Conform With the Wind

Processing short-term ensemble forecasts with conformal based methods for probabilistic wind-speed forecasting

Simon Althoff

Master's thesis
2023:E61

Lund University

Faculty of Engineering
Centre for Mathematical Sciences
Mathematical Statistics

Centrum Scientiarum Mathematicarum

Conform With the Wind

Processing short-term ensemble forecasts with conformal based methods for probabilistic wind-speed forecasting

Simon Althoff

2023-06-26

LUND UNIVERSITY

LTH

FACULTY OF ENGINEERING

Master's Thesis in Mathematical Statistics

Faculty of Engineering, LTH
Center for Mathematical Sciences, Lund University

Supervisor: Erik Lindström

Examinor: Andreas Jakobsson

Abstract

Forecasting wind has always been an interesting subject, and as large parts of the world are relying more on wind for power production it is becoming even more important to have reliable forecasts. Probabilistic forecasts, where distributions are predicted in contrast to deterministic forecasts, are important for informed decision making. We apply two methods based on conformal prediction for processing ensemble forecasts to well calibrated probability distributions. Conformal prediction is a relatively modern method for quantified uncertainty analysis within machine learning. These methods are compared to the quantile regression forest algorithm, which has been well tested in literature for probabilistic ensemble post processing. Ensemble forecasting is a method based on running several numerical weather prediction models simultaneously, creating an array of forecasts. The conformal methods rely on an additional point forecast, supplied by another model, for producing the distributions while the quantile regression forest works directly on the ensemble. The methods were tested using a teaching schedule which determines the best configuration of parameters, from a predefined set, based on the continuous ranked probability score metric before making each prediction. This is a way of simulating how the methods perform over time. For the conformal methods we employ a normalized version of conformal predictive distribution systems and a non-exchangeable conformal prediction method. For the non-exchangeable case we suggest a method of stacking confidence intervals to produce distributions. We also suggest a normalized version of this algorithm. Both methods show promising results, both able to produce significantly better distributions than the raw ensemble and as good or better calibrated distributions compared to the quantile regression forest. Though the conformal methods are supplied external forecasts and the quantile regression forest is not using an optimal configuration. The conformal methods also produce well calibrated predictions consistently over different setups of the algorithms.

Acknowledgments

There are several people I would like to thank, though it was I who wrote this thesis, it is far from a lonely effort. Firstly I want to thank all the lovely people at Algorithma AB, I am grateful for the warm welcome and great support from you all. Especially, I would like to thank Lars Carlsson for the supervision and all the valuable input during the project and thank you for pushing me to publish an article to COPA. Also thank you Jonathan Anderson and Johan Hallberg Szabadváry for your support, input and good vibes. To my supervisor at Lund University, Erik Lindström, thank you for coming with insightful input throughout the process. I also want to send a thank you to Christoffer Hallgren at Uppsala University, for pointing me in the right direction regarding data early in the project and for teaching me about meteorology.
To all my family and friends who have supported me through this thesis and my time at LTH, thank you, none of this would have been possible without you!

Contents

Chapter 1

Introduction

Wind is a phenomenon with significant impact on our societies. No matter if it is an enabling or disabling factor, we always wish to predict it as well as possible. With the rise of initiatives and policies pushing for the transition to fossil-free energy, wind forecasting has become a very hot topic. As wind energy represents a larger portion of energy production, reliable wind forecasting is paramount for planning other types of energy production [24]. It is also highly important for energy trading markets. While deterministic forecasting still has an important role to play, probabilistic forecasting has risen in popularity. These types of forecasts allow for more flexibility in decision making, enabling one to make optimal choices according to certain probabilities. A lot of research investigating probabilistic forecasting techniques for wind power production has been published in recent years [5]. Several of these are ensemble methods, where one processes ensemble forecasts to produce prediction intervals or probability distributions. Ensemble forecasting is a method of running several numerical models in parallel with slightly different input data, effectively producing as many forecasts as the number of models. The ensembles themselves can naturally act as probabilistic forecasts, by producing empirical distributions from them, though these types of forecasts a typically overly confident [22]. This is where post-processing plays an important role. The post processing can be performed through parametric methods, such as *ensemble model output statistics* (EMOS), or non-parametric methods, such as the *quantile regression forest* (QRF). Many machine learning methods can also extend beyond the ensemble, relying on more variables to create reliable forecasts. The use of different kinds of neural networks has become more common in the past couple of years. Many of these kinds of methods require a lot of computational power and may thus be difficult to employ on a wider scale. Uncertainty analysis and probabilistic and set predictions is also an emerging field within machine learning. *Conformal prediction* [23], which is a somewhat new technique, produces set predictions for both classification and regression problems with guarantees to validity. Conformal predictions can also be extended to produce distribution predictions, they are called *conformal predictive distribution systems* (CPDS). These methods are

typically employed as a supplement to an underlying deterministic forecasting model. It is a straight forward technique that, for some versions, will add very little additional complexity on top of the underlying model, to produce probabilistic forecasts [23]. This is then potentially a way of getting both the benefits of the accuracy of a deterministic forecast and the flexibility of a probabilistic forecast, with very little extra computational complexity. Further, this could perhaps allow for effective employment of probabilistic forecasting on a wider scale, by supplementing it with already existing deterministic models.

1.1 Purpose

The application of conformal prediction and corresponding extensions to wind forecasting seems to be a relatively unexplored subject. It has been used in predicting electricity price in power markets with some success [13]. The purpose of this report is to do an initial analysis of the effectiveness of conformal prediction methods to day ahead probabilistic wind speed forecasting as an ensemble post processing technique. It will be compared to the quantile regression forest method, which is well tested for this application [22]. The conformal methods will be supplied external deterministic forecasts while the QRF will work directly on the ensemble, as it has been used historically. This study is a continuation of the work done in [2].

1.2 Structure

In chapter 2 we will introduce conformal prediction, CPDS and a relevant extension to these as well as the theory behind them. We will also briefly introduce the QRF as well as evaluation metrics. Chapter 3 will contain descriptions of how we implement the methods presented above and how we evaluate their performance. Finally, in chapter 4 we will present the results and in chapter 5 we will discuss these as well as draw conclusions.

Chapter 2

Preliminaries

In this chapter we will introduce main concepts of this study, conformal prediction, and the theory behind it. We will also introduce conformal predictive distribution systems and non-exchangeable conformal prediction. These are the two versions of this concept that are used in the rest of the study. We will also present the baseline method, the quantile regression forest, before finally introducing the two main metrics used for evaluation.

2.1 Conformal prediction and extensions

Conformal prediction, the main point of focus of this report, is a method that quantifies uncertainties in connection to predictions. When encountering new concepts it is always useful to have a mental picture of what is going on. In the case of conformal prediction this is especially suitable since it is a mathematical representation of what one might do in real life. Assume a situation we might encounter regularly where we would predict something. An example could be to predict the temperature outside by how the weather looks like from the inside. Each time we make such a prediction, we could also note how strange the example in question is, based on the outcome of the actual temperature. An example where it is sunny outside and it turns out to be warm might be considered not-so-strange. While an example of sunny weather and cold temperature might be considered the opposite. With sufficient number of examples we could create a picture of what types of situations are easy to predict, which are difficult, how likely certain outcomes are and generally how good our predictive method is.

2.1.1 Exchangeability

A fundamental concept to understand regarding conformal prediction is exchangeability, since this is assumed of the data to ensure validity of the predictions. Exchangeability can be seen as a slightly relaxed version of the standard independent and identically distributed assumption. For exchangeability to hold

all permutations of the ordering of the data needs to have the same joint probability distribution [1]. Formally assume a data set $\{z_1, z_2, \ldots, z_m\}$ drawn from some distribution over the space $\mathbf{Z}^m$. If for any permutation of the order $\pi(i)$ we have that

$$P(z_1, \ldots, z_m) = P(z_{\pi(1)}, \ldots, z_{\pi(m)})$$

then the data is exchangeable. Distribution drifts and time series which have temporal dependencies are examples of non-exchangeable data. We will later introduce a version of conformal prediction that reduces the problems caused by non-exchangeable data.

2.1.2 Conformal Prediction

The base version of conformal prediction produces set predictions. It can be applied to both classification and regression type problems. In the classification case it will produce a set of classes in which the actual label of an example lies with some probability. In regression, which we will stick to in this report, the output is an interval on the real line. Formally we would, for some object $x_n \in \mathbf{X}$, like to produce a prediction range $\Gamma^\epsilon \subseteq \mathbb{R}$ where we can expect the label $y_n \in \mathbf{Y}$ to lie in with probability $1 - \epsilon$. The examples $z_i = (x_i, y_i)$ are assumed to be drawn from the example space $\mathbf{Z} = \mathbf{X} \times \mathbf{Y}$ through some probability distribution Q [23]. The value ϵ marks a significance level of the ratio of errors to tolerate. The prediction range for a test object z_n will be based on a set of training examples $z_i = (x_i, y_i),\ i = 1, \ldots, n-1$. For this we must first define what is called a (non-)conformity function, A, which is a measurable and order invariant (in terms of the training examples) function on $\mathbf{Z}^{(*)} \times \mathbf{Z} \to \mathbb{R}$ ($\mathbf{Z}^{(*)}$ having arbitrary dimension size) that maps an example z_i together with the other examples and a potential test object (x_n, y), to a nonconformity score

$$\alpha_i = A(\{z_1, \ldots, z_{n-1}, (x_n, y)\} \setminus z_i, z_i) \in \mathbb{R}.$$

Here we can choose A as increasing with strange examples, nonconformity, or increasing with non-strange examples, conformity. In the sections below we will use the nonconformity version. The results are the same for the conformity case, though some inequalities have to be flipped. As we will see in a later result, the validity of the prediction range, meaning the amount of actual errors compared to what we can tolerate, is not dependent on how we define A. The efficiency however, meaning the size of the range, may be affected and could thus improve if A is chosen well. After defining nonconformity scores for all training examples we can predict the range for a test object by quantifying how it conforms to the training examples. We produce *p-values* for this test object through a *conformal transducer* f, where

$$p_n^y := f(z_1, \ldots, z_{n,1}, x_n, y) = \frac{|\{i = 1, \ldots n : \alpha_i \geq \alpha_n\}|}{n}.$$

We can then subsequently use this conformal transducer to produce the range prediction

$$\Gamma^\epsilon = \{y \in \mathbf{Y} : p_n^y > \epsilon\}.$$

We can see that a natural way of using this system is with an underlying predictive machine learning model. The nonconformity scores could then be constructed as

$$\alpha_i = |\hat{y}_i - y_i|$$

where $\hat{y}_i$ is the prediction of y_i from the underlying model, which is trained on all the other examples. This version of the algorithm is however computationally costly since it requires the recomputing of all nonconformity scores for new values of y, as well as retraining of the underlying algorithm for each example in the data set. It is therefore common to use an *inductive* version, sometimes called split-conformal, instead [19]. In that case we train the underlying model on a set of data separate from the ones we use to calculate the nonconformity scores. These two sets are commonly called the *proper training set* and the *calibration set*. For the inductive conformal method we present the following property.

Theorem 2.1. *Suppose $z_i = (x_i, y_i)$, $i = 1, \ldots, n-1$ and z_n are exchangeable with corresponding nonconformity scores α_i from nonconformity function $A(z_{past}, z_i)$, z_{past} being a proper training set. We define the p-value for (x_n, y) as*

$$p_n^y = \frac{|\{i = 1, \ldots, n : \alpha_i \geq \alpha_n\}|}{n},$$

where $\alpha_n = A(z_{past}, (x_n, y))$, and the prediction range as

$$\Gamma^\epsilon = \{y \in \mathbf{Y} : p_n^y > \epsilon\}.$$

Then,

$$P(y_n \in \Gamma^\epsilon) \geq 1 - \epsilon.$$

Proof. Here we consider the case where α_i are distinct with probability 1. We also assume that the nonconformity scores are ordered $\alpha_1 < \cdots < \alpha_{n-1}$, without loss of generality. We require that $\epsilon \geq \frac{1}{n}$, otherwise we put $\Gamma^\epsilon = \mathbf{Y}$ which satisfies the theorem. Firstly we observe the following statement

$$y_n \in \Gamma^\epsilon \iff p_n^{y_n} > \epsilon \iff \frac{|\{i = 1, \ldots, n : \alpha_i \geq \alpha_n\}|}{n} > \epsilon$$

where, in this case, $\alpha_n = A(z_{past}, (x_n, y_n))$. Those conditions are true if and only if

$$\alpha_n \leq \alpha_{\lceil n(1-\epsilon) \rceil}.$$

Given the exchangeability of $z_1, \ldots, z_n$ we can state that

$$P(\alpha_n \leq \alpha_k) = \frac{k}{n}$$

for any integer $k \leq n - 1$, i.e. α_n falls between any of the calibration examples $\alpha_1, \ldots, \alpha_{n-1}$ with equal likelihood. This gives us the final statement that

$$P(\alpha_n \leq \alpha_{\lceil n(1-\epsilon) \rceil}) = \frac{\lceil n(1-\epsilon) \rceil}{n} \geq 1 - \epsilon.$$

□

The above proof is a modified version of the proof for Theorem D.1 in [3]. We also have a result regarding the upper limit of validity, in this case we require the joint distribution of the (non-)conformity scores to be continuous. This will avoid ties, just as we assumed in the above proof, the difference is that there exists a general proof for theorem 2.1 where this condition is not required. The general version is however seldom needed in practice since we could simply add a small amount of noise to the scores.

Theorem 2.2. *Assume all conditions from theorem 2.1 are met as well as that the joint distribution of the nonconfirmity scores α_i is continuous. If the prediction range Γ^ϵ is also constructed as in theorem 2.1 then*

$$P(y_n \in \Gamma^\epsilon) \leq 1 - \epsilon + \frac{1}{n}.$$

Proof. Here we observe as in the above case that

$$y_n \notin \Gamma^\epsilon \iff \alpha_n > \alpha_{\lceil n(1-\epsilon) \rceil}.$$

Since all the conditions of the above proof applies we know that

$$P(\alpha_n > \alpha_{\lceil n(1-\epsilon) \rceil}) = 1 - P(\alpha_n \leq \alpha_{\lceil n(1-\epsilon) \rceil}) =$$
$$= 1 - \frac{\lceil n(1-\epsilon) \rceil}{n} > 1 - \frac{n(1-\epsilon)+1}{n} = \epsilon - \frac{1}{n}$$

which means that

$$P(\alpha_n > \alpha_{\lceil n(1-\epsilon) \rceil}) > \epsilon - \frac{1}{n} \iff P(\alpha_n \leq \alpha_{\lceil n(1-\epsilon) \rceil}) \leq 1 - \epsilon + \frac{1}{n}.$$

□

This proof is a modified version of the proof of Theorem 2.2 in [14]. There are similar results for the transductive or full-conformal case, but we will only use the inductive version in this report.

