SafeWind

Collaborative project funded by the European Commission
under the 7th Framework Program, Theme 2007-2.3.2: Energy

"Multi-scale data assimilation, advanced wind modelling & forecasting with emphasis to extreme weather situations for a safe large-scale wind power integration"

Grant Agreement N°: 213740

DELIVERABLE Dp-5.1

"Documentation of evaluation suite and probabilistic skill scores"

DOCUMENT TYPE	Deliverable
DOCUMENT NAME:	swind.deliverable_Dc-5.1_EvaluationSuite_v2.1.pdf
VERSION:	V2.1(*)
DATE:	2009.10.22
CLASSIFICATION:	R0: General public
STATUS:	Released

Abstract: This Deliverable of the SafeWind project presents the evaluation suite used at ECMWF. This suite is dedicated to the evaluation of ensemble forecasts with a limited number of single deterministic predictions (ensemble members). It further describes the deterministic and probabilistic scores that are calculated and displayed by the suit. This report is closely related to Deliverable Dp-6.2 "Methodology for the evaluation of probabilistic forecasts" (McSharry et al., 2009) and aims to complement the descriptions of forecast evaluation given by the wind power forecasting community from the meteorological point of view.

AUTHORS[1], REVIEWERS				
MAIN AUTHOR/EDITOR:	M. Denhard			
AFFILIATION:	ECMWF			
ADDRESS:	Shinfield Park, Reading, RG2 9AX, United Kingdom.			
TEL.:	+44-118.949.9664			
EMAIL:	michael.denhard@ecmwf.int			
FURTHER AUTHORS:	R. Hagedorn, T. Petroliagis			
PEER REVIEWERS:	P. Pinson			
REVIEW APPROVAL:	Approved :	**X**	Rejected (improve as indicated below) :	
SUGGESTED IMPROVEMENTS:				
APPROVER:	G. Kariniotakis			

VERSION HISTORY			
VERSION[2]:	DATE:	COMMENTS, CHANGES, STATUS:	PERSON(S):
0.1	09/10/2009	Draft for comments	M. Denhard, T. Petroliagis
0.2	13/10/2009	Draft for comments	M.Denhard, R.Hagedorn
0.3	16/10/2009	Draft for comments	M.Denhard, R. Hagedorn
1.0	20/10/2009	Pending for review	M.Denhard, R.Hagedorn
1.1	22/10/2009	Modifications according to review of P. Pinson, Pending for review	M.Denhard
1.2	23/10/2009	Approved by the reviewer (P.Pinson)	P. Pinson

STATUS, CONFIDENTIALITY, ACCESSIBILITY									
STATUS:				CONFIDENTIALITY:				ACCESSIBILITY:	
S0	Approved/Released	X		**R0**	General public	**X**		Private web site	
S1	Reviewed			**R1**	Restricted to project members			Public web site	**X**
S2	Pending for review			**R2**	Restricted to European Commission			Paper copy	
S3	Draft for comments			**R3**	Restricted to WP members + PL				
S4	Under preparation			**R4**	Restricted to Task members +WPL+PL				

PL: *Project leader* **WPL:** *Work package leader* **TL:** *Task leader*

[1] **The authors of this document are solely responsible for its content, which does not represent the opinion of the European Community and the European Community is not responsible for any use that might be made of data appearing therein.**

[2] **VERSION NAMING : V0.x** draft before peer-review approval, **V1.0** at the approval, **V1.x** minor revisions, **V2.0** major revision

Contents

1. Introduction

The general aim of WP 5 in the SafeWind project is the development of an optimized Ensemble Prediction System (EPS) that provides sharp and well calibrated probabilistic forecasts from numerical weather prediction models. For an overview of the numerical forecast products from the European Centre for Medium-Range Weather Forecasts (ECMWF) and other global modelling centres see http://tigge.ecmwf.int/models.html. Giebel et al. 2009 and the references therein also provide information about the high and very high resolution Limited Area Models (LAM) running operationally at the weather services in Europe.

The evaluation of results from different models and configurations of EPS constitutes a major part of the development of improved forecasts. This is mainly due to the complexity of verifying probabilistic forecasts and the variety of atmospheric parameters, regions and forecast ranges that need to be considered. Comprehensive overviews of verification methods in meteorology can be found e.g. in Bougeault (2003), Casati et al. (2008) and Atger et al. (2009).

