{N_i \times N_r}}{arg\ min} \frac{1}{2} \left\|\mathbf{SW} - \mathbf{y}^*\right\|^2 + \frac{\lambda}{2} \left\|\mathbf{W}\right\|^2 = (\mathbf{S}^T\mathbf{S} + \lambda\mathbf{I})^{-1}\mathbf{S}^T\mathbf{y}^* \tag{10}$$

where $\mathbf{W} = [\mathbf{W}_i^o \mathbf{W}_r^o]^T$ and $\lambda \in \mathbb{R}^+$ is the $L_2$ regularization coefficient.

# 8 Bayesian Regression

## 8.1 Introduction

In general, there are two approaches when it comes to statistical inference, the two being the frequentist approach and the Bayesian approach (Mushore, 2018). The frequentist approach is the more commonly used one and draws its conclusions only from the sample data using known experiments (Hoijtink, Klugkist, & Boelen, 2008), while Bayesian inference in addition to this includes the use of a subjective belief. The results of frequentist inference are deterministic, meaning even if repeated an infinite number of times, the results will not change. How significant these findings are is usually measured by a p-value or a confidence interval. These results make it possible to perform hypothesis testing, which results in finite conclusions, like rejecting or accepting an alternate hypothesis.

Furthermore, the parameters in the frequentist approach are fixed, even if they are unknown. Whereas in the Bayesian approach, the parameters are not fixed. They get updated as more information becomes available. They are also assigned a prior probability distribution before being updated that can add information about the parameters.

In contrast to frequentist inference, the advantage of Bayesian inference is the extreme flexibility offered. With the Bayesian approach, it is straightforward to fit realistic models to complex data sets with measurement errors and censored or missing observations (Dunson, 2001). Also, the ability to provide further information through a prior distribution is advantageous as this allows the user to quantify one's prior beliefs about the likely values of the unknown parameter. However, the use of subjective priors has been the more controversial aspects of Bayesian inference (Dunson, 2001). To combat this, one can use vague priors that avoid this issue altogether.

## 8.2 Bayesian Inference defined

The Bayesian inference approach can be split into three parts: the prior, the posterior, and the likelihood. If we have unknown parameters $\boldsymbol{\theta} = (\theta_1, ..., \theta_m)$ and data $\mathbf{y} = (y_1, ..., y_n)$, given the random variables $\mathbf{y}$ and $\boldsymbol{\theta}$ let $\pi(\cdot)$ denote the probability distribution function of a random variable. The likelihood function is then given by:

$$L(\boldsymbol{\theta}|\mathbf{y}) = \pi(\mathbf{y}|\boldsymbol{\theta}) = \prod_{i=1}^{n} \pi(y_i|\boldsymbol{\theta}) \tag{11}$$

This likelihood function is the sample data's density function, and the observations $y_1, .., y_n$ are thought to be independent given the unknown parameters $\boldsymbol{\theta}$ and can thus be written as the product in this equation. Here the prior distribution is given as $\pi(\boldsymbol{\theta})$ as a subjective belief on $\boldsymbol{\theta}$, this gives an indication of how uncertain $\boldsymbol{\theta}$ might be. Whereas the posterior distribution reflects the uncertainty of $\boldsymbol{\theta}$ after the data $\mathbf{y}$ has been observed. The posterior distribution is a conditional distribution, and by applying Bayes theorem to continuous distributions, one achieves this equation:

$$\pi(\boldsymbol{\theta}|\mathbf{y}) = \frac{\pi(\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\theta})}{\pi(\mathbf{y})} = \frac{\pi(\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\theta})}{\int \pi(\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\theta})d\boldsymbol{\theta}}. \tag{12}$$

$\pi(\mathbf{y}|\boldsymbol{\theta})\pi(\mathbf{y})$ represent the joint density of $\mathbf{y}$ and $\boldsymbol{\theta}$ while $\pi(\mathbf{y})$ is the normalizing constant, it ensures that $\pi(\boldsymbol{\theta}|\mathbf{y})$ is a proper posterior distribution. However, it is normally not of interest as we are usually more interested in the proportion of the product of the prior and the likelihood, this is then written as

$$\pi(\boldsymbol{\theta}|\mathbf{y}) \propto \pi(\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\theta}). \tag{13}$$

Also, this normalizing constant is often intractable. However, a very popular sampling technique, Markov Chain Monte Carlo (MCMC), avoids the computation of it.

The posterior distribution is usually used to acquire summary statistics such as the mean, variance, and quantiles of the posterior and represents a compromise of the data $\mathbf{y}$ and our belief of $\boldsymbol{\theta}$. By using this posterior distribution, it is possible to find credible regions for $\boldsymbol{\theta}$ or intervals for elements of $\boldsymbol{\theta}$. The difference between regular confidence intervals and credible intervals is that confidence intervals depend only on the data and are given as random variables for fixed parameters (Mushore, 2018). Whereas credible intervals are quantiles for the density of the parameter in question, which is dependent on both the data and the prior. A 100(1- $\alpha$)% credible interval is then defined as:

$$\int_{c_l}^{c_u} \pi(\theta|\mathbf{y})d\theta = 1 - \alpha, \quad \alpha \in (0,1) \tag{14}$$

where $c_u$ and $c_l$ are the quantiles that gives the specified probability.

The two most common types of credible intervals are the highest posterior density (HPD) and equi-tailed intervals. The HPD approach works by finding the sample space of $\boldsymbol{\theta}$ that makes up the 100(1- $\alpha$)% interval beginning at the peak of the posterior density function. The region then defines the HPD credible interval by this equation:

$$R(c) = \{\theta : \pi(\theta|\mathbf{y}) \geq c\} \tag{15}$$

Where c is the largest constant that fulfills this equation

$$\int_{\theta \in R(c)} \pi(\theta|\mathbf{y}) = 1 - \alpha \tag{16}$$

The equi-tailed interval is simply an interval where we choose $c_l = \frac{\alpha}{2}$ and $c_u = 1 - \frac{\alpha}{2}$. If the posterior distribution is symmetrical then the HPD interval and equi-tailed interval will be equal.

## 8.3 Sampling For Bayesian Regression

Regarding calculating the posterior distribution, a standard method is the sampling class MCMC as one can not typically express the distribution in an analytical form (Mushore, 2018). MCMC sampling work by generating irreducible and aperiodic Markov chains, which then should converge to the target posterior distribution. The accuracy of this sample will grow as the chain grows in the number of iterations.

Gibbs sampling is an MCMC method and is the specific case of the Metropolis-Hastings algorithm where all the proposals are accepted (Maklin, 2020). This sampling technique is applicable as the conditional distributions are known, but the joint distribution is not known or is difficult to sample from. The algorithm goes as follows:

**Algorithm 1:** Gibbs Sampling.

```
Initialize $Y^0, X^0$
for $j = 1, \dots$ do
    Sample $X^j \sim p(X|Y^{j-1})$
    Sample $Y^j \sim p(Y|X^j)$
```

Here we start by choosing an initial value of the random variables X and Y, and then we sample from the conditional distribution of X|$Y_{j-1}$. We use this value next to sample from Y|$X_j$ and then repeat n-1 times, alternating between sampling from X|Y and $Y|X$ using the current value of the other random variable.

Due to Gibbs sampling using the conditional distributions, it naturally lends itself to Bayesian regression. The posterior distribution for the regression parameters given an input and output from which we can sample then becomes:

$$P(\beta|y,X) = \frac{P(y|\beta,X) * P(\beta|X)}{P(y|X)} \tag{17}$$

where X is our explanatory variable, Y is our dependent variable, and $\beta$ are our coefficients. This means that $P(\beta|X)$ is our prior, $P(y|X)$ the normalizing constant and $P(y|\beta,X)$ being the likelihood. By utilizing Gibbs sampling for Bayesian linear regression, we receive as many different regression models as our number of iterations. These models can, in turn, be used to make predictions and construct our prediction intervals.

With the sampling done, inferences about each regression parameter can be made. More importantly, as this thesis focuses on probabilistic forecasting, each sample can be used to make a prediction, thus giving us samples from the posterior predictive distribution. With samples from the posterior predictive distribution, one can simply take whatever quantile is desired to produce a prediction interval. The posterior predictive distribution is sampled using this equation:

$$\hat{y}_n^j = \boldsymbol{\beta}^j \mathbf{X}_{new} + N(0,\sigma^j) \tag{18}$$

where $\hat{y}_n^j$ is the predicted value from sample j using the regression coefficients $\boldsymbol{\beta}^j$ multiplied with the new data $\mathbf{X}_{new}$ and adding the error term that is normally distributed with mean zero and variance $\sigma^j$.

## 8.4 MCMC - Diagnostic

In order to ascertain whether the sampling sufficiently samples from the target distribution, the sampling is done in what is called chains. This simply means that sampling is done twice; by sampling in chains, one can use plots like trace plots and density plots to assess whether the chains converge towards the same distribution or not.

It is essential that the two chains converge towards the same distribution as MCMC samples from the posterior distribution, as any inference could be radically different depending on which chain is used if the chains do not converge. This can mean drawing the wrong conclusions about the parameters, leading to improper decision-making for the model user. In the case of probabilistic forecasting, this can manifest itself as very different prediction intervals. Therefore, trace plots in combination with density plots for MCMC are a valuable tool to assess the convergence of the MCMC sampler.

### 8.4.1 Trace plot

Trace plots are a useful diagnostic tool that can identify problems in the sampling, such as an insufficient burn-in period or serial correlations, meaning that the beginning part of the sampling is discarded or a sample is somewhat dependent on the previous samples, respectively.

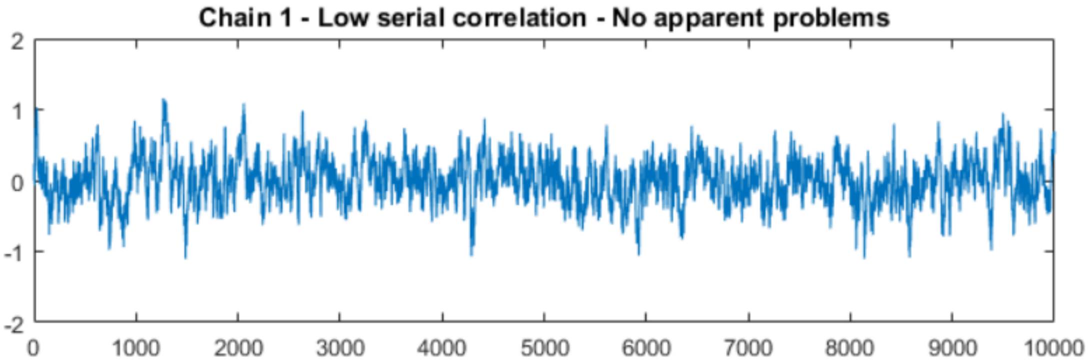


Figure 2: Showing an MCMC chain with sufficient burn-in and low serial correlations. Source: (Taboga, 2021).

In figure 2 it is shown how a trace plot ideally should look like where the values vary around the mean of the sample, indicating that the MCMC sampling are samples from a similar distribution as the target distribution (Taboga, 2021).

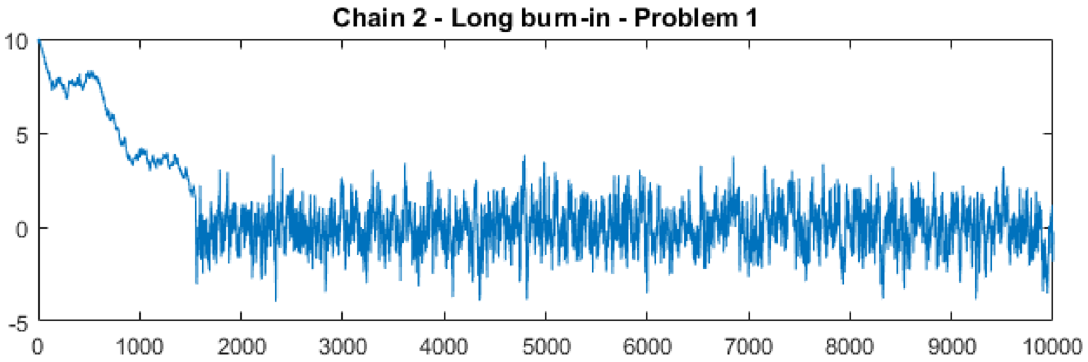


Figure 3: Showing trace plot of MCMC sampling where burn-in is required. Source: (Taboga, 2021).

Figure 3 show an MCMC chain where burn-in is required. It is categorized by drastically different values until the 1500 iterations mark, where it switches over to oscillating around the mean. By discarding the first 2500 samples, the trace plot more closely resembles the trace plot shown in figure 2. This is shown in figure 4.

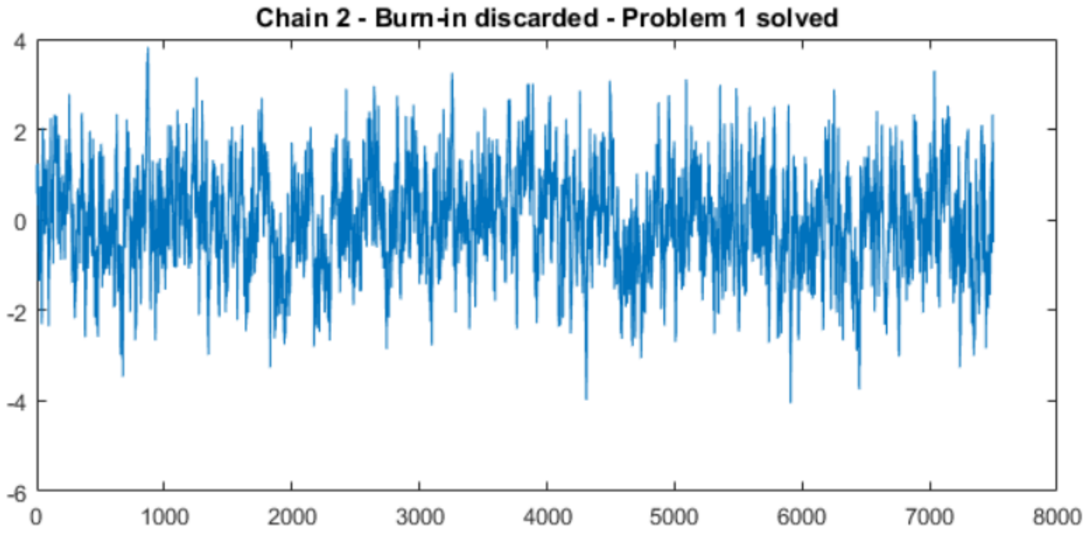


Figure 4: Showing trace plot of MCMC sampling after discarding the burn-in period of 2500 samples. Source: (Taboga, 2021).

High serial correlations between the samples mean that the chain is slow in exploring the sample space. This can be countered by increasing the number of iterations giving the chain more samples to sufficiently explore the sample space (Taboga, 2021). Figure 5 shows an MCMC chain where there is high serial correlation present.

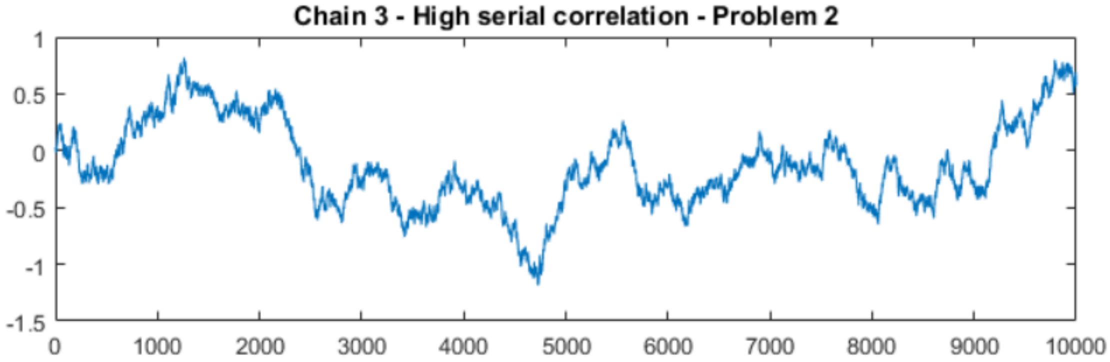


Figure 5: Showing trace plot of MCMC sampling where high serial correlations are present. Source: (Taboga, 2021).

By increasing the number of iterations, the trace plot shown in figure 6 more closely resembles the one in figure 2, thus making the sampling more closely resemble the target distribution.

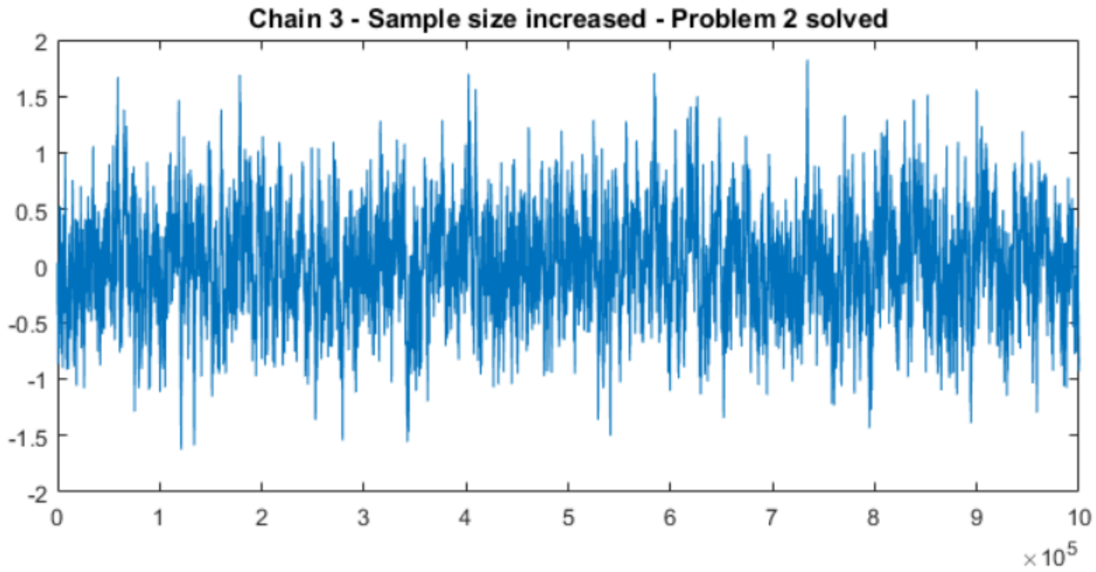


Figure 6: Showing trace plot of MCMC sampling where serial correlations are present but with increased iterations. Source: (Taboga, 2021).

### 8.4.2 Density plot

In addition to trace plots, density plots also offer a way to diagnose problems in the sampling. Density plots are simply smoothed histograms of the MCMC chain, making it possible to discern whether the chains in the MCMC sampling converge towards the same distribution and what kind of distribution it is, for example, if the sampling distribution is bi-modal or normally distributed.

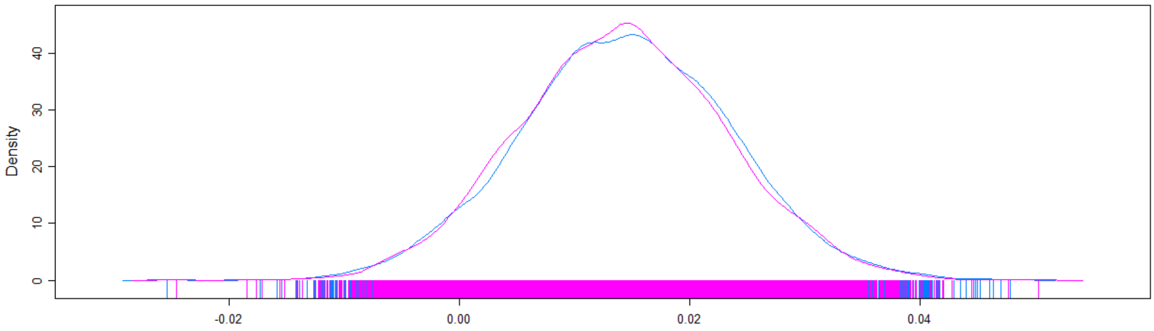


Figure 7: Showing density plot of MCMC sampling where two chains converge into a normal distribution.

Figure 7 show two MCMC chains that converge onto the same distribution, indicating good sampling choices that lead to sufficient sampling from the target distribution.

However, should the prior or the sampling parameters be wrongly chosen, namely the burn-in period and the number of iterations, this can lead to sampling chains that do not converge to the same distribution as shown in figure 8. In this figure, one chain resembles a bi-modal distribution while the other chain more resembles a skewed distribution indicating subpar sampling from the target distribution.

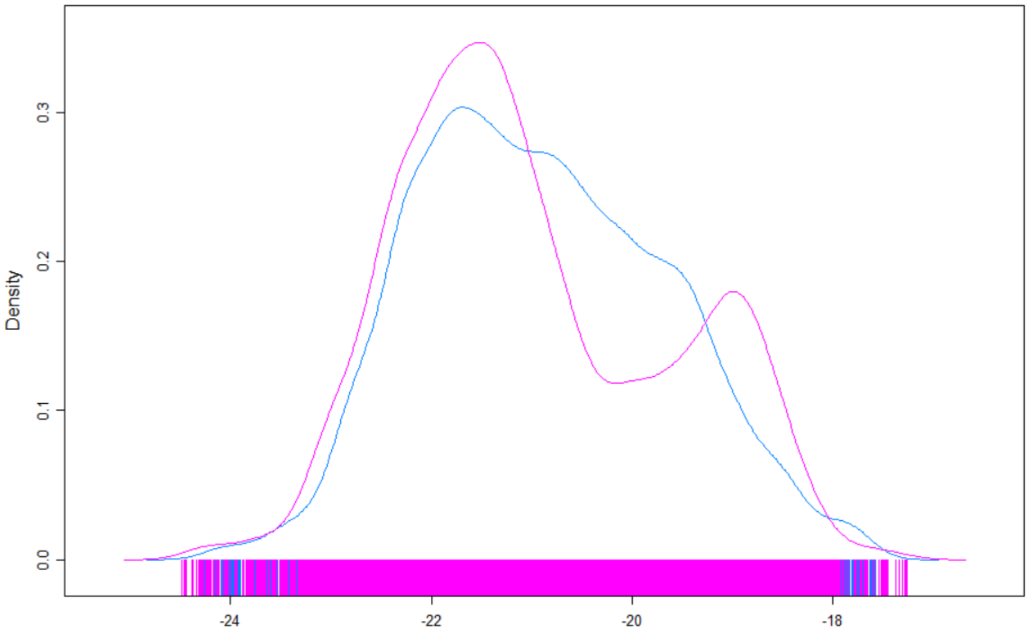


Figure 8: Showing density plot of MCMC sampling where two chains do not converge.

# 9 Principal Component Analysis

Principal component analysis (PCA) is a dimensionality reduction technique which is statistically motivated (Bianchi et al., 2020). The data is projected onto the orthonormal basis which preserves as much variance as possible in the input signal, while also keeping the individual components uncorrelated (Bianchi et al., 2020). The basis vectors are called principal components and thus gives PCA its name. PCA works by applying the linear transformation $\mathbf{Y} = \mathbf{E}^T\mathbf{X}$, where $\mathbf{Y}$ is the data projected on the principal components, $\mathbf{X}$ is our input signal and $\mathbf{E}$ is the orthogonal eigenvector matrix. For this to work we let $\mathbf{X} \in \mathbb{R}^p$ be a random vector and $\Sigma_x = \mathbf{E}\mathbf{\Lambda}\mathbf{E}^T$ be its covariance matrix, also $\mathbf{\Lambda} = \text{diag}(\lambda_i)$ is the diagonal eigenvalue matrix. Doing this transformation ensures that the covariance matrix of $\mathbf{Y}$ is $\mathbf{\Sigma}_{\boldsymbol{y}} = \mathbf{\Lambda}$, it is then clear that the components of $\mathbf{Y}$ are uncorrelated. It follows then that:

$$\sum_{i=1}^{p} Var(\mathbf{x}_i) = \sum_{i=1}^{p} Var(\mathbf{Y}_i) = \sum_{i=1}^{p} \lambda_i \tag{19}$$

Now it is possible to reduce the dimensionality to d where $d < p$ by projecting the data onto d eigenvectors with the largest eigenvalues. This means the dimension reduction step becomes:

$$\hat{\mathbf{Y}} = \mathbf{E}_d^T\mathbf{x} \tag{20}$$

This preserves most of the variance of $\mathbf{X}$ as $\mathbf{E}_d = (e_1, e_2, .., e_d)$ is the truncated eigenvector matrix with the eigenvalues sorted like this $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_d$. Thus making the reduction step to choose how many of these $e_i$ to include.

# 10 Platt Scaling

## 10.1 Introduction

Faithfully assessing uncertainty in specific applications can be as crucial as obtaining a high accuracy with predictions. Therefore, having a properly calibrated model is highly desirable (Kuleshov et al., 2018). Unfortunately, in a Bayesian setting, uncertainty estimates regularly fail to capture the true data distribution; see figure 9 below. When this occurs, a model is referred to as miscalibrated; an example of this is a prediction interval containing more or less than its intended coverage. This can happen for several reasons, such as a model bias or a predictor not being expressive enough to assign the correct probability to every credible interval.

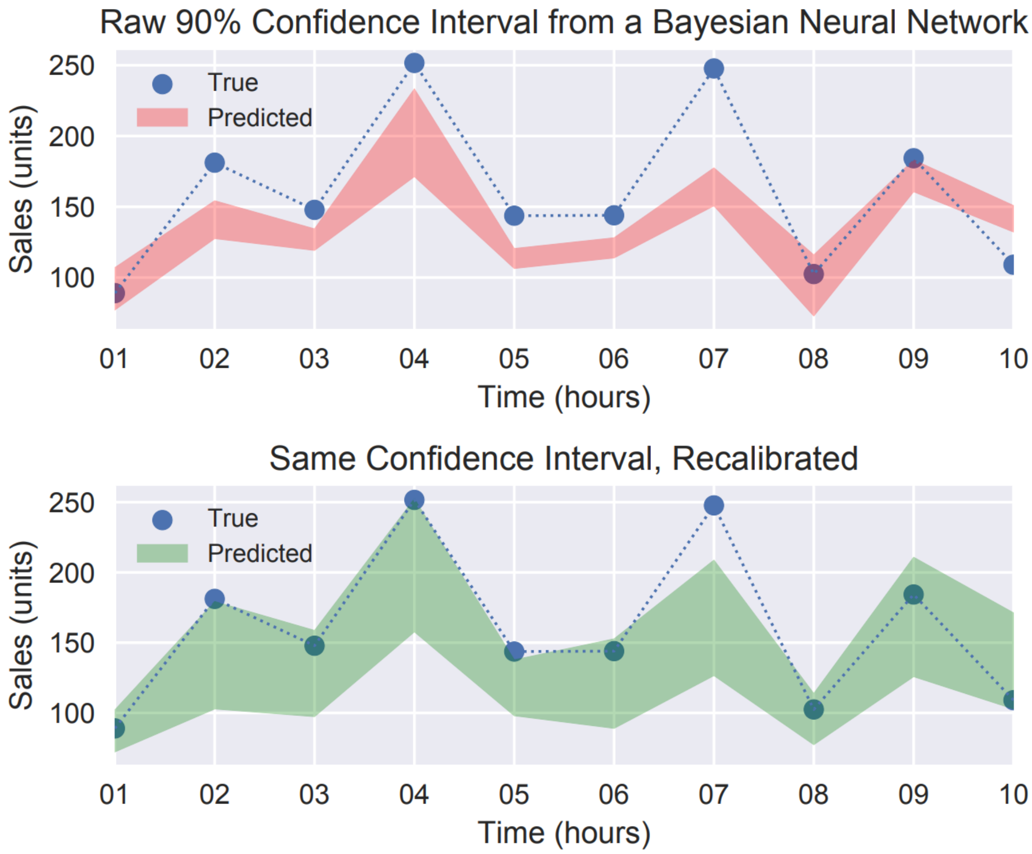


Figure 9: Showing properly and improperly calibrated confidence intervals. Source: (Kuleshov et al., 2018).