2.1.3 Conformal Predictive Distribution Systems

Conformal prediction creates a framework for producing valid set and range predictions for given significance levels. However, it can be extended to do more, such as producing whole probability distributions as predictions. For this we again use the conformal transducer to produce p-values. However, instead of using the conformity function to measure how an example conforms to the data set, we will measure how it conforms to the property of being large, perhaps larger than the underlying prediction. We could thus potentially define a score according to

$$\alpha_i = \hat{y}_i - y_i$$

to achieve this. Formally we use the transducer as a function Π to arrange p-values such that they mimic a distribution, thus the following criteria must be met:

1. $\Pi((z_1, \ldots, z_{n-1}), (x_n, y))$ is a monotonically increasing function of $y \in \mathbb{R}$
2. $\lim_{y \to -\infty} \Pi((z_1, \ldots, z_{n-1}), (x_n, y)) = 0$
3. $\lim_{y \to \infty} \Pi((z_1, \ldots, z_{n-1}), (x_n, y)) = 1$

In short it needs to fulfill the criteria of a cumulative distribution function in y. To ensure that these criteria are met, as well as to get better validity (exact to be precise, see [23] for details), we add a uniform randomization term $\tau \in [0, 1]$ to our transducer

$$\Pi((z_1, \ldots, z_{n-1}), (x_n, y), \tau) = \frac{|\{i = 1, \ldots n : \alpha_i > \alpha_n\}|}{n} + \\ + \frac{\tau |\{i = 1, \ldots n : \alpha_i = \alpha_n\}|}{n}.$$

The p-values will by definition be monotonically increasing in τ. If we fix $\tau = 0$ as $y \to -\infty$ or $\tau = 1$ as $y \to \infty$ both criteria 2 and 3 are met, given that criteria 1 holds for all calibration sets. These distributions are valid, assuming the data is independent and identically distributed (IID), under the notion that the produced p-values are uniformly distributed on $[0, 1]$, i.e. they are calibrated in probability. An important fact in this case is the following Lemma from [23].

Lemma 2.3. *Let Y be a random variable distributed as a continuous distribution function F on $\mathbb{R}$. If $\Pi : \mathbb{R} \to \mathbb{R}$ is monotonically increasing and the distribution $\Pi(Y)$ is uniformly distributed on $[0, 1]$, then $\Pi = F$.*

Given that the previously mentioned criteria are met then it is natural that the CPDS method produces valid distributions. We refer to [23] for theoretical details. For the inductive version of this algorithm an additional randomization term has to be added to the transducer according to

$$\Pi(z_{past}, (z_1, \ldots, z_{n-1}), (x_n, y), \tau) = \frac{|\{i = 1, \ldots n : \alpha_i > \alpha_n\}|}{n} + \\ + \frac{\tau |\{i = 1, \ldots n : \alpha_i = \alpha_n\}| + \tau}{n}.$$

to attain theoretical validity [23]. However, in practice we can often get good enough results without randomization, especially for large data sets. Though for certain choices of underlying model, randomization is required.

2.1.4 Non-exchangeable conformal prediction

Few real world process can be claimed to be truly exchangeable, however a process might be exchangeable enough to allow for valid predictions. In the

case where the underlying data is not exchangeable enough, we might look to reduce the dependency on this property through some mechanism. Recently there has been an surge of ideas published to tackle this problem. One such idea is, quite naturally, to assign weights to the examples. These weight $w_i \in [0, 1]$ are assigned such that more important or more trusted examples are given a higher weight than other examples [4]. We will call this the non-exchangeable conformal prediction method, or NECP for short. How the weights are assigned is dependent on the specific problem, however in a time-series setting it might be natural to use exponential decay over time. These weights need to be normalized according to

$$\tilde{\omega}_i = \frac{\omega_i}{\omega_1 + \cdots + \omega_{n-1} + 1}, i = 1, \ldots, n-1$$
$$\tilde{\omega}_n = \frac{1}{\omega_1 + \cdots + \omega_{n-1} + 1}$$

before we make a prediction. We can then produce an interval Γ^ϵ as

$$\Gamma^\epsilon = \hat{y}_n \pm \mathbf{Q}_{1-\epsilon}\left(\sum_{i=1}^{n-1} \tilde{\omega}_i \cdot \delta_{R_i} + \tilde{\omega}_n \cdot \delta_{+\infty}\right) \tag{2.1}$$

given an underlying deterministic prediction $\hat{y}_n$ of y_n. The term $\mathbf{Q}_{1-\epsilon}$ means the $1-\epsilon$ quantile of the argument, keeping in line with notation from [4]. The expression δ_{R_i} in (2.1) represents the Dirac delta function in residual $R_i = |\hat{y}_i - y_i|$. Thus, we pick a size of residual such that the sum of the weights of the corresponding previous residuals that are smaller than the new residual, is at least $1-\epsilon$. Since this method assumes a pre-trained underlying predictive model, this is an inductive version of the non-exchaneangle algorithm called the non-exchangeable split conformal method in [4]. The residuals of the predictions act as nonconformity scores in this occasion. Much like in the exchangeable case there are validity guarantees, however now according to

$$\mathbb{P}\{y_n \in \Gamma^\epsilon\} \geq 1 - \epsilon - \sum_{i=1}^{n-1} \tilde{\omega}_i \cdot d_{TV}(R_{splitCP}(Z), R_{splitCP}(Z^i)) \tag{2.2}$$

$$\mathbb{P}\{y_n \in \Gamma^\epsilon\} < 1 - \epsilon + \tilde{\omega}_n + \sum_{i=1}^{n-1} \tilde{\omega}_i \cdot d_{TV}(R_{splitCP}(Z), R_{splitCP}(Z^i)) \tag{2.3}$$

where d_{TV} is the total variation distance between distributions. Meaning that we can limit validity issues caused by non-exchangeability, by assigning low weights to the examples where d_{TV} is high. Here, $R_{splitCP}(Z)$ is a vector function with entries $R_{splitCP}(Z)_i = |y_i - \hat{y}_i|$ and $R_{splitCP}(Z^i)$ is the same but with the ith entry exchanged for the nth one. For clarification, Z is the sequence of random variables for the calibration and test set $\{Z_1, \ldots, Z_n\}$ and Z^i is the same sequence but with the ith and nth points exchanged

$$\{Z_1, \ldots, Z_{i-1}, Z_n, Z_{i+1}, \ldots, Z_i\}.$$

The total variation distance is defined by the largest difference in probability on any subset of the sample space, more formally

$$d_{TV}(\mu, \beta) := \sup_{A \in \Omega} |\mu(A) - \beta(A)|$$

where μ and β are two probability measures defined on the same sample space. If the data Z is exchangeable then the total variation distance is 0, since the conditional distribution, no matter the ordering, is the same in that case. The property (2.3) holds only, like in the standard case, when the residuals are distinct with probability 1. Another interesting result gained from (2.2) and (2.3) is that if we set $\omega_i = 1$ for all i, then we get a picture for what happens to the validity of the intervals of the standard inductive conformal prediction algorithm, if the data is non-exchangeable. The conditions also hold for general (non-)conformity scores, not only the absolute residuals stated here [4].

2.2 Quantile Regression Forest

The quantile regression forest (QRF) algorithm [16] generalizes the random forest [10] algorithm from predicting conditional expectations to conditional quantiles. It is, similar to conformal prediction, a non-parametric method. To understand the QRF algorithm it is useful to first know how the standard random forest works. It is based on a number of decision trees, each of which sort some of the examples $z_i = (x_i, y_i),\ i = 1, \ldots, n-1$ in the data set into different classes based on a random selection θ of features of x_i. The selection of examples to use for each tree comes from bagging the data set. The classes for each tree t takes the form of leafs l_{tj} which corresponds to some subset $\mathbf{X}_{l_{tj}} \subseteq \mathbf{X}$. The leaf in which an example i is put into is a function of the object x_i and the random selection of features θ_t. Passing a new object $x_n \in \mathbf{X}$, the label of which we wish to predict, we get a corresponding leaf $l_t(x_n, \theta_t)$ for each tree in the forest. Weights $\bar{\omega}_i(x_n)$ are then assigned to each training example through

$$\omega_{ti}(x_n) = \frac{\mathbf{1}_{\{x_i \in \mathbf{X}_{l_t(x_n, \theta_t)}\}}}{|\{k : x_k \in \mathbf{X}_{l_t(x_n, \theta_t)}\}|}$$

$$\bar{\omega}_i(x_n) = \frac{1}{T} \sum_{t=1}^{T} \omega_{ti}(x_n)$$

where T is the number of trees in the forest. The prediction of conditional expectation of y_n is then

$$\hat{y}_n = \sum_{i=1}^{n-1} \bar{\omega}_i(x_n) y_i. \tag{2.4}$$

To extend this to produce quantiles we only need a small change to (2.4) according to

$$\hat{F}(y|X = x_n) = \sum_{i=1}^{n-1} \bar{\omega}_i(x_n) \mathbf{1}_{\{y_i \leq y\}}.$$

The QRF is a tried and tested non-parametric method to post-process ensemble forecasts into probabilistic predictions. It has been shown to improve raw ensemble forecasts as well as outperform another baseline method, ensemble model output statistics, which is a parametric method [22].

2.3 Probabilistic Evaluation

We will in this section present two ways of evaluating probabilistic forecasts which will be used later in the report. These are far from the only ways of evaluating these types of predictions and each has their respective drawbacks.

2.3.1 Continuous Ranked Probability Score

A common technique for evaluating probabilistic predictions is the continuous ranked probability score (CRPS) [15]. It is defined through the formula

$$CRPS(F_n, y_n) = \int_{-\infty}^{y_n} F_n(u)^2 du + \int_{y_n}^{\infty} (1 - F_n(u))^2 du \tag{2.5}$$

with F_n being the estimated CDF and y_n the label of the prediction. The best possible value is 0, only attained when F is the Heaviside step-function in y_n, i.e. we are 100 % confident in our point prediction and that prediction turns out to be correct. This method gives a score for each example, so to extend it to a set of predictions one typically calculates the mean CRPS of these [6]. Further in this report CRPS is used interchangeably with the mean CRPS. The reason why CRPS is used is that it gives a general picture of performance while being quick to estimate.

2.3.2 Probability Integral Transform

The CRPS score is a quantitative evaluation, meaning it can tell us the performance between predictions but not the quality of them. The probability integral transform (PIT) evaluation is qualitative, meaning it tells us about the quality of the predictions [6]. From a label y_n and forecast density f_n we attain the PIT s_n as

$$s_n = \int_{-\infty}^{y_n} f_n(u) du = F_n(y_n).$$

We know from Lemma 2.3 that if the random variable S, from which we sample s_n, is uniformly distributed on $[0, 1]$, then f_n is correctly calibrated. Thus to determine the quality of predictions we produce a histogram of PIT scores for a set of predictions which then should, given that the predictions are good, resemble a uniform distribution density function.

Chapter 3

Method

We begin this chapter with explaining the data we use for the study and how it was gathered. We continue on by describing how each method was implemented as well as describing the two naive methods that serve as baseline beyond just the QRF. After that we describe the teaching schedule, which is the algorithm used for testing the methods, and we present two versions of parameter selection used in this algorithm. Finally, we present a description of how we evaluate the results.

3.1 Data

The data we use in this study can be split into three categories, measurements, ensemble forecasts and deterministic forecasts. They span the time January 2, 2022 to January 23, 2023, each measurement made at noon of the given day with forecasts made 24-hours in advance. All data samples were gathered around a small island outside the west coast of Sweden, Måseskär, the coordinates of each type are presented in Table 3.1. The data was subject to some cleaning based on each data type, these procedures are described in the following sections together with information about the data. The final size of the data set after cleaning was 367 points.

Table 3.1: Location coordinates for the different data-types

Data type	Longitude	Latitude
Measurement	58.0937	11.3312
Ensemble	58.101800	11.309300
Deterministic Forecast	58.099915	11.327288

3.1.1 Measurements

This data was gathered from the Swedish Meteorological and Hydrological Institute (SMHI) and their weather station on Måseskär.[1] The choice of station was based on the hope that the local topography would affect the wind minimally, though this could be an interesting topic for further research. Here we gathered the average wind speed (10 meters above ground over 10 minutes) in meters per second for the time-points in question. In the case where a measurement was missing, that time-point was considered useless and was thus discarded.

3.1.2 Ensemble Forecasts

The ensemble forecasts used in the study come from the MetCoOp Ensemble Prediction System [17] (MEPS). It is a collaboration between the meteorological institutes of some of the countries within the coverage region which is a rectangle over the Scandinavian Countries, Finland, the Baltic States and parts of the North Sea and Atlantic Ocean. It is based on the AROME model developed by météo-France and runs 30 numerical weather prediction (NWP) models in parallel, each simulating the atmosphere with slightly different initial conditions. The system thus produces 30 different forecasts and does so every 6 hours. Each forecast contains predictions for every hour up to 61 hours ahead. The system produces predictions for a large amount of atmospheric variables from which we used the following:

1. `x_wind_10m`: average (over 10 min) zonal wind speed in m/s at 10 m
2. `y_wind_10m`: average (over 10 min) meridional wind speed in m/s at 10 m
3. `surface_air_pressure`: Pa
4. `air_temperature_0m`: surface air temperature in K
5. `wind_speed_of_gust`: wind speed of gust in m/s at 10 m

These variables will be referenced as `x_wind`, `y_wind`, `pressure`, `temperature` and `gust` respectively for the remainder of this report. The grid, over which predictions are produced, has a horizontal resolution of 2.5 km where we used one of the closest grid points to the weather station on Måseskär (again see table 3.1 for exact location). We fetched this data from the Norwegian Meteorological institute (MET Norway) and their open data thredds server.[2] At certain points there were missing values in the forecasts. If a variable had missing values corresponding to more than 25 % of the total, i.e. more than 8 points were missing, then that time-point was deemed unusable. If a variable had missing values but less than 25 %, then the missing values were sampled from the non-missing ones.

[1] https://www.smhi.se/data/meteorologi/vind gathered under CC BY 4.0 SE license
[2] https://thredds.met.no/thredds/metno.html gathered under CC BY 4.0 license

3.1.3 Deterministic Forecasts

The conformal prediction methods are most often used as an application on top of a deterministic machine learning algorithm. In this case we used the deterministic forecasts from MET Norway, which are post processed from the ensemble forecasts, as the underlying model. This is a global model, meaning they produce forecasts over the entire region, so we could perhaps create local models that are more accurate. However, they are likely good enough for our use, and are potentially better than what we could produce with very limited data. This means that we must assume inductive versions of the conformal algorithms without explicitly knowing what data the underlying model is trained on. If the model is updated on the data we use to form conformity scores, this could hurt the validity of the results from the conformal methods. However, if this is the case, we can suspect that the potential changes might be small enough to still allow for valid results, since the underlying model is global. The post processed forecasts have a finer grid than the ensemble, 1 km in resolution to be precise, we thus chose a point closer to the weather station (see Table 3.1). From this forecast we only gathered the wind speed variable, again averaged over 10 min at 10 m height. Time-points that had missing deterministic forecasts were, just as in the measurement case, determined to be useless so if that was the case the time-point in question was discarded.