In order to facilitate the evaluation of ensemble forecasts a diagnostic software package, also called evaluation suite EnsVeriPy, has been developed at ECMWF. This evaluation suite enables an efficient production of a comprehensive range of scores to diagnose the performance of different models and forecast system configurations. Chapter 2 describes the evaluation suite and gives technical details. It further describes some deterministic scores calculated by EnsVeriPy. The probabilistic evaluation capacities of EnsVeriPy can be found in Chapter 3.

This report is closely related to Deliverable Dp-6.2 “Methodology for the evaluation of probabilistic forecasts” (McSharry et al., 2009) and aims to complement the descriptions of forecast evaluation given by the wind power forecasting community from the meteorological point of view.

2. ECMWF evaluation suite

2.1 Design and Structure

The evaluation suite has been designed in a modular structure in order to enable continuous update of the software components and easy implementation of additional features. The core software components are written in the script language Meteorological Python (MetPy), which is a library of tools for meteorological data acquisition, decoding and processing in the Python language. Python is a programming language available from the open source community (www.python.org). MetPy extends Python, enabling it to decode file formats used in Numerical Weather Prediction (NWP) such as GRIB, BUFR, ODB and netCDF, retrieve and manipulate data, compute statistics and perform other data handling tasks necessary for verification.

EnsVeriPy runs under the Supervisor Monitor Scheduler (SMS), a software tool written in the Meteorological Applications Section at ECMWF. SMS is an application that enables users to run a large number of programs which may have dependencies on one another, and in time, in a controlled environment with reasonable tolerance of both hardware and software failures, combined with good restart capabilities. SMS submits tasks (jobs) and receives acknowledgments from the tasks when they change status and when they send events. SMS knows the relationships between tasks, and is able to submit dependent tasks when a given task changes its status, for example when it finishes. Users talk to SMS using either the command and display program (CDP), or its X-windows equivalent XCDP. A screenshot of the evaluation suite running under SMS / XCDP can be found in Appendix A.

The evaluation suite is easily set up for every new user of ECMWF’s computing environment. By adapting the generic launch script ‘SAFEWIND_verify.ksh’ (see Appendix B) to the specific needs of a user and submitting this script under UNIX, a new and personal evaluation suite can be launched. The main parameters and switches a user might want to modify are:

- Models to be scored
- Verification data to be used
- Dates to be scored
- Variables to be scored

As detailed in the next sections, the main tasks of the evaluation suite are to retrieve the forecast data, calculate scores and archive the results.

2.2 Data input

Meteorological forecasts can be verified either against global analyses, available at every gridpoint of the model grid, or against observations, available only at single observation stations. The evaluation suite described here is designed to facilitate the verification against analysis, whereas the verification against the observational data base is currently realised in an external module linked to a software package on the Combined Prediction System (CPS) developed in Task 5.5 of the SafeWind project. A future extension of EnsVeriPy to incorporate verification against observations directly into the evaluation suite is underway; however, it will not change the main structure and design of the suite. Therefore, this extension will not be part of this documentation which describes the main design and scores incorporated up to now.

The choice which verifying analysis is used in the evaluation suite is determined by setting the parameter ‘classANA’ in the launch script (see Appendix A). Currently implemented choices are:

- ECMWF’s operational analysis: classANA=od
- ERA-40 re-analysis: classANA=e4
- ERA-interim analysis: classANA=ei
- TIGGE mullti-model analysis: classANA=ti

The parameters ‘originANA’, ‘expverANA’, and ‘streamANA’ are further sub-identifiers to be set according to the main choice of ‘classANA’.

The calculation of probabilistic scores requires the definition of event thresholds. The evaluation suite enables to define such thresholds with respect to the climatological distribution of the variable under investigation rather than using absolute threshold values. The evaluation suite uses pre-calculated daily climatologies for each variable, consisting of daily fields of the mean, standard deviation, and quantiles of the distribution, derived from the ERA-40 reanalysis dataset. The climatologies are based on the years 1979–2001, and the statistics for each day are based on a 61-day window centred on the day of interest.

The choice which forecast data to score in the evaluation suite is determined by the parameters ‘originEPS’, ‘expverEPS’, ‘streamEPS’, and ‘classEPS’, which are parameters to identify the dataset to be scored in ECMWF’s data archive. In addition to setting the main choice of forecast dataset, users have to choose which variables on which levels to be scored by setting the parameters ‘PARAMS’, ‘LTYPES’, and ‘LEVELS’. The dates to be scored are chosen by setting the parameters ‘D1’ and ‘D2’ (first date and last late to be scored) or by the parameter ‘DLIST’ (list of dates to be scored).