In a classification setting, this means a forecaster correctly classifying samples 90% of the time with a confidence level of $\alpha = 0.1$.

## 10.2 Platt Scaling Defined

For classifying into two classes, there is usually a class-separating feature that can help classify samples into the two classes. Figure 10 for an example of this.

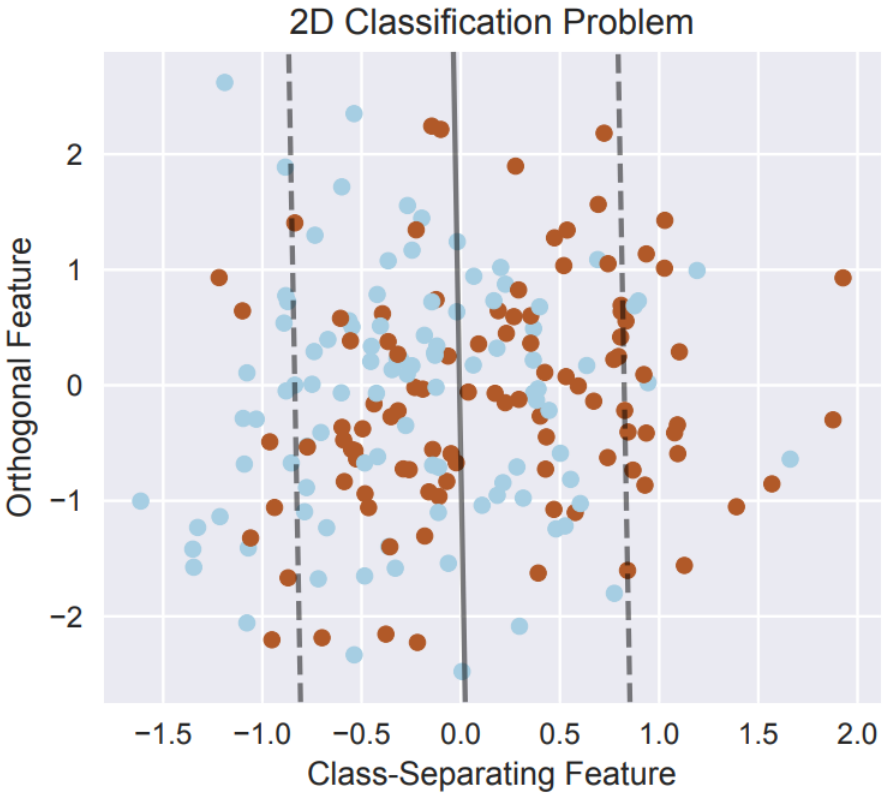


Figure 10: Showing two classes separated by a hyperplane in 2D. Source: (Kuleshov et al., 2018).

A binary classification forecaster H is perfectly calibrated if equation 21 holds true as T → ∞.

$$\frac{\sum_{t=1}^{T} y_t \mathbb{I}\{H(x_t) = p\}}{\sum_{t=1}^{T} \mathbb{I}\{H(x_t) = p\}} \rightarrow p \; for \; all \; p \in [0,1], \tag{21}$$

Here H$(x_t)$ denotes the probability of event $y_t = 1$ while $\mathbb{I}$ is an indicator function, meaning that the numerator is the sum of all the correct classifications and the denominator is the sum of both incorrect and correct classifications. A sufficient condition for a calibrated model when $x_t, y_t$ are identical and independent distributed realizations of the random variables X, Y ∼ $\mathbb{P}$ is:

$$\mathbb{P}(Y = 1 \mid H(X) = p) = p \; for \; all \; p \in [0,1]. \tag{22}$$

As most classification algorithms do not come perfectly calibrated as standard (Kuleshov et al., 2018), recalibration methods train a secondary model R : [0,1] → [0,1] on top of the already trained forecaster H such that R ∘ H is calibrated. If $x_t$ and $y_t$ are sampled i.i.d from $\mathbb{P}$ then recalibration can be seen as estimating the conditional density $R(p) = \mathbb{P}(Y = 1 \mid H(X) = p)$.

One of the most commonly used recalibration techniques is Platt scaling (Platt et al., 1999), which approximates the conditional density with a sigmoid (Kuleshov et al., 2018). The reason for approximating with a sigmoid is that margins between densities are apparently exponential (Platt et al., 1999). Therefore, Bayes' rule on two exponentials suggests a parametric form of a sigmoid (Hastie & Tibshirani, 1997).

$$\mathbb{P}(Y = 1 \mid H(X) = p) = \frac{1}{1 + e^{AH(X)+B}} \tag{23}$$

Where A and B are two parameters trained discriminatively, and the output of the forecaster is proportional to the log odds of a positive example. Platt scaling assumes the forecaster is made using support vector machines. The parameters A and B are fit using maximum likelihood estimation from a training set, separate from the one used to train the forecaster.

In essence, the objective is to estimate the empirical probability of observing a given class as a function of x, see figure 11.

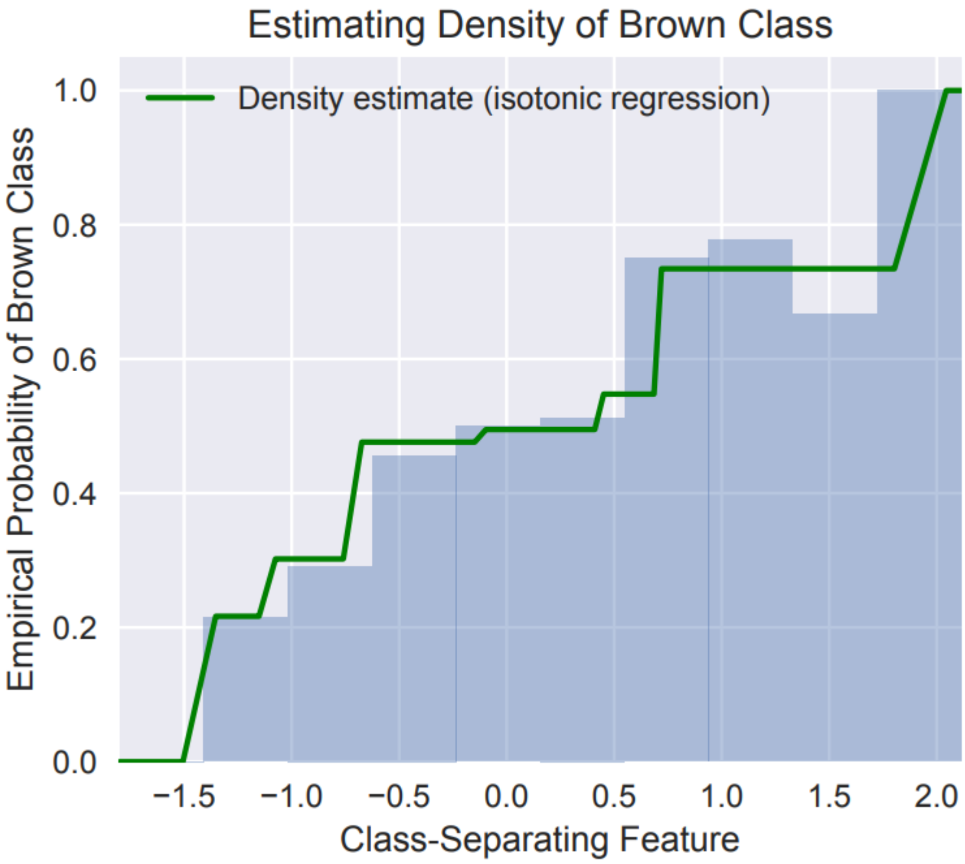


Figure 11: Showing empirical probability of observing a given class. Source: (Kuleshov et al., 2018).

To evaluate the recalibration a plot such as figure 12 is used, where each line should ideally follow the gray stipulated line as that represents perfect calibration. The uncalibrated and recalibrated lines are made by binning the prediction into ten intervals ([0,0.1], (0.1,0.2],...), and plotting the predicted versus the observed frequency of the brown class in each interval (Kuleshov et al., 2018).

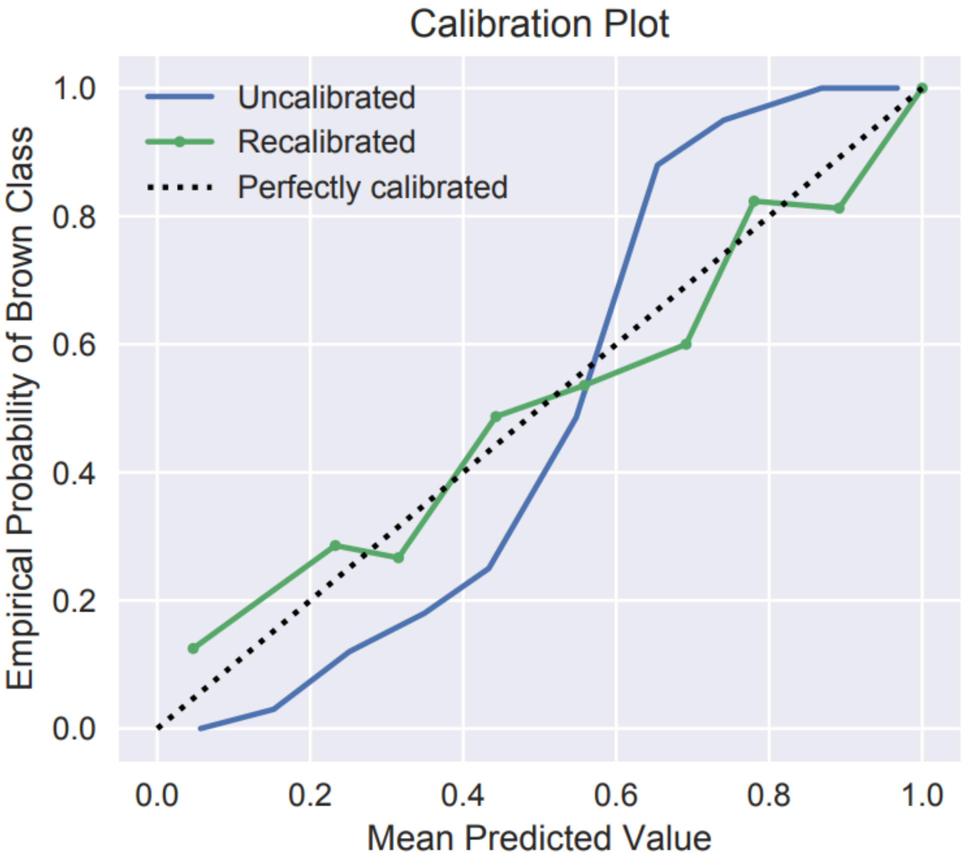


Figure 12: Showing recalibration plot using Platt scaling. Source: (Kuleshov et al., 2018).

For a further and more in-depth explanation of Platt scaling, the reader is referred to the paper by (Platt et al., 1999), which first introduced the concept.

# Part III / Proposed Method

This part of the thesis will present the proposed method, which is inspired by the works *Accurate Uncertainties for Deep Learning Using Calibrated Regression* by (Kuleshov et al., 2018) and *Deep echo state networks with uncertainty quantification for spatio-temporal forecasting* by (McDermott & Wikle, 2019). These two works tackle the two separate parts of the proposed method, namely quantifying uncertainty in an ESN model and recalibrating a Bayesian prediction interval. The two following sections, 11 and 12 will introduce these subjects. Section 13.2 will present the combined and proposed method to construct a probabilistic wind power and electricity load forecasts with accurate estimates of uncertainty. In addition to this, the hyperparameter search will be discussed in section 13.3.

## 11 Echo State Network Uncertainty Quantification with Bayesian Regression

(McDermott & Wikle, 2019) present an algorithm for Bayesian deep ensemble ESN; however, in this thesis, standard Bayesian ESN is performed, but it is still using the foundation laid out by (McDermott & Wikle, 2019). As stated by the authors, the motivation behind this approach is to provide the opportunity for uncertainty quantification, specifically in long-lead forecasting of environmental processes literature. Their paper also specifies that the method also applies to standard ESN. The training algorithm used differs from traditional ESN models; as traditional ESN models typically use Ridge regression, the Bayesian ESN model utilizes Bayesian regression as the training algorithm of choice.

According to the authors of this paper, standard and deep ensemble ESN does not account for the errors in estimating the regression or residual variance parameters in the training stage of the ESN algorithm nor for the truncation error in the basis expansion. This, however, can be changed by implementing Bayesian estimation at the training stage of the ESN model. The model is defined as:

$$\textbf{Data stage:}\quad \mathbf{Z}_t|\boldsymbol{\alpha}_t \sim D(\boldsymbol{\mu}(\boldsymbol{\alpha}_t), \boldsymbol{\Theta}), \tag{24}$$

$$\textbf{Output stage:}\quad \boldsymbol{\alpha}_t = \frac{1}{n_{res}} \sum_{j=1}^{n_{res}} [\boldsymbol{\beta}_1^{(j)} + \sum_{l=2}^{L} \boldsymbol{\beta}_l^{(j)} g_h(\tilde{\mathbf{h}}_{t,l}^{(j)})] + \boldsymbol{\eta}_t, \tag{25}$$

here D denotes an unspecified distribution with a known $n_z \times n_z$ spatial covariance matrix $\boldsymbol{\Theta}$, $\boldsymbol{\alpha}$ is our forecast. $g_h$ is the activation function, the same as $\phi$ in Equation (3) and $\boldsymbol{\eta}_t \sim N(\mathbf{0}, \sigma_\eta^2 \mathbf{I})$. What makes this model an ensemble is the $n_{res}$ part and is referring to the fact that there is $n_{res}$ number of models and is then averaged, hence why it is divided by that in equation 25. $\boldsymbol{\beta}$ are our regression matrices which have now been acquired through some form of sampling, the (j) is in reference from the j-th sampled $\tilde{\mathbf{h}}_{t,l}^{(j)}$ in $n_{h,1}$ dimensional vector from an $n_{h,1} \times T$ dynamical reservoir. These are defined for $l = 1, ..., L$ and j $= 1, .., n_{res}$ where L is the number of hidden layers, these hidden layers are what makes it deep instead of standard ESN with only one hidden layer. The matrices are then:

$$\boldsymbol{\beta}_l^{(j)} \equiv \begin{bmatrix} \boldsymbol{\beta}_{l,1}^{(j)\prime} \\ \cdot \\ \cdot \\ \cdot\boldsymbol{\beta}_{l,n_b}^{(j)\prime} \end{bmatrix} \tag{26}$$

This implies that if all the $\boldsymbol{\beta}_l^{(j)}$ terms for $l \leq 2$ are set to zero, this becomes the standard ESN model in an ensemble setting. Also the $\tilde{\mathbf{h}}_{t,l}^{(j)}$ terms come the from the deep nature of this type of model.

The paper by (McDermott & Wikle, 2019) also suggests using a stochastic search variable selection (SSVS) prior for the Bayesian sampling. This is because, as is stated in their paper, "This model is clearly overparameterized." One can use a multitude of variable selection priors from the Bayesian variable selection literature. (McDermott & Wikle, 2019) chose an SSVS prior as it can shrink a large percentage of the regression parameters to zero or close to zero while simultaneously leaving the remaining parameters unchanged. A stochastic search variable selection prior is defined in the paper as:

$$\beta_{l,b,k_l} | \gamma_l^{\beta_l} \sim \gamma_l^{\beta_l} N(0, \sigma_{\beta_l,0}) + (1 - \gamma_l^{\beta_l}) N(0, \sigma_{\beta_l,1}), \tag{27}$$

$$\gamma_l^{\beta_l} \sim Bernoulli(\pi_{\beta_l}). \tag{28}$$

Here the $k_l$ indexes the hidden units for a particular layer and $\sigma_{\beta_l,0} \gg \sigma_{\beta_l,1}$. $\pi_{\beta_l}$ is the probability of including each regression parameter.

Because of the conditional nature of the SSVS prior, (McDermott & Wikle, 2019) suggest using Gibbs sampling as the distributions are straightforward to sample from and their natural convenience with hierarchical models, due to being defined conditionally. At the same time, sampling $n_{res}$ reservoirs that are generated are treated as fixed covariates.

To finish the prior specifications, the variance parameter is chosen as an inverse-gamma prior, this way the variance parameter $\sigma_\eta^2$ is distributed according to $IG(\alpha_\eta, \beta_\eta)$. In regards to hyper-parameter search the authors suggest using the genetic algorithm for the hyper-parameters defined in the ESN model, while $\sigma_{\beta_l,0}, \sigma_{\beta_l,1}, \pi_{\beta_l}, \alpha_\eta \ and \ \beta_\eta$ are problem specific.

## 11.1 Discussion

The subject of this subsection will be to briefly discuss the application of Bayesian principles to the ESN architecture. The reader is referred to the paper by (McDermott & Wikle, 2019) should a more thorough and in-depth discussion be desired.

The approach introduced by (McDermott & Wikle, 2019) certainly achieves its goal of implementing uncertainty estimates with ESN models and is specifically made for use with spatio-temporal data sets. This methodology is then able to construct prediction intervals, which are not readily available from the ESN architecture. The authors also find their approach to give more accurate predictions than traditional ESN model, whether the model is "deep" or not. The authors note that the advantage of the deep architecture comes from allowing different time scales in the predictors. That is a positive for their experiments as they look at long-lead forecasting. However, that might not be as relevant regarding wind and electricity load forecasts as the forecast horizons are usually an hour or 24 hours ahead.

In the experiments, (McDermott & Wikle, 2019) note that the deep Bayesian model produces considerably better uncertainty estimates than just the deep model, which acquires its estimate from bootstrapping. They also mention that computing times are reasonably short, at least compared to traditional deep learning models. Most of the computing times come in the form of Bayesian sampling.

# 12 Accurate Uncertainties for Deep Learning Using Calibrated Regression

## 12.1 Introduction

(Kuleshov et al., 2018) present in their paper algorithms to accurately assess the uncertainty of forecasts. It is applied to both classification and regression. There are other ways to recalibrate prediction intervals (Kuleshov et al., 2018); however, their algorithm is unique in the fact that it is model-agnostic and that it can be applied to any regression model. This is possible it is applied as a post-processing step on the intervals produced by the trained model. The only requirement is that the model outputs a cumulative distribution function (CDF) for each observation. The algorithm boils down to simply shifting the quantiles used in making the prediction interval up or down to achieve correct coverage. Their algorithm is inspired by Platt scaling, which can only be used on classification problems, while their algorithm is applied to regression problems as well. Algorithm 2 is written verbatim from their paper in section 12.3; it can be a bit unclear what to do from reading the algorithm. In practice, it is pretty simple and computationally cheap, at least in a Bayesian setting, as the data needed is readily available.

## 12.2 Calibrated Regression Defined

In a regression setting, a calibrated prediction interval simply means a (1-$\alpha$)100% interval should contain (1-$\alpha$)100% of the true value. I.e with a 90% prediction interval 90% of $y_t$ should be inside. Mathematically formulated as:

$$\frac{\sum_{t=1}^{T} \mathbb{I}\{y_t \leq F_t^{-1}(p)\}}{T} \rightarrow \ p \ for \ all \ p \in [0, 1], \tag{29}$$

where $F_t^{-1}$ is the quantile function at time step t, p is our level of confidence, and T → $\infty$. This means that the predicted and empirical CDF should converge as the data set increases in length if the model is calibrated. If $x_t$ and $y_t$ are i.i.d realizations of the random variable X and Y, respectively, then a sufficient condition for this will be:

$$\mathbb{P}(Y \leq F_x^{-1}(p)) = p \; for \; all \; p \in [0, 1] \tag{30}$$

where $F_x$ denotes our forecast at X. This implies for every $p_1$ and $p_2 \in$ [0, 1]

$$\frac{\sum_{t=1}^{T} \mathbb{I}\{F_t^{-1}(p_1) \leq y_t \leq F_t^{-1}(p_2)}{T} \rightarrow p_2 - p_1 \tag{31}$$

where $p_1$ and $p_2$ are our lower and upper prediction interval quantiles.

As a visualization of a model with uncalibrated confidence intervals in comparison to calibrated confidence intervals, see figure 13. Here it is possible to see a model with too low coverage where the width is overly narrow, causing more than the desired amount to be outside the interval. While after recalibration, the interval is wider with more of the true values inside the confidence interval.

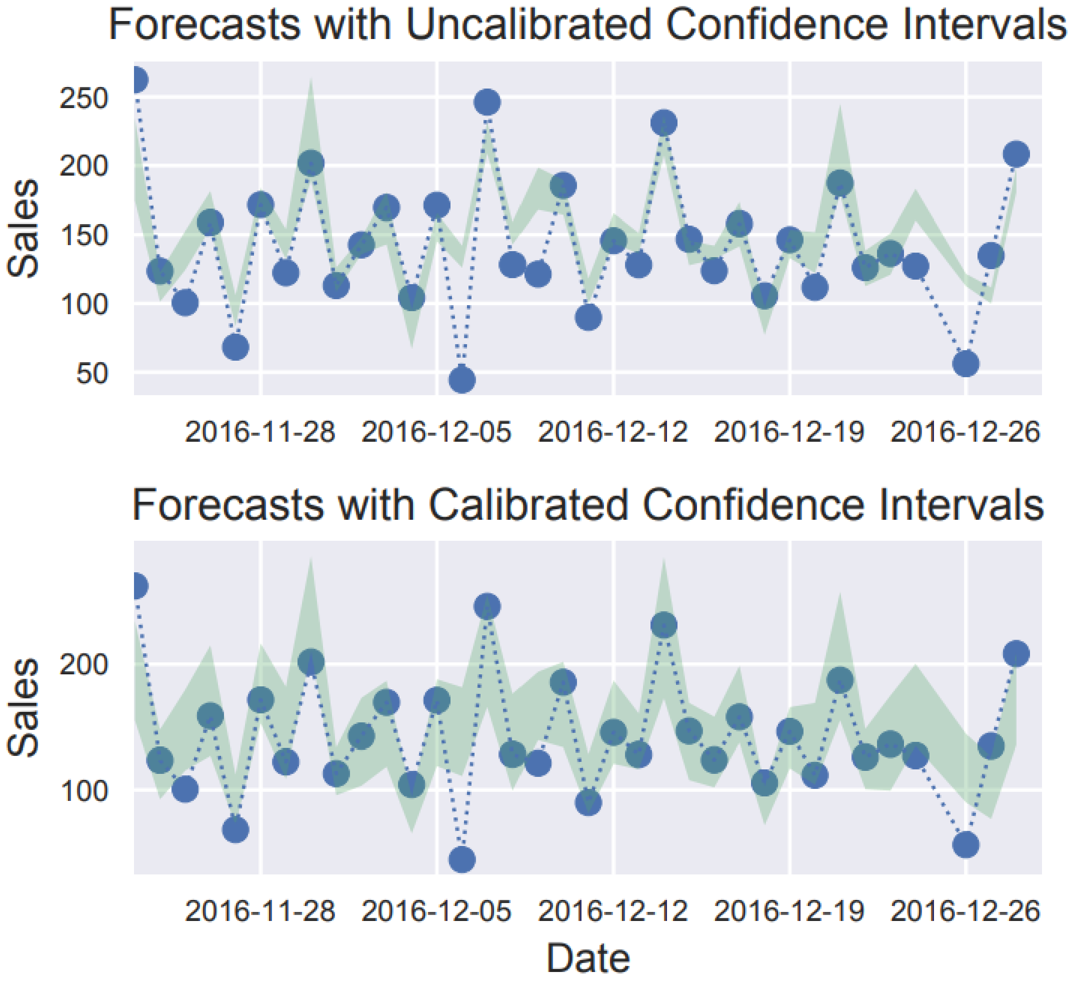


Figure 13: Difference between calibrated and uncalibrated prediction intervals. Source (Kuleshov et al., 2018).

## 12.3 Performing Calibration

However, simply a properly calibrated confidence interval is not desirable; it should also be sharp. Meaning the interval should be as narrow as possible while still containing the requisite number of observations. Explicitly, var($F_t$) should be small, where $F_t$ is the cumulative distribution function for the Bayesian predictions at time step t (Kuleshov et al., 2018).

This calibration technique resembles Platt scaling on which it is based. It relies on an auxiliary model R: [0, 1] → [0, 1] such that the R ∘ $F_t$ forecasts are calibrated. The algorithm is simple in nature and can be applied to most regression models, specifically applied to Bayesian regression is used in this thesis. The algorithm goes as follows:

**Algorithm 2:** Recalibration of Regression Models.

```
input : Uncalibrated model $H : X \to (Y \to [0,1])$ and calibration set
        $S = \{(x_t, y_t)\}_{t=1}^{T}$.
output: Auxiliary recalibration model $R : [0,1] \to [0,1]$.
1 Construct a recalibration dataset: D = $\{([H(x_t)](y_t), \hat{P}([H(x_t)](y_t)))\}_{t=1}^{T}$ where
    $\hat{P}(p) = \left|\{y_t | [H(x_t)](y_t) \leq p, t = 1, ..., T\}\right| / T$
2 Train a model R on D
```

By using algorithm 2 one achieves a new model where inputting the desired confidence level returns the quantile to use, which theoretically should give calibrated prediction intervals.

## 12.4 Calibrated Regression In Practice

Performing recalibration in practice means first obtaining this data set D, called recalibration set. This is done by making all the probabilistic predictions using a separate data set from the training data set. After that, it is possible to make the predictive CDF part of the data set. One does that by averaging how many predicted values are below the true value at each time step; this is the $\hat{P}([H(x_t)](y_t))$ part of the recalibration set. With this predictive CDF, obtaining the empirical CDF simply becomes the average of the predictive CDF values that are less than each predictive CDF value at each time step; this is the $[H(x_t)](y_t)$ part of the recalibration set. To visualize the nomenclature used in the paper by (Kuleshov et al., 2018), see figure 14. We notice that in practice, the validation set, used to optimize the model's hyperparameters, can also be used as the recalibration set D.

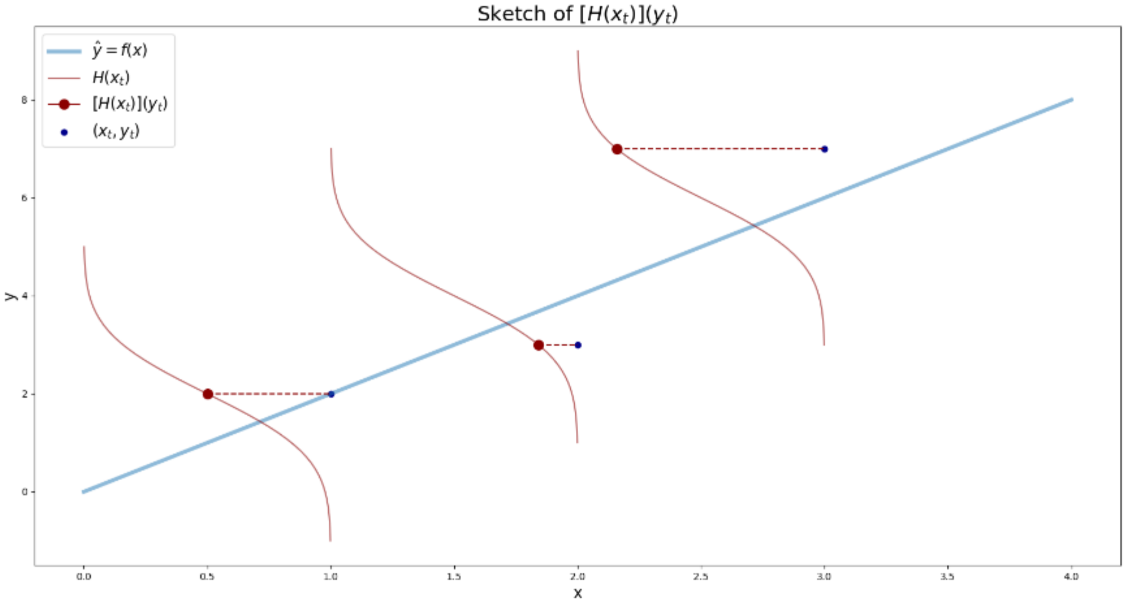


Figure 14: Visualization of what the different nomenclatures represent. Source (Rentsch & Vasishth, 2019).