3.2 Models

Below we will describe how each model was implemented, as well as our contributions. There were some things that were done universally for the models. The models all produced a set of evenly distributed quantiles of the cumulative distribution function (CDF). To gain consistency we introduced a lower limit of 0 and an upper limit of 100 to these quantiles, since we know wind speed is non-negative and we do not expect it to be above 100 m/s. If a method produced quantiles below zero, these were all set to zero which produced a CDF that started above 0, see Figures 3.1 to 3.3 for visual representations. This naturally produces incorrect distributions, however this method was chosen as to not skew the median or give incorrect central confidence intervals of the distributions. If necessary, 0 and 100 were added to the ends of the distributions. Finally, they were up or down sampled to 200 points, through linear interpolation, for consistency between methods. This is because the software used for the QRF always supplies a specified number of quantiles. All the visualized predictions in this section were performed with only `x_wind` and `y_wind` as input from the ensemble.

3.2.1 CPDS

Here we used the library Crepes [8] available for python to implement the model. The library implements several versions of both conformal prediction and CPDS. We used the normalized version which scales (non-)conformity scores according

to some difficulty measurement of the prediction. In this case we used a supplied k-nearest neighbors technique to scale the scores [18]. The assumption here is that difficult predictions will have similar predictors x_i and analogously, easy predictions should also have similar predictors. This need not be the case, and as such we might see a decrease in sharpness of the forecasts due to this. The conformity scores were constructed as

$$\alpha_i = \frac{y_i - \hat{y}_i}{\kappa_i + \gamma}$$

where κ_i is the mean of the absolute residuals of the k-nearest neighbors of x_i and $\gamma = 0.01$. The distribution was then constructed through

$$\Pi_n = \hat{y}_n + (\kappa_n + \gamma)\vec{\alpha}$$

where $\vec{\alpha}$ is the array of (non-)conformity scores sorted in ascending order. The number k nearest neighbors to include we set as a hyperparameter to the system. The length of data window, i.e. the amount of past examples to include in the calibration before a prediction, was also set as a hyperparameter. This choice comes from the assumption that there might be a distribution drift in time, meaning that we might benefit from excluding past examples beyond a certain point. The predicted distribution of the 56th example in the data set after calibrating on all the previous examples, using $k = 5$, is shown in Figure 3.1.

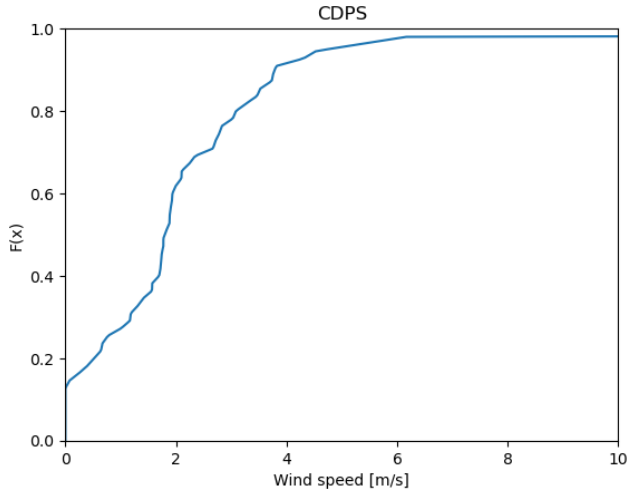


Figure 3.1: Visualization of the CDF for point 56, produced by the CPDS system using $k = 5$ and calibrating on all previous examples.

3.2.2 NECP

The NECP method, in its current form, only produces range predictions so to get distributions we need to stack prediction intervals of increasing confidence

levels. We produced $m = \min[\lfloor (n-1)/2 \rfloor, 100]$ intervals from $1/m$ in confidence to $1 - 1/m$. Apart from the conditions (2.2) and (2.3) for the intervals, we do not have any theoretical guarantees for this kind of distribution though they might prove to be useful in practice. Since we are dealing with a phenomenon which might have a seasonal component we employed exponential decay for the weights, meaning that the weights were defined through

$$\omega_i = \lambda^{n-i}, \ \lambda \in [0, 1]$$

where λ is a forgetting factor and $i = 1, \ldots, n-1$. We thus assume that there is a distribution drift in the wind and that more recent observations represent the underlying distribution better. There might be better options available, such as a sinusoidal scheme which also gives high weights to observations of the same season from previous years. However, that is beyond the scope of this report, especially since we have limited data. If there is no or very little drift in distribution, the employed scheme might instead worsen predictions since we put higher trust in fewer observations. Above only using the residuals as nonconformity score, we also developed our own score which takes inspiration from the normalized (non-)conformity scores from [20] and [18]. These are defined through

$$\alpha_i = |y_i - \hat{y}_i|(1 + \beta^T \hat{\sigma}[\mathbf{x_i}])$$

with β being an array of scaling factors and $\hat{\sigma}[\mathbf{x_i}]$ being a function $\mathbf{X} \to \mathbb{R}^v$ with v being the number of variables in the ensemble. The function $\hat{\sigma}[\mathbf{x_i}]$ should produce a value for each variable signifying the difficulty of a certain prediction. A natural choice (and the reason for the notation) would be the standard deviation of the given ensemble variable. We assume here that a larger deviation to the variables signifies instability in the weather, meaning the prediction should be more difficult to make. If this is not the case, we will likely see a decrease in the sharpness of the forecasts, though theoretically validity should still hold. The array β is an additional hyperparameter to λ. Technically $\hat{\sigma}[\mathbf{x_i}]$ would also be a hyperparameter but we kept it consistent over all tests, producing the standard deviations of the ensemble variables using the maximum likelihood estimate. We call this version of the algorithm the non-exchangeable conformal prediction normalized (NECP-N). To attain an interval for a given confidence level we

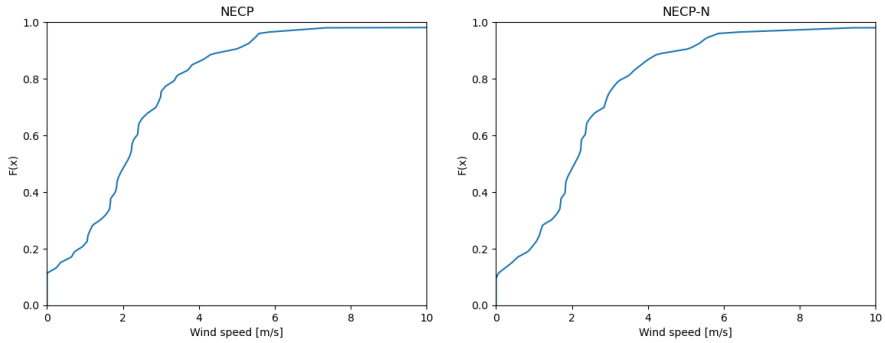


Figure 3.2: Visualization of the CDF for point 56, produced by NECP and NECP-N respectively with $\lambda = 0.99$ and in the case of NECP-N $\beta = [0.1, 0.1]$ with $\hat{\sigma}[\mathbf{x_i}]$ being the standard deviation of the two wind components in the ensemble.

need the following expression

$$\Gamma^{\epsilon} = \{y \in \mathbf{Y} : A(\{z_1, \ldots, z_{n-1}\}, z_n) \leq \mathbf{Q}_{1-\epsilon}\left(\sum_{i=1}^{n-1} \tilde{\omega}_i \cdot \delta_{\alpha_i} + \tilde{\omega}_n \cdot \delta_{+\infty}\right)\} \implies$$

$$|y_i - \hat{y}_i|(1 + \beta^T \hat{\sigma}[\mathbf{x_i}]) \leq \mathbf{Q}_{1-\epsilon}\left(\sum_{i=1}^{n-1} \tilde{\omega}_i \cdot \delta_{\alpha_i} + \tilde{\omega}_n \cdot \delta_{+\infty}\right) \implies$$

$$|y_i - \hat{y}_i| \leq \frac{1}{(1 + \beta^T \hat{\sigma}[\mathbf{x_i}])}\left(\mathbf{Q}_{1-\epsilon}\left(\sum_{i=1}^{n-1} \tilde{\omega}_i \cdot \delta_{\alpha_i} + \tilde{\omega}_n \cdot \delta_{+\infty}\right)\right) \implies$$

$$\Gamma^{\epsilon} = \hat{y}_i \pm \frac{1}{(1 + \beta^T \hat{\sigma}[\mathbf{x_i}])}\left(\mathbf{Q}_{1-\epsilon}\left(\sum_{i=1}^{n-1} \tilde{\omega}_i \cdot \delta_{\alpha_i} + \tilde{\omega}_n \cdot \delta_{+\infty}\right)\right).$$

Produced distributions for point 56 in the data set from both NECP and NECP-N are shown in Figure 3.2. Here $\lambda = 0.99$ for both and in the case of NECP-N, $\beta = [0.1, 0.1]$ and $\hat{\sigma}[\mathbf{x_i}]$ is the estimated standard deviations of `x_wind` and `y_wind`, respectively.

3.2.3 QRF

Implementation of the QRF algorithm was handled by the quantile-forest library by Zillow.[3] The QRF allows for a very large amount of hyperparameters but for simplicity we kept most at default, only modifying the number of trees in the forest. Above the QRF's own hyperparameter we also implemented a data window length parameter, like for the CPDS. The library does it's own interpolation, thus we only requested 198 evenly spaced quantiles from 1/200 to

[3] https://github.com/zillow/quantile-forest

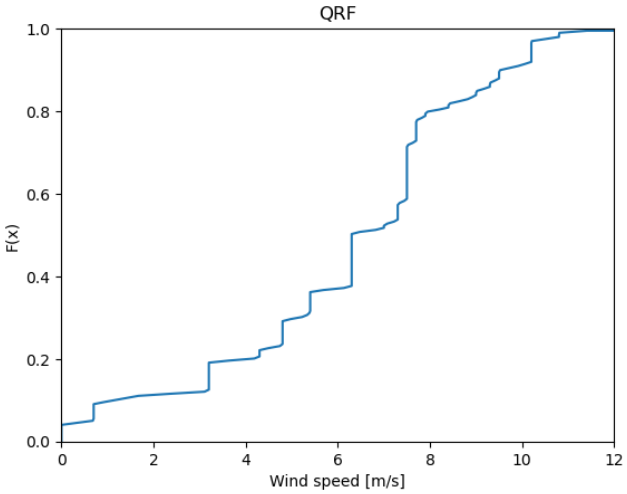


Figure 3.3: QRF with 100 trees trained on first 55 examples predicting the CDF of the 56th example.

$1 - 1/200$ and added on the endpoints ourselves. With 100 trees trained on the first 55 examples, the predicted CDF of point 56 is presented in Figure 3.3.

3.2.4 Naive methods

To attain a frame of reference for the studied methods we additionally employed two naive methods which serve as baseline above the QRF. The first is based on constructing distributions from the raw ensemble. This was done by first calculation the resultants from the ensemble variables `x_wind` and `y_wind`, which then were sorted in ascending order. The resultants then formed a distribution, and to match the other methods, the endpoints were added on and linear interpolation was performed. This method thus serves as a post-processing baseline, showing performance with no post-processing and is subsequently called the raw method. The other method is a completely naive method which forms predictions with none of the input. Constructing such methods can be done in many different ways, in this case we chose to use the current wind speed as our prediction for the next day. This creates a point forecast, so we needed to extend it to produce a distribution. We did so by doing linear interpolation between 0 and our point prediction, creating 100 evenly distributed quantiles. Since the final distribution has 200 quantiles, this means that we predict that there is a 50 % chance the actual wind will be below the point forecast. Similarly we interpolated the second half of the distribution. However, since speeds close the maximum value are very rare, we interpolated 95 points to the speed 20 m/s and the remaining 5 points were evenly interpolated between 20 and 100. If however, the point forecast was above 20, the entire second half was simply interpolated

evenly to 100. We call this method the Naive method for the remainder of the report.

3.3 Teaching schedule

To mimic a real world scenario we tested different models through a teaching schedule [23]. It works by, before each new prediction, determining which combination of hyperparameters, from some set, performs best on the already known data. Then it uses that combination to make the prediction and the process is repeated. To determine which combination is the best some metric is used which in this case was CRPS. The algorithm can be summarized according to the following structure:

1. Split the data into an initial training set and a test set.
2. Use a parameter selection algorithm to determine which combination of hyperparameters performs best on the data in the training set.
3. Train the best model on the training data, predict the first point in the test set and save the prediction.
4. Move the first point in the test set to the end of the training set.
5. Repeat steps 2 to 4 until the test set is empty.

We implemented two versions of parameter selection algorithms, based on the fact that some of the models required retraining when encountering new data. The NECP and NECP-M methods used what we call sequential parameter selection, which mimics what is done in the teaching schedule. The CPDS and QRF methods instead used a block parameter selection, which predicts blocks of data instead of single data points. These procedures are explained in detail below.

3.3.1 Sequential parameter selection

This version mimics what is done in the over arching teaching schedule, by predicting one example at a time in chronological order. This was used for the NECP and NECP-N models, since we had control over that implementation we could add on new training examples without having to retrain the system from scratch. This algorithm trains the model on some initial data which we call the training subset (to make a distinction from the training set in the teaching schedule). It then makes a prediction, and then adds the predicted example to the training subset. This procedure is performed until all points in the current training set, from the teaching schedule, lies in the training subset. The CRPS is calculated for the predictions and the algorithm is run again for the next combination of hyperparameters. The combination with the lowest CRPS is the one selected as the best. The algorithm can be summarized through:

1. Split the available data into some initial training subset and test subset.
2. Train the model on the training subset.
3. Predict the first point in the test subset and save the prediction.
4. Move the predicted point from the test subset to the end of the training subset.
5. Repeat steps 2 to 4 until the test subset is empty.
6. Calculate the CRPS of the predictions.
7. Repeat steps 1 to 6 for all combinations of hyperparameters.
8. Select the combination which gave the lowest CRPS.

The initial training subset was always chosen to be small, in this case $5 \mod n-1$, which has a parallel to what we do in the block parameter selection. This might give very poor predictions in the beginning, however since it was only used for parameter selection it was deemed acceptable.

3.3.2 Block parameter selection

We used this version of the parameter selection algorithm for the CPDS and QRF cases. These algorithms were retrained each time they encountered new training data, which would make the sequential parameter selection very computationally costly and would thus limit testing. This algorithm works similarly to the first, the difference being that the available data is split into k blocks. It then trains on the all other data and predicts the entire block in one go. This algorithm can be summarized through:

1. Split the available data into k equal size blocks.
2. Concatenate the data before the first block to the end of the data after the block to form a training subset.
3. Train the model on the training subset, then predict the examples in the block and save those predictions.
4. Repeat steps 2 to 3 for all blocks.
5. Calculate the CRPS of the predictions.
6. Repeat steps 1 to 5 for all combinations of hyperparameters.
7. Select the combination which gave the lowest CRPS.

The rest of the initial split i.e. $k \mod n-1$ was always used for training. In testing k was always set to 5.

3.4 Evaluation

3.4.1 CRPS

We used the CRPS score for both parameter selection and model evaluation. It requires the estimated CDF which is attained from

$$\hat{F}(x) = \sum_{i=0}^{m-1} \frac{1}{m} \cdot \mathbf{1}_{\{Q_{i/m} \leq x\}} \tag{3.1}$$

since the models produced estimated quantiles of the distribution. Here m is the number of produced quantiles i.e. 200 and $Q_{i/m}$ is the i/mth quantile of the distribution. The CRPS was then calculated by inserting (3.1) into (2.5) and numerically integrating through the use of the trapezoidal rule.