2.3 Calculation of scores

In the evaluation suite both deterministic and probabilistic scores are calculated. The full set of diagnostics consists of calculating the:

Deterministic:

- mean absolute error (MAE)
- root mean squared error (RMSE)
- anomaly correlation coefficient (ACC)

Probabilistic:

- Spread of ensemble forecasts (SDEV)
- Brier Score (BS)
- Ranked Probability Score (RPS)
- Continous Ranked Probability Score (CRPS)
- Ignorance Score (IGN)
- Rank Histogram (RH)
- Reliability Diagram (RD)
- Relative Operating Characteristics (ROC)

In most cases the scores are calculated for a sequence of forecasts f which are initialized at the reference times $t \in [t_1, t_N]$ and validate at lead times $t+k$ at a given location (e.g. a met-mast). Scores may also be calculated for a geographical region adding all local forecast observation pairs to the sample. In the following we will therefore use the index i to count the sample of spatial and temporal forecast observations pairs.

The prediction error of a deterministic forecast is defined as the difference between the predicted and the observed value.

$$\varepsilon_{i,k} = f_{i,k} - o_i \,, \tag{2.1}$$

where o is the observation. There are two basic criteria for illustrating a predictor performance: the Mean Absolute Error (*MAE*) and the Root Mean Squared Error (*RMSE*). The mean absolute error is defined as

$$MAE(k) = \frac{1}{N}\sum_{i=1}^{N}\left|\varepsilon_{i,k}\right|, \tag{2.2}$$

while the root mean squared error is

$$RMSE(k) = \sqrt{\frac{\sum_{i=1}^{N}\left(\varepsilon_{i,k}\right)^2}{N}}\,, \tag{2.3}$$

Both systematic and random errors contribute to the *MAE* and *RMSE* criteria. Statistically the values of the *MAE* are associated with the first moment of the prediction error while the values of the *RMSE* are associated with the second order moment, and hence are associated to the variance of the prediction error. For the *RMSE* large prediction errors have the largest effect.

The Anomaly Correlation Coefficient (*ACC*) is the correlation between the forecast and analysed deviations from climate ($a_{i,k}^f = f_{i,k} - c_i$ and $a_i^o = o_i - c_i$):

$$ACC(k) = \frac{\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(a_{i,k}^f - \overline{a}^f\right)\left(a_i^o - \overline{a}^o\right)}}{\sigma(a^f)\,\sigma(a^o)}\,, \tag{2.4}$$

where $\sigma(x)$ denotes the standard deviation. The ACC can be regarded as a skill score with reference to the climate and is therefore sensitive to similarities in forecasts and analysed patterns, rather than their absolute values. Note that the climatological values c_i may vary with location and time of the year.

For a description of the probabilistic scores that can be calculated with the ECMWF evaluation suite see chapter 2. For further information on the use and interpretation of all the diagnostic tools we refer to Deliverable 6.2: “Methodology for the evaluation of probabilistic forecasts”.

2.4 Comparing forecast systems

The evaluation suite is based on the quantification of the characteristic properties of ensemble forecasts relative to the climatological distribution of the variable under consideration (e.g. wind, temperature). This gain, in general terms denoted as an improvement with respect to the considered reference forecast system (benchmark), is measured by "Skill Scores" which are defined as:

$$\mathrm{SkillScore} = \frac{ScoreREF - Score}{ScoreREF} = 1 - \frac{Score}{ScoreREF}, \qquad 2.5$$

where Score is the considered evaluation criterion, which can be either a deterministic or a probabilistic measure. The reference forecast $ScoreREF$ can be an unskilled forecast such as random chance, persistence (defined as the most recent set of observations, "persistence" implies no change in condition), or climatology.

2.5 Data output

In order to facilitate easy access to the output of the evaluation suite the results are archived in netCDF files. This enables efficient storage and access to metadata, describing the forecast and verification datasets itself as well as details of the verification procedure applied. These netCDF files are used by further post-processing scripts in order to calculate average scores over seasons or other specific periods of interests and to display the results of the evaluation suite.