Now that both parts of the calibration set are in place, all that remains is to perform the isotonic regression with the empirical CDF as input and the predictive CDF as output. This, in turn, allows us to input the desired confidence level into the function, and the output is the readjusted quantile level needed for the model to be properly calibrated, in theory. The authors suggest using isotonic regression for this part as the regression line fit is a free-form line. Isotonic regression captures the variance seen in a plot of predicted versus empirical CDF; also, it does not force a straight line onto the points (Kuleshov et al., 2018). Another reason for using isotonic regression is its non-decreasing nature, which is essential as a $P(Y \leq F_x^{-1}(p))$ is monotonically increasing. Therefore, for the recalibration to work, it is crucial that the fitted line is non-decreasing.

In short, the method proposed by (Kuleshov et al., 2018) is isotonic regression fit on the predicted cumulative distribution and the empirical cumulative distribution. Thus recalibrating each theoretical quantile to the empirical quantiles on the recalibration dataset. Figure 15 shows the curve the isotonic regression is fit on.

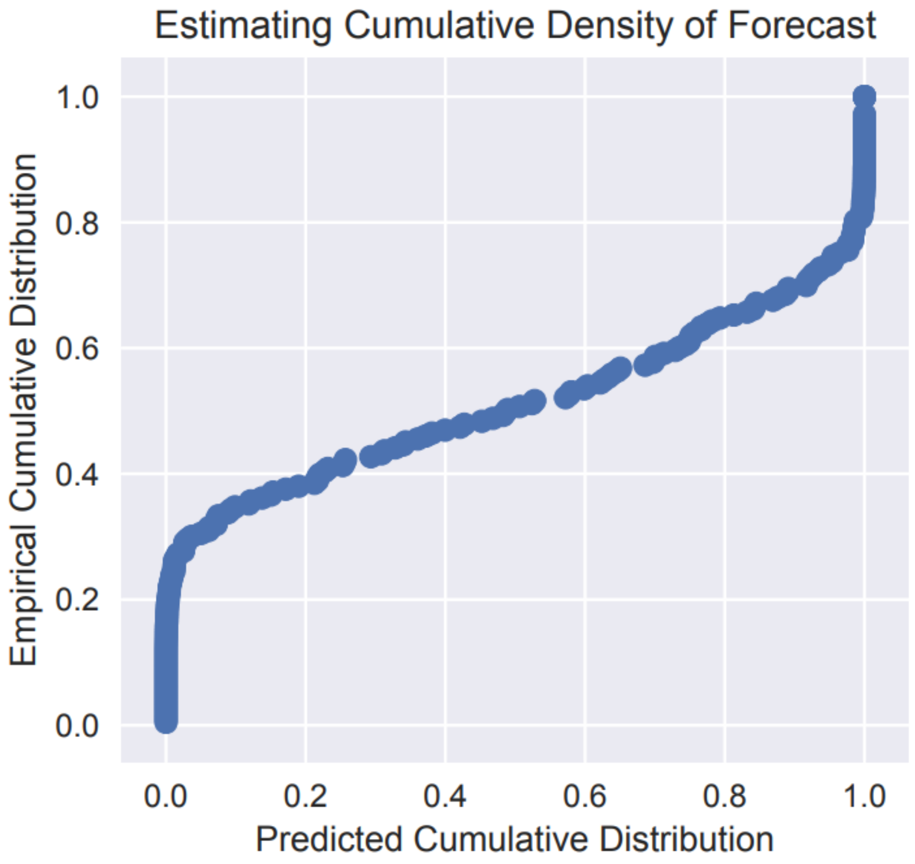


Figure 15: Showing a plot of the predicted cumulative distribution against the empirical cumulative distribution, the function that is fit by isotonic regression. Source (Kuleshov et al., 2018).

## 12.5 Diagnostic

It is recommended that this recalibration set is made using a separate dataset from the regular regression training to reduce overfitting. A diagnostic tool for this method is the calibration plot below, see figure 16.

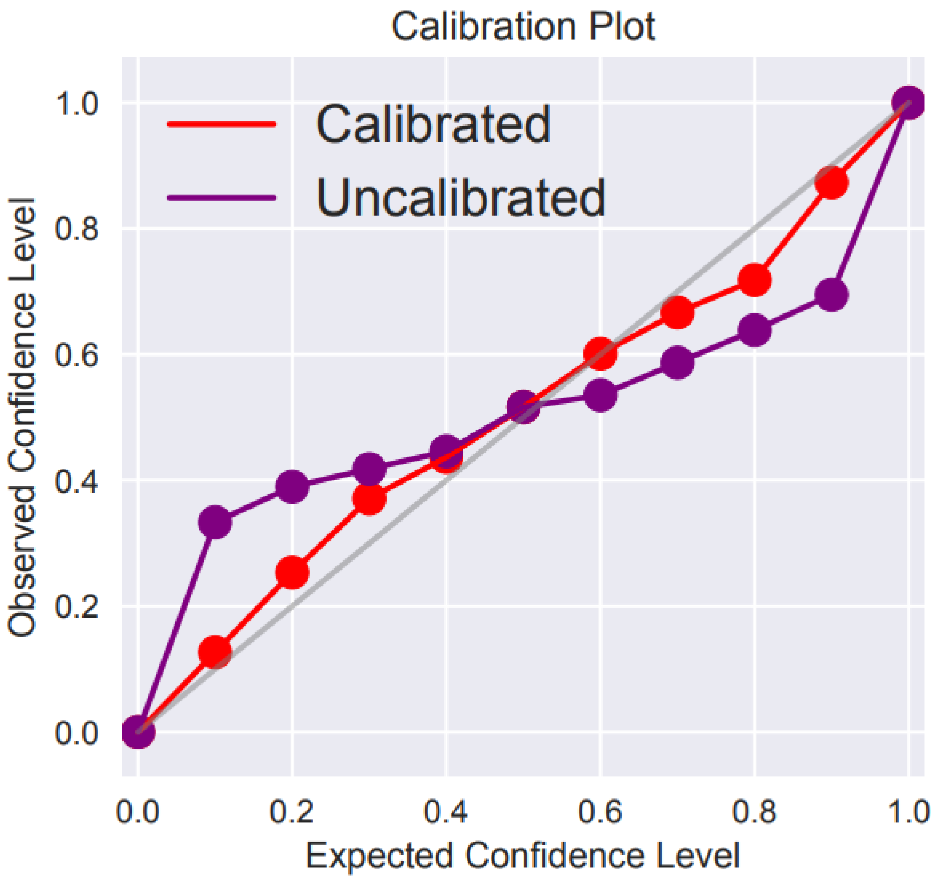


Figure 16: Showing diagnostic plot. Source: (Kuleshov et al., 2018).

This plot indicates how close to a perfectly calibrated model we receive from recalibrating. It should ideally match the gray line where the expected confidence level is the same as the observed one. Each point in figure 16 represents the observed confidence level attained using the expected confidence level; the expected confidence level starts from 0 and goes up to 1. The expected confidence level is just the intended coverage level, meaning the coverage one expects from the interval. The observed confidence level is the coverage attained using $-\infty$ as the lower bound and the expected confidence level as the upper bound. In other words, the probability of observing an outcome $y_t$ in a set of ten intervals $(-\infty$, F(p)] for p $= 0, 0.1, ..., 1$ (Kuleshov et al., 2018), calculated according to equation 30.

## 12.6 Discussion

The subject of this subsection will be to briefly discuss recalibration of a prediction interval. The reader is referred to the paper by (Kuleshov et al., 2018) should a more thorough and in-depth discussion be desired.

In the experiments (Kuleshov et al., 2018) achieve somewhat or significantly better calibration. They used eight different UCI data sets and two different regression algorithms, Bayesian linear regression and dense neural network. The authors note that this form of recalibration preserves the accuracy of point estimates given sufficient data for recalibration. However, the prediction intervals can become less sharp if the model underestimates the uncertainty and is too tight around the mean prediction before recalibration. This should come as no surprise, as the interval naturally grows as the upper and lower quantiles approach 1 and 0, respectively.

The approach introduced and formalized by (Kuleshov et al., 2018) appears to be successful for recalibrating in both the classification and continuous case. They also introduced visualization tools such as figure 16 to assess performance.

# 13 Proposed Method: Calibrated Uncertainty Estimates For Echo State Networks

## 13.1 Motivation

Simply having an accurate forecast is not enough in many situations, and although ESN forecasts can be incredibly accurate, it does not inherently provide uncertainty estimates (Bianchi et al., 2020). The fact that the approach made by (McDermott & Wikle, 2019) can be applied to any ESN model provides the proposed method opportunity to be used within a wide array of different time series forecasting tasks. However, the prediction intervals provided by the Bayesian ESN model are not guaranteed to provide adequate coverage and are very computationally expensive as dimensionality can be high using optimized hyperparameters. Through the use of dimensionality reduction techniques such as principal component analysis, the Bayesian ESN model becomes a more feasible model to implement.

Recalibrating the quantiles from the Bayesian sampling should, in theory, make the prediction intervals at least approximately marginally valid with the algorithm provided by (Kuleshov et al., 2018). The effect of this is more helpful prediction intervals that can better guide decision-making as uncertainty should then be faithfully represented in the prediction intervals.

By combining the modifications done to the traditional ESN model to produce probabilistic forecasts with recalibrating the corresponding prediction intervals to accurately quantify the uncertainty in the predictions while maintaining fast computations time through the use of dimensionality reduction techniques, the proposed method hopefully will be valuable.

## 13.2 Model Overview

As introduced above, the proposed method combines Bayesian regression as the regression algorithm with PCA being applied to the reservoir states and then recalibrating the quantiles. The steps for performing the proposed method thus become:

- **Step 1:** The first step is to initialize the ESN model and retrieve the training reservoir states; PCA is then applied to these states.

- **Step 2:** After reducing the dimensionality, the Gibbs sampling is applied to give us $n$ regression weight samples. With the regression weights in place, prediction of the dataset's recalibration part can occur. To do this, reservoir states from the recalibration dataset must be retrieved, and this is then reduced down to the same dimensionality as the training by applying PCA with the same rotation matrix as in step 1.

- **Step 3:** The next step is performing the recalibration. With each sample from the Gibbs sampling, a prediction is made, thus giving us $n$ predictions at each predicted time step. By applying the recalibration algorithm, we attain the isotonic regressed function.

- **Step 4:** The final step is inputting the desired confidence level into the recalibration function, which then gives the quantile levels to get the right amount of coverage. These quantiles are then used to produce the final prediction interval using the testing data, and these prediction intervals are predicted the same way as the recalibration ones in step 3.

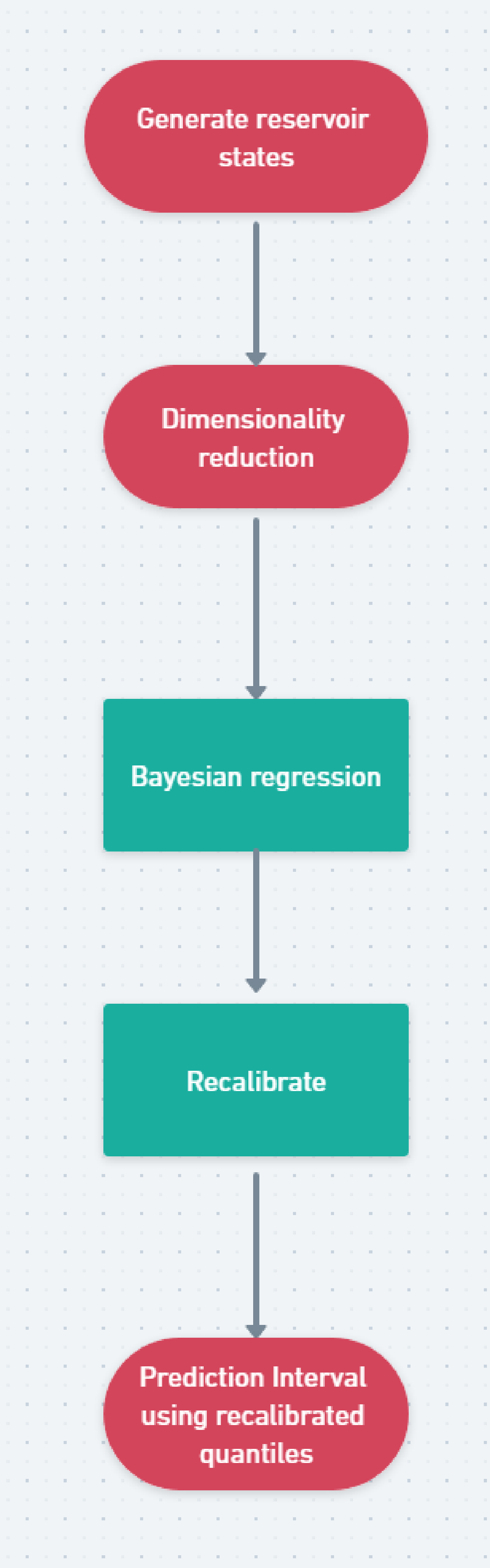

Generate reservoir states
Dimensionality reduction
Bayesian regression
Recalibrate
Prediction Interval using recalibrated quantiles

Figure 17: Flowchart for the steps in the proposed method.

**Algorithm 3:** Proposed method algorithm

```
input : Training data ${(x_i, y_i)}_{i=1}^{T_1}$, confidence level $\alpha \in (0,1)$, recalibration data
        ${(x_i, y_i)}_{t=T_1+1}^{T_1+T_2}$ and test data ${(x_i, y_i)}_{t=T_2+1}^{T_2+T_3}$.
output: Recalibrated prediction intervals
Initialize ESN model
Retrieve reservoir states from training data
Apply PCA to said reservoir states according to equation 20
Perform Bayesian regression using Gibbs sampling with algorithm 1, sample $n$ iterations
Retrieve reservoir states from recalibration data
Apply PCA to those reservoir states
Using the samples from the Gibbs step, make $n$ predictions for each time step
Construct recalibration dataset to train auxiliary model in accordance with algorithm 2
Train said auxiliary model with isotonic regression
Input desired confidence level into auxiliary model
Retrieve reservoir states from testing data
Apply PCA to those reservoir states
Using the samples from the Gibbs step, make $n$ predictions for each time step
Use the output of step 10 as the quantiles for the prediction intervals
```

## 13.3 Hyperparameter Search

Choosing bad hyperparameters in the ESN model can significantly impact the performance of the model. Especially parameters like spectral radius and input scaling are critical to tune for the model to perform to the best of its abilities (Maat, Gianniotis, & Protopapas, 2018). The hyperparameters are, in practice, chosen by evaluating the model's performance on a validation dataset, or in our case, the recalibration dataset, with a specific network configuration. The hyperparameter optimization techniques can include the likes of manual search or grid search.

These methods work by selecting hyperparameters from a user-defined search space, either systematically in the case of grid search or randomly in the case of random search (Jensen, 2021). This space is a volume where each dimension represents a hyperparameter, and each of the points within represents a particular network configuration (Brownlee, 2018). Systematically searching this space is termed grid search, whereas randomly selecting configurations within this space is referred to as random search. In this thesis, the technique used will be random search

As stated in the paragraph above, random search is used in this thesis. Each configuration is initialized ten times, and the mean MSPE is noted for the configuration. Each hyperparameter is sampled from the uniform distribution such that the minimum and maximum values inside the hyperparameter space are reasonable. An exception is made for the number of units as it needs to be an integer; sampling from the uniform distribution does not work. Instead, a random number generator is used. The search is done using 1000 configurations which should provide ample opportunity to find a good network configuration.

Table 1: Values used in the random search for optimizing hyperparameters.

| Random search parameters | | |
|---|---|---|
| | Minimum | Maximum |
| Number of units | 500 | 1500 |
| Spectral radius | 0.15 | 1.55 |
| Input scaling | 0.05 | 0.95 |
| Density | 0.05 | 0.35 |

Table 1 shows the minimum and maximum values each hyperparameter can take. Obviously, these values exclude potentially better network configurations that are outside this hyperparameter space. However, these values should form a solid hyperparameter space in which the search can be conducted.

# Part IV / Experiments

In this part of the thesis, the proposed method will be applied to several different data sets to assess the performance. The proposed method is tested on both univariate and multivariate electricity load time series and a multivariate wind power data set. The performance of the recalibrated prediction interval will be compared to the uncalibrated one. In addition, comparisons will be made to traditional methods of time series forecasting such as ARIMAX. As discussed in the methodology section, the proposed method combines Bayesian regression with the standard ESN model to compute prediction intervals. The prediction intervals are then recalibrated to improve performance. Finally, the comparisons will be made to assess performance and whether the added computation cost and increased complexity are rewarding. As mentioned previously, standard ESN and ARIMAX will be used as reference models. Standard ESN is naturally chosen as the proposed method is an augmented ESN model, and ARIMAX is chosen as the ARIMA family of models are some of the most widely used models within time series forecasting (Jensen, Bianchi, & Anfinsen, 2022).

Section 14 will introduce the three data sets used in the experiments, in addition to the preprocessing steps of the time series. Afterward, section 15 will describe the architectures and implementation details of the different models. Then, in section 16 the different evaluation metrics are introduced in order to assess the performance. Lastly, in section 17.1 the results of the experiments will be discussed.

## 14 Datasets

This section will present three different real-world data sets, where two of them are electricity load data sets, and the third is a wind power production data set. A pre-analysis will be performed on all the data sets to determine whether the time series exhibit trends or seasonalities, which is essential to understand the data at hand in order to interpret the results and perform helpful preprocessing steps. The latter can include normalizing the data, detrending, or deseasonalizing. If any preprocessing steps are taken, the reverse transformation needs to be applied as a post-processing step to transform the data back to its original format to evaluate results properly.

### 14.1 ACEA - Univariate

The first dataset that is analyzed is the electricity consumption registered by Azienda Comunale Energia e Ambiente (ACEA). ACEA is the company that provides electricity to Rome and some of the neighboring regions with a power grid covering 10 490km of medium voltage lines and 11 120km of low voltage lines. Their distribution network comprises of backbones of uniform sections that expand radially. Therefore, the distribution network has the ability to counter-supply if a branch is out of order. These backbones are each fed by two distinct primary stations. Also, through the use of breakers, each half-line is protected against faults.

The dataset originally introduced in (Bianchi, De Santis, et al., 2015), consists of a time series of the amount of electricity supplied and is measured on a medium voltage feeder from the distribution network in Rome. The data was collected from 2009 to 2011 with a measurement taken every 10 minutes for 954 days of activity, resulting in 137,444 measurements. The models will be trained to predict the electricity load 24 hours ahead; this means predicting 144-time steps ahead. As for exogenous time series, no exogenous time series is provided; however, the electricity load 24 hours in the past is used. The dataset is split into three parts: the training part is chosen to be the first three months, the 4th month is used as a validation/recalibration set, and the 5th month for testing the validity and accuracy of the final model.

According to (Bianchi, Maiorino, Kampffmeyer, Rizzi, & Jenssen, 2017) "*In the ACEA time series there are no missing values, but 742 measurements (which represent 0.54% of the whole dataset) are corrupted*". The corrupted values are replaced by fitting a cubic spline to the dataset and then replacing the corrupted entries with the corresponding values from the fitted spline. In doing so, the imputation should factor in the local variations of the load better (Bianchi, Maiorino, et al., 2017).

### 14.1.1 Data Preprocessing

Table 2: Descriptive statistics of the load profile in kiloVolts (KV) of the electricity consumption.

| Descriptive Statistics for ACEA load | | | |
|---|---|---|---|
| | Training | Recalibrating | Testing |
| Length | 12960 | 4320 | 4320 |
| Mean | 54.3 | 55.8 | 61.1 |
| Std | 14.5 | 16.1 | 17.4 |
| Min | 17.3 | 25.2 | 19.1 |
| Max | 109.1 | 96.1 | 102.2 |

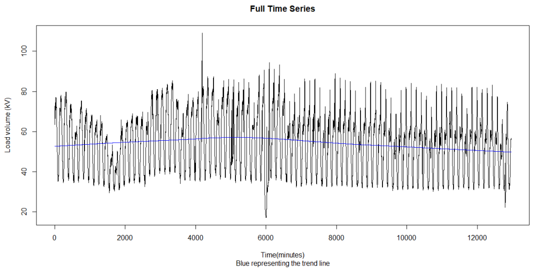


Figure 18: Electricity load over the entire time period.

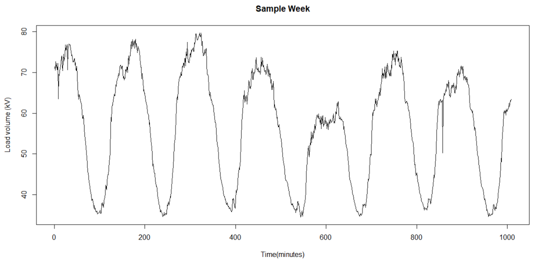


Figure 19: Electricity load over week 1 of the time period.

To identify trends or seasonalities, one can look at the plotted time series such as figure 18. Here there is some indication of a weak trend that seems to dissipate midway into the time series. However, by looking at figure 19, one can clearly see a seasonality emerging every 144-time intervals. This is expected as 144-time intervals correspond to precisely a day, and electricity consumption naturally follows a daily cycle. By analyzing the autocorrelation function and the partial autocorrelation function, one can further identify trends or seasonalities.

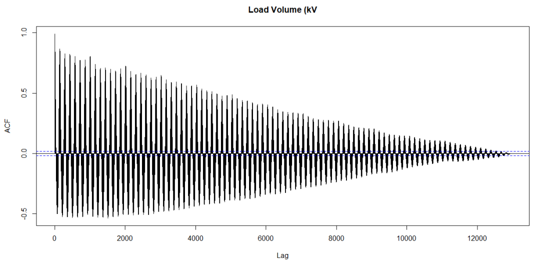


Figure 20: Autocorrelation function up to the maximum lag.

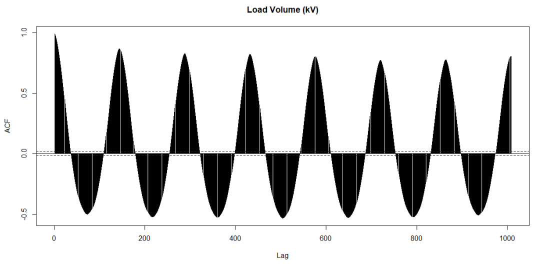


Figure 21: Autocorrelation function up to lag 1008 (a weeks worth).

The ACF plots confirm the findings from plotting the time series; there is clear seasonality every 144-time steps by looking at the weekly ACF from figure 21. The seasonality can be dealt with by differencing the time series at lag 144, which also removes the trend.

The partial autocorrelation function indicates which correlations are indirect and which are direct. For example, figure 22 shows a few statistically significant correlations; they are however, relatively small, barring the one at lag 1 and a few around lag 144 .

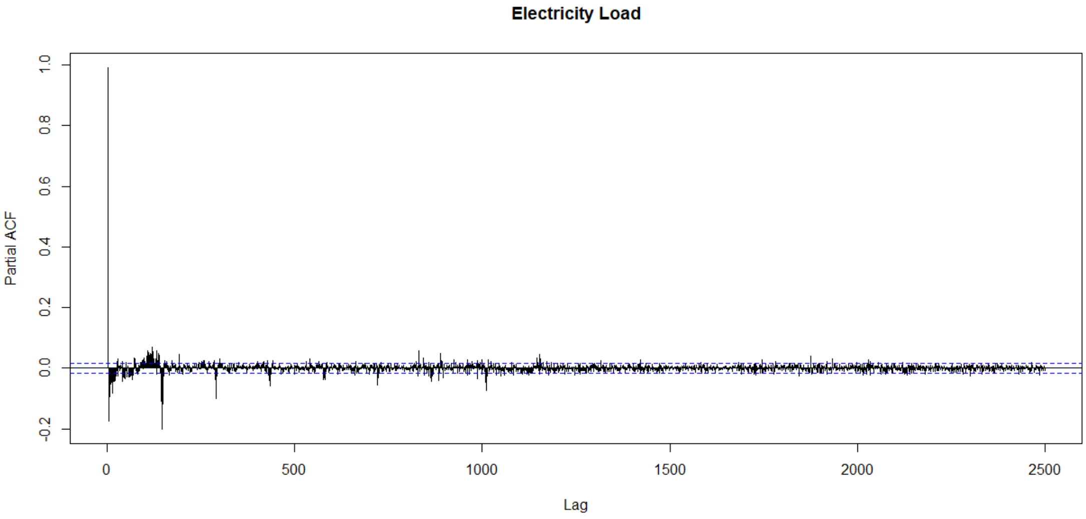


Figure 22: Partial autocorrelation function for the electricity load.

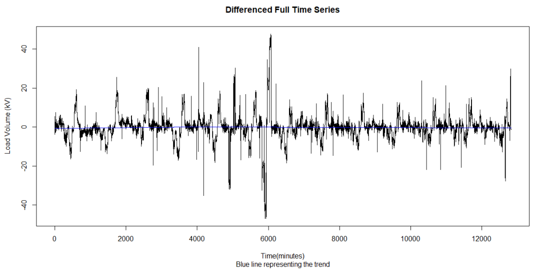


Figure 23: Differenced at time lag 144 electricity load plot.

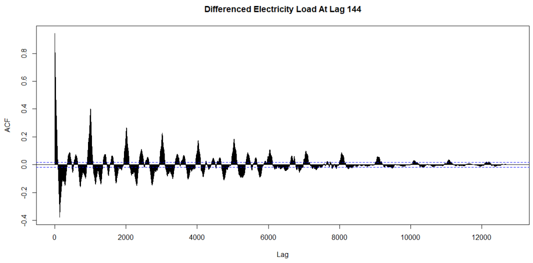


Figure 24: Differenced at time lag 144 ACF plot.

The differenced electricity load plot now shows no trend, and there is still some seasonality. Notably, the 1008 time steps frame shows some seasonality; this corresponds to a weekly seasonality. This can be removed with a second differentiation. However, because of the long periodicity of the time cycle, the models would require large amounts of memory to store information for a longer time interval (Bianchi, Kampffmeyer, Maiorino, & Jenssen, 2017). Another thing to consider with a second differentiation is that the models would have to be trained on the load residuals on the same day and time for two consecutive weeks. That is why only the differentiation at lag 144 is applied.

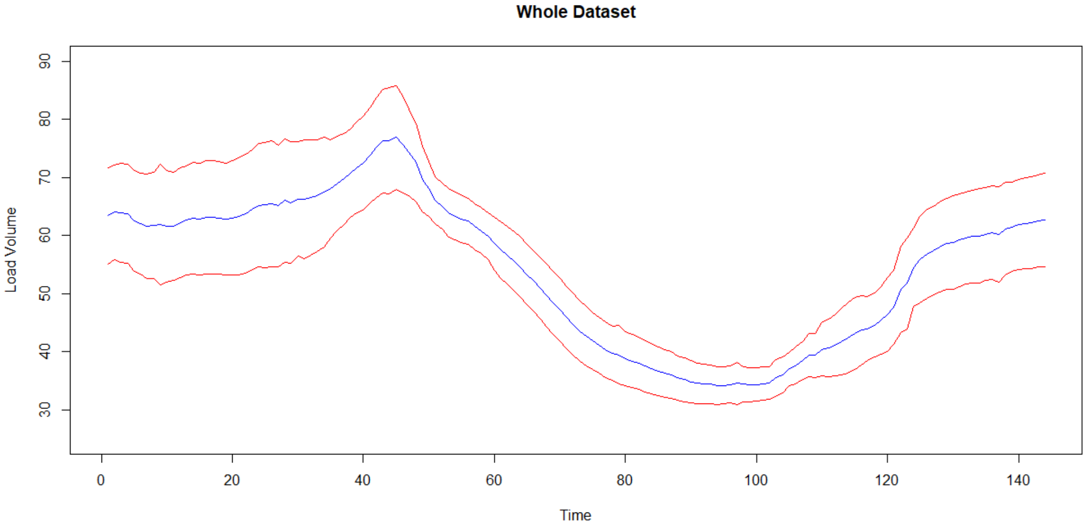


Figure 25: Showing the mean load (blue line) and the mean ± standard deviation (red line).