3.4.2 Validity and width of intervals

While the CRPS score is a quantified measure for the performance of the distributions, it does not give any details about how these distributions look. Thus, we supplement the score by looking at the validity and width of two symmetric prediction intervals around the median of the distributions. This gives a sample picture of how well the predicted quantiles match the observations. In this study we chose to observe the 50 % and 90 % prediction intervals. The intervals were constructed by selecting the area between the corresponding quantiles. So, in the case of the 50 % interval, we selected the area between $Q_{0.25}$ and $Q_{0.75}$, i.e. between the points below which we can expect 25 % and 75 % of observations to fall, respectively. Since the estimated distributions had a limited number of quantiles, the closest available quantiles to the desired ones were chosen. The 90 % prediction intervals were chosen in a similar manner with $Q_{0.05}$ and $Q_{0.95}$. To test the validity of these intervals we observe the ratio of observations that fall within them. If a distribution is well calibrated, we expect this ratio to be close to the corresponding confidence level. Further, to gain a picture of the sharpness of the distributions, meaning how narrow they are, we calculate the average width of these intervals over the predictions.

3.4.3 PIT histogram

For each teaching schedule we produced a PIT histogram over the corresponding set of predictions. This was done by simply inserting the label of an example y_i into it's estimated CDF (3.1). The produced values were then put into a histogram of 20 equal size bins. Each bin, under the assumption that the PIT distribution is uniform, can be viewed as coming from a binomial distribution $B(m, 1/20)$, with m being the number of predictions. The collection of bins could thus be tested for uniformity through a χ^2 test, with degrees of freedom $m - 1$ [7], since we did not estimate any parameters of the distribution.

Chapter 4

Results

This chapter presents the results of the study. It begins by introducing the configurations used for the different methods as well as the statistics from the best performing of these. We also supply a more comprehensive description of the results for each of the main methods. Then we present a visualization of the predictions from each of the methods. Lastly, we present the results regarding calibration in probability with plots of some of the PIT histograms as well as the χ^2 test statistics.

4.1 Method configurations and statistics

Below follows the results of the analysis of the different methods. Certain methods were analyzed more extensively than others due to the stark difference in computation complexity between certain methods, and that some of the methods gave promising results when adjusting the input and parameters. The initial data split for the teaching schedule was January 2, 2022 to March 1, 2022 for the training set and the rest for the testing set. That gave the initial training set 55 examples and the test set 312 examples. The complete results for each method is presented in Appendix A. There, for each method we supply two tables, one presenting the configurations tested and one presenting the result statistics of each of these. The configurations are defined through input variables from the ensemble as well as the combinations of hyperparameters used in parameter selection. Each configuration is stated with a Ratio and Time parameter. The Ratio represents the ratio of predictions performed by the corresponding hyperparameter combination in the teaching schedule, i.e. the ratio of times that combination was chosen in the parameter selection. This metric thus shows what hyperparameters, given the input, performs best in the CRPS metric for that set. The Time parameter is the execution time for the teaching schedule for that configuration.[1] Note here that the NECP(-N) case uses a different parameter selection method than the CPDS and QRF cases as explained in 3.3.

[1] Performed on a Macbook Air M1 2020 on a single core.

The statistics presented are what we call Val 0.9 and 0.5, which is the coverage of the corresponding prediction intervals, Width 0.9 and 0.5 which is the average width of those intervals and lastly the CRPS score over the predictions. They are all presented with respective 95 % confidence intervals. An important note regarding the validity statistics is that, due to how the distributions are constructed, the target coverages are actually 0.8995 and 0.4975 instead of 0.9 and 0.5 respectively. The results of the best configuration of each method for each of the statistics Val 0.9, Val 0.5 and CRPS are presented in Table 4.1 while their configurations are presented in Table 4.2. The validity metrics were chosen over width since width is only interesting in the case where the predictions are valid. The results of the baseline methods, Raw and Naive, are also presented in Table 4.1.

Table 4.1: Statistics with 95 % confidence intervals from the naive methods and the best performing configuration of each method for each of the metrics Val 0.9, Val 0.5 and CRPS. The metric in which each configuration performs best is written in bold. For the full description of each configuration as well as all results, see Appendix A.

Method	Val 0.9	Width 0.9	Val 0.5	Width 0.5	CRPS
Raw	0.827 ± 0.042	4.091 ± 0.190	0.378 ± 0.054	1.513 ± 0.082	0.753 ± 0.054
Naive	1 ± 0	18.603 ± 0.018	0.837 ± 0.041	10.242 ± 0.010	2.349 ± 0.093
CPDS 6	$\mathbf{0.910} \pm 0.032$	6.541 ± 0.291	$\mathbf{0.500} \pm 0.055$	2.060 ± 0.096	0.881 ± 0.071
CPDS 8	0.926 ± 0.029	6.198 ± 0.148	0.513 ± 0.055	2.023 ± 0.052	$\mathbf{0.863} \pm 0.070$
NECP(-N) 1	0.929 ± 0.028	6.094 ± 0.038	0.503 ± 0.055	2.042 ± 0.010	$\mathbf{0.864} \pm 0.070$
NECP(-N) 3	$\mathbf{0.923} \pm 0.030$	6.080 ± 0.039	0.516 ± 0.055	2.053 ± 0.015	0.873 ± 0.072
NECP(-N) 5	0.936 ± 0.027	6.255 ± 0.057	$\mathbf{0.500} \pm 0.055$	2.033 ± 0.016	0.870 ± 0.070
QRF 1	$\mathbf{0.891} \pm 0.034$	5.924 ± 0.221	0.526 ± 0.055	2.467 ± 0.135	0.906 ± 0.075
QRF 4	0.869 ± 0.037	4.814 ± 0.141	$\mathbf{0.506} \pm 0.055$	1.943 ± 0.083	0.785 ± 0.062
QRF 5	0.869 ± 0.037	4.622 ± 0.141	0.510 ± 0.055	1.902 ± 0.082	$\mathbf{0.776} \pm 0.061$

The methods were all tested with the following combinations of ensemble variables as input:

1. `x_wind`, `y_wind`
2. `x_wind`, `y_wind`, `pressure`
3. `x_wind`, `y_wind`, `pressure`, `temperature`
4. `x_wind`, `y_wind`, `pressure`, `temperature`, `gust`
5. `x_wind`, `y_wind`, `gust`

For the CPDS and QRF methods, combination 5 was also tested in a reduced version. In this case only the quantiles $Q_{0.1}$, $Q_{0.5}$ and $Q_{0.9}$ of each variable was supplied as input, a method inspired by [22].

4.1.1 CPDS

The CPDS method was first tested with fixed $k = 5$ for the k-nearest neighbors difficulty estimate but with varying data window length (WL). The tested data

Table 4.2: Input and hyperparameter configurations of the best performing method configurations according to metrics in Table 4.1, together with execution time of each teaching schedule. The ratio of each parameter combination used for prediction in the teaching schedule are presented in the Ratio column. The input in red() means reduced to the first, fifth and ninth deciles.

Method	Input	Sets of Parameters	Ratio	Time
CPDS 6	red(`x_wind`) red(`y_wind`) red(`gust`)	$\{\text{WL} = \text{all}, k = 5\}$ $\{\text{WL} = 200, k = 5\}$ $\{\text{WL} = 100, k = 5\}$ $\{\text{WL} = 50, k = 5\}$	0.647 0.096 0.240 0.016	05:20
CPDS 8	red(`x_wind`) red(`y_wind`) red(`gust`)	$\{\text{WL} = \text{all}, k = 15\}$ $\{\text{WL} = 200, k = 15\}$ $\{\text{WL} = 100, k = 15\}$ $\{\text{WL} = 50, k = 15\}$	0.330 0.179 0.385 0.106	05:26
NECP(-N) 1	`x_wind` `y_wind`	$\{\lambda = 1, \beta = 0\}$ $\{\lambda = 0.995, \beta = 0\}$ $\{\lambda = 0.99, \beta = 0\}$ $\{\lambda = 0.98, \beta = 0\}$ $\{\lambda = 0.97, \beta = 0\}$	1 0 0 0 0	07:43
NECP(-N) 3	`x_wind` `y_wind`	$\{\lambda = 0.99, \beta = [0, 0]\}$ $\{\lambda = 0.99, \beta = [0.05, 0.05]\}$ $\{\lambda = 0.99, \beta = [0.1, 0.1]\}$	1 0 0	05:01
NECP(-N) 5	`x_wind` `y_wind`	$\{\lambda = 0.99, \beta = [0.05, 0.05]\}$ $\{\lambda = 0.99, \beta = [0.1, 0.1]\}$ $\{\lambda = 0.999, \beta = [0.05, 0.05]\}$ $\{\lambda = 0.999, \beta = [0.1, 0.1]\}$	0.115 0 0.885 0	06:36
QRF 1	`x_wind` `y_wind`	$\{\text{WL} = 100, T = 200\}$ $\{\text{WL} = \text{all}, T = 100\}$ $\{\text{WL} = \text{all}, T = 200\}$	0.237 0.006 0.756	43:39

QRF 4	`x_wind`
	`y_wind`
	`temperature`
	`pressure`
	`gust`
		$\{\text{WL} = 100, T = 200\}$
		$\{\text{WL} = \text{all}, T = 100\}$
		$\{\text{WL} = \text{all}, T = 200\}$
			0.391
			0.038
			0.571
				1:07:39

QRF 5	`x_wind`
	`y_wind`
	`gust`
		$\{\text{WL} = 100, T = 200\}$
		$\{\text{WL} = \text{all}, T = 100\}$
		$\{\text{WL} = \text{all}, T = 200\}$
			0.417
			0.048
			0.535
				49:01

windows were 200, 100, 50 as well as an `all` option which had no limit on the window length. The `all` option was generally most favored by the parameter selection followed by 100, 200 and last 50. For the reduced data case the same window lengths were tested but with different values to k which were 5, 10, 15, 20 and 30. The method responded well to the reduced data and performed the best in that case according to the Validity and CRPS statistics. The configuration with $k = 5$ was best in both Val 0.9 and 0.5 while $k = 15$ gave the best results in CRPS. These results are presented in Table 4.1 as CPDS 6 and 8 respectively with the configurations in Table 4.2. The longest execution time was 05:27 (min:s) for four combinations in the parameter selection giving a time of about 01:22 per combination.

4.1.2 NECP(-N)

Firstly the pure NECP method, which is independent of input variables, was tested with the forget-factors 1, 0.995, 0.99, 0.98 and 0.97. The configuration with forget-factor 1 i.e. the standard conformal method, was exclusively favored in the parameter selection. This was the case for the NECP-N version as well, exclusively picking the configurations with $\lambda = 1$ and $\beta = \mathbf{0}$ anytime it was included in the set of possible configurations. It did so since this performed best in CRPS, the results of which is thus presented in Table 4.1 as NECP(-N) 1. Configurations with forced forget factors 0.99 and 0.999 were also tested. These were tested with three different values to β namely 0, 0.05 and 0.1, meaning β could take the form of $[0, 0]$, $[0.05, 0.05]$ or $[0.1, 0.01]$ but with varying lengths according to input. Also in this case the parameter selection exclusively favored $\beta = \mathbf{0}$. However, with forget factor 0.99 the NECP model performed the best in Val 0.9 which is represented as NECP(-N) 3 in Table 4.1. A forced normalized configuration was also tested with β values 0.05 and 0.1 paired with forget factors 0.99 and 0.999, meaning four possible combinations. The favoring of the parameter selection was mixed between the forget factors though it exclusively picked 0.05 for beta. This version, with only the wind components as input, performed best in Val 0.5 which is represented as NECP(-N) 5 in Table 4.1. Additionally a forced version with β values 0.5 and 1 was tested for the case with wind components and gust as input, though it did not perform better in the used statistics. The specifications of the referenced configurations can be found in Table 4.2. The slowest execution for this teaching schedule was 1:44 per combination.

4.1.3 QRF

The QRF was tested across the board with three different hyperparameter combinations. Two combinations with 200 trees (T) and data window lengths of 100 and `all` respectively. The third combination had 100 trees and `all` in window length. The combination with 200 trees and no limit to data was generally favored in the parameter selection, with the other 200 tree combination coming in second. With only wind components as input it performed best in Val 0.9,

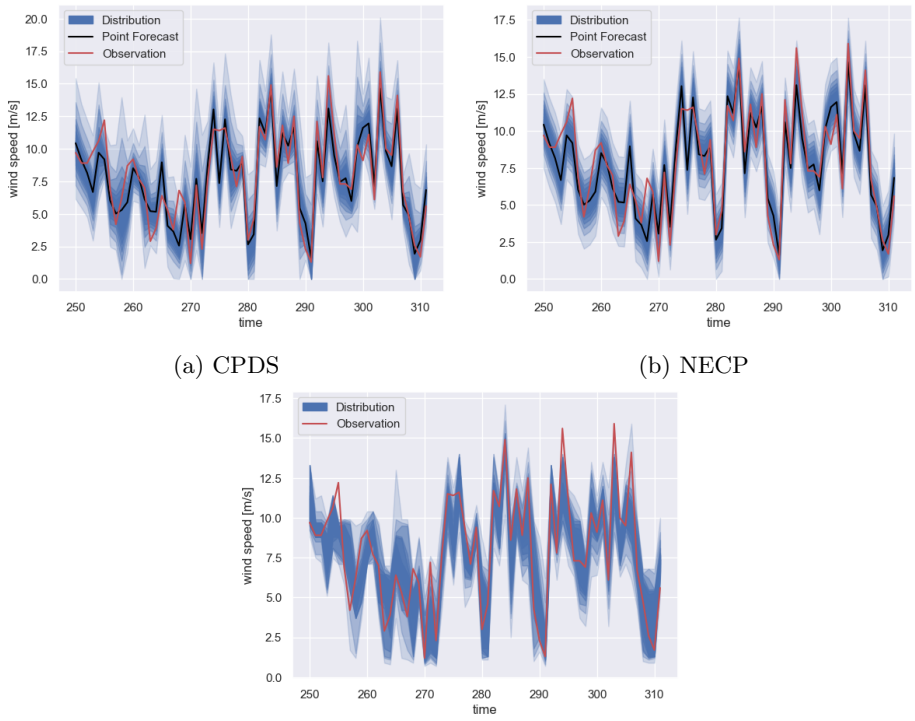
and when all the variables were included it performed best in Val 0.5. These are QRF 1 and 4 in Table 4.1 respectively. Only wind and gust as input gave the best performance in CRPS, presented as QRF 5. The setups of these configurations can also be found in Table 4.2. The QRF was also tested with the reduced version of this input, but it performed significantly worse. The execution time for the QRF was highly dependent on the number of input variables, with the slowest being the full input with 22:33 per combination and the quickest the reduced version with 7:39 per combination.

4.2 Visualization

For a visual reference we present plots of the observed wind together with predictions from the three main methods of the study CPDS, NECP and QRF. The used configurations are the first of each, using only the wind components (see Appendix A for details). The displayed points are test point 250 and beyond, since these predictions are based on the most amount of data. The plots are all displayed in Figure 4.1. The plotted distributions are sequentially larger central predictions intervals, constructed as the ones described in 3.4.2.