2.6 Display of results

In order to display the results of the evaluation suite two main MetPy scripts are available:

• plot_scores.py
This script can be used to display lead time dependent scores and skill scores (see Appendix C).

• plot_fields.py
This script can be used to display global fields of scores (see Appendix D).

Both scipts are written in MetPy and use ECMWF's graphics library MAGICS (Meteorological Applications Graphics Integrated Colour System). MAGICS provides a set of graphics routines tailored for the meteorological world e.g.field contouring, observation plotting, etc. The main parameters to be modified by users of the MetPy scripts plot_scores.py and plot_fields.py are the parameters determining the data input to be displayed, i.e. lists of fields, dates, scores and diagnostics to be used as input of the plotting programme.

2.7 Future Developments

The modular structure of the evaluation suite allows for an easy extension of the basic features to include further scores or diagnostics as well as other forecast or verification datasets. Such new features can be easily added by incorporating new modules into the software package and then simply adding parameter flags into the launch script to drive the application of such additional diagnostics.

3. Probabilistic Skill Scores

A probabilistic forecast gives a probability of an event occurring, with a value between 0 and 1 (or 0% and 100%), but the event itself is categorical. It either occurs (o=1) or does not occur (o=0), which is known after the measurement. Probabilistic scores in general quantify the difference between some special property of a forecasted probability distribution and the same property of the verifying event. The probabilistic evaluation of forecast skill provides useful results only if the scores are calculated for a large sample of forecast/observation pairs. Murphy (1993) described nine aspects (called

"attributes") that contribute to the quality of a forecast. The most important are reliability, sharpness resolution and uncertainty. For a detailed description see e.g. WWRP/WGNE (2009), McSharry et al. (2009). The statistical measures (scores) described hereafter try to quantify the forecasts quality according to these different aspects.

3.1 Spread of an ensemble forecast

The spread of an ensemble forecast $f_{i,k}$ at lead time k measures the differences between the forecasted values of the ensemble members $m = 1,...,M$. Using the ensemble mean $\bar{f}_{i,k}$ defined as:

$$\bar{f}_{i,k} = \frac{1}{M}\sum_{m=1}^{M} f_{i,k,m} \qquad 3.1$$

the spread or the standard deviation $\sigma_{i,k}^{f}$ of the ensemble forecast $f_{i,k}$ is:

$$\sigma_{i,k}^{f} = \sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(f_{i,k,m} - \bar{f}_{i,k}\right)^2} \; . \qquad 3.2$$

The spread is an indicator for the uncertainty of the ensemble forecast. If the ensemble is well calibrated, a large spread should indicate a large uncertainty of the forecast. The spread also indicates what is not likely to happen, which at times might be as important as knowing what is likely to happen. Only when the spread might cover most of the climatological range nothing can be deduced from the forecast about the significant deviations from the climatological normal.

When evaluating ensemble forecasts, it is useful to combine the RMSE of the ensemble mean (RMSerror) and the root mean squared spread (RMSspread) of the evaluation sample

$$\sigma_{k}^{f} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\sigma_{i,k}^{f}\right)^2} \qquad 3.3$$

in one graph. In the mean, the RMSerror should be equal to the RMSspread. Otherwise, if $RMSerror < \sigma_{k}^{f}$, the ensemble has too much spread and if $RMSerror > \sigma_{k}^{f}$, there is not enough spread. An example is given in Figure 3.1 showing the RMSerror and the RMSspread at different forecast lead times.

An evaluation of the RMSerror and RMSspread relation conditional on the forecasted spread can be performed by ranking the forecast observation pairs of the sample according to their forecasted spread values. From this ranking subclasses of equal size are separated and for each class the average RMSerror and RMSspread values are calculated. The resulting diagrams show the mean relation between spread and skill for different spread classes and therefore indicate the reliability of the spread. That is why these diagrams are sometimes called spread-reliability diagrams (see e.g. Leutbecher & Palmer, 2008).

It can be seen from Figure 3.2 that for short lead times (1day) most of the ensembles have not enough spread and the deviations from the diagonal line depend on the forecasted spread. It is interesting to see that the combination of the four ensembles (black line) is located much closer to the diagonal than the single systems but now indicating some more spread than necessary (black line below the diagonal). Obviously, the combination of the ensembles cause a positive change in the spread-reliability compared to the single systems. For longer lead times (7 days) the spread is becoming more reliable and the graphs of most systems are close to the expected diagonal line. This is due to the fact that the spread of most of the global ensemble systems is optimized for the medium range (3-5 days).