For studying the variance, the average daily load for the 144-time intervals has been calculated across the whole dataset, and the standard deviation for each time interval. As the standard deviation is relatively small in this case, a non-linear transformation for stabilizing the variance might not be necessary. Therefore, we conclude that standardization is an adequate procedure for normalizing the data in this case. The standardization step is done to limit the oversaturation as the activation function used in the ESN model reaches its upper and lower limits at around ± 2.

In summary, the data preprocessing applied are standardization and seasonal differentiation at lag 144. These transformations are reversed after computing the forecasts to obtain predictions on the correct scale.

## 14.2 GEFCom2012 - Bivariate

The second dataset that is considered is the time series of electricity consumption from the Global Energy Forecasting Competition 2012 (GEFCom2012) (Silva, 2014). The dataset covers one year during 2006 of electricity load from a US energy supplier measured hourly; in addition, the dataset includes a time series of the temperature in the same area as where the electricity consumption is measured. The electricity load varies from 10 000kWh to

200 000kWh and is represented as the average hourly load. The GEFCom 2012 dataset is measured from 20 different feeders in the same geographical area and then aggregated.

As with the ACEA dataset, the GEFCom 2012 is divided into three parts. The training set covers the first ten months, and the $11^{th}$ month is used as validation/recalibration set while the $12^{th}$ month is used for testing the accuracy of the different models. The time horizon for the forecasts is 24 hours ahead of the aggregated time series. The temperature time series is used as an exogenous input in this case.

### 14.2.1 Data Preprocessing

Table 3: Descriptive statistics of the load profile in kilowatt hours (kWh) of the electricity consumption.

| Descriptive Statistics for GEFCom 2012 load | | | |
|---|---|---|---|
| | Training | Recalibrating | Testing |
| Length | 7272 | 720 | 720 |
| Mean | 1037412 | 1071306 | 1086948 |
| Std | 351148.8 | 337812.8 | 202733 |
| Min | 519507 | 561629 | 528534 |
| Max | 2942993 | 1966244 | 1762396 |

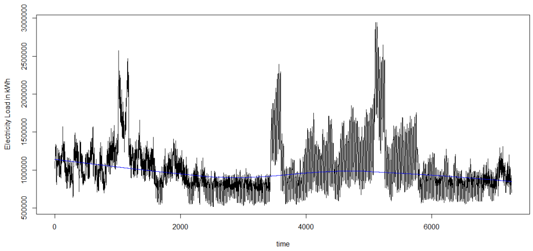


Figure 26: Aggregate electricity load profile GEFCom 2012 where the blue line represents the trend.

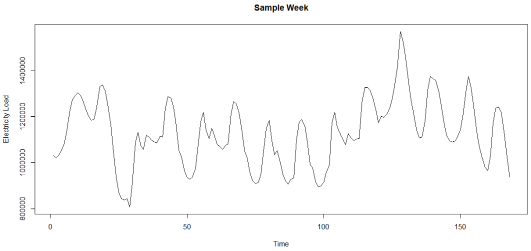


Figure 27: Aggregate electricity load the first week of the GEFCom 2012 dataset.

By plotting the time series over the first ten months, we can see a clear trend depending on whether it is summer or winter, see figure 26. As expected, the electricity consumption is higher during the winter due to increased household heating demands. In addition, it is also possible to observe higher energy consumption during the day than at night. This effect becomes more apparent by looking at a single week in the time series like in figure 27 instead.

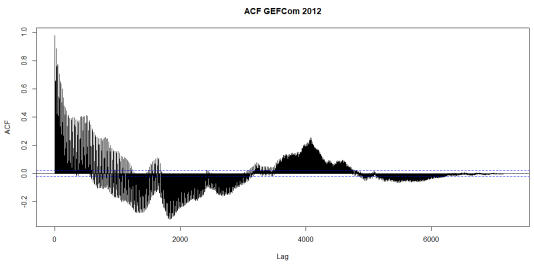


Figure 28: Autocorrelation function up to the maximum lag.

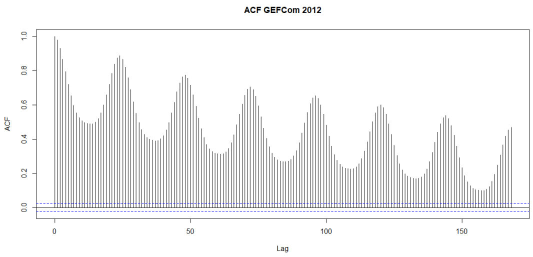


Figure 29: Autocorrelation function up to lag 168 (a week).

To study the seasonality of the dataset, the ACF for the first ten months as well as the first week, see figure 28 and 29. Here the 24-hour seasonality is quite apparent, especially in figure 29. To remove the daily seasonality, differencing at lag 24 is performed. Such a differentiation produced the time series in figure 30 and the corresponding autocorrelation function plot in figure 31.

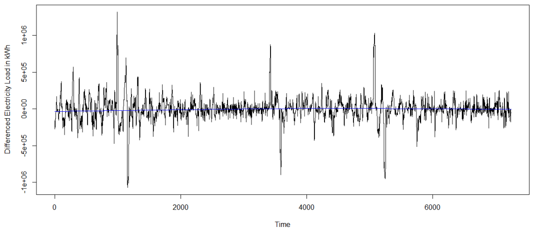


Figure 30: Differenced electricity load at lag 24.

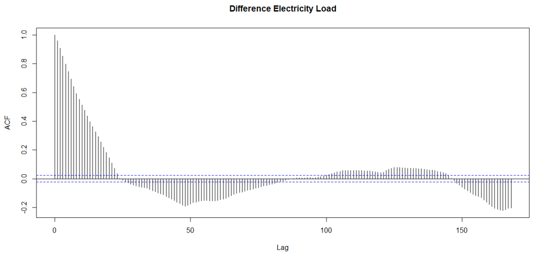


Figure 31: Autocorrelation function of differenced electricity load at lag 24.

The partial autocorrelation in figure 32 shows two large correlations followed by minor but still statistically significant correlations. Larger lags also show two statistically significant correlations every 24 lags, which amounts to 1 day.

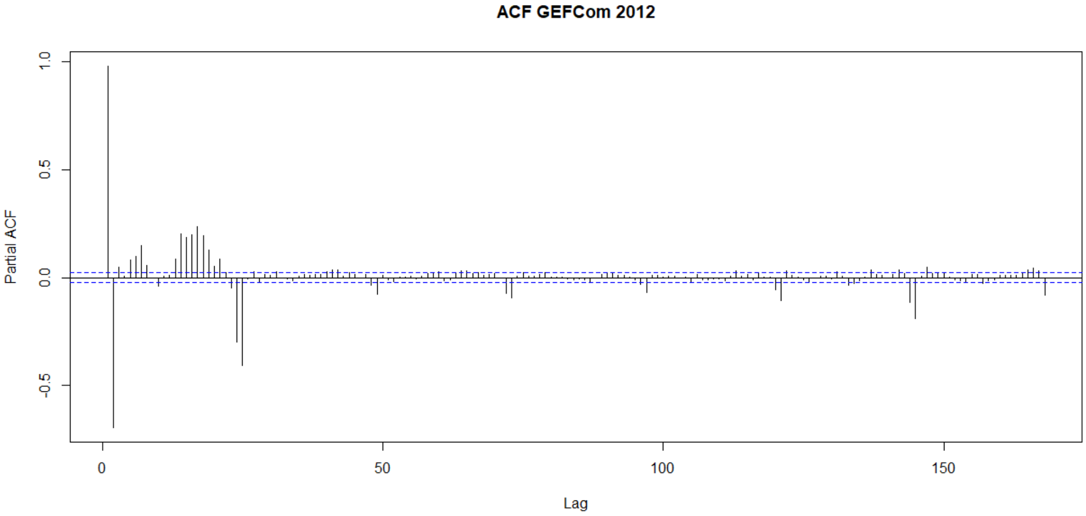


Figure 32: Partial autocorrelation function for the electricity load.

As with the ACEA dataset, in order to study the variance, the mean and standard deviation for the aggregate load of each hour in a day have been calculated and plotted in figure 33. The standard deviation appears to vary more later during the day when consumption typically is higher. The GEFCom 2012 dataset is normalized similarly to the ACEA dataset, i.e., with standardization. Standardization is done to prevent oversaturation as the activation function for the ESN model, tanh(x), relies on small values as the upper and lower limits are reached at around ± 2.

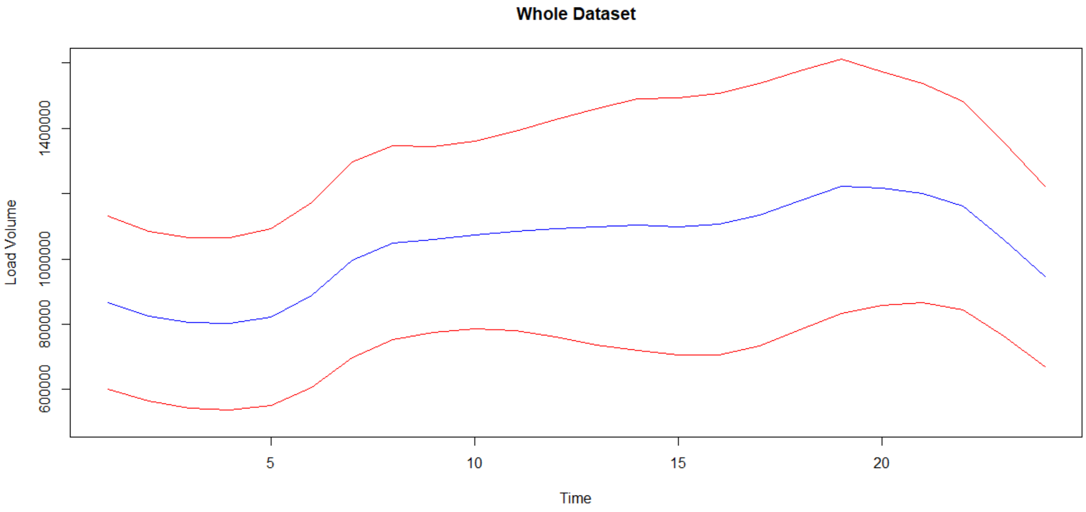


Figure 33: Showing the mean load (blue line) and the mean ± the standard deviation (red line) for each hour in a day.

The main difference between GEFCom 2012 compared to ACEA is the presence of an exogenous time series. As stated earlier, the exogenous time series is the temperature, as the electricity consumption rises with the use of household heating or cooling in the colder or warmer months, respectively. However, the relationship between electricity consumption and temperature can not be captured by linear correlation as electricity consumption rises with both increases and decreases in temperature. Indeed, the correlation between the two time series is only 0.2. The relationship, however, is stronger than what the correlation would indicate. Figure 34 makes this evident as the V-shape denotes an increase in electricity consumption when the temperature rises or falls from the mean of $62^o F$.

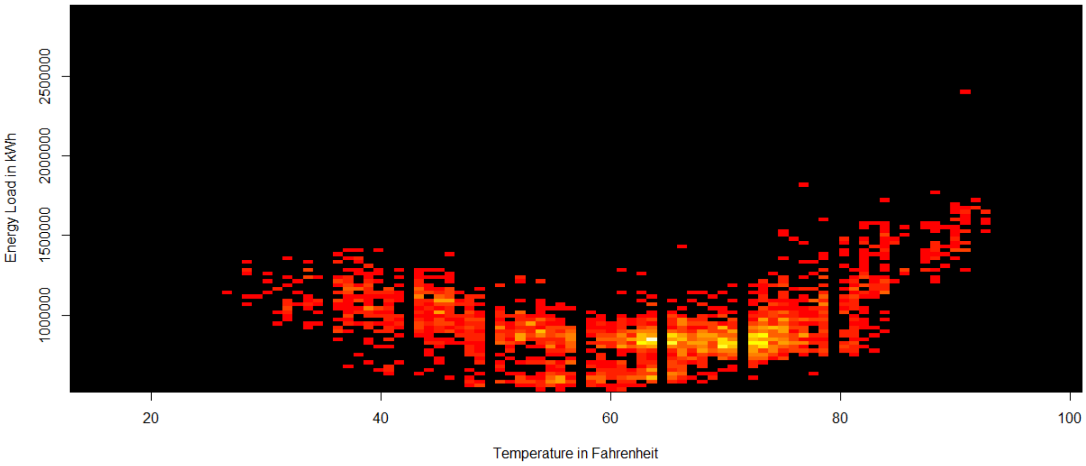


Figure 34: 2-dimensional histogram of the temperature and the aggregated electricity consumption where brighter areas indicate more populated bins.

To summarize, the preprocessing steps taken for GEFCom2012 are seasonal differentiation at lag 24 and standardization. After the forecasts are obtained, the transformations are reversed to map the predictions to the correct scale.

## 14.3 Fakken Wind Power production - Multivariate

The last dataset concerns the power production of a wind farm in the northern part of Norway. This wind farm is called "Fakken" and consists of 18 turbines, each capable of producing up to 3 MW under the right circumstances. This gives the farm a maximum power output of 54 MW. The area in which Fakken is situated upon has a complex topography, which has a significant impact on each of the turbines' power production (Eikeland et al., 2022). The altitude in this area varies from 0 to more than 1000 meters above sea level (MASL) as shown by figure 35.

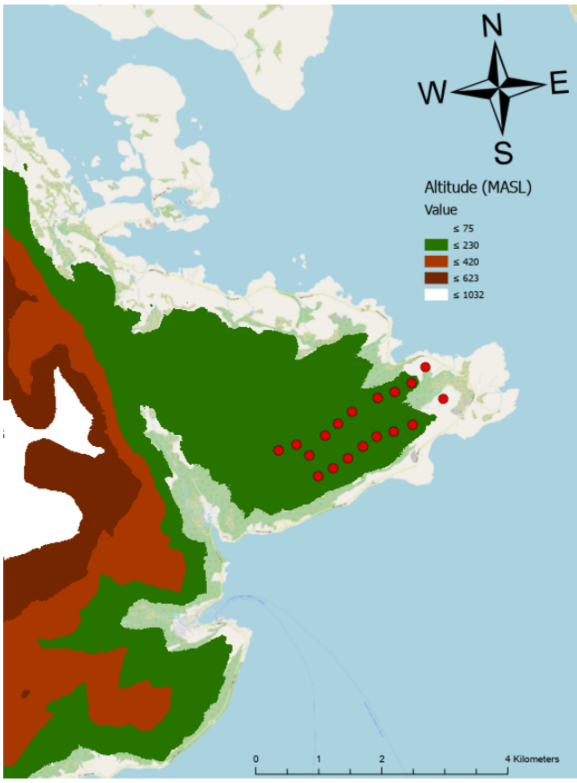


Figure 35: Altitude map to show the topology of the region. The red circles are the 18 wind turbines in the farm. The colors are for each altitude level, with green being the most predominant ranging from 75 to 230 MASL. Source: (Eikeland et al., 2022).

Due to rapidly changing altitude, the wind turbines can experience vastly different wind speeds, and, according to the owner of Fakken, the leftmost turbine produced just 75% of the power that the rightmost produced during 2020 (Eikeland et al., 2022). The main reason for this difference is the number of obstacles in the wind's path. The turbines on the left are somewhat protected from the wind by nearby large mountains, while the ones on the right are closer to the ocean with less protection. The result of this can be increased difficulty in making accurate predictions since the aggregate power production vary more than comparable wind farms in flat areas as the weather conditions will be more uniform for the farm as a whole. For this reason, predictions will be made for individual turbine power production instead of aggregating the power production.

The dataset spans 2021, yielding 8,784 power production observations for each of the turbines. In addition to the power production, the wind speed and direction measurements were taken with the same temporal resolution as the power for each of the 18 turbines. In particular, the measurements are taken from weather stations mounted on each turbine at 1-hour resolution by the Troms Kraft Power company. Weather forecasts are also used to provide further accuracy in predicting, rather than just historical power and wind. These weather predictions come from a numerical weather prediction (NWP) model called the AROME-Arctic model, which was created by the meteorological institute of Norway (MET) [3]. The weather predictions made by AROME-Arctic come in the form of hourly forecasts with a 2.5 km spatial resolution. As with the wind measurements, the AROME-Arctic predictions come as wind speed and wind direction. However, contrary to the

[3] https://www.met.no/en/projects/The-weather-model-AROME-Arctic

weather measurements that are specific to each wind turbine, the predictions are split into two different cells, one containing 12 turbines while the other contains the remaining 6, see figure 36.

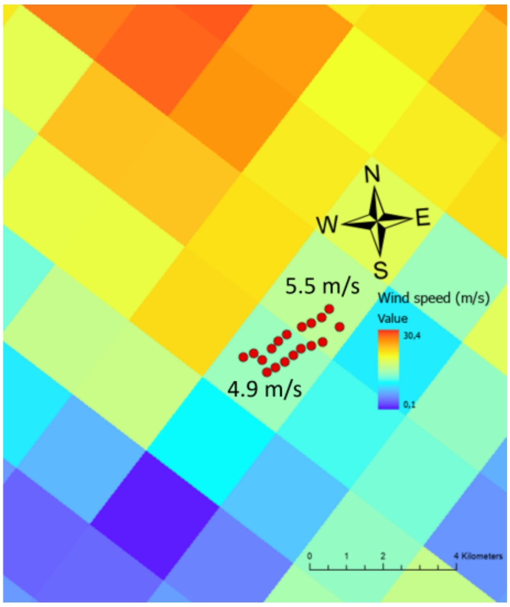


Figure 36: Weather simulation map from the AROME-Arctic model with each cell representing a 2.5 $km^2$ area and a red dot for each turbine. Source: (Eikeland et al., 2022).

The AROME-Arctic map highlights the differing wind speeds that occur in the region. It ranges from upwards of 30 m/s to zero in just a few kilometers. In addition to the actual measurements collected by the weather stations, these weather predictions should aid in the accuracy of the power generation forecasts.

As a final note about the turbines, the cut-in and cut-off wind speeds for the turbines are around 4 m/s and 25 m/s, respectively. Power production starts at this cut-in speed and continually climbs until the wind speed reaches about 12-13 m/s; at this wind speed, the power production is at its peak of 3 MW. On the other hand, if the wind speed increases to the cut-off speed of around 25 m/s, the turbine will stop, as can be seen in figure 37. This is due to a safety mechanism that prevents the wind turbine from rotating too fast as it can get damaged (Eikeland et al., 2022).

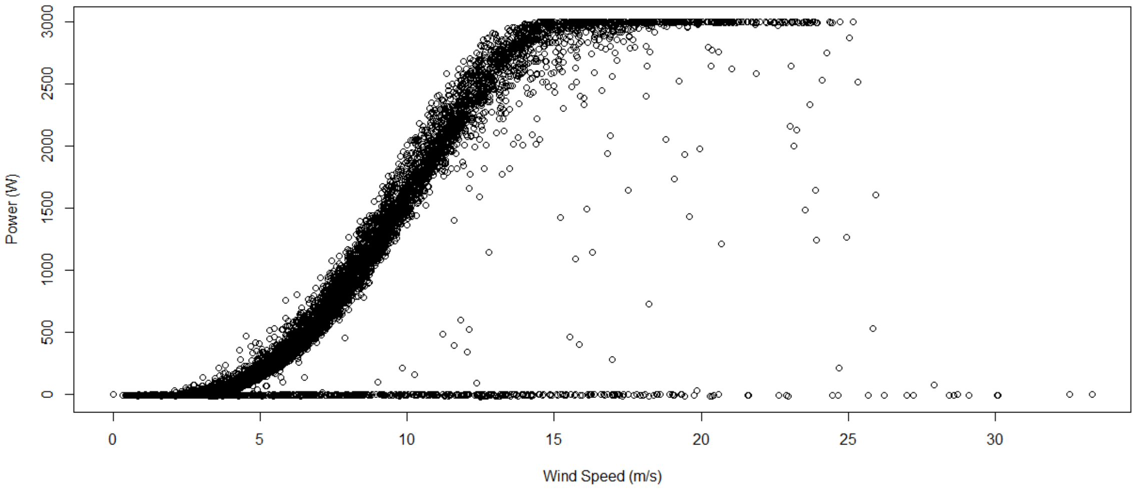


Figure 37: Plot showing the power curve from one of the turbines.

This dataset is split into three parts, like the ACEA and GEFCom 2012 datasets. A training part, a validation/recalibration part, and a testing part. The training part covers the first ten months, while validation/recalibration and testing cover the $11^{th}$ and $12^{th}$, respectively. The forecasting horizon is set to 36 hours ahead; the reason is that the electricity market requires participants to submit their final bids by 12:00 regarding the expected amount of power generation the next day, therefore the 36 hours (24 hours + 12 hours). As for exogenous time series, historical power, wind speed, wind direction, and predicted wind speed and wind direction are used. Several different subsets of these covariates will be considered in section 14.3.1.

### 14.3.1 Data Preprocessing

Table 4: Descriptive statistics for the power production of one of the 18 turbines.

| Descriptive Statistics for power production in kW | | | |
|---|---|---|---|
| | Training | Recalibrating | Testing |
| Length | 7320 | 720 | 744 |
| Mean | 830.68 | 1226.8 | 1075 |
| Std | 988.3 | 1030.5 | 1023.2 |
| Min | -23.7 | -14.26 | -15,9 |
| Max | 3000.9 | 3000.2 | 3000.6 |

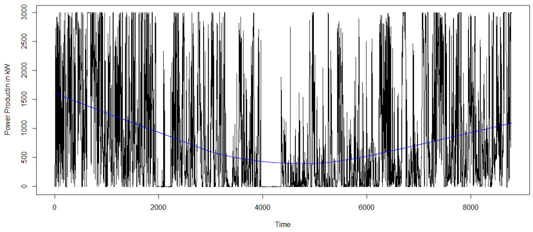


Figure 38: Power production of one of the turbines with the blue line representing the trend in kW.

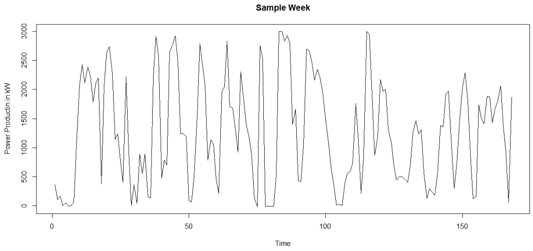


Figure 39: Power production of the same turbine as in figure 38, but for a week instead in kW.

In figure 38 it is quite clear the erratic nature of the power production, at least on that time scale. With figure 39 it becomes a bit clearer; the power production varies wildly in just a few time steps because of the non-linear dependence with the wind speed.

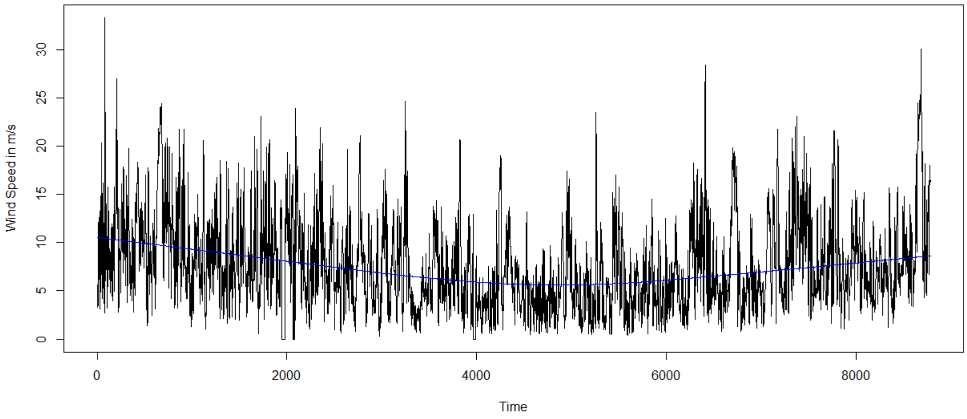


Figure 40: Wind speed plot for the same turbine as in figure 38, the blue line representing the trend.

There is a trend towards less power production during the summer months. This is due to less windy conditions during the summer, as can be seen in figure 40. The reason for the continuous lack of power production that lasts approximately 300-time steps at around time step 4000 is unclear as the wind rarely exceeds the 25 m/s threshold in that period. A reason could be that the turbines are shut down for maintenance; however, this is just speculation.

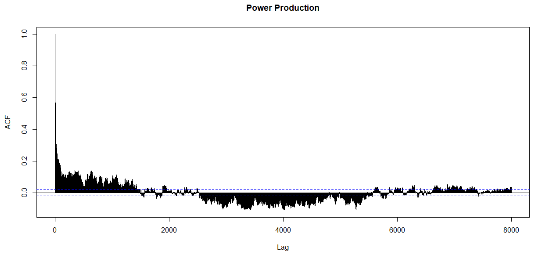


Figure 41: Autocorrelation function up to the maximum lag.

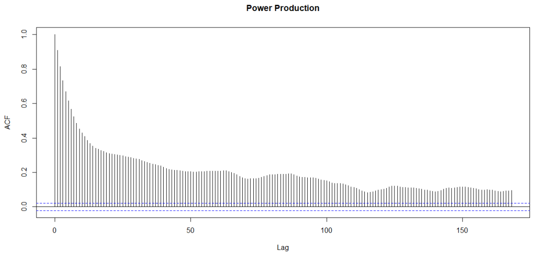


Figure 42: Autocorrelation function up to lag 168 (a week).

The autocorrelation functions in figure 41 and 42 show an abundance of short-term linear dependencies; however there seems to be a lack of seasonality in the time series. This is confirmed in figure 43.

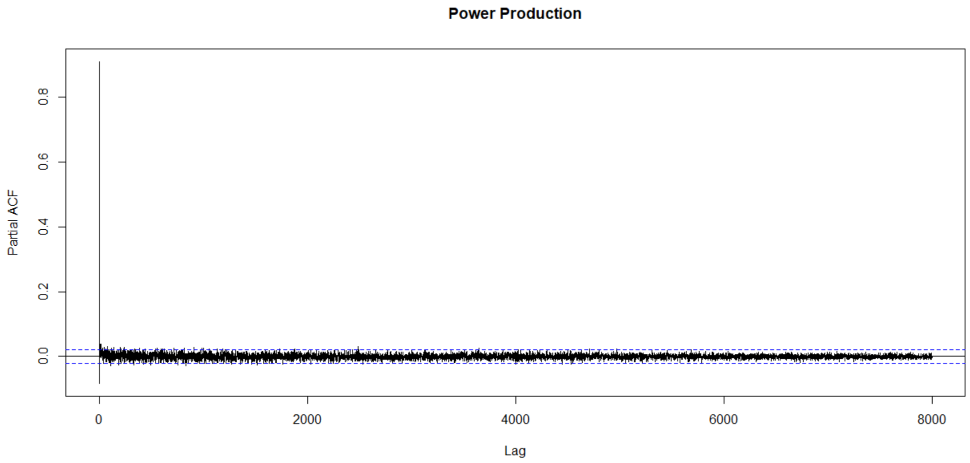


Figure 43: Partial autocorrelation function with the blue lines representing the upper and lower limits of a 95% confidence interval.

As only the first two lags are outside the confidence interval in figure 43 it is safe to say the correlations in figure 41 are indirect correlations that can be explained by the first two time steps. These sorts of dependencies are also exhibited in the wind speed time series.

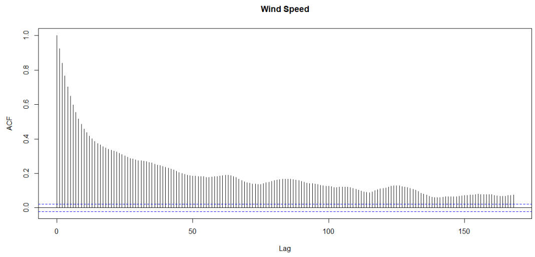


Figure 44: Autocorrelation function.

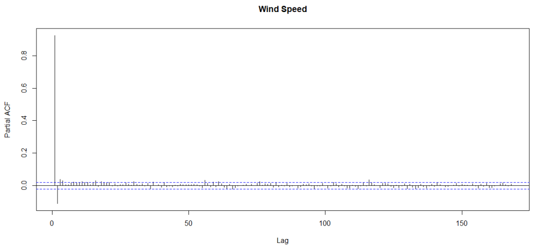


Figure 45: Partial autocorrelation function.