4.3 Calibration in probability

Calibration in probability is measured in PIT histograms in this report. It supplements the statistics in the previous presented with a visualization of how well the produced distributions represent the actual measurements. The PIT histograms were constructed with 20 bins. The expected value should then be $312/20 = 15.6$. We can also construct a 95% confidence interval by calculating x such that $P(X \leq x) = 0.025$ and $P(X \leq x) = 0.975$ if $X \sim B(312, 1/20)$, which is 9 and 24 respectively. These levels are represented by red dashed lines in the plots. Here we have selected a handful of PIT histograms of the methods from the previous section based on interesting characteristics, see Appendix B for all histograms. The χ^2 statistics and p-values for all configurations are presented in Table 4.3. The p-values represent the hypothesis test with H_0 that the bins of each histogram comes from a uniform distribution, based on the χ^2 test.
We start with the resultants of the naive methods which are presented in Figure 4.2.
From the CPDS method we present PIT histograms for configurations 6 and 8 which were the best performers in Table 4.1. We have also chosen to include the histograms for 4 and 5 as well, since these performed the worst and best in χ^2 respectively. Configuration 4 is the one that uses all input variables and 5 only uses wind and gust. These are presented in Figure 4.3. Note that none of the configurations of the CPDS histograms can be discarded with 95% confidence in the hypothesis test.
For NECP(-N) we present configurations 1, 3 and 5, again the best performers in Table 4.1. No other configuration gave a significantly worse χ^2 score, though



(a) CPDS

(b) NECP

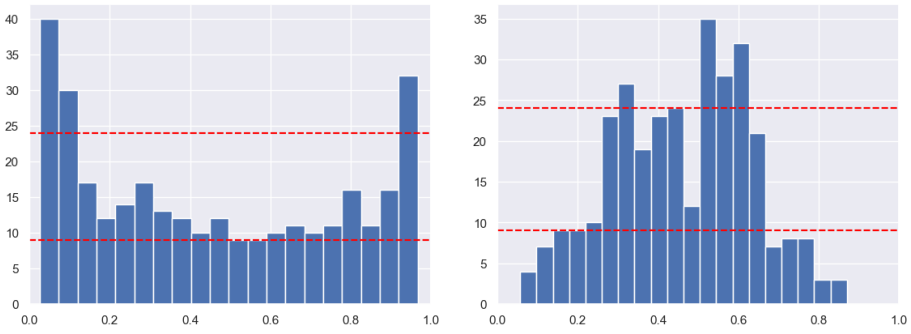
(c) QRF

Figure 4.1: Visualizations of the predictions made by CPDS, NECP and QRF all using corresponding configuration 1.

configuration 22 with significantly increased β, gave a much lower score and is thus also included. These histograms are presented in Figure 4.4. Again note that none of the configurations in this case can be discarded.
Finally for the QRF we once again present the best performers from Table 4.1 which are 2, 4 and 5. Here we note that it is only configuration 6, with the reduced input, that can be discarded as non-uniform with 95 % confidence. However, we can also note that configurations 1 and 3 are relatively close to that threshold. The relevant histograms are presented in Figure 4.5.

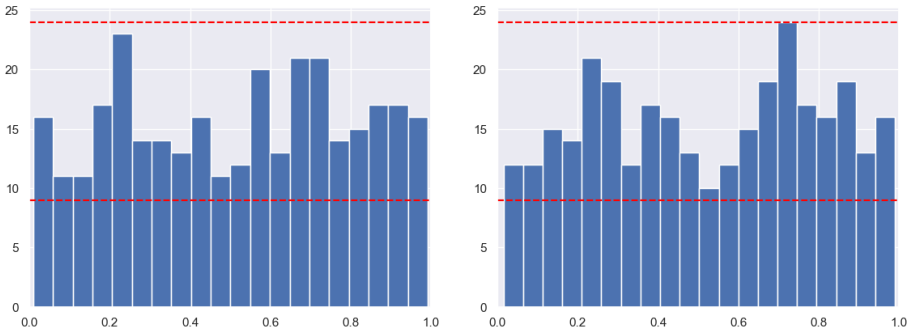
Table 4.3: χ^2 statistic with p-values of all configurations from the PIT histograms. NECP configurations 1,3 and 4 share results with the configurations not represented. Note that the Raw configuration on the QRF line represents the Raw ensemble i.e. not a QRF model. See appendix A for details about each configuration.

CPDS	1	2	3	4	5	6	7
χ^2	23.128	18.256	27.744	28.641	13.769	15.179	24.667
p	0.232	0.505	0.088	0.072	0.797	0.711	0.172
CPDS	8	9	10				
χ^2	15.308	16.462	28.513				
p	0.703	0.626	0.074				
NECP(-N)	1	3	4	5	9	13	17
χ^2	18.128	17.359	16.718	18.000	15.308	15.436	18.256
p	0.514	0.566	0.609	0.522	0.703	0.695	0.505
NECP(-N)	21	22					
χ^2	19.667	11.718					
p	0.415	0.897					
QRF	1	2	3	4	5	6	
χ^2	28.897	23.128	29.154	21.718	25.949	53.513	
p	0.068	0.232	0.064	0.299	0.132	$4 \cdot 10^{-5}$	
Other	Raw	Naive					
χ^2	87.744	128.000					
p	$8 \cdot 10^{-11}$	$3 \cdot 10^{-18}$					



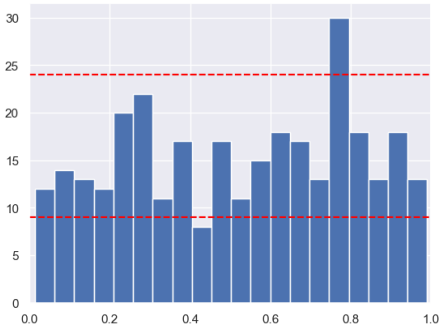
(a) Raw (b) Naive

Figure 4.2: PIT histograms of the distributions created from the raw ensemble and the Naive predictor.

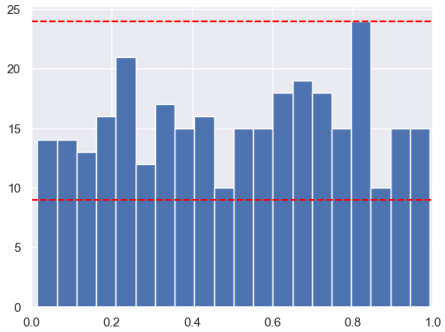


(a) CPDS 6

(b) CPDS 8

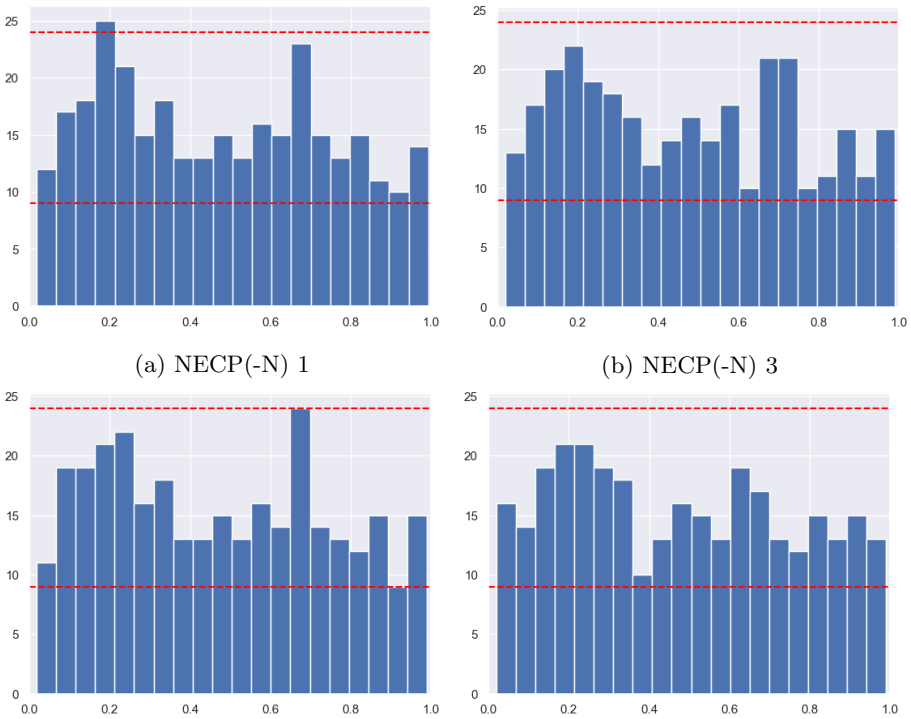


(c) CPDS 4



(d) CPDS 5

Figure 4.3: PIT histograms from CPDS configurations 6, 8, 4 and 5 details of which can be found in Table A.1.



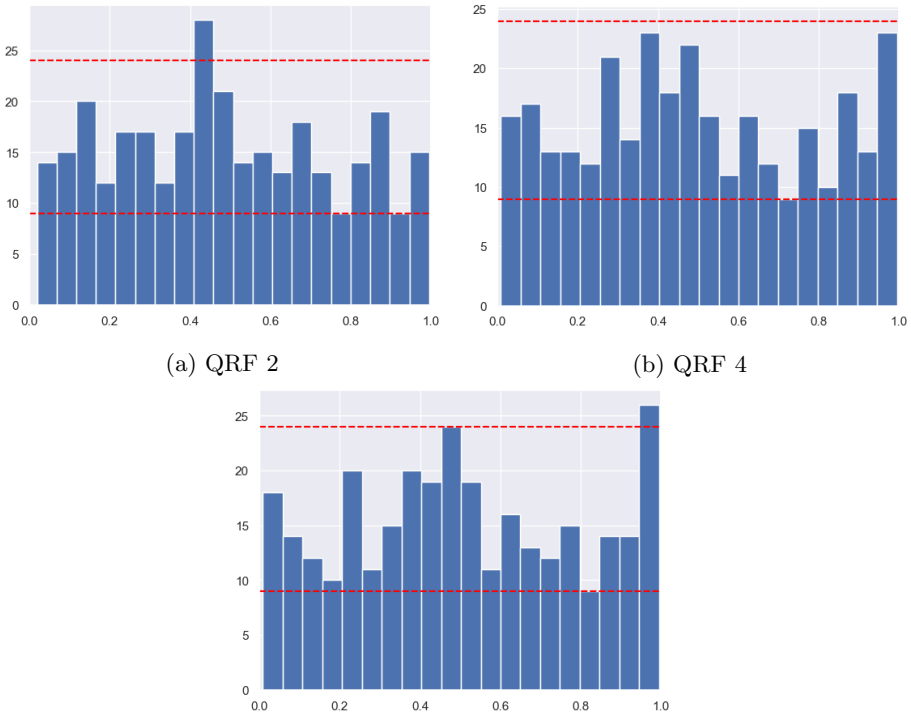
(a) NECP(-N) 1

(b) NECP(-N) 3

(c) NECP(-N) 5

(d) NECP(-N) 22

Figure 4.4: PIT histograms of NECP(-N) configurations 1, 3, 5 and 22 details of which can be found in Table A.3.



(a) QRF 2

(b) QRF 4

(c) QRF 5

Figure 4.5: PIT histograms from QRF configurations 3, 4 and 5 details of which can be found in Table A.5.

Chapter 5

Discussion

This final chapter presents the analysis of the results as well as concluding remarks. We begin by discussing the CPDS and NECP(-N) methods, their results and how they compare to the other methods. Then we take a quick review of how the QRF could be improved for further testing. We also conduct a discussion of the potential pitfalls of using CRPS as a metric in this study. Finally, we give suggestions for further research before drawing the overall conclusions of the study.

5.1 CPDS

The CPDS method is an interesting alternative to current methods since it offers theoretical guarantees of correct distributions. These guarantees of course only hold under the assumption of IID (implicitly exchangeable) data, something we cannot ensure in this case. Further, since we use an inductive approach, we assume that the underlying model is not trained on the calibration data and that it handles the data symmetrically. These are yet mire assumptions we cannot guarantee, which again might hurt the validity of the results. The best configuration in validity, CPDS 6, seems to give very good intervals for both 0.9 and 0.5, though the intervals are somewhat wider than other methods. Looking at the results in Table A.2 we see that the method generally has good coverage for these intervals. The method is somewhat conservative in the 0.9 case, sometimes the confidence interval does not cover the target value, while the results are mixed for 0.5, though in this case the target is always covered by the intervals. In general we can expect conformal based systems to be conservative in validity according to Theorems 2.1 and 2.2. However, with randomization we can expect exact validity theoretically [23], although Crepes [8] does not implement this. Looking at the PIT histograms in Figure 4.3 we can conclude that 6 and 8 look good. Configuration 4 seems to perform significantly worse in calibration, in line with the χ^2 score in Table 4.3. While CPDS 5 does not seem to improve calibration noticeably when looking at the plots in Figure 4.3. The

choice of (non-)conformity score should in theory not affect the validity of the algorithm, and although there is a clear difference in the histograms, all versions still hold under the hypothesis test. We might in this case see less difference in calibration asymptotically, while for a limited data set as this is, might notice larger differences between scores. At least we can see that reducing the data to quantiles seems to improve performance, at least in the CRPS and validity cases. This could suggest that nearest neighbors algorithm can become over saturated with predictors. The case where temperature and pressure is added seems to reduce performance in both CRPS and calibration (see Tables A.1 and A.2). This might also suggest that these variables do not add significant information about the difficulty of the forecast. There are of course other ways of defining scores that might be beneficial here. Crepes for instance includes a function that normalizes scores according to the variance of the input, much like in our NECP-N method. Another way is through Mondrian conformal prediction [9], which sorts examples into categories from which predictions are drawn.
In comparison to other methods we can first say that it definitely improves predictions from the naive methods across the board, except for CRPS against the raw ensemble, a fact we will return to. Compared to the QRF we might say it performs slightly better in Calibration in Probability when looking at the χ^2 scores alone, even though the results are mixed. However, looking at the histograms it is difficult to make any clear conclusions about which is better, if we discard the reduced configuration for the QRF. One thing to note it due to the inherent randomness of the QRF, results could vary significantly between tests. The QRF also has a lot of potential for improvement, like increasing the amount of trees in the forest. The computational complexity between the methods is stark however, with the QRF taking more than 10 times longer to complete a teaching schedule in the worst case. The CPDS method does however have an advantage in this case since it relies on an underlying predictive model to produce point forecasts, which are already supplied. If we had to train an underlying model as well, the systems would be more equal in complexity and the outcome would depend heavily on the choice of this model.
Coming back to the results, the point still stands that even with an external underlying model, over which we have no control, we still see promising results. This might be connected to the fact that the underlying model is global and that we employ the CPDS method locally. One possible implication here is that CPDS might be a viable and very efficient tool to supplement forecasts from a global model, with probabilistic forecasts locally. This might require measuring stations nearby, but if we consider a wind turbine park for instance, then there probably are measurements available.