The autocorrelation function and the partial autocorrelation for the wind speed as depicted in figure 44 and 45 closely resemble the ones for the power production. This is not surprising as the correlation between the two is relatively high (0.85). Obviously, this makes sense as the wind speed directly dictates the power production, and the measurements are taken at the specific wind turbine.

To remove the trend shown in figure 38 differentiation can be applied like in the other datasets; however, in this case, it is sufficient to do it at lag 1. The same procedure is applied to the wind speed time series. The result of this are figure 46 and 47 showing a flat trend line.

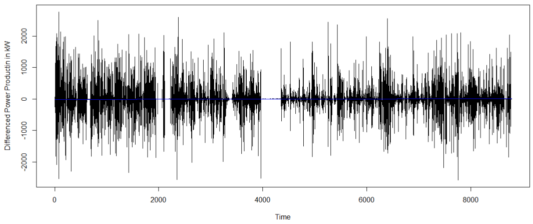


Figure 46: Differenced power production time series with the blue line representing the trend.

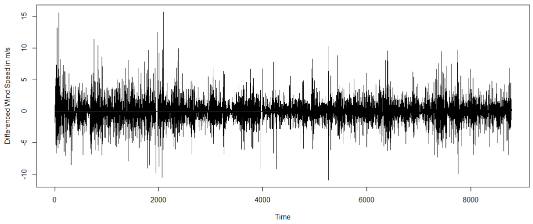


Figure 47: Differenced wind speed time series with the blue line representing the trend.

The resulting ACF for the power production and wind speed show little correlation after lag 1.

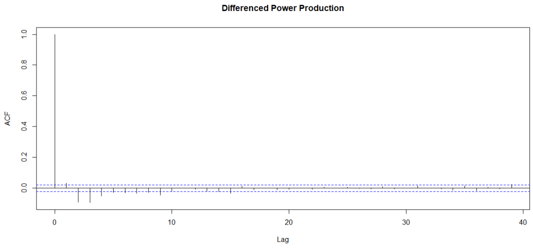


Figure 48: Differenced autocorrelation function for the power production time series.

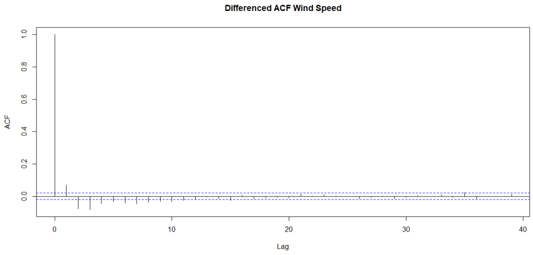


Figure 49: Differenced autocorrelation function for the wind speed time series.

As the measurement units in the wind direction time series are degrees, the time series is rapidly changing with no fundamental changes in direction at 360 and 0. In other words, if the direction changes between 350 and 10, it is a 340 degrees change while being only 20 in reality. The nature of this problem can be seen in figure 50.

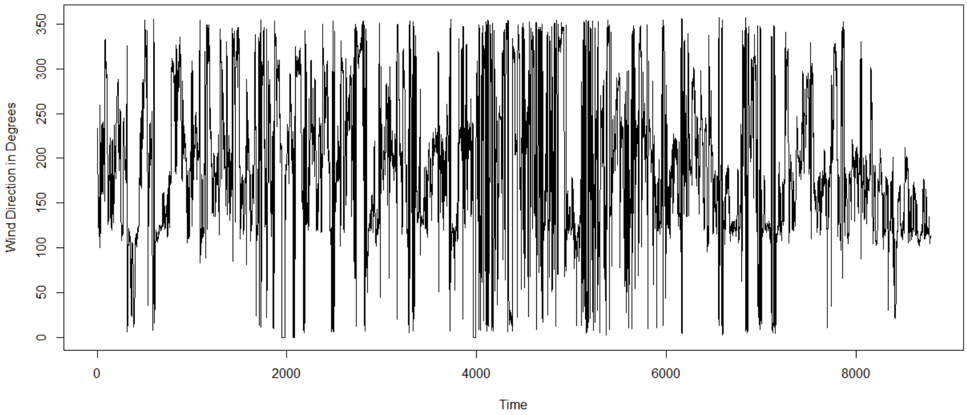


Figure 50: Plot of the wind direction time series showing the rapidly changing direction in degrees.

Steps to remedy this can be made. The procedure chosen here transforms the wind speed and direction from polar coordinates to Cartesian coordinates. The time series for the X and Y component then becomes as shown in figure 51 and 52 respectively.

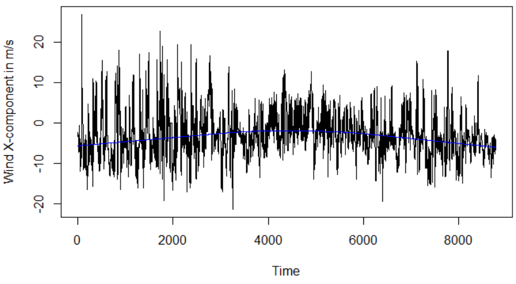


Figure 51: Plot showing the X-component of the wind time series.

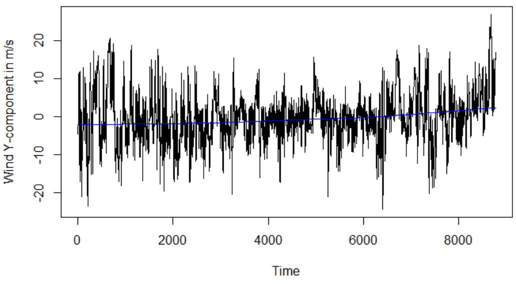


Figure 52: Plot showing the Y-component of the wind time series.

The accompanying autocorrelation functions and partial autocorrelation functions then become as shown in figure 53, 54, 55 and 56.

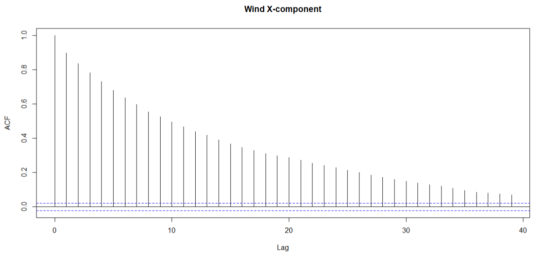


Figure 53: ACF for the X-component of the wind time series.

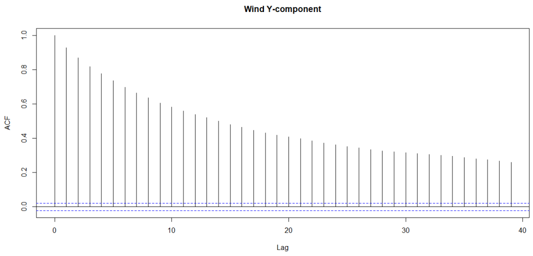


Figure 54: ACF for Y-component of the wind time series.

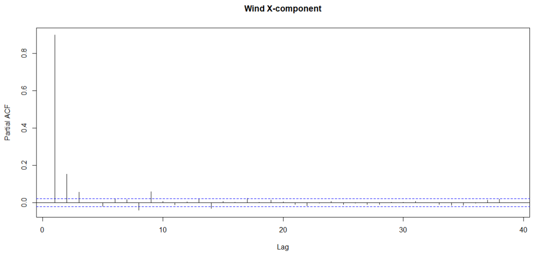


Figure 55: PACF for X-component of the wind time series.

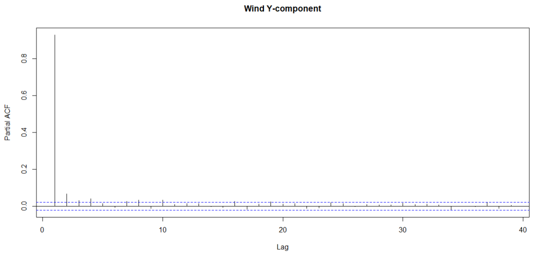
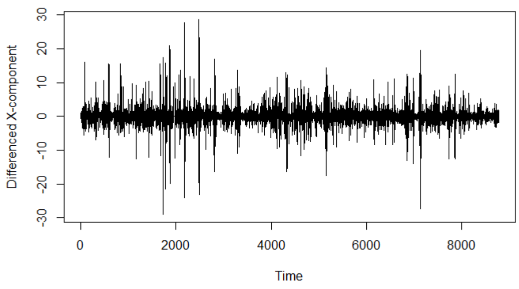


Figure 56: PACF for the Y-component of the wind time series.

The partial autocorrelation functions and autocorrelation functions resemble the ones for the wind speed time series, which makes sense. There is a clear lack of seasonality and the correlations in figure 53 and 54 can be explained as indirect correlations as they are gone in the partial autocorrelation functions in figure 55 and 56.

These time series can also be differentiated at lag 1 to remove the correlations in figure 53 and 54 and the trends shown in figure 51 and 52. With this differentiation done, the plot of the differenced wind components become as shown in figure 57 and 58.



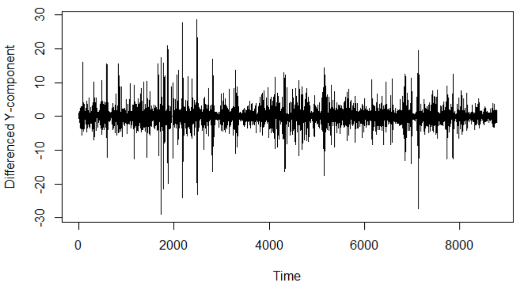
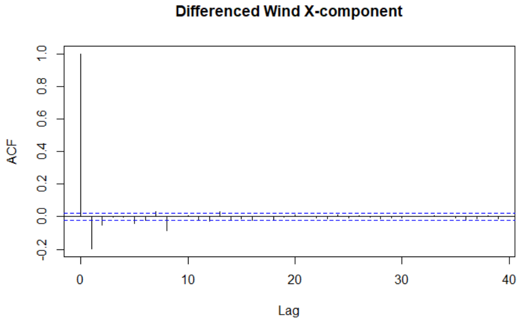


Figure 57: Plot showing the differenced X-component of the wind time series.

Figure 58: Plot showing the differenced Y-component of the wind time series.

Accompanying the time series plots are the PACF and ACF plots, which show no significant correlations after lag 2 in figure 59 and 60.



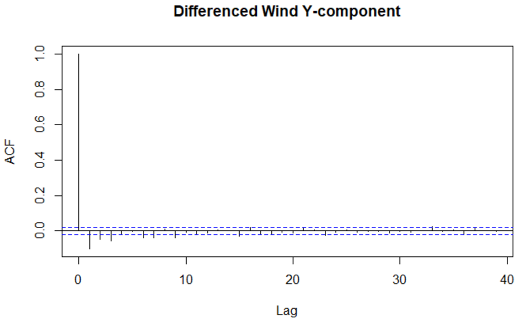
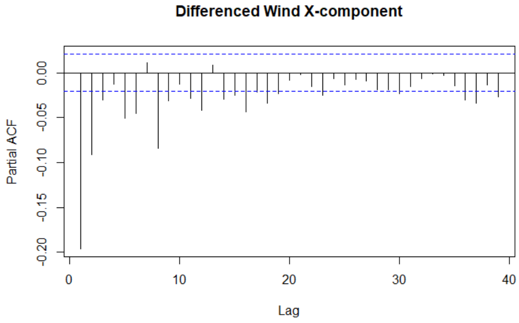


Figure 59: ACF for the differenced X-component of the wind time series.

Figure 60: ACF for the differenced Y-component of the wind time series.

The PACF plots shown in figure 61 and 62 show some statistically significant correlation after the first few lags.



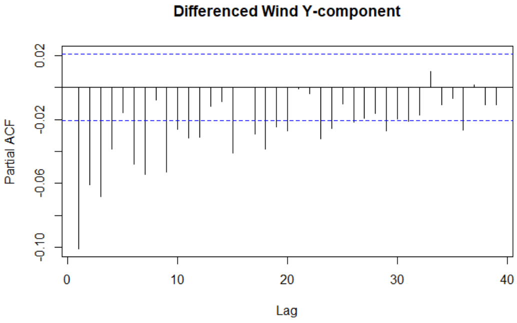


Figure 61: PACF for the differenced X-component of the wind time series.

Figure 62: PACF for the differenced Y-component of the wind time series.

As a final note to the wind direction and speed, the same procedures have been applied to the predicted wind direction and speed from the AROME-Arctic model.

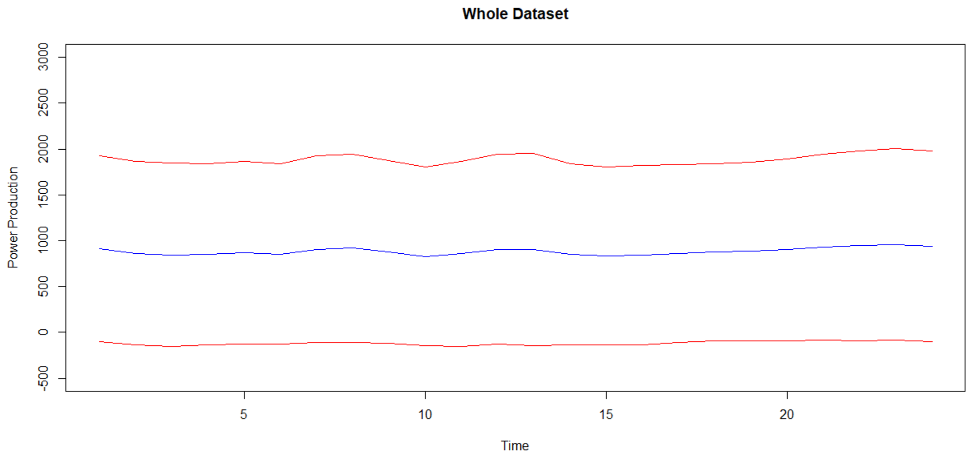


Figure 63: Plot showing the mean (blue) and mean ± standard deviation (red) for the power production in kW.

As the time of day does not generally impact the wind speeds, the mean is relatively stagnant throughout the day, i.e., the mean is a rather straight line as seen in blue in figure 63 around the 1000 kW mark. There is, however, quite a large amount of variation as just one standard deviation covers two-thirds of the possible power production spectrum.

The final preprocessing step taken to these time series is standardization. This is mainly done to prevent oversaturation in the ESN as the activation function tanh(x) reaches its limits shortly after ±2. This makes the ESN not sensitive to differences in lower and larger values.

To summarize, the preprocessing steps taken are differentiation at time lag 1, decomposing the wind speed and direction into X and Y components, and standardization. Section 15 will cover different combinations of exogenous variables such as decomposed wind, historical wind and power, and predicted wind as inputs to achieve the best model to apply the proposed method. The transformations will be reversed when making the final predictions with the models. Although only one turbine has been shown in this section, the analysis of the time series for the 17 other turbines produced similar results.

# 15 Model Setup

The models presented in this section will each construct a 90% prediction interval, meaning $\alpha = 0.1$. The time horizon for the prediction intervals will be 24 hours for all the datasets with the exclusion of the Fakken dataset, where it is 36 hours as explained in section 14.3.

Sections 15.1 and 15.2 will cover the parameter configurations for each model. Firstly the implementation and configuration of the ARIMAX models will be presented; this is necessary to have a baseline to compare the ESN models to. After that, the ESN hyperparameter configuration will be covered as well as the hyperparameter choice in the Gibbs sampling step of the proposed method.

## 15.1 ARIMAX Models

The ARMIAX models are implemented using the auto.arima function from the forecast package in R[4]. The auto.arima function fits an ARIMA model using the training data provided. It is of the class ”ARIMA,” which allows the use of the forecast function from the same package to predict a specified number of out-of-sample forecasts. For the multivariate datasets, the historical data and the exogenous variables are presented to the model. In regards to choosing the model orders, two approaches are possible. The first is the standard of determining it by analyzing the ACF and PACF; however, the auto.arima function searches to choose the model that minimizes the Akaike information criterion and thus might be preferential in situations where it is hard to determine the orders of the model.

As touched upon in section 5 the uncertainty in prediction obtained from multi-step forecasting increases as the forecasting horizon increases, and thus the prediction interval width typically widens. This is because multi-step forecasting treats the previous forecast as actual observations when making the following prediction. The result of this is often an accumulation of errors that further widen the prediction interval.

This thesis focuses on multi-step forecasting on a 24-hour time horizon for the ACEA and GEFCom 2012 datasets and 36 hours for the Fakken dataset. These time series, as stated earlier, are divided into training, recalibration, and testing, where the recalibration set is also used as a validation set for hyperparameter optimization of the ESN models. Contrary to this hyperparameter optimization, the ARIMA models do not use cross-validation as they only use the ACF, PACF, and AIC to select the orders. Therefore, in order for the ARIMA models to have similar prediction setups, the recalibration set was not used for these models.

One thing to note, there should ideally be fitted a new ARIMA model after each forecasting horizon (Jensen, 2021), i.e., a new one after 24 or 36 hours, depending on the dataset. This is to obtain optimal results but is very much infeasible due to the length of the datasets as well as the sheer number of forecasting horizons.

In comparison with the ESN models, the ARIMA models are deterministic and, therefore, only need the results from a single run for each model. Table 5 shows the orders for each of the models. One thing of note here is that the Fakken model is made with historical power production, wind speed, direction, and the X and Y components, in addition to the NWP data that also contains the wind speed and direction with its decomposed elements as the exogenous variables. As can be noticed in table 5, the AIC is significantly higher for the Fakken data, and this is reflected in our predictions in section 17.1.3.

Table 5: ARIMA model parameters, (p,d,q).

| ARIMA Models | | | |
|---|---|---|---|
| | ACEA | GEFCom 2012 | Fakken |
| Model | ARIMAX(2,0,3) | ARIMAX(4,0,5) | ARIMAX(2,1,2) |
| AIC | -16981.53 | -10729.64 | 7441.73 |

[4] https://cran.r-project.org/web/packages/forecast/index.html

## 15.2 Echo State Network Model

The ESN-based models are initialized using the rESN package [5] which is then modified to include the parameter input scaling. Modifications were also made to extract the reservoir states to perform the Bayesian regression. As stated in section 13.3, the hyperparameter search is carried out by random search with ten initialization per configuration to find an approximately ideal configuration to apply the proposed method. The parameters in question are the number of units, spectral radius, the density of non-zero connections, and input scaling.

### 15.2.1 Choice Of Exogenous Variables For Fakken Data

In regards to choosing a good combination of exogenous variables such as the temperature in the GEFCom 2012 dataset or wind speed in the Fakken dataset, the mean squared prediction error (MSPE) is used. The goal is to choose the combination that produces the best predictions. It is much quicker to do as we can train using Ridge regression instead of performing the Bayesian regression part of the proposed algorithm. Ridge regression foregoes the sampling, which is comparatively more computationally expensive, thus making it quicker to make these comparisons. The data used for these comparisons is the recalibration part, just like the hyperparameter optimization. This is done to choose a model and model configuration that best fit the prediction task while also not being too specific for the testing part, as that is kept separate until the final predictions. Also, the MSPE is calculated using standardized data.

When choosing the combination of exogenous variables, one run of each is not enough as the predictions can vary wildly between each run, even with the same configuration and exogenous variables. This is due to the fact that each ESN model is randomly initialized. To combat this, each combination is initialized 200 times, and the mean MSPE for each combination is calculated. The different combinations are chosen, totaling 12, divided into two groups, 6 using the differenced time series and 6 using the regular time series. Do note that the MSPE for the differenced models is calculated after reverting the differencing on the predictions.

The configuration is kept constant for all the combinations and every initialization. This is to provide a baseline MSPE and isolate the effect of the combinations. Due to this, the combination chosen might be the right one based on the testing, but in reality, it may not be the best one for this task. Table 6 shows the configuration chosen for choosing the combination.

Table 6: Network configuration used for finding the combination of exogenous variables.

| Network configuration | |
|---|---|
| Number of units | 500 |
| Spectral radius | 0.85 |
| Regularization | 1 |
| Input scaling | 1 |
| Density | 0.2 |

[5] https://github.com/jaredhuling/rESN

The different exogenous variables are termed as such:

- Predicted Wind (PW), for the numerical wind speed and direction prediction from AROME-Arctic.
- Historical Power (HP), the historical power production for the wind turbine.
- Historical Wind (HW), the historical wind speed and direction for the wind turbine.
- Predicted Decomposed Wind (PDW), the decomposed X and Y components of the numerical wind speed and direction prediction from AROME-Arctic in addition to the same wind speed prediction.
- Historical Decomposed Wind (HPDW), the decomposed X and Y components of historical wind speed and direction in addition to the historical wind speed.
- All Historical Power(AHP), the historical power production for all the 18 turbines.

When using historical power or wind, the time series lagged 36-time steps behind the time step we want to predict. In theory, we only have access to the historical wind or power up to 36 hours before making a prediction. To make it a bit clearer, see figure 64.

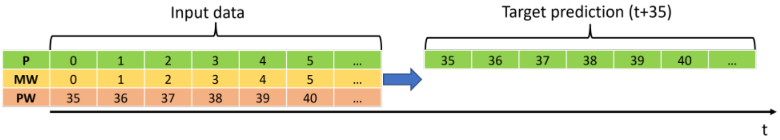


Figure 64: Showing how predicted and measured data are used in training and predicting with historical wind power generation (P), historically measured weather (MW) and predicted weather (PW). Source (Eikeland et al., 2022).

All these combinations give us the following MSPE as seen in table 7. In addition to calculating the MSPE of each combination, the standard deviation can also be calculated and help inform the correct choice of combination. A minor standard deviation is desirable as the proposed method does not include an ensemble. Thus, the variation with the same model configuration and exogenous variable has more impact on the final predictions.

Table 7: Mean squared prediction error for the 12 combinations where regular are before applying differentiation at lag 1 and differenced after differencing at lag 1.

| Mean Squared Prediction Error | | |
|---|---|---|
| | Regular | Differenced |
| PW | 1.01 (0.027) | 3.08 (0.32) |
| PW+HP | **0.83** (0.029) | 10.6 (3.47) |
| PDW | 0.88 (0.053) | 11.4 (3.6) |
| PDW+HP | 0.89 (0.046) | 3.2 (0.96) |
| PDW+AHP | 0.87 (0.063) | 10.2 (9.75) |
| PDW+HDW+HW+PW+HP | 0.86 (0.077) | 10.6 (6.65) |

By using table 7, we can see that the combination that gives us the lowest MSPE is the one with predicted wind speed and direction with the addition of historical power. The increase in MSPE while using the historical wind speed and direction and its derivatives might be the high correlation between it and the historical power, 0.85 to be exact. Also, why using the differenced time series increases the MSPE to such a large extent is unclear. This increase MSPE is also reflected in its standard deviation, which is significantly higher for the ones using differenced data as seen in table 7.

To further identify the optimal combination, we used the Friedman test to ascertain whether there is a difference between the combinations or if the difference in MSPE is purely due to chance. Here the null hypothesis is that MSPE does not differ between combination pairs, and the alternative hypothesis is that there is a difference in MSPE between the combination pairs. Figure 65 shows the p-value for each pair; most of the pairs differ significantly from each other, with the exception of the pairs PDW & PDW+AHP, PDW & PDW+HDW+HW+PW+HP, PDW+AHP & PDW+HDW+HW+PW+HP. The differenced pairs are not that interesting, but PW & PDW+HP, PW+HP & PDW+HDW+HW+PW+HP, and PDW & PDW+HDW+HW+PW+HP do not significantly differ. One note here is that all the regular pairs that do not differ use decomposed wind as an exogenous variable.

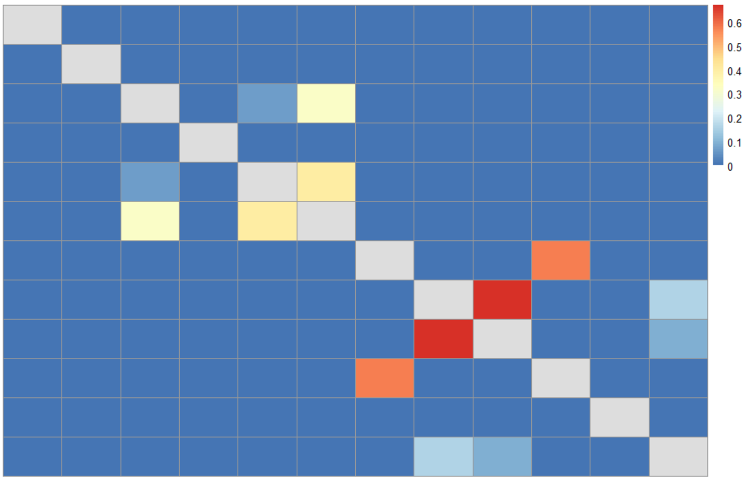


Figure 65: Heatmap for the p-values for each combination from the Friedman test, the gray diagonal being paired with itself. The order is going from regular combinations and then differenced combinations in the order from table 7.

The reasonable conclusion based on average MSPE, the standard deviation, in addition to being significantly different from any other combination, is that predicted wind speed and direction with historical power should be the combination used going forward. It is, however, essential to note that while this might be the best combination with the data available, it does not exclude the possibility that another combination is better for actually predicting. This is due to the MSPE being calculated using the recalibration part of the dataset and not the actual part we want to predict. Also, the model configuration is kept constant for all the models and every initialization. Therefore there might be a combination and configuration that outperforms this. In fact, it is safe to say that even the network configuration used for the PW+HP model is not ideal. For this reason, hyperparameter optimization is performed using the PW+HP combination to find an approximately ideal configuration for the model.

### 15.2.2 Hyperparameters

To achieve a good selection of hyperparameters for the ESN model, the algorithm laid out in section 13.3 is used to find these hyperparameters. The optimization will be applied to all three datasets. While several different configurations might perform roughly the same, some care should be taken not to choose a needlessly complex configuration when a more straightforward model performs equally well. An example of this is the Fakken optimization, where 1550 units perform barely better than 570, but because of this increase in units, the complexity rises while not gaining additional performance. Therefore, if the two configurations perform approximately the same, the one with fewer units is chosen. This decrease in complexity also impacts the PCA part of the proposed algorithm as the dimensionality is not reduced as heavily. One note is that what is deemed a good trade-off in terms of MSPE to the number of units is mainly arbitrary. However, among the 1000 different combinations, the difference between the two usually comes down to the $4^{th}$ decimal.

Table 8: Network configuration for the three datasets.

| Network Configuration | | | |
|---|---|---|---|
| | ACEA | GEFCom 2012 | Fakken |
| Number of units | 1165 | 559 | 570 |
| Spectral radius | 1.52 | 0.99 | 0.175 |
| Input scaling | 0.065 | 0.225 | 0.517 |
| Density | 0.26 | 0.26 | 0.093 |
| MSPE | 0.17 | 0.32 | 0.63 |

Table 8 present the different optimized network configurations for each dataset used in the final prediction in section 17. Also, in this table, the corresponding MSPE achieved during the hyperparameter search is presented. As noted earlier, this is with ten initializations each. With the MSPE for each dataset, the increasing difficulty of predicting is quite apparent, especially with the Fakken dataset, which has almost five times the MSPE as the ACEA dataset. This, in theory, should be reflected in the relative width of each prediction interval, as MSPE is a metric for error in the predictions.

### 15.2.3 PCA

As laid out in section 9, PCA is a dimensionality reduction tool that conserves the greatest amount of variation. To determine the number of components used going forward, two factors are taken into account, how much variance each component contains and how computationally complex we want our model to be. In regards to the variance, a scree plot is used. A scree plot is simply a line plot of the variance each component represents and is strictly non-increasing due to the components being ordered from largest to smallest. This way, we can judge at what point each additional component brings no significant increase in explanatory power. Also considered is the added complexity and corresponding computation time each component brings, where we use more components than strictly needed if the sampling computation time can be kept short just as an insurance policy.