5.2 NECP(-N)

Again much like in the CPDS case, we assume that the underlying model is not trained on the data we use, an assumption we cannot guarantee. However, the validity does still seem to be decent, though the 0.9 intervals are generally

more conservative than the CPDS. Since the distributions in this case are just stacked prediction intervals, the intervals should be close to valid. Interestingly, it performs very well in calibration in probability, at least according to the χ^2 scores. A natural question to ask then is if the non-exchangeable version of the algorithm improves the prediction, i.e. is the data non-exchangeable? Or a better question is perhaps if there is a distribution drift in time, since we employ exponential decay for the weights. According to CRPS we do not gain anything from this weight scheme, since the parameter selection consistently favors $\lambda = 1$. Comparing configurations 1 and 3 in Table 4.1 we can conclude that validity and width gets slightly better, in 0.9 confidence, with $\lambda = 0.99$, while it gets slightly worse in 0.5. In calibration in probability all configurations perform well, so it is difficult to state anything about the forgetting factor here. Next question is if normalizing the conformity scores proportional to the standard deviation of the input variables improves our predictions? Again according to CRPS they give no improvement however, NECP(-N) 5 which is normalized, gives the best validity in 0.5. Looking at the validity and width of the configurations with forced $\beta > 0$ in Table A.4 the results are mixed. Part of this might be due to the forced forgetting factor below 1 though. However, in configuration 22, with much larger β, the χ^2 statistic drops significantly which could indicate that calibration in probability improves with this normalization. The drop in χ^2 is paired with an increase in CRPS though, which indicates that the CRPS metric does not improve with better calibration. This will be discussed further below. It would be a good idea to keep testing with larger β to determine if this result came by chance or if it improves calibration in probability consistently.
One certainty is that it performs significantly better than the naive methods, especially in calibration in probability and in validity (again we can note that CRPS for the raw ensemble is better). Compared to CPDS it performs similarly in case of validity, though it is a little more conservative across the board, at least in the 0.9 case. In calibration in probability it seems to outperform CPDS in general, according to the χ^2 statistic, at least the forced non-exchangeable configurations. The CPDS should in theory give better p-values here (the ones used to form the distribution) which might indicate some non-exchangeability of the data. However, since none of the configurations for either method could be discarded under the hypothesis test, we cannot say with confidence that NECP is better in this regard. Against the QRF it similarly performs better in calibration in probability looking at the χ^2 scores. The QRF in this case also seems to be less consistent in validity and width statistics. This might be due to the fact the NECP methods has an underlying predictive model to rely on. Computationally it is not as easy to compare NECP and QRF due to different parameter selection. But since the sequential selection should be more computationally heavy than the block selection we can say with confidence that, given a pre-trained underlying model, the NECP method is significantly less computationally complex than the QRF.
It is clear that the NECP(-N) method has similar potentials as CPDS. To gain a better picture of how the non-exchangeability and normalization affects performance, more research with more data should be conducted. It is possible

that with shorter forecasting lead times, temporal dependencies might increase and the non-exchangeable algorithm will exceed more in performance.

5.3 Improving the baseline

In this report we have focused on making an initial comparison between conformal based methods and the established QRF method for ensemble post-processing. Hence very little time was spent on optimizing the QRF as a baseline. For further research it would be beneficial to compare the conformal methods to an improved version. For instance, increasing the number of trees in the QRF to 400 might improve results [22]. This would increase computational complexity significantly, which is partly why we kept it low in this study. The major difference between the conformal based methods and the QRF in this case is the supplying of point forecasts for the former. Effectively the conformal methods thus creates distributions on the residuals of the predictions while the QRF is used to simply process the ensemble. This is typically how the QRF has been used and why we kept it this way in this study. The advantage the conformal methods then have, apart from computational complexity, is that they are built on top of an already existing and likely well trained model. However, that model is global and will likely not take local variations into account, which might be an advantage for the QRF in this case. It is difficult to say which is more advantageous without more testing. The QRF could be further improved by supplying more or better information as input, see [22] for examples. Naturally, this could similarly be done for the conformal methods by integrating other variables in the (non-)conformity scores. Though in that case one would have to know, or guess, how each variable affects the difficulty (or property) of a forecast unless one finds a system that does so automatically.
A further thing to note is due to the randomness of the QRF algorithm, the results might vary somewhat between instances of the algorithm, though it is likely reduced thanks to the teaching schedule. Running the algorithm several times and looking at the distribution of the results might give a better picture of the general performance. However, this is out of scope for this study.

5.4 CRPS as metric

Using CRPS as a metric for evaluation is certainly an important discussion. It seems to be a common metric within the field of weather forecasting. Comparing results in Table 4.1 it seems the CRPS favors narrow distributions even though they are lacking in validity. Though the distribution should likely be somewhat centered around the measurement to receive a good score. However, the point stands that CRPS as used in this study, might not give a proper picture of the desired performance. It would thus be beneficial to instead use the fair version of this score [11]. Given that the score might favor overly confident forecasts, this might have a detrimental effect on parameter selection in the teaching schedule.

Thus we should not trust blindly that the teaching schedule has consistently chosen the best models, at least from a validity point of view. There are of course other scores up for consideration such as the log-score or variogram score [6]. These generally perform better than CRPS in the multivariate case, but at least the log-score can definitely be useful in the univariate case as well. For further testing, it might be beneficial to employ several scoring rules.

5.5 Further research

We have already mentioned several points of possible further research such as improving the QRF, adding more variables to the (non-)conformity scores and investigating the non-exchangeability further. Looking at other metrics than the CRPS is another potential subject for further research, which has been discussed slightly already. Here the fair version of the CRPS [11], which punishes under dispersed forecasts more, could be a first option. Additionally one could introduce the log-score in evaluation, though to do that the CDFs would have to be transformed into probability density functions first. This score would be beneficial if we wanted to extend to multivariate forecasting [6], for instance predicting both speed and direction, which is yet another potential continuation of this study. Potentially, we could also look at metrics connected to real world use cases, such as the return when using the predictions on power market trading. Further, there are other topics within the field of conformal prediction which might be of interest in the application of ensemble post processing and weather forecasting in general. For instance extending to time-series forecasting one might consider the work done in [21] where the inductive conformal technique is used with recurrent neural networks. Or it might be effective to combine the techniques of conformal prediction and quantile regression like the work in [12], also for the time-series setting. Perhaps a more interesting topic for further testing is the non-exchangeable case, to understand what aspects improve predictions of this algorithm. Here it would further be interesting to look at different lead times in forecasting. Another extension for further research would be to see if it is possible to add a weighting scheme to CPDS, to handle non-exchangeability in that case.

5.6 Conclusions

We have tested two conformal based methods, conformal predictive distribution systems and non-exchangeable conformal prediction (with normalization), for post-processing ensemble forecasts to wind speed distributions and compared them to the QRF method, the raw ensemble as well as a completely naive method. We have seen that the conformal methods can create well calibrated distributions, significantly better than the raw ensemble and perhaps better than QRF, for 24-hour lead time forecasts. They do so consistently for different combinations of hyperparameters and with much less computational

complexity, given that the methods are supplied an external deterministic forecast. More testing is required, especially against better configurations of the QRF, to determine the long term usefulness of these methods within the field. Using more fair metrics for evaluation would be an essential part of continued research. However, the results show a lot of promise for using conformal based methods for probabilistic wind-speed forecasting. Especially for supplementing global deterministic models with local probabilistic predictions and doing this at low computational cost.

Bibliography

[1] David J. Aldous. "Exchangeability and related topics". In: *École d'Été de Probabilités de Saint-Flour XIII — 1983*. Ed. by P. L. Hennequin. Berlin, Heidelberg: Springer Berlin Heidelberg, 1985, pp. 1–198. ISBN: 978-3-540-39316-0.

[2] Simon Althoff et al. "Evaluation of conformal-based probabilistic forecasting methods for short-term wind speed forecasting". In: *Proceedings of the Twelfth Symposium on Conformal and Probabilistic Prediction and Applications*. Ed. by Harris Papadopoulos et al. Vol. 204. Proceedings of Machine Learning Research. [in press]. PMLR, 2023.

[3] Anastasios N. Angelopoulos and Stephen Bates. *A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification*. 2022. arXiv: 2107.07511 [cs.LG].

[4] Rina Foygel Barber et al. *Conformal prediction beyond exchangeability*. 2023. arXiv: 2202.13415 [stat.ME].

[5] Ioannis Bazionis, Panagiotis Karafotis, and Pavlos Georgilakis. "A review of short-term wind power probabilistic forecasting and a taxonomy focused on input data". In: *IET Renewable Power Generation* 16 (Jan. 2022). DOI: 10.1049/rpg2.12330.

[6] Mathias Blicher Bjerregård, Jan Kloppenborg Møller, and Henrik Madsen. "An introduction to multivariate probabilistic forecast evaluation". In: *Energy and AI* 4 (2021), p. 100058. ISSN: 2666-5468. DOI: https://doi.org/10.1016/j.egyai.2021.100058. URL: https://www.sciencedirect.com/science/article/pii/S2666546821000124.

[7] Gunnar Blom et al. *Sannolikhetsteori och statistikteori med tillämpningar*. 7th ed. Lund: Studentlitteratur, 2017. ISBN: 9789144123561.

[8] Henrik Boström. "crepes: a Python Package for Generating Conformal Regressors and Predictive Systems". In: *Proceedings of the Eleventh Symposium on Conformal and Probabilistic Prediction and Applications*. Ed. by Ulf Johansson et al. Vol. 179. Proceedings of Machine Learning Research. PMLR, 2022.

[9] Henrik Boström, Ulf Johansson, and Tuwe Löfström. "Mondrian conformal predictive distributions". In: *Proceedings of the Tenth Symposium on Conformal and Probabilistic Prediction and Applications*. Ed. by Lars Carlsson et al. Vol. 152. Proceedings of Machine Learning Research. PMLR, Aug. 2021, pp. 24–38. URL: https://proceedings.mlr.press/v152/bostrom21a.html.

[10] Leo Breiman. "Random Forests". In: *Machine Learning* 45 (Oct. 2001), pp. 5–32. DOI: 10.1023/A:1010933404324.

[11] C. A. T. Ferro. "Fair scores for ensemble forecasts". In: *Quarterly Journal of the Royal Meteorological Society* 140.683 (2014), pp. 1917–1923. DOI: https://doi.org/10.1002/qj.2270. eprint: https://rmets.onlinelibrary.wiley.com/doi/pdf/10.1002/qj.2270. URL: https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.2270.

[12] Vilde Jensen, Filippo Maria Bianchi, and Stian Anfinsen. "Ensemble Conformalized Quantile Regression for Probabilistic Time Series Forecasting". In: *IEEE Transactions on Neural Networks and Learning Systems* PP (Nov. 2022), pp. 1–12. DOI: 10.1109/TNNLS.2022.3217694.

[13] Christopher Kath and Florian Ziel. "Conformal prediction interval estimation and applications to day-ahead and intraday power markets". In: *International Journal of Forecasting* 37.2 (Apr. 2021), pp. 777–799. DOI: 10.1016/j.ijforecast.2020.09.006.

[14] Jing Lei et al. *Distribution-Free Predictive Inference For Regression*. 2017. arXiv: 1604.04173 [stat.ME].

[15] James E. Matheson and Robert L. Winkler. "Scoring Rules for Continuous Probability Distributions". In: *Management Science* 22.10 (1976), pp. 1087–1096. ISSN: 00251909, 15265501. URL: http://www.jstor.org/stable/2629907 (visited on 03/24/2023).

[16] Nicolai Meinshausen. "Quantile Regression Forests". In: *Journal of Machine Learning Research* 7.35 (2006), pp. 983–999. URL: http://jmlr.org/papers/v7/meinshausen06a.html.

[17] Malte Müller et al. "AROME - MetCoOp : A Nordic convective scale operational weather prediction model". In: *Weather and Forecasting* 32 (Jan. 2017). DOI: 10.1175/WAF-D-16-0099.1.

[18] Harris Papadopoulos, Vladimir Vovk, and Alex J. Gammerman. "Regression Conformal Prediction with Nearest Neighbours". In: *CoRR* abs/1401.3880 (2014). arXiv: 1401.3880. URL: http://arxiv.org/abs/1401.3880.

[19] Harris Papadopoulos et al. "Inductive Confidence Machines for Regression". In: *Machine Learning: ECML 2002*. Ed. by Tapio Elomaa, Heikki Mannila, and Hannu Toivonen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 345–356. ISBN: 978-3-540-36755-0.

[20] Harris Papadopoulus, Alex Gammerman, and Volodya Vovk. "Normalized nonconformity measures for regression conformal prediction". In: ed. by Alex Gammerman. Vol. 152. Proceedings of AIA 2008. ACTA Press, Feb. 2008, pp. 64–69.

[21] Kamile Stankeviciute, Ahmed M. Alaa, and Mihaela van der Schaar. "Conformal Time-series Forecasting". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 6216–6228. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/312f1ba2a72318edaaa995a67835fad5-Paper.pdf.

[22] Maxime Taillardat et al. "Calibrated Ensemble Forecasts Using Quantile Regression Forests and Ensemble Model Output Statistics". In: *Monthly Weather Review* 144 (6 2016), pp. 2375–2393. DOI: 10.1175/MWR-D-15-0260.1.

[23] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*. New York: Springer, 2022, pp. 22, 29–38, 181–187, 210–216. ISBN: 0-387-00152-2.

[24] Erotokritos Xydas et al. "Probabilistic wind power forecasting and its application in the scheduling of gas-fired generators". In: *Applied Energy* 192 (2017), pp. 382–394. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2016.10.019. URL: https://www.sciencedirect.com/science/article/pii/S030626191631443X.

Appendix A

Method configurations and results

Below follows the all testing setups and corresponding results. They are sorted under corresponding method sections.

A.1 CPDS

Table A.1: Input and hyperparameter configurations of the CPDS method and execution time of each teaching schedule. The ratio of each parameter combination used for prediction in the teaching schedule are presented in the Ratio column.

Number	Input	Set of Hyperparameters	Ratio	Time
1	`x_wind` `y_wind`	$\{\text{WL} = \text{all}, k = 5\}$	0.468	02:55
		$\{\text{WL} = 200, k = 5\}$	0.135	
		$\{\text{WL} = 100, k = 5\}$	0.282	
		$\{\text{WL} = 50, k = 5\}$	0.115	
2	`x_wind` `y_wind` `pressure`	$\{\text{WL} = \text{all}, k = 5\}$	0.510	03:00
		$\{\text{WL} = 200, k = 5\}$	0.224	
		$\{\text{WL} = 100, k = 5\}$	0.196	
		$\{\text{WL} = 50, k = 5\}$	0.071	
3	`x_wind` `y_wind` `pressure` `temperature`	$\{\text{WL} = \text{all}, k = 5\}$	0.465	02:56
		$\{\text{WL} = 200, k = 5\}$	0.103	
		$\{\text{WL} = 100, k = 5\}$	0.330	
		$\{\text{WL} = 50, k = 5\}$	0.103	

4	x_wind y_wind pressure temperature gust	$\{\text{WL} = \text{all}, k = 5\}$ $\{\text{WL} = 200, k = 5\}$ $\{\text{WL} = 100, k = 5\}$ $\{\text{WL} = 50, k = 5\}$	0.401 0.247 0.263 0.090	03:05
5	x_wind y_wind gust	$\{\text{WL} = \text{all}, k = 5\}$ $\{\text{WL} = 200, k = 5\}$ $\{\text{WL} = 100, k = 5\}$ $\{\text{WL} = 50, k = 5\}$	0.628 0.112 0.202 0.058	02:56
6	red(x_wind) red(y_wind) red(gust)	$\{\text{WL} = \text{all}, k = 5\}$ $\{\text{WL} = 200, k = 5\}$ $\{\text{WL} = 100, k = 5\}$ $\{\text{WL} = 50, k = 5\}$	0.647 0.096 0.240 0.016	05:20
7	red(x_wind) red(y_wind) red(gust)	$\{\text{WL} = \text{all}, k = 10\}$ $\{\text{WL} = 200, k = 10\}$ $\{\text{WL} = 100, k = 10\}$ $\{\text{WL} = 50, k = 10\}$	0.497 0.199 0.247 0.058	05:21
8	red(x_wind) red(y_wind) red(gust)	$\{\text{WL} = \text{all}, k = 15\}$ $\{\text{WL} = 200, k = 15\}$ $\{\text{WL} = 100, k = 15\}$ $\{\text{WL} = 50, k = 15\}$	0.330 0.179 0.385 0.106	05:26
9	red(x_wind) red(y_wind) red(gust)	$\{\text{WL} = \text{all}, k = 20\}$ $\{\text{WL} = 200, k = 20\}$ $\{\text{WL} = 100, k = 20\}$ $\{\text{WL} = 50, k = 20\}$	0.471 0.167 0.327 0.035	05:27
10	red(x_wind) red(y_wind) red(gust)	$\{\text{WL} = \text{all}, k = 30\}$ $\{\text{WL} = 200, k = 30\}$ $\{\text{WL} = 100, k = 30\}$ $\{\text{WL} = 50, k = 30\}$	0.436 0.192 0.346 0.026	04:53

Table A.2: Statistics with 95 % confidence intervals of the CPDS model configurations presented in Table A.1.

Number	Val 0.9	Width 0.9	Val 0.5	Width 0.5	CRPS
1	0.933 ± 0.028	7.528 ± 0.329	0.532 ± 0.055	2.102 ± 0.097	0.887 ± 0.069
2	0.936 ± 0.027	7.457 ± 0.336	0.484 ± 0.055	2.046 ± 0.091	0.879 ± 0.070
3	0.933 ± 0.028	7.572 ± 0.328	0.494 ± 0.055	2.111 ± 0.096	0.906 ± 0.071
4	0.936 ± 0.027	7.254 ± 0.296	0.487 ± 0.055	2.065 ± 0.093	0.890 ± 0.069
5	0.920 ± 0.030	7.182 ± 0.312	0.510 ± 0.055	2.087 ± 0.100	0.884 ± 0.071
6	0.910 ± 0.032	6.541 ± 0.291	0.500 ± 0.055	2.060 ± 0.096	0.881 ± 0.071
7	0.917 ± 0.031	6.164 ± 0.202	0.480 ± 0.055	1.981 ± 0.066	0.871 ± 0.071
8	0.926 ± 0.029	6.198 ± 0.148	0.513 ± 0.055	2.023 ± 0.052	0.863 ± 0.070
9	0.920 ± 0.030	6.059 ± 0.132	0.515 ± 0.055	2.003 ± 0.044	0.867 ± 0.072
10	0.917 ± 0.031	6.057 ± 0.108	0.484 ± 0.055	2.012 ± 0.035	0.865 ± 0.072

A.2 NECP(-N)

Table A.3: Input and hyperparameter configurations of the NECP and NECP-N methods and execution time of each teaching schedule. The ratio of each parameter combination used for prediction in the teaching schedule are presented in the Ratio column.

Number	Input	Set of Hyperparameters	Ratio	Time
1	x_wind y_wind	$\{\lambda = 1, \beta = 0\}$	1	07:43
		$\{\lambda = 0.995, \beta = 0\}$	0	
		$\{\lambda = 0.99, \beta = 0\}$	0	
		$\{\lambda = 0.98, \beta = 0\}$	0	
		$\{\lambda = 0.97, \beta = 0\}$	0	
2	x_wind y_wind	$\{\lambda = 1, \beta = [0, 0]\}$	1	09:32
		$\{\lambda = 1, \beta = [0.05, 0.05]\}$	0	
		$\{\lambda = 1, \beta = [0.1, 0.1]\}$	0	
		$\{\lambda = 0.99, \beta = [0, 0]\}$	0	
		$\{\lambda = 0.99, \beta = [0.05, 0.05]\}$	0	
		$\{\lambda = 0.99, \beta = [0.1, 0.1]\}$	0	
3	x_wind y_wind	$\{\lambda = 0.99, \beta = [0, 0]\}$	1	05:01
		$\{\lambda = 0.99, \beta = [0.05, 0.05]\}$	0	
		$\{\lambda = 0.99, \beta = [0.1, 0.1]\}$	0	
4	x_wind y_wind	$\{\lambda = 0.999, \beta = [0, 0]\}$	1	05:06
		$\{\lambda = 0.999, \beta = [0.05, 0.05]\}$	0	
		$\{\lambda = 0.999, \beta = [0.1, 0.1]\}$	0	
5	x_wind y_wind	$\{\lambda = 0.99, \beta = [0.05, 0.05]\}$	0.115	06:36
		$\{\lambda = 0.99, \beta = [0.1, 0.1]\}$	0	
		$\{\lambda = 0.999, \beta = [0.05, 0.05]\}$	0.885	
		$\{\lambda = 0.999, \beta = [0.1, 0.1]\}$	0	
6	x_wind y_wind pressure	$\{\lambda = 1, \beta = [0, 0, 0]\}$	1	09:59
		$\{\lambda = 1, \beta = [0.05, 0.05, 0.05]\}$	0	
		$\{\lambda = 1, \beta = [0.1, 0.1, 0.1]\}$	0	
		$\{\lambda = 0.99, \beta = [0, 0, 0]\}$	0	
		$\{\lambda = 0.99, \beta = [0.05, 0.05, 0.05]\}$	0	
		$\{\lambda = 0.99, \beta = [0.1, 0.1, 0.1]\}$	0	
7	x_wind y_wind pressure	$\{\lambda = 0.99, \beta = [0, 0, 0]\}$	1	05:02
		$\{\lambda = 0.99, \beta = [0.05, 0.05, 0.05]\}$	0	
		$\{\lambda = 0.99, \beta = [0.1, 0.1, 0.1]\}$	0	
8	x_wind y_wind pressure	$\{\lambda = 0.999, \beta = [0, 0, 0]\}$	1	05:06
		$\{\lambda = 0.999, \beta = [0.05, 0.05, 0.05]\}$	0	
		$\{\lambda = 0.999, \beta = [0.1, 0.1, 0.1]\}$	0	
9	x_wind y_wind pressure	$\{\lambda = 0.99, \beta = [0.05, 0.05, 0.05]\}$	0.734	06:47
		$\{\lambda = 0.99, \beta = [0.1, 0.1, 0.1]\}$	0	
		$\{\lambda = 0.999, \beta = [0.05, 0.05, 0.05]\}$	0.266	
		$\{\lambda = 0.999, \beta = [0.1, 0.1, 0.1]\}$	0	
10	x_wind y_wind pressure temperature	$\{\lambda = 1, \beta = [0, 0, 0, 0]\}$	1	10:09
		$\{\lambda = 1, \beta = [0.05, 0.05, 0.05, 0.05]\}$	0	
		$\{\lambda = 1, \beta = [0.1, 0.1, 0.1, 0.1]\}$	0	
		$\{\lambda = 0.99, \beta = [0, 0, 0, 0]\}$	0	
		$\{\lambda = 0.99, \beta = [0.05, 0.05, 0.05, 0.05]\}$	0	
		$\{\lambda = 0.99, \beta = [0.1, 0.1, 0.1, 0.1]\}$	0	

11	x_wind y_wind temperature pressure	$\{\lambda = 0.99, \beta = [0, 0, 0, 0]\}$ $\{\lambda = 0.99, \beta = [0.05, 0.05, 0.05, 0.05]\}$ $\{\lambda = 0.99, \beta = [0.1, 0.1, 0.1, 0.1]\}$	1 0 0	05:04
12	x_wind y_wind temperature pressure	$\{\lambda = 0.999, \beta = [0, 0, 0, 0]\}$ $\{\lambda = 0.999, \beta = [0.05, 0.05, 0.05, 0.05]\}$ $\{\lambda = 0.999, \beta = [0.1, 0.1, 0.1, 0.1]\}$	1 0 0	05:05
13	x_wind y_wind temperature pressure	$\{\lambda = 0.99, \beta = [0.05, 0.05, 0.05, 0.05]\}$ $\{\lambda = 0.99, \beta = [0.1, 0.1, 0.1, 0.1]\}$ $\{\lambda = 0.999, \beta = [0.05, 0.05, 0.05, 0.05]\}$ $\{\lambda = 0.999, \beta = [0.1, 0.1, 0.1, 0.1]\}$	0.737 0 0.263 0	06:42
14	x_wind y_wind pressure temperature gust	$\{\lambda = 1, \beta = [0, 0, 0, 0, 0]\}$ $\{\lambda = 1, \beta = [0.05, 0.05, 0.05, 0.05, 0.01]\}$ $\{\lambda = 1, \beta = [0.1, 0.1, 0.1, 0.1, 0.1]\}$ $\{\lambda = 0.99, \beta = [0, 0, 0, 0, 0]\}$ $\{\lambda = 0.99, \beta = [0.05, 0.05, 0.05, 0.05, 0.05]\}$ $\{\lambda = 0.99, \beta = [0.1, 0.1, 0.1, 0.1, 0.1]\}$	1 0 0 0 0 0	10:23
15	x_wind y_wind temperature pressure gust	$\{\lambda = 0.99, \beta = [0, 0, 0, 0, 0]\}$ $\{\lambda = 0.99, \beta = [0.05, 0.05, 0.05, 0.05, 0.05]\}$ $\{\lambda = 0.99, \beta = [0.1, 0.1, 0.1, 0.1, 0.1]\}$	1 0 0	05:08
16	x_wind y_wind temperature pressure gust	$\{\lambda = 0.999, \beta = [0, 0, 0, 0, 0]\}$ $\{\lambda = 0.999, \beta = [0.05, 0.05, 0.05, 0.05, 0.05]\}$ $\{\lambda = 0.999, \beta = [0.1, 0.1, 0.1, 0.1, 0.1]\}$	1 0 0	05:10
17	x_wind y_wind temperature pressure gust	$\{\lambda = 0.99, \beta = [0.05, 0.05, 0.05, 0.05, 0.05]\}$ $\{\lambda = 0.99, \beta = [0.1, 0.1, 0.1, 0.1, 0.1]\}$ $\{\lambda = 0.999, \beta = [0.05, 0.05, 0.05, 0.05, 0.05]\}$ $\{\lambda = 0.999, \beta = [0.1, 0.1, 0.1, 0.1, 0.1]\}$	0.247 0 0.753 0	06:45
18	x_wind y_wind gust	$\{\lambda = 1, \beta = [0, 0, 0]\}$ $\{\lambda = 1, \beta = [0.05, 0.05, 0.05]\}$ $\{\lambda = 1, \beta = [0.1, 0.1, 0.1]\}$ $\{\lambda = 0.99, \beta = [0, 0, 0]\}$ $\{\lambda = 0.99, \beta = [0.05, 0.05, 0.05]\}$ $\{\lambda = 0.99, \beta = [0.1, 0.1, 0.1]\}$	1 0 0 0 0 0	10:08
19	x_wind y_wind gust	$\{\lambda = 0.99, \beta = [0, 0, 0]\}$ $\{\lambda = 0.99, \beta = [0.05, 0.05, 0.05]\}$ $\{\lambda = 0.99, \beta = [0.1, 0.1, 0.1]\}$	1 0 0	05:05
20	x_wind y_wind gust	$\{\lambda = 0.999, \beta = [0, 0, 0]\}$ $\{\lambda = 0.999, \beta = [0.05, 0.05, 0.05]\}$ $\{\lambda = 0.999, \beta = [0.1, 0.1, 0.1]\}$	1 0 0	05:04
21	x_wind y_wind gust	$\{\lambda = 0.99, \beta = [0.05, 0.05, 0.05]\}$ $\{\lambda = 0.99, \beta = [0.1, 0.1, 0.1]\}$ $\{\lambda = 0.999, \beta = [0.05, 0.05, 0.05]\}$ $\{\lambda = 0.999, \beta = [0.1, 0.1, 0.1]\}$	0.045 0 0.955 0	06:38
22	x_wind y_wind gust	$\{\lambda = 0.99, \beta = [0.5, 0.5, 0.5]\}$ $\{\lambda = 0.99, \beta = [1, 1, 1]\}$ $\{\lambda = 0.999, \beta = [0.5, 0.5, 0.5]\}$ $\{\lambda = 0.999, \beta = [1, 1, 1]\}$	0.308 0 0.692 0	06:46

Table A.4: Statistics with 95 % confidence of the NECP(-N) model configurations presented in Table A.3.

Number	Val 0.9	Width 0.9	Val 0.5	Width 0.5	CRPS
1	0.929 ± 0.028	6.094 ± 0.038	0.503 ± 0.055	2.042 ± 0.010	0.864 ± 0.070
2	0.929 ± 0.028	6.094 ± 0.038	0.503 ± 0.055	2.042 ± 0.010	0.864 ± 0.070
3	0.923 ± 0.030	6.080 ± 0.039	0.516 ± 0.055	2.053 ± 0.015	0.873 ± 0.072
4	0.929 ± 0.028	6.084 ± 0.039	0.510 ± 0.055	2.045 ± 0.010	0.864 ± 0.070
5	0.936 ± 0.027	6.255 ± 0.057	0.500 ± 0.055	2.033 ± 0.016	0.870 ± 0.070
6	0.929 ± 0.028	6.094 ± 0.038	0.503 ± 0.055	2.042 ± 0.010	0.864 ± 0.070
7	0.923 ± 0.030	6.080 ± 0.039	0.516 ± 0.055	2.053 ± 0.015	0.873 ± 0.072
8	0.929 ± 0.028	6.084 ± 0.039	0.510 ± 0.055	2.045 ± 0.010	0.864 ± 0.070
9	0.929 ± 0.028	7.483 ± 0.287	0.513 ± 0.055	2.108 ± 0.080	0.906 ± 0.069
10	0.929 ± 0.028	6.094 ± 0.038	0.503 ± 0.055	2.042 ± 0.010	0.864 ± 0.070
11	0.923 ± 0.030	6.080 ± 0.039	0.516 ± 0.055	2.053 ± 0.015	0.873 ± 0.072
12	0.929 ± 0.028	6.084 ± 0.039	0.510 ± 0.055	2.045 ± 0.010	0.864 ± 0.070
13	0.929 ± 0.028	7.419 ± 0.285	0.513 ± 0.055	2.104 ± 0.080	0.905 ± 0.069
14	0.929 ± 0.028	6.094 ± 0.038	0.503 ± 0.055	2.042 ± 0.010	0.864 ± 0.070
15	0.923 ± 0.030	6.080 ± 0.039	0.516 ± 0.055	2.053 ± 0.015	0.873 ± 0.072
16	0.929 ± 0.028	6.084 ± 0.039	0.510 ± 0.055	2.045 ± 0.010	0.864 ± 0.070
17	0.936 ± 0.027	7.738 ± 0.296	0.526 ± 0.055	2.139 ± 0.080	0.906 ± 0.068
18	0.929 ± 0.028	6.094 ± 0.038	0.503 ± 0.055	2.042 ± 0.010	0.864 ± 0.070
19	0.923 ± 0.030	6.080 ± 0.039	0.516 ± 0.055	2.053 ± 0.015	0.873 ± 0.072
20	0.929 ± 0.028	6.084 ± 0.039	0.510 ± 0.055	2.045 ± 0.010	0.864 ± 0.070
21	0.936 ± 0.027	6.353 ± 0.066	0.503 ± 0.055	2.026 ± 0.019	0.870 ± 0.070
22	0.933 ± 0.028	7.427 ± 0.224	0.503 ± 0.055	1.976 ± 0.059	0.900 ± 0.072

A.3 QRF

Table A.5: Input and hyperparameter configurations of the QRF method and execution time of each teaching schedule. The ratio of each parameter combination used for prediction in the teaching schedule are presented in the Ratio column.