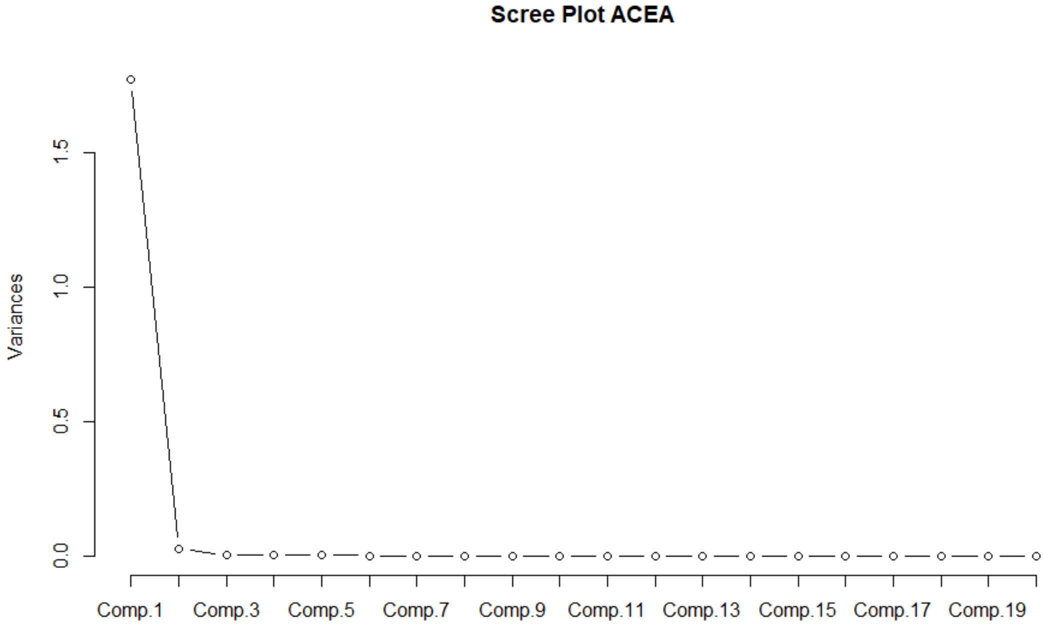


Figure 66: Scree plot for the components of the ACEA ESN model.

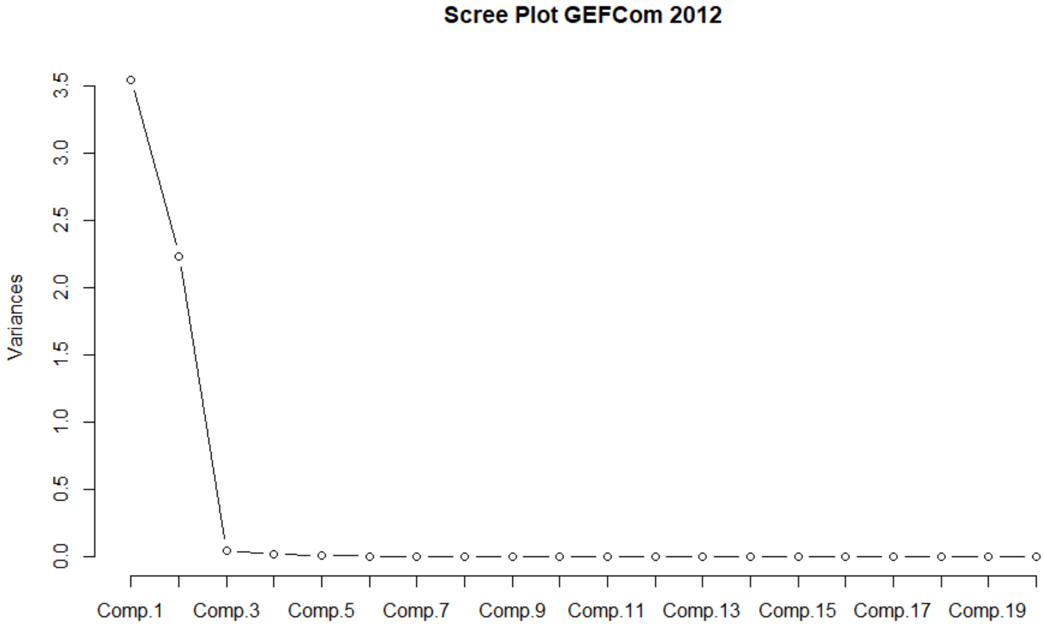


Figure 67: Scree plot for the components of the GEFCom 2012 ESN model.

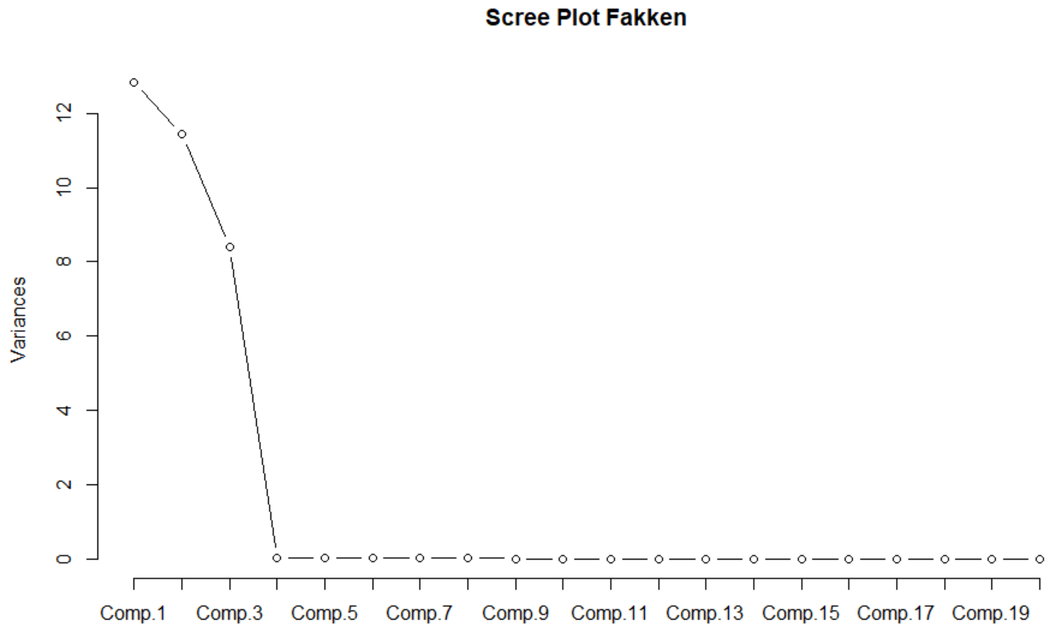


Figure 68: Scree plot for the components of the Fakken ESN model.

Figure 66, 67 and 68 show the ACEA, GEFCom 2012 and Fakken scree plots, respectively. The ACEA scree plot indicates that most of the variance is contained in just the first two components. The GEFCom 2012 scree plot increases to 3 components before the significant drop-off variance. Lastly, the Fakken should require four components to explain most of the variance. Table 9 shows the chosen amount of components for each dataset, and it is chosen by judging the scree plots in combination with choosing an amount that offers some redundancy. It should be noted that these choices are mainly subjective. However, this amount of components should be a solid compromise between the total amount of variance explained and computation times. This dimensionality reduction reduces the computation time of 99% when comparing the original GEFCom 2012 with a dimension of 559. Obviously, a more significant reduction is achieved with the other models as they have even more units.

Table 9: The amount of components chosen to move forward with.

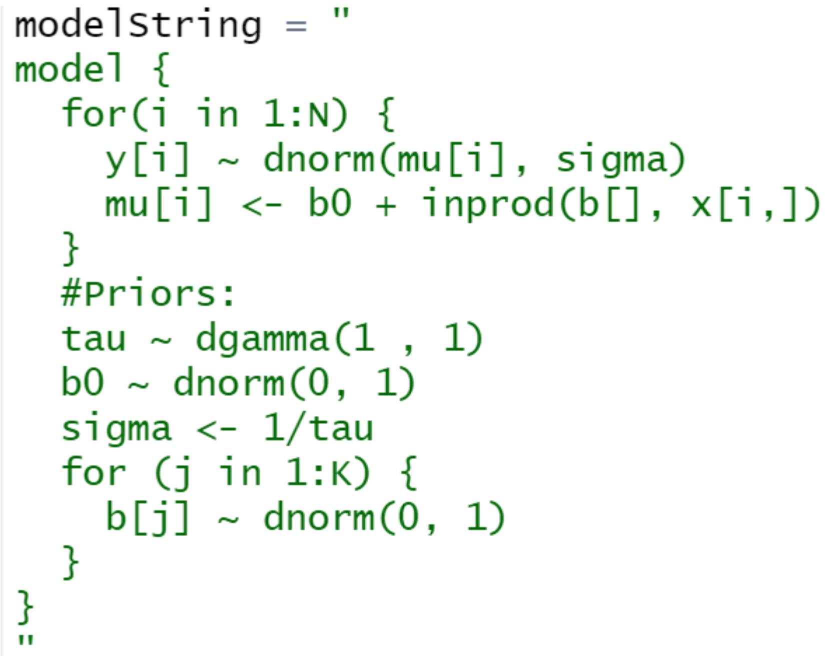
| PCA components choice | | | |
|---|---|---|---|
| | ACEA | GEFCom 2012 | Fakken |
| Components | 10 | 10 | 10 |

### 15.2.4 Bayesian Regression

The Bayesian regression part is done using the Just Another Gibbs Sampler (JAGS) with the interface package rJAGS [6] to allow the use of JAGS inside R. A vital thing to note

[6] https://cran.r-project.org/web/packages/rjags/index.html

about JAGS is that it uses precision instead of variance, with precision being the inverse of the variance. Therefore, higher precision specified in the initialization of the sampler equals less variance. When sampling, the values for the intercept and the regression coefficients are recorded in addition to the variance from each sample. Burn-in is used when sampling as Gibbs sampling requires a few samples before properly sampling from the target distribution and means simply discarding the first iterations of the sampling with however many iterations specified as burn-in. Thinning is another parameter used in Gibbs sampling, referring to how many iterations are thrown out, with 1 being none of the iterations thrown out and 10 being nine iterations thrown out while one is saved. As stated in section 8.4, density and trace plots are used to check whether the Markov chains converge towards the same distribution or not. If the two chains do not converge to the same distribution, that can indicate a wrong choice of priors or simply not enough samples with too little burn-in. Therefore, ample amounts of sampling have been provided for each dataset, however, at the cost of some computing time. This increased computing time is not detrimental as PCA is applied before sampling.



```
modelString = "
model {
  for(i in 1:N) {
    y[i] ~ dnorm(mu[i], sigma)
    mu[i] <- b0 + inprod(b[], x[i,])
  }
  #Priors:
  tau ~ dgamma(1 , 1)
  b0 ~ dnorm(0, 1)
  sigma <- 1/tau
  for (j in 1:K) {
    b[j] ~ dnorm(0, 1)
  }
}
"
```

Figure 69: Showing model specification for using JAGS.

Figure 69 shows an example of what a model specification for Bayesian linear regression could be using rJAGS. Here it is possible to see what priors are used for the regression, with dnorm(0,1) being a normal distribution with mean zero and precision equal to one for all the regression parameters from $\beta_0$ to $\beta_j$. However, as noted earlier, precision is equal to 1 divided by the variance, which means the variance will also be one in the case of precision equal to one. The prior for sigma is an inverse gamma distribution as sigma is $\frac{1}{tau}$ and tau is distributed according to a gamma distribution with scale and shape parameters equal to one. Higher precision here indicates a more informative prior, while lower precision indicates a vaguer prior. Inverse gamma is chosen as a prior as it is a conjugate prior for the normal distribution, meaning that prior and posterior share the same functional form. This means that the prior for the regression parameters $\beta$ and the variance $\sigma^2$ is a normal-inverse-gamma distribution. The priors for this regression are then:

$$
\begin{aligned}
&p(\beta, \sigma^2) = p(\beta|\sigma^2)p(\sigma^2)\\
&where\\
&\beta|\sigma^2 \sim N(0, \sigma^2)
\end{aligned}
$$

As the prior for $\beta$ is conditioned on $\sigma^2$, it is sometimes called hierarchical (Gundersen, 2020). Conjugate prior is useful as it allows easier sampling as the prior and posteriors distributions share the same form. However, note in figure 69 sigma is specified in the model, but sigma in actuality means $\sigma^2$. This is due to the way rJAGS works.

Table 10 shows the different parameters used for the Gibbs sampling. These were chosen through a mix of trial and error and reasonable assumptions.

Table 10: The Bayesian hyperparameters used in the Gibbs sampler.

| Bayesian Regression Parameters | | | |
|---|---|---|---|
| | ACEA | GEFCom 2012 | Fakken |
| Burn-in | 2000 | 2000 | 2000 |
| Samples | 10000 | 10000 | 10000 |
| Prior | Gamma(50,1) | Gamma(1,1) | Gamma(50,1) |
| Precision | 10 | 350 | 1 |
| Thinning | 1 | 1 | 1 |
| Computation time (minutes) | 13.7 | 5.2 | 5.4 |

As stated in the above paragraph, density and trace plots are used to be certain of convergence of the sampling from the target distribution. In addition to this sampling with two chains, meaning sampling twice, this is to compare where each chain converges if it indeed converges. Below are the plots concerning the convergence of the regression parameters for each dataset to give an insight into whether our hyperparameters shown in table 10 work for this given task.

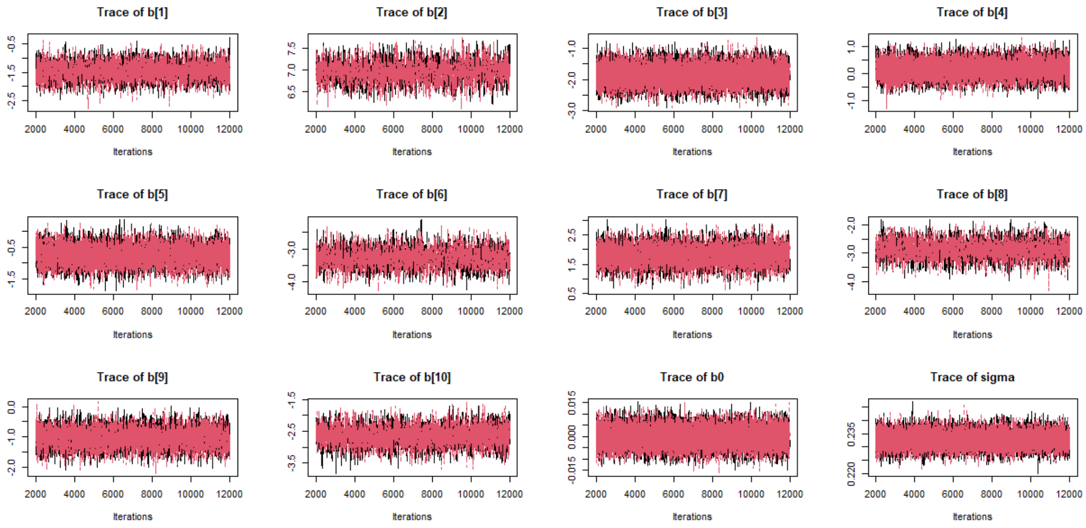


Figure 70: Trace plot from the sampling for the ACEA dataset, where black and red represent each chain.

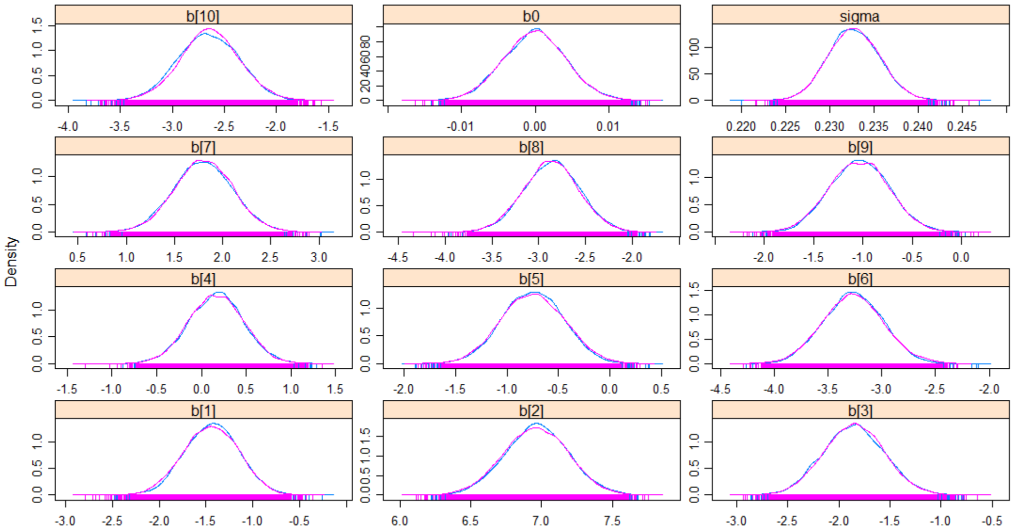


Figure 71: Density plot from the sampling for the ACEA dataset, where blue and purple represent each chain.

Here it is possible to see that both of these chains converge to the same distribution, with the trace plot oscillating between the mean of each variable. This is good as this indicates that our sampling assumptions hold up. Note that sigma is the variance used in the predictions later on, and each b is a regression coefficient with b0 being the intercept.

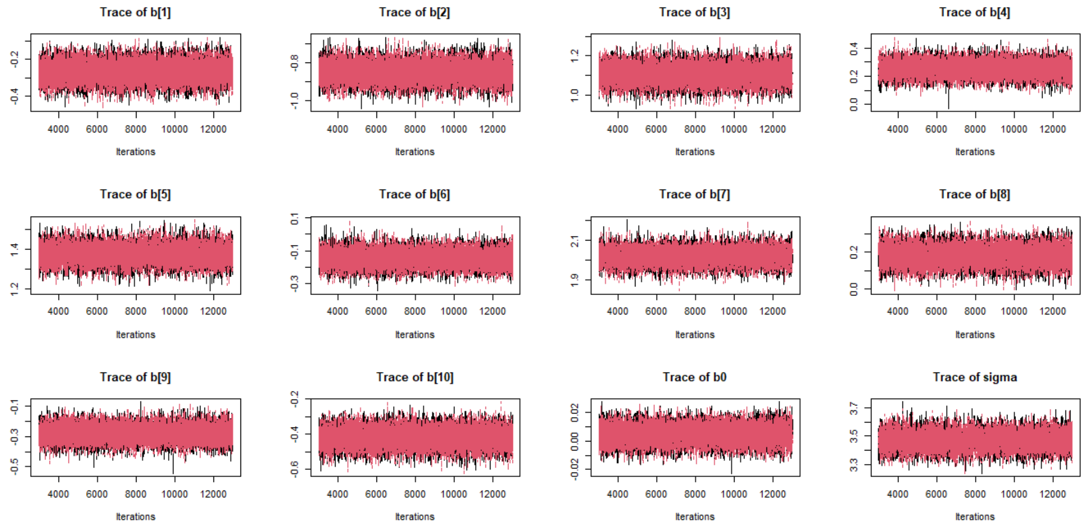


Figure 72: Trace plot from the sampling for the GEFCom 2012 dataset, where black and red represent each chain.

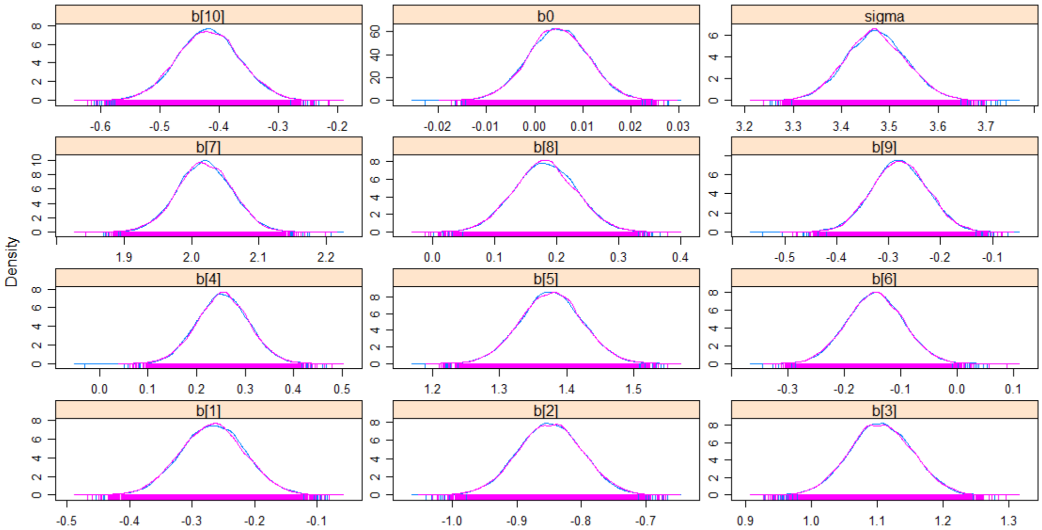


Figure 73: Density plot from the sampling for the GEFCom 2012 dataset, where blue and purple represent each chain.

As with the figures 70 and 71, figures 72 and 73 show similar results where the densities converges to the same distribution for the most part.

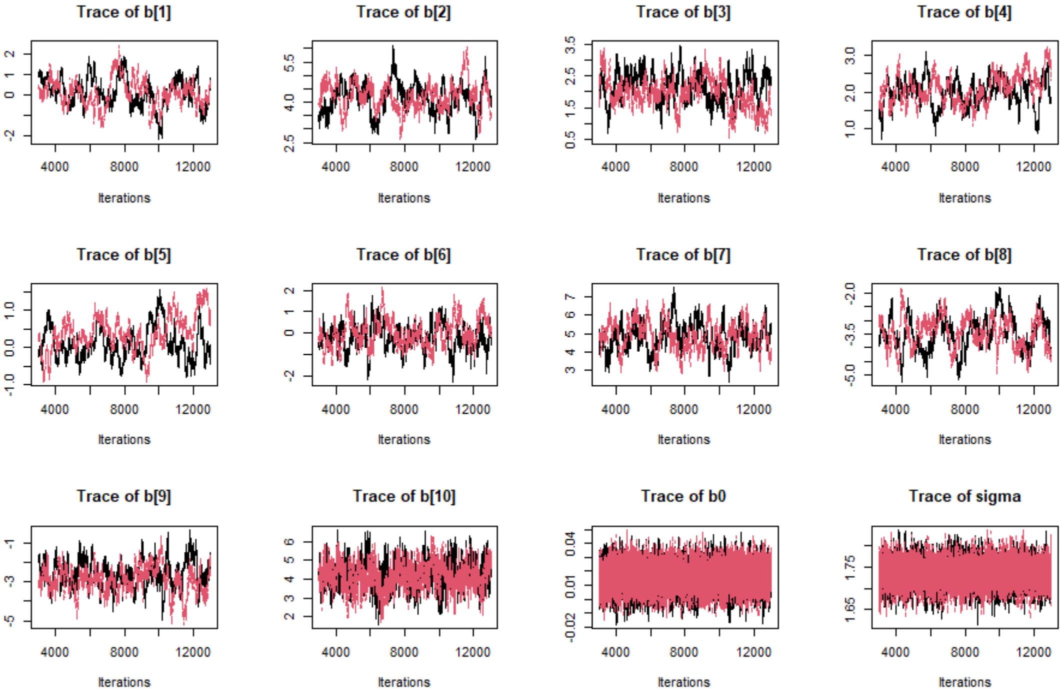


Figure 74: Trace plot from the sampling for the Fakken dataset, where black and red represent each chain.

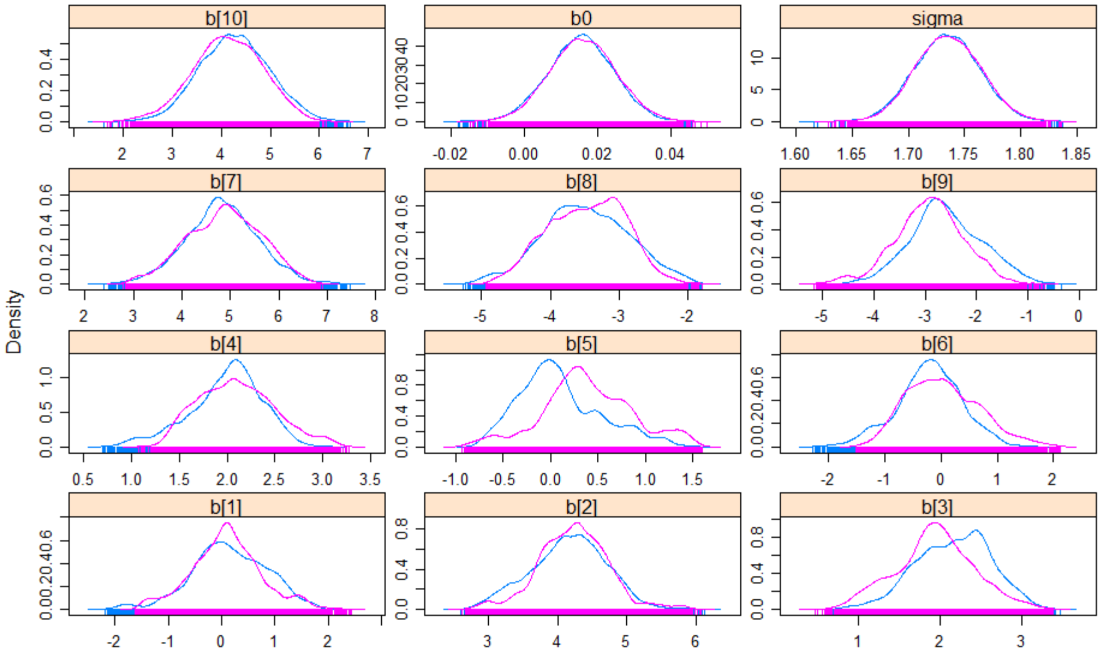


Figure 75: Density plot from the sampling for the Fakken dataset, where blue and purple represent each chain.

Again, figures 74 and 75 show that, for the most part, our sampling assumptions hold, however, with a bit more deviation in the densities as seen in figure 75. This is also reflected in the trace plots in figure 74 where the chains seem to differ a bit for coefficients like b[1] and b[5]. b[5] in particular have pretty different densities depending on the chain. This coefficient does not converge to the same value for the mean or any different quantile and could change upon redoing the sampling.

Table 11: Comparison of the the resulting MSPE using Ridge regression and Bayesian regression in combination with PCA.

| MSPE Comparison | | | |
|---|---|---|---|
| | ACEA | GEFCom 2012 | Fakken |
| Ridge MSPE | 0.16 | 0.32 | 0.63 |
| Bayes MSPE | 0.17 | 0.33 | 0.73 |
| Difference | 0.01 | 0.01 | 0.1 |

Judging by these figures, our choice of parameters seems to be good enough of a fit to move forward with. However, simply having the sampling converge as one would expect is not enough. As the mean prediction should converge to the ordinary least squares given enough data, a check of this could be to plot both predicted time series and calculate the MSPE as, in theory, they should not differ significantly from each other. This difference can be seen in table 11. In addition, this difference can be seen in figures 76, 77 and 78 below. These figures show that the Gibbs sampling is working as intended and can thus be used in the recalibration part of the proposed method in section 17. Do note that the

Ridge regression predictions made below are calculated before applying PCA; therefore, this also serves to see if the PCA step significantly changes the predictions.