Number	Input	Set of Hyperparameters	Ratio	Time
1	`x_wind` `y_wind`	$\{\text{WL} = 100, T = 200\}$ $\{\text{WL} = \text{all}, T = 100\}$ $\{\text{WL} = \text{all}, T = 200\}$	0.237 0.006 0.756	43:39

2	`x_wind`
	`y_wind`
	`pressure`
		$\{\text{WL} = 100, T = 200\}$
		$\{\text{WL} = \text{all}, T = 100\}$
		$\{\text{WL} = \text{all}, T = 200\}$
			0.167
			0.029
			0.804
				52:38

3	`x_wind`
	`y_wind`
	`temperature`
	`pressure`
		$\{\text{WL} = 100, T = 200\}$
		$\{\text{WL} = \text{all}, T = 100\}$
		$\{\text{WL} = \text{all}, T = 200\}$
			0.141
			0.010
			0.849
				1:03:39

4	`x_wind`
	`y_wind`
	`temperature`
	`pressure`
	`gust`
		$\{\text{WL} = 100, T = 200\}$
		$\{\text{WL} = \text{all}, T = 100\}$
		$\{\text{WL} = \text{all}, T = 200\}$
			0.391
			0.038
			0.571
				1:07:39

5	`x_wind`
	`y_wind`
	`gust`
		$\{\text{WL} = 100, T = 200\}$
		$\{\text{WL} = \text{all}, T = 100\}$
		$\{\text{WL} = \text{all}, T = 200\}$
			0.417
			0.048
			0.535
				49:01

6	red(`x_wind`) red(`y_wind`) red(`gust`)	$\{\text{WL} = 100, T = 200\}$ $\{\text{WL} = \text{all}, T = 100\}$ $\{\text{WL} = \text{all}, T = 200\}$	0.663 0.026 0.311	22:57

Table A.6: Statistics with 95 % confidence of the QRF model configurations presented in Table A.5.

Number	Val 0.9	Width 0.9	Val 0.5	Width 0.5	CRPS
1	0.891 ± 0.034	5.924 ± 0.221	0.526 ± 0.055	2.467 ± 0.135	0.906 ± 0.075
2	0.926 ± 0.029	6.142 ± 0.221	0.532 ± 0.055	2.481 ± 0.144	0.909 ± 0.080
3	0.889 ± 0.035	6.295 ± 0.219	0.526 ± 0.055	2.566 ± 0.132	0.962 ± 0.089
4	0.869 ± 0.037	4.814 ± 0.141	0.506 ± 0.055	1.943 ± 0.083	0.785 ± 0.062
5	0.869 ± 0.037	4.622 ± 0.141	0.510 ± 0.055	1.902 ± 0.082	0.776 ± 0.061
6	0.785 ± 0.046	3.863 ± 0.143	0.397 ± 0.054	1.588 ± 0.101	0.904 ± 0.132

Appendix B

PIT histograms

Here follows the PIT histogram plots for all the tested configurations in Appendix A. The histograms for each method are sorted under corresponding sections below.

B.1 CPDS

(a) CPDS 1

(b) CPDS 2

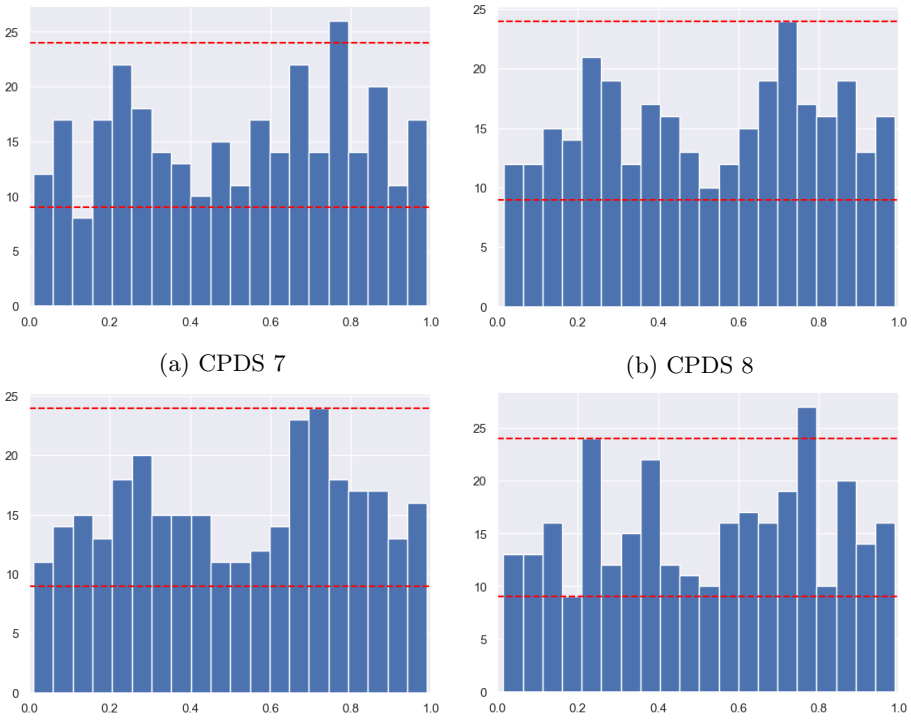
(c) CPDS 3

(d) CPDS 4

(e) CPDS 5

(f) CPDS 6

Figure B.1: PIT histograms from the CPDS configurations 1 through 6 in Table A.1.



(a) CPDS 7

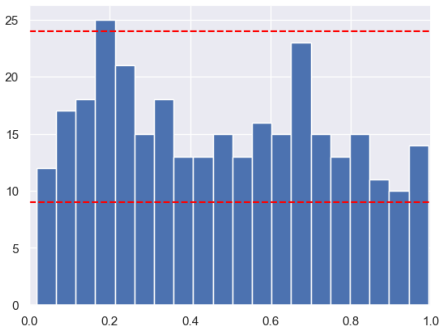
(b) CPDS 8

(c) CPDS 9

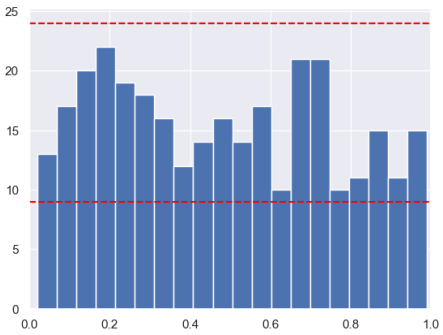
(d) CPDS 10

Figure B.2: PIT histograms from the CPDS configurations 7 through 10 in Table A.1.

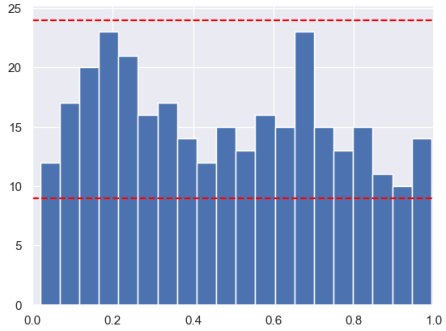
B.2 NECP(-N)



(a) NECP(-N) 1, 2, 6, 10, 14, 18

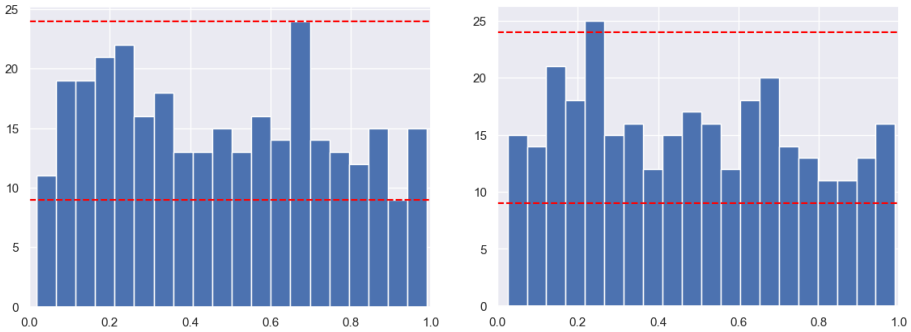


(b) NECP(-N) 3, 7, 11, 15, 19



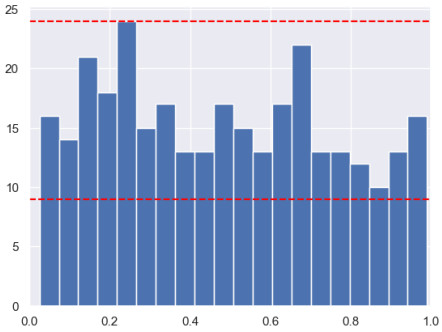
(c) NECP(-N) 4, 8, 12, 16, 20

Figure B.3: PIT histograms from the NECP(-N) configurations with duplicate results in Table A.3.

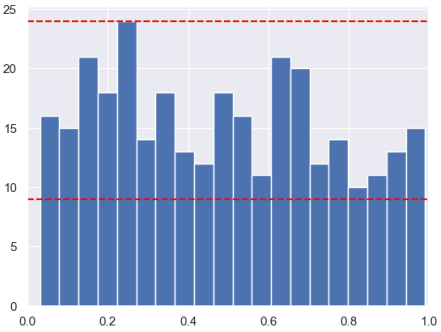


(a) NECP(-N) 5

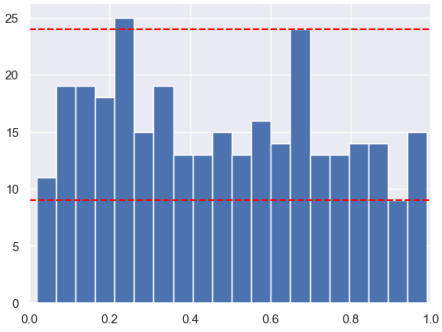
(b) NECP(-N) 9



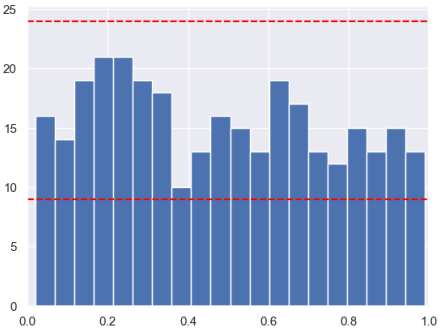
(c) NECP(-N) 13



(d) NECP(-N) 17



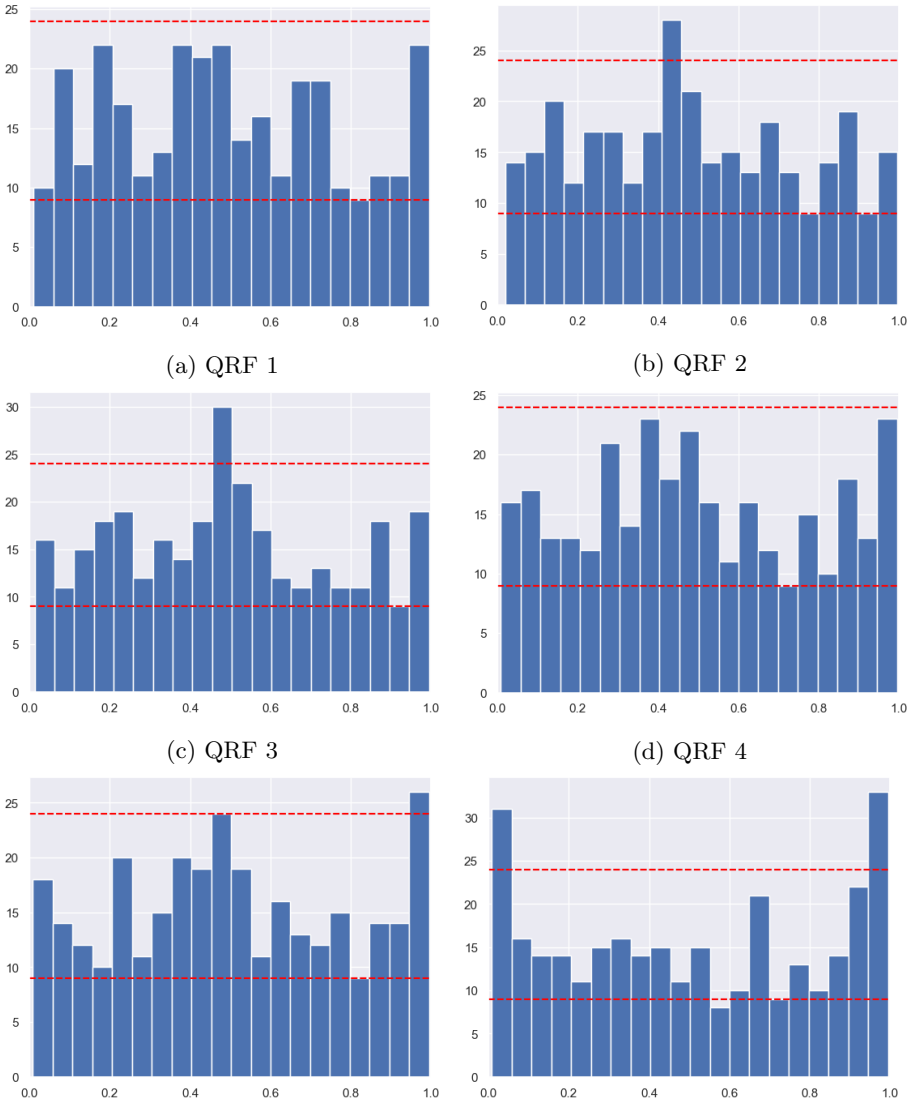
(e) NECP(-N) 21



(f) NECP(-N) 22

Figure B.4: PIT histograms from the NECP(-N) configurations 5, 9, 13, 17, 21 and 22 in Table A.3.

B.3 QRF



(a) QRF 1

(b) QRF 2

(c) QRF 3

(d) QRF 4

(e) QRF 5

(f) QRF 6

Figure B.5: PIT histograms from the QRF configurations A.5.

Master's Theses in Mathematical Sciences 2023:E61
ISSN 1404-6342

LUTFMS-3488-2022

Mathematical Statistics
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden

http://www.maths.lu.se/