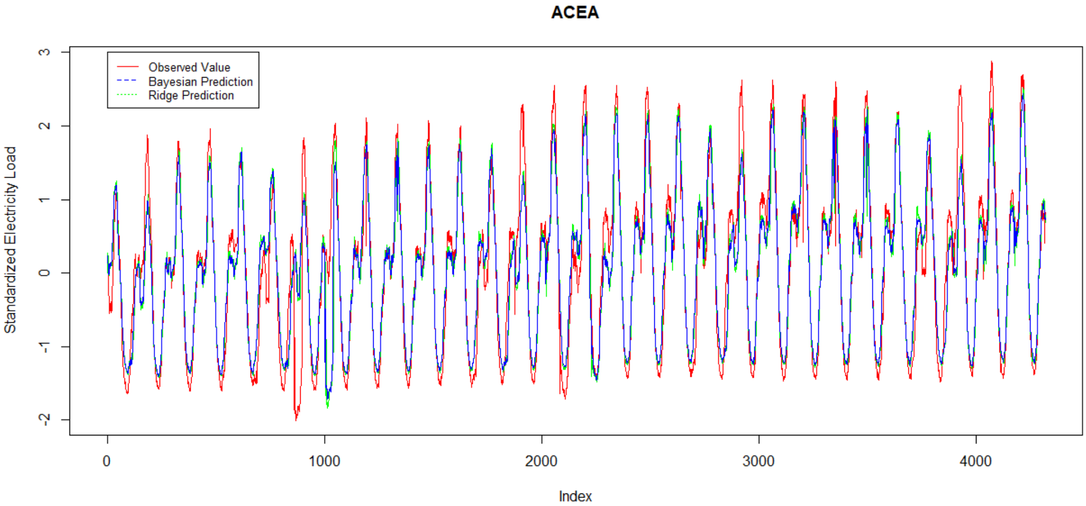


Figure 76: Showing the mean Bayesian prediction in blue and the Ridge regression with $\lambda$ being 1 in green for the ACEA dataset, the red one is the true value from the recalibration part of the dataset.

Figure 76 confirms our conclusions from the density and trace plots as there is, for all intents and purposes, no difference between the Ridge regression prediction and the mean Bayes prediction showing that our sampling assumptions are holding up.

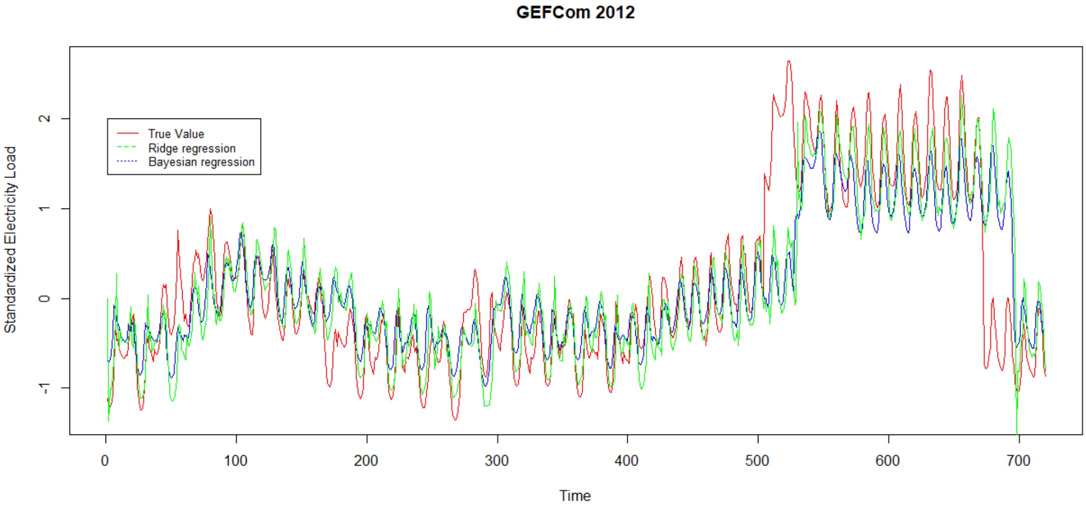


Figure 77: Showing the mean Bayesian prediction in blue and the Ridge regression with $\lambda$ being 1 in green for the GEFCom 2012 dataset, the red one is the true value from the recalibration part of the dataset.

The mean prediction for the GEFCom 2012 dataset, as seen in figure 77 shows an almost identical prediction as the standard Ridge regression prediction. This means that our sampling assumptions are indeed holding up.

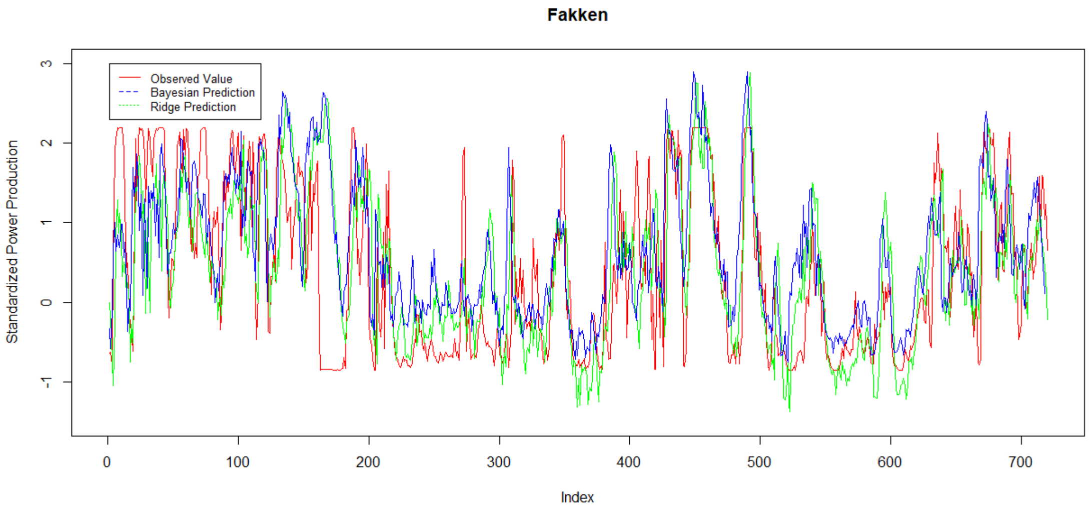


Figure 78: Showing the mean Bayesian prediction in blue and the Ridge regression with $\lambda$ being 1 in green for one turbine in the Fakken dataset, the red one is the true value from the recalibration part of the dataset.

Figure 78 show more deviation from the Ridge regression predictions; however, the predictions are generally in the same vicinity, with the mean Bayes prediction not being as extreme as the Ridge predictions. This can be due to a multitude of reasons, such as lack of data. More probably, the sampling parameters could be tuned better. Especially, the precision could be better tuned due to the high variance of this dataset. This could also be due to the PCA step in the proposed algorithm, and further experimenting might prove helpful. The precision chosen for the Fakken sampling was chosen due to the fact that increasing precision leads to a more negligible difference between the chains at the cost making the prediction more conservative, i.e., being more centered around the mean, thus increasing the difference between the mean Bayes prediction and the Ridge prediction. Therefore the precision landed at one as shown in table 10 and serves as a middle ground between the chains converging and prediction close to the Ridge prediction.

These three figures show that both the PCA and sampling steps are working as intended as beside figure 78 the mean Bayes prediction is very close to the Ridge regression prediction. Meaning that not much information is lost during the PCA step, and sampling is done from the target distribution.

# 16 Evaluation Metrics

To properly assess the performance of differing forecasters, it is crucial to have metrics that properly rank the predictive power of each forecasting approach (Gneiting & Raftery, 2007). Two useful metrics to look at regarding probabilistic forecasting in the form of prediction intervals are the width and the coverage. Using merely one of these is usually insufficient as an interval with the correct coverage but is wider than it needs to be is undesirable. Likewise, having the width be narrow is a good thing, but if it is too narrow to achieve the correct coverage, that is also undesirable. Therefore both the width and the coverage in unison should be considered when assessing the performance of a prediction interval.

Section 12 put forth the idea of sharpness and calibration as ideally a prediction interval must maximize the sharpness of the predictive distributions, subject to calibration (Gneiting & Katzfuss, 2014). As stated earlier, the sharpness is how tightly the interval covers the true distribution. In contrast, a prediction interval is valid if coverage of a new observation is guaranteed to be greater or equal to the desired confidence level (Jensen, 2021).

To quantify the performance of each model in section 15, two of the more frequently used metrics in probabilistic forecasting are prediction interval coverage probability (PICP) (Khosravi, Nahavandi, & Creighton, 2010) and prediction interval normalized average width (PINAW) (Shepero, Van Der Meer, Munkhammar, & Widén, 2018), is used.

The reasoning behind PICP is that whether the prediction interval covers a single data point is binary, the data point lies within the prediction interval, or it does not. To assess the coverage of a prediction interval for several data points, such as a time series, the coverage must be averaged over the entire predicted time series. The PICP is simple to calculate as it is the sum of all the observations inside the prediction interval divided by the number of observations. Formulated mathematically, this becomes:

$$\text{PICP} = \frac{1}{n_t}\sum_{i=1}^{n_t} c_i, \quad c_i = \begin{cases} 1, & y_t \in [L_i, U_i] \\ 0, & y_t \notin [L_i, U_i] \end{cases} \tag{32}$$

Here, $L_i$ and $U_i$ are the lower and upper bounds of the prediction interval at time step i. While $n_t$ is the number of observations in the data set. The resulting PICP score will be a number between 0 and 1, where 0 is a prediction interval containing none of the observations while 1 contains every observation.

Whereas PICP pertains to the coverage, PINAW is the normalized metric for how wide a prediction interval is. The average prediction interval width is the difference between the upper and lower limits of the interval for every time step predicted, divided by the total number of observations in the dataset. PIAW is the non-normalized prediction interval average width, formulated as such

$$PIAW = \frac{1}{n_t}\sum_{i=1}^{n_t}(U_i - L_i) \tag{33}$$

As the PIAW is not normalized, it is dependent on the scale and unit of the predictions. To make comparisons between prediction interval widths easier, it is usually normalized. This is done by dividing the PIAW by a normalizing constant. The PINAW then turns into

$$PINAW = \frac{1}{n_t R}\sum_{i=1}^{n_t}(U_i - L_i) \tag{34}$$

Where the normalizing constant R = $y_{max} - y_{min}$ is the difference between the maximum and minimum of the observations for the predicted variable.

As stated earlier, a prediction interval should have the correct coverage for the desired level while being as narrow as possible since very wide intervals are uninformative. This makes the use of the PICP score insufficient to evaluate the performance of a prediction interval. Therefore, one should include the use of PINAW when evaluating the performance of a prediction interval.

As the focus of this thesis is to construct prediction intervals for calibrated ESN forecasters, these two metrics are used to compare against the baseline method. However, comparing the width without the coverage being at least nearly equal is not a good comparison. Therefore the PICP is first used, then the PINAW, where the best model is the one with the closest PICP to its designed level while simultaneously minimizing the PINAW.

# 17 Experimental Results

In this section, the results from the experiments are presented and discussed. First, the performance using ARIMA models will be presented in table 12 and then the performance of the ESN models in table 13 and table 14, the specifics of each dataset will be gone over in section 17.1. The predictions made in section 17 are using the testing part of the dataset.

Table 12: Results from the ARIMA models establishing a baseline performance.

| ARIMA Performance | | | |
|---|---|---|---|
| | ACEA | GEFCom 2012 | Fakken |
| PICP | 0.89 | 0.97 | 0.996 |
| PINAW | 0.53 | 0.55 | 1.4 |
| MSPE | 0.96 | 0.18 | 0.88 |

Table 13: Performance before performing recalibration.

| Bayesian ESN – Uncalibrated Performance | | | |
|---|---|---|---|
| | ACEA | GEFCom 2012 | Fakken |
| PICP | 0.93 | 1.0 | 0.98 |
| PINAW | 0.28 | 1.75 | 1.41 |
| MSPE | 0.19 | 0.19 | 0.8 |

From table 12 and table 13, we can see that the Bayesian ESN model vastly outperforms the ARIMA model for the ACEA set with the MSPE reduction of over 75%. Sadly this performance increase is not the same for the two other datasets, where the Fakken dataset only offers a 10% reduction in MSPE. In contrast, the GEFCom 2012 dataset actually sees an increase in MSPE. The performance increase in PICP and PINAW is also evident, at least for the ACEA dataset; even if there is no significant difference in

PICP, the PINAW is almost half that of the ARIMA prediction interval compared to the Bayesian ESN prediction interval. This means that the Bayesian ESN model has greater coverage before recalibrating while maintaining substantially lower PINAW. The GEFCom 2012 prediction interval is excessively wide for both the Bayesian ESN and the ARIMAX intervals. The PINAW for the Bayesian model is extreme and thus very uninformative as the true values are all within the interval. However, the Fakken dataset sees a decrease in PICP with a slight increase in PINAW, meaning that the coverage is down, but the interval is wider. This will be explored more in-depth in section 17.1.3.

Table 14: Performance after performing recalibration.

| Bayesian ESN – Recalibrated Performance | | | |
|---|---|---|---|
| | ACEA | GEFCom 2012 | Fakken |
| PICP | 0.89 | 0.90 | 0.88 |
| PINAW | 0.23 | 0.42 | 0.81 |
| MSPE | 0.19 | 0.19 | 0.8 |

The recalibration shows promising results. Here the PICP is closer to its intended value for the ACEA and Fakken datasets, with a significant narrowing in the width of the prediction intervals seen in the PINAW score. GEFCom 2012 shows the most dramatic difference, and it goes from covering every single observation with an extremely wide prediction interval to producing a PICP score of 0.9, which is perfect. The PINAW is still relatively large even though there is a reduction of 76%. Overall the results are excellent as all the PICP for all the datasets are within 0.02 of their designed level.

## 17.1 Performance on Individual Datasets

### 17.1.1 ACEA Dataset

ACEA was the dataset that showed the most promising results switching from an ARIMAX model to a Bayesian ESN model. The resulting reduction in MSPE is nothing short of extreme, as it goes from 0.96 to 0.19. The ARIMAX model is unable to deal with reproducing the prominent daily seasonality. However, the ESN model reproduces this very well, and therefore the errors are pretty minor, resulting in a substantial decrease in MSPE.

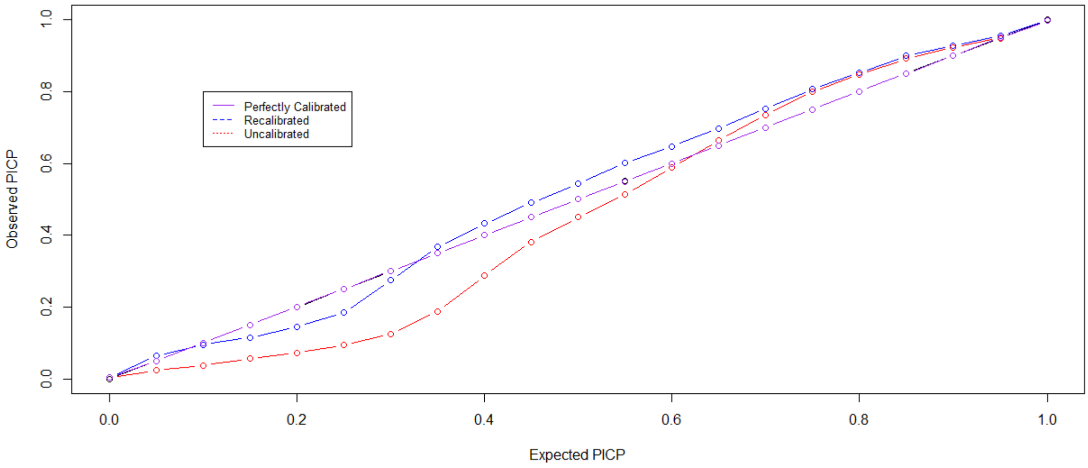


Figure 79: Diagnostic plot for the recalibration with the blue line being the recalibrated and red being the original with purple as perfectly calibrated using the test data.

Figure 79 confirms the improvements made by recalibrating as the recalibrated line much more closely resembles the perfectly calibrated model compared to the uncalibrated one. As the data used is the testing, some deviation is expected to be seen in figure 79. Ideally, when using such a plot, the recalibrated line should closely resemble the perfectly straight purple line, which would indicate a perfectly calibrated model. There is, however, still some amount of deviance, resulting in a model that is not perfectly calibrated but is still an improvement over the uncalibrated model. In particular, it still provides undercoverage in the 0.2 region while also providing overcoverage shortly after that remains until the expected PICP reaches 0.95. The recalibration still yielded good performance as it is very close at the 0.05 and 0.95 quantiles, which are the quantiles used to make the prediction intervals. Both before and after recalibration, the performance line exhibits some of the same behavior and sort of mimics each other. There is some undercoverage around the 0.25 region which then transitions into overcoverage later on. Albeit, the deviations are considerably more significant before recalibration.

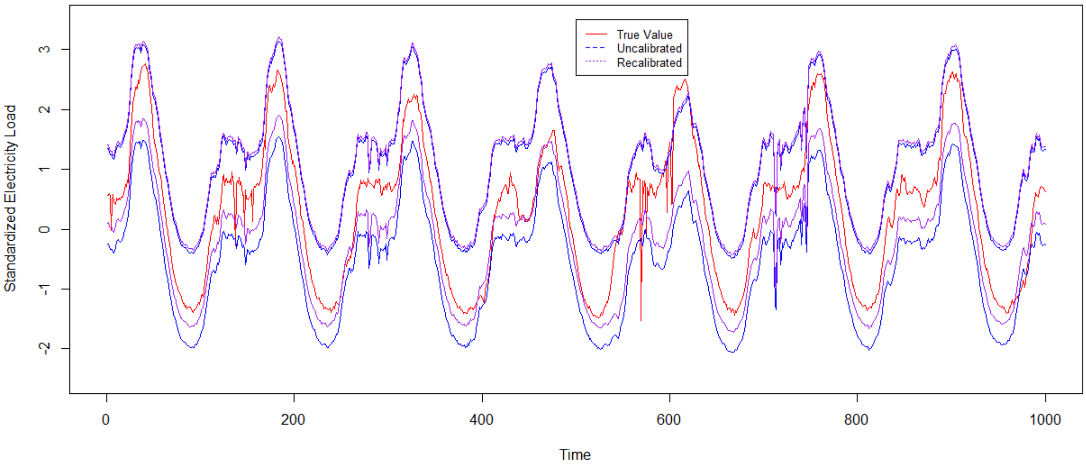


Figure 80: Plot showing the uncalibrated and recalibrated prediction intervals for the first 1000 predictions, also showing the observed values.

Figure 80 shows the narrowing of the prediction interval. While the PICP is not precisely 0.9 as intended, it got pretty close and is an improvement compared to the uncalibrated prediction interval. One can easily observe here that the upper bound remains almost utterly unchanged while the lower band is shifted slightly upwards due to the recalibration.

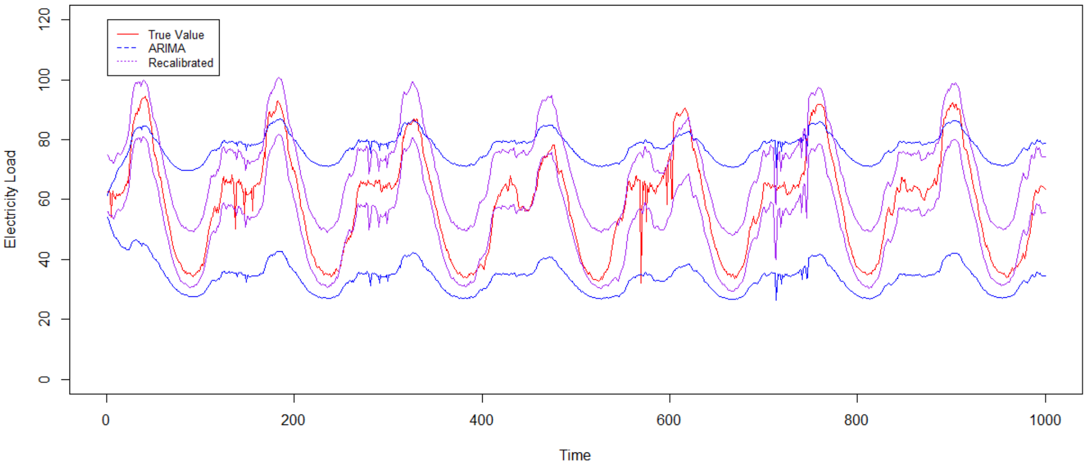


Figure 81: Plot showing the recalibrated prediction interval for the first 1000 predictions as well as the ARIMAX prediction interval, also showing the observed values. This is after reverting the standardization step.

With figure 81 the performance gained by using the proposed method as opposed to ARIMAX becomes evident. As noted earlier, ARIMAX's failure to capture the daily electricity load in the ACEA dataset leads to uninformative prediction intervals. It is only in the upper band that observations land outside the interval. The recalibrated Bayesian ESN prediction interval follows the ACEA dataset fluctuations while maintaining a PICP score very close to the intended level, making the model vastly superior to the ARIMAX model. Overall the recalibration yielded good results for ACEA.

### 17.1.2 GEFCom 2012 Dataset

The GEFCom 2012 saw a giant performance leap where the coverage reached its designed level. This is also confirmed with figure 82 which shows that the recalibrated is near-perfectly calibrated with only a slight deviation for a few PICP levels. It is also apparent from this figure that before recalibration shows extreme coverage as there is only a narrow band where the coverage actually changes.

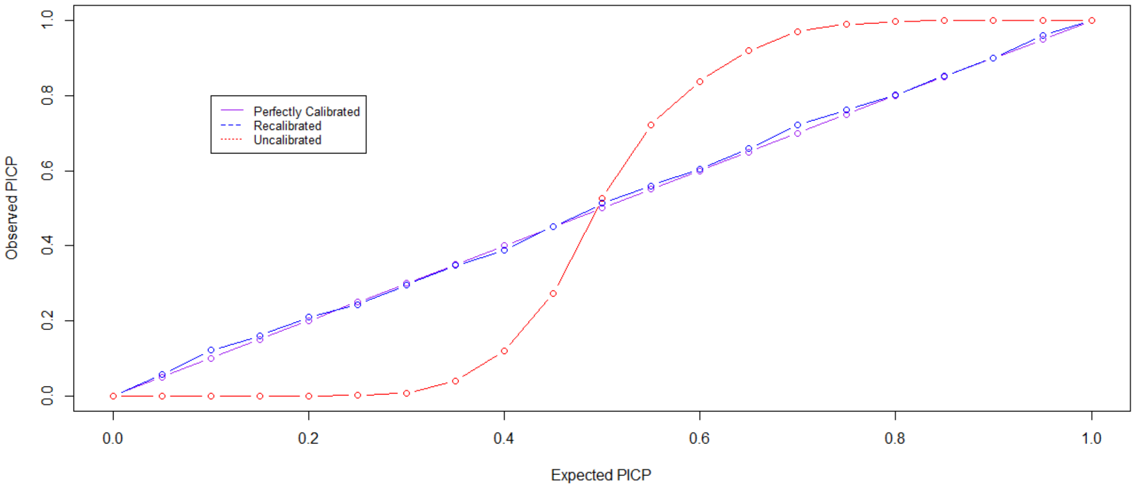


Figure 82: Diagnostic plot for the recalibration with the blue line being the recalibrated and red being the original with purple as perfectly calibrated using the test data.

The extent of this overcoverage becomes apparent in figure 83, the chasm between both the upper and lower band and the true value is extreme as even the highest value in the lower bound is still significantly lower than the lowest true value. Recalibration tightens this gap to a large extent and provides approximately valid prediction intervals compared to the uncalibrated prediction interval, which produced abysmal results even compared to the ARIMAX model.

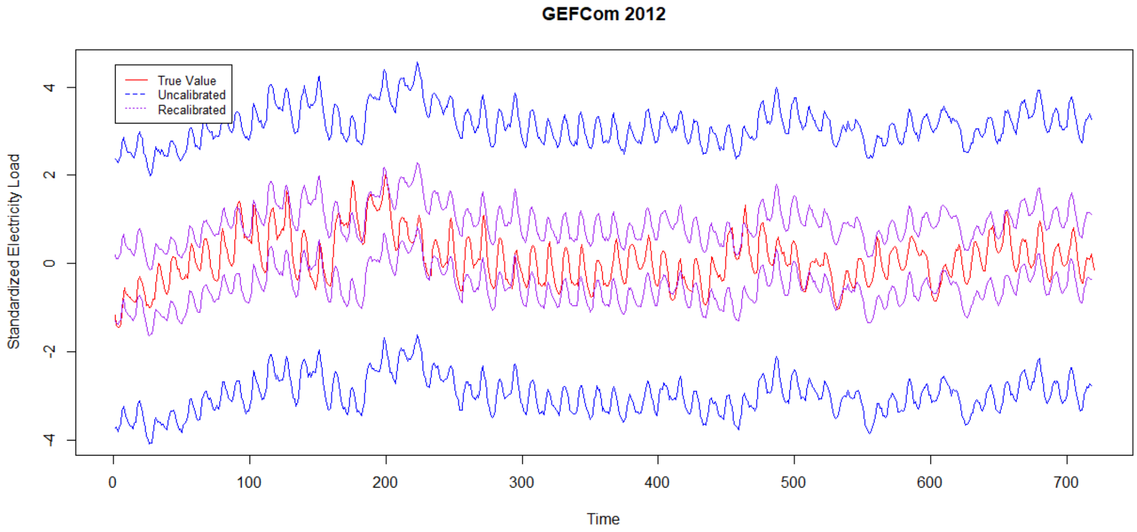


Figure 83: Plot showing the uncalibrated and recalibrated prediction intervals, also showing the observed values.

Figure 84 shows the difference between the ARIMAX prediction interval and the recalibrated Bayesian ESN prediction interval. Here, the difference between the upper bounds remains relatively small. However, in the lower bound, the recalibrated prediction interval starts to pull ahead in performance by following the true observations more closely, thus providing more narrow prediction intervals. However, both prediction intervals are quite wide, and the mean Bayesian prediction and ARIMAX predictions perform roughly the same, given that the MSPE for both is within 0.01 of each other.

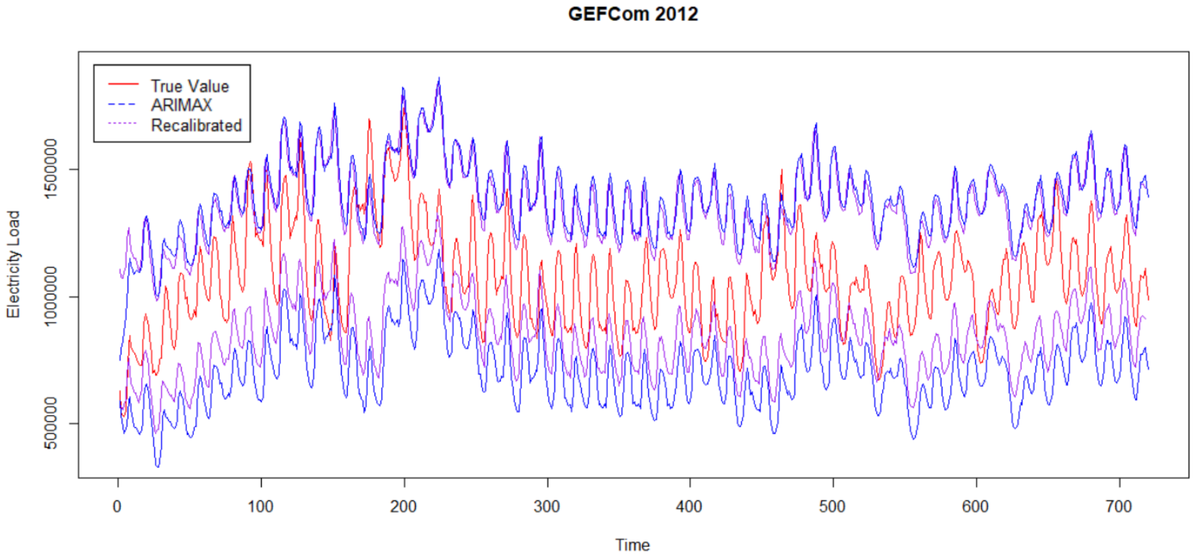


Figure 84: Plot showing the recalibrated prediction intervals as well as the ARIMAX prediction interval, also showing the observed values. This is after reverting the standardization step.

### 17.1.3 Fakken Dataset

In the Fakken dataset, after recalibration, the PICP is only slightly lower than the desired coverage level of 0.9. The PINAW was also significantly reduced from 1.41 to 0.81 and is a clear improvement compared to both the ARIMAX model and the uncalibrated Bayesian ESN model.

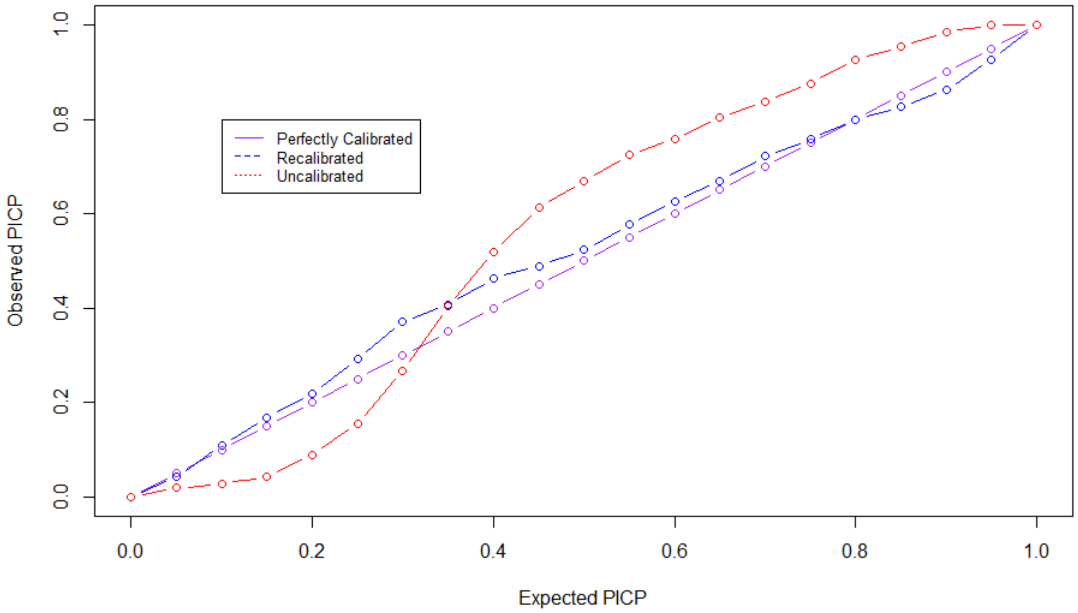


Figure 85: Diagnostic plot for the recalibration with the blue line being the recalibrated and red being the original with purple as perfectly calibrated using the test data.

The diagnostic plot as shown in figure 85 confirm the results shown in table 13 and table 14. As is evident in this plot, the recalibrated model adheres much closer to a model with perfectly calibrated PICP. There is barely any deviance with the exception of roughly 0.3 to 0.5, where there is some amount of overcoverage; however, it performs very well with the PICP desired in most situations.

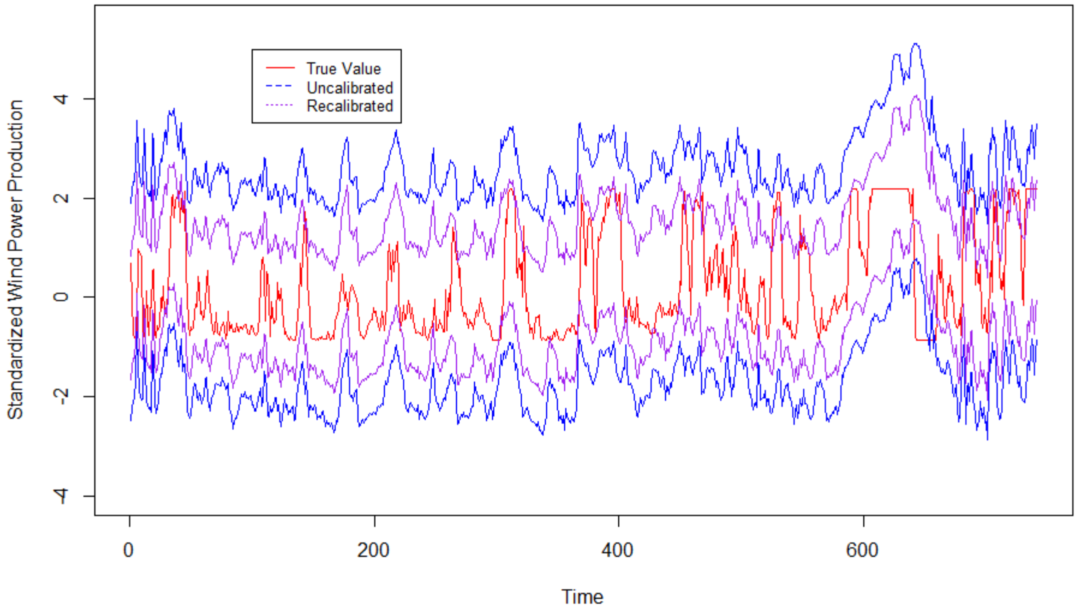


Figure 86: Plot showing the uncalibrated and recalibrated prediction intervals, also showing the observed values.

Figure 86 shows the narrowing of the prediction interval while getting closer to the intended PICP score. Even though PINAW is greatly reduced, it still remains rather large, which means that the prediction interval is not very sharp and thus is quite uncertain.

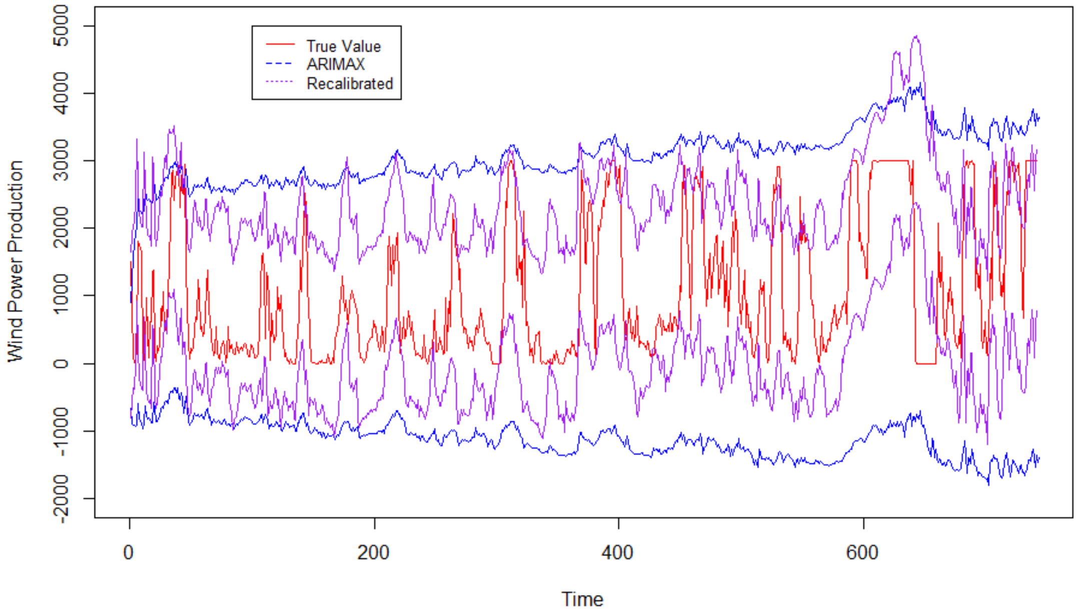


Figure 87: Plot showing the recalibrated prediction intervals as well as the ARIMAX prediction interval, also showing the observed values. This is after reverting the standardization step.

In figure 87, the improvements made by the recalibrated Bayesian ESN model become rather straightforward, especially when moving further along the time axis. The ARIMAX prediction is even worse, considering the turbine has a maximum power output of 3000kW and a minimum of 0. The lower interval bound is always lower than 0, making it effectively useless and the upper bound goes past 3000kW without going below it again after about 350-time steps. This means that after about 350-time steps, there is no possibility of the true values not being contained in the ARIMAX prediction interval. While the recalibrated prediction interval is better in this regard, it still leaves a lot to be desired, especially around the 600-time step mark. The upper bound exceeds the turbine's maximum capacity by a wide margin during this period. The lower bound of the recalibrated prediction interval also exhibits the same behavior as the ARIMAX, at least somewhat during the times when the actual power production hovers around 0.

Of course, the quality of the interval could be improved by manually adjusting the interval to 0 when the lower bound assumes negative values and by decreasing the upper bound to 3000kW when it goes above 3000kW.

# 18 Discussion

From table 12, 13 and 14, it can be seen that the proposed method produced the sharpest approximately valid prediction intervals, however depending on the coverage achieved before recalibration, as with the ACEA data, the performance gained might not be huge.

The PINAW was also seen to be varying hugely, depending on the dataset, as Fakken produced incredibly wide prediction intervals no matter the model used. Thus the PICP reached nearly 1 for both models before recalibrating. The reason for this appears to be the considerable variation seen in the dataset coupled with the lack of production seen in figure 34. One can reasonably assume this will impact the uncertainty in the predictions and the MSPE, which is very high for both the ARIMAX and the Bayesian ESN models.

In contrast to the Fakken data, the GEFCom 2012 data saw good performance from the ARIMAX model with an MSPE slightly lower than the one from the Bayesian ESN model. The PICP and PINAW also show better performance for the ARIMAX model than the Bayesian ESN model, at least before recalibration. Speculations can be made about this; the biggest reason for this massively large prediction interval is how the predictions are made. The Bayesian predictions are made by using the regression weights from a given sample and adding an error term that is normally distributed with a mean equal to 0 and variance equal to the sigma from the same given sample; this sigma was incredibly large for the GEFCom 2012 data, hovering around 3.3-7 depending on the sample as can be seen in the trace plots in figure 72.

Gibbs sampling with GEFCom proved to be tricky as there was a trade-off to be had here; by increasing the precision, sigma converged to a smaller value. However, this came at the cost of increasing the MSPE as the mean prediction then became less informative, see figure 88. Thus the choice became to move forward with a high sigma, and this meant receiving extremely wide prediction intervals as the PICP went to 1 and no true value ever came close to the upper and lower bounds.

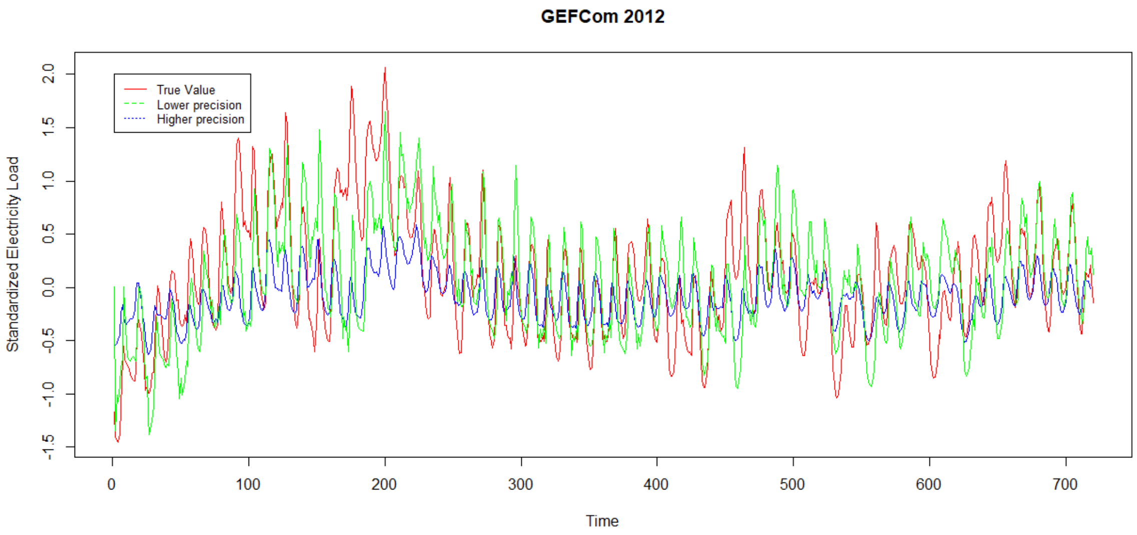


Figure 88: Plot showing how increasing precision decreases the usefulness of the predictions.

However, the proposed method was able to deal with this extreme overcoverage, producing an almost perfectly calibrated model, see figure 82. Through recalibration, all the coverage goes from the narrow band from circa 0.4 to 0.6 to only showing a very slight deviation from a perfectly calibrated. This shows the ability of the proposed method to produce an approximately valid prediction interval given grossly overcovering models.

# Part V / Conclusions

This thesis focused on producing marginally valid and calibrated prediction intervals in the field of probabilistic electricity load and power production forecasting by applying Bayesian inference to an echo state network model. While also reducing the dimensionality of the reservoir states to shorten computation times and recalibrating the quantiles to produce said marginally valid and calibrated prediction intervals. The proposed method has been shown to produce calibrated prediction intervals with approximately valid marginal coverage using two real-world electricity load datasets in conjunction with one real-world electricity production dataset.

Additionally, the proposed method has been compared to the statistical time series forecasting method autoregressive integrated moving average as well as the performance gains offered by recalibrating the quantiles. The experimental results show the coverage being significantly closer to the designed level after recalibration, resulting in narrower and more informative prediction intervals as all but one model's prediction interval being too wide with coverage of the prediction interval close to 100% when 90% is intended. While simultaneously reducing computation times by a massive margin without sacrificing point estimate performance. The dimensionality reduction resulted in a whopping 99% reduction in computation time for the dataset with the least amount of dimensions. The conclusions can overall be drawn as follows:

- Reduction of the dimensionality of the reservoir states saw little to no performance decrease, but has a tremendous impact in reducing the computational time for training the Bayesian regression.
- All recalibrated Bayesian models constructed approximately marginally valid and calibrated prediction intervals as they in theory should.
- The major problem of uncalibrated Bayesian models is that they systematically overcover, i.e., they produce prediction intervals that are too wide. After recalibration, the prediction interval become sharper and the coverage is reduced to match the designed confidence level.
- Compared to the statistical-based models, the neural networks before recalibrating performed as good or better, and then the performance was further improved after recalibration.

To conclude this thesis, both research questions have been answered. The dimensionality reduction greatly reduced the computation time while maintaining the model's performance even with a large number of connections in the reservoir, and the prediction intervals were recalibrated to provide calibrated predictions intervals. This held for all three datasets, regardless of the model's performance or coverage before recalibration. Thus, it provides evidence for the use case of the proposed method as it works for various problems.

Future work can be applying the proposed method to different types of recurrent networks or any model that requires regression to train the model, in particular to more complex versions of the echo state network models that might improve the baseline point estimate performance, such as deep echo state network models (Gallicchio et al., 2017), or perhaps by making it an ensemble echo state network model (Rigamonti et al., 2018). Another direction future work can take is in trying to make the prediction intervals sharper for a model that is well calibrated through some kind of post processing algorithm such as the recalibration algorithm used in this thesis.

# References

Adhikari, R., & Agrawal, R. K. (2013). An introductory study on time series modeling and forecasting. *arXiv preprint arXiv:1302.6613*.

Almeshaiei, E., & Soltan, H. (2011). A methodology for electric power load forecasting. *Alexandria Engineering Journal*, *50*(2), 137–144.

Barber, R. F., Candes, E. J., Ramdas, A., & Tibshirani, R. J. (2019). The limits of distribution-free conditional predictive inference. *arXiv preprint arXiv:1903.04684*.

Bianchi, F. M., De Santis, E., Rizzi, A., & Sadeghian, A. (2015). Short-term electric load forecasting using echo state networks and pca decomposition. *Ieee Access*, *3*, 1931–1943.

Bianchi, F. M., Gallicchio, C., & Micheli, A. (2022). Pyramidal reservoir graph neural network. *Neurocomputing*, *470*, 389–404.

Bianchi, F. M., Kampffmeyer, M., Maiorino, E., & Jenssen, R. (2017). Temporal overdrive recurrent neural network. In *2017 international joint conference on neural networks (ijcnn)* (pp. 4275–4282).

Bianchi, F. M., Livi, L., & Alippi, C. (2016). Investigating echo-state networks dynamics by means of recurrence analysis. *IEEE transactions on neural networks and learning systems*, *29*(2), 427–439.

Bianchi, F. M., Livi, L., & Alippi, C. (2018). On the interpretation and characterization of echo state networks dynamics: a complex systems perspective. In *Advances in data analysis with computational intelligence methods* (pp. 143–167). Springer.

Bianchi, F. M., Livi, L., Alippi, C., & Jenssen, R. (2017). Multiplex visibility graphs to investigate recurrent neural network dynamics. *Scientific reports*, *7*(1), 1–13.

Bianchi, F. M., Livi, L., Jenssen, R., & Alippi, C. (2017). Critical echo state network dynamics by means of fisher information maximization. In *2017 international joint conference on neural networks (ijcnn)* (pp. 852–858).

Bianchi, F. M., Maiorino, E., Kampffmeyer, M. C., Rizzi, A., & Jenssen, R. (2017). An overview and comparative analysis of recurrent neural networks for short term load forecasting. *arXiv preprint arXiv:1705.04378*.

Bianchi, F. M., Scardapane, S., Løkse, S., & Jenssen, R. (2017). Bidirectional deep-readout echo state networks. *arXiv preprint arXiv:1711.06509*.

Bianchi, F. M., Scardapane, S., Løkse, S., & Jenssen, R. (2020). Reservoir computing approaches for representation and classification of multivariate time series. *IEEE transactions on neural networks and learning systems*, *32*(5), 2169–2179.

Bianchi, F. M., Scardapane, S., Uncini, A., Rizzi, A., & Sadeghian, A. (2015). Prediction of telephone calls load using echo state network with exogenous variables. *Neural Networks*, *71*, 204–213.

Bianchi, F. M., & Suganthan, P. N. (2020). Non-iterative learning approaches and their applications. *Cognitive Computation*, *12*(2), 327–329.

Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time series analysis: forecasting and control.* John Wiley & Sons.

Brockwell, P. J., Brockwell, P. J., Davis, R. A., & Davis, R. A. (2016). *Introduction to time series and forecasting.* Springer.

Brownlee, J. (2018). *Deep learning for time series forecasting: Predict the future with mlps, cnns and lstms in python.* Machine Learning Mastery.

Chen, Y., Kang, Y., Chen, Y., & Wang, Z. (2020). Probabilistic forecasting with temporal convolutional neural network. *Neurocomputing*, *399*, 491 - 501.

Choi, C., Bianchi, F. M., Kampffmeyer, M., & Jenssen, R. (2020). Short-term load forecasting with missing data using dilated recurrent attention networks.

Dalal, N., Mølnå, M., Herrem, M., Røen, M., & Gundersen, O. E. (2020). Day-ahead forecasting of losses in the distribution network. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 34, pp. 13148–13155).

Dang-Ha, T., Bianchi, F. M., & Olsson, R. (2017). Local short term electricity load forecasting: Automatic approaches. In *2017 international joint conference on neural networks (ijcnn)* (p. 4267-4274). doi: 10.1109/IJCNN.2017.7966396

De Gooijer, J. G., & Hyndman, R. J. (2006). 25 years of time series forecasting. *International Journal of Forecasting*, *22*(3), 443-473. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0169207006000021` (Twenty five years of forecasting) doi: https://doi.org/10.1016/j.ijforecast.2006.01.001

Deihimi, A., & Rahmani, A. (2017). Application of echo state network for harmonic detection in distribution networks. *IET Generation, Transmission & Distribution*, *11*(5), 1094–1101.

Dickey, D. A., & Fuller, W. A. (1979). Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American statistical association*, *74*(366a), 427–431.

Dunson, D. B. (2001). Commentary: practical advantages of bayesian analysis of epidemiologic data. *American journal of Epidemiology*, *153*(12), 1222–1226.

Eikeland, O. F., Hovem, F. D., Olsen, T. E., Chiesa, M., & Bianchi, F. M. (2022). Probabilistic forecasts of wind power generation in regions with complex topography using deep learning methods: An arctic case. *arXiv preprint arXiv:2203.07080*.

Gallicchio, C., & Micheli, A. (2010). Graph echo state networks. In *The 2010 international joint conference on neural networks (ijcnn)* (pp. 1–8).

Gallicchio, C., & Micheli, A. (2011). Architectural and markovian factors of echo state networks. *Neural Networks*, *24*(5), 440–456.

Gallicchio, C., & Micheli, A. (2013). Tree echo state networks. *Neurocomputing*, *101*, 319–337.

Gallicchio, C., Micheli, A., & Pedrelli, L. (2017). Deep reservoir computing: A critical experimental analysis. *Neurocomputing*, *268*, 87–99.

Gallicchio, C., Micheli, A., & Pedrelli, L. (2018). Design of deep echo state networks. *Neural Networks*, *108*, 33–47.

Gasparin, A., Lukovic, S., & Alippi, C. (2019). Deep learning for time series forecasting: The electric load case. *arXiv preprint arXiv:1907.09207*.

Gasthaus, J., Benidis, K., Wang, Y., Rangapuram, S. S., Salinas, D., Flunkert, V., & Januschowski, T. (2019). Probabilistic forecasting with spline quantile function rnns. In *The 22nd international conference on artificial intelligence and statistics* (pp. 1901–1910).

Gilks, W. R., Richardson, S., & Spiegelhalter, D. (1995). *Markov chain monte carlo in practice*. CRC press.

Gneiting, T., & Katzfuss, M. (2014). Probabilistic forecasting. *Annual Review of Statistics and Its Application*, *1*, 125–151.

Gneiting, T., & Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, *102*(477), 359–378.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. (`http://www.deeplearningbook.org`)

Gundersen, G. (2020). *Bayesian linear regression.* Retrieved Accessed: 2022-05-

08, from https://gregorygundersen.com/blog/2020/02/04/bayesian-linear-regression/

Hastie, T., & Tibshirani, R. (1997). Classification by pairwise coupling. *Advances in neural information processing systems*, *10*.

Hoijtink, H., Klugkist, I., & Boelen, P. A. (2008). *Bayesian evaluation of informative hypotheses.* Springer.

Hong, T., & Fan, S. (2016). Probabilistic electric load forecasting: A tutorial review. *International Journal of Forecasting*, *32*(3), 914–938.

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: principles and practice.* OTexts.

Infield, D., & Freris, L. (2009). *Renewable energy in power systems.* John Wiley & Sons.

Jensen, V. (2021). *Probabilistic load forecasting with deep conformalized quantile regression* (Unpublished master's thesis). UiT The Arctic University of Norway.

Jensen, V., Bianchi, F. M., & Anfinsen, S. N. (2022). Ensemble conformalized quantile regression for probabilistic time series forecasting. *arXiv preprint arXiv:2202.08756*.

Keren, G., Cummins, N., & Schuller, B. (2018). Calibrated prediction intervals for neural network regressors. *IEEE Access*, *6*, 54033–54041.

Khosravi, A., Nahavandi, S., & Creighton, D. (2010). Construction of optimal prediction intervals for load forecasting problems. *IEEE Transactions on Power Systems*, *25*(3), 1496–1503.

Kuleshov, V., Fenner, N., & Ermon, S. (2018, 10–15 Jul). Accurate uncertainties for deep learning using calibrated regression. In J. Dy & A. Krause (Eds.), *Proceedings of the 35th international conference on machine learning* (Vol. 80, pp. 2796–2804). PMLR.

Livi, L., Bianchi, F. M., & Alippi, C. (2017). Determination of the edge of criticality in echo state networks through fisher information maximization. *IEEE transactions on neural networks and learning systems*, *29*(3), 706–717.

Løkse, S., Bianchi, F. M., & Jenssen, R. (2017). Training echo state networks with regularization through dimensionality reduction. *Cognitive Computation*, *9*(3), 364–378.

Lunn, D. J., Thomas, A., Best, N., & Spiegelhalter, D. (2000). Winbugs-a bayesian modelling framework: concepts, structure, and extensibility. *Statistics and computing*, *10*(4), 325–337.

Maat, J. R., Gianniotis, N., & Protopapas, P. (2018). Efficient optimization of echo state networks for time series datasets. In *2018 international joint conference on neural networks (ijcnn)* (pp. 1–7).

Maiorino, E., Bianchi, F. M., Livi, L., Rizzi, A., & Sadeghian, A. (2017). Data-driven detrending of nonstationary fractal time series with echo state networks. *Information Sciences*, *382*, 359–373.

Maklin, C. (2020). *Gibbs sampling yet another mcmc method.* Retrieved Accessed: 2022-04-19, from https://towardsdatascience.com/gibbs-sampling-8e4844560ae5#:~:text=The%20Gibbs%20Sampling%20is%20a,we%20always%20accept%20the%20proposal

McDermott, P. L., & Wikle, C. K. (2019). Deep echo state networks with uncertainty quantification for spatio-temporal forecasting. *Environmetrics*, *30*(3), e2553.

Mushore, G. S. T. (2018). *Bayesian analysis of temporal and spatial trends of house prices in norway* (Unpublished master's thesis). UiT Norges arktiske universitet.

Noori, N. S., Waag, T. I., & Bianchi, F. M. (2020). Condition monitoring system for internal blowout prevention (ibop) in top drive assembly system using discrete event systems and deep learning approaches.

Platt, J., et al. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, *10*(3), 61–74.

Quan, H., Srinivasan, D., & Khosravi, A. (2013). Short-term load and wind power forecasting using neural network-based prediction intervals. *IEEE transactions on neural networks and learning systems*, *25*(2), 303–315.

Racca, A., & Magri, L. (2021). Robust optimization and validation of echo state networks for learning chaotic dynamics. *Neural Networks*, *142*, 252–268.

Rentsch, A., & Vasishth, A. (2019). *Accurate uncertainties for deep learning using calibrated regression.* Retrieved Accessed: 2022-04-19, from `https://github.com/AnthonyRentsch/calibrated_regression/blob/master/FinalProjectReport.ipynb`

Rigamonti, M., Baraldi, P., Zio, E., Roychoudhury, I., Goebel, K., & Poll, S. (2018). Ensemble of optimized echo state networks for remaining useful life prediction. *Neurocomputing*, *281*, 121–138.

Romano, Y., Patterson, E., & Candès, E. J. (2019). Conformalized quantile regression. *arXiv preprint arXiv:1905.03222*.

Shepero, M., Van Der Meer, D., Munkhammar, J., & Widén, J. (2018). Residential probabilistic load forecasting: A method using gaussian process designed for electric load data. *Applied Energy*, *218*, 159–172.

Shumway, R. H., & Stoffer, D. S. (2017). *Time series analysis and its applications: with r examples.* Springer.

Silva, L. (2014). A feature engineering approach to wind power forecasting: Gefcom 2012. *International Journal of Forecasting*, *30*(2), 395–401.

Skowronski, M. D., & Harris, J. G. (2007). Automatic speech recognition using a predictive echo state network classifier. *Neural networks*, *20*(3), 414–423.

Taboga, M. (2021). *Markov chain monte carlo (mcmc) diagnostics.* Retrieved Accessed: 2022-04-30, from `https://www.statlect.com/fundamentals-of-statistics/Markov-Chain-Monte-Carlo-diagnostics`

Taieb, S. B., Bontempi, G., Atiya, A. F., & Sorjamaa, A. (2012). A review and comparison of strategies for multi-step ahead time series forecasting based on the nn5 forecasting competition. *Expert systems with applications*, *39*(8), 7067–7083.

Taylor, J. W., & McSharry, P. E. (2007). Short-term load forecasting methods: An evaluation based on european data. *IEEE Transactions on Power Systems*, *22*(4), 2213-2219.

Tchakoucht, T. A., & Ezziyyani, M. (2018). Multilayered echo-state machine: a novel architecture for efficient intrusion detection. *IEEE Access*, *6*, 72458–72468.

Thiede, L. A., & Parlitz, U. (2019). Gradient based hyperparameter optimization in echo state networks. *Neural Networks*, *115*, 23–29.

Tsay, R. S. (2014). *Multivariate time series analysis with r and financial applications.* Wiley.

Variengien, A., & Hinaut, X. (2020). A journey in esn and lstm visualisations on a language task. *arXiv preprint arXiv:2012.01748*.

Xu, C., & Xie, Y. (2020). Conformal prediction interval for dynamic time-series. *arXiv preprint arXiv:2010.09107*.

Xu, D., Lan, J., & Principe, J. C. (2005). Direct adaptive control: an echo state network and genetic algorithm approach. In *Proceedings. 2005 ieee international joint conference on neural networks, 2005.* (Vol. 3, pp. 1483–1486).

Yang, Y., Wu, J., Chen, Y., & Li, C. (2013). A new strategy for short-term load forecasting. In *Abstract and applied analysis* (Vol. 2013).

Zhang, G. P. (2001). An investigation of neural networks for linear time-series forecasting. *Computers & Operations Research*, *28*(12), 1183–1202.