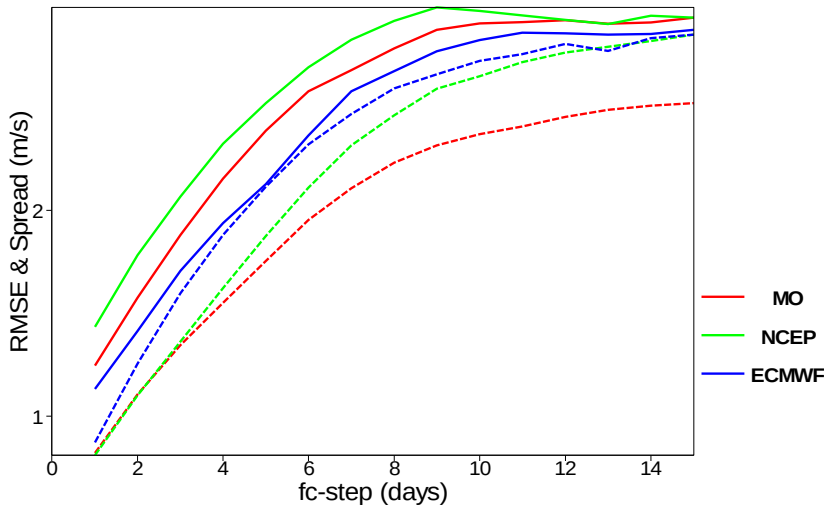

Figure 3.1: *RMSE of the ensemble mean (solid lines) and mean spread (dashed lines) of the ensemble forecasts for 10m wind speed predictions of three global EPS (UK MetOffice: red, US National Center for Environment Prediction: green, ECMWF: blue). Scores calculated for forecasts started in the period DJF 2008/09 and averaged over all grid points in the Northern Hemisphere (20°N - 90°N)*

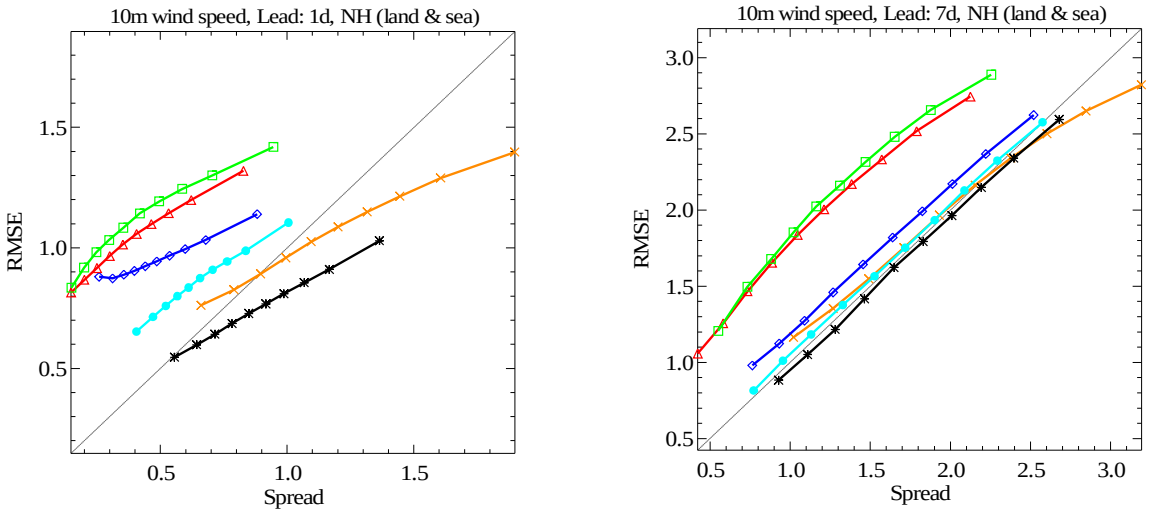

Figure 3.2: *Spread reliability diagrams for lead times of 1 and 7 days. Green: NCEP, red: MetOffice, blue: ECMWF, orange: CanadianMetCenter, light blue: calibrated ECMWF using reforecasts, and black: TIGGE combination of NCEP, MO, ECMWF and CMC for forecasts started in the period DJF 2008/09 and averaged over all grid points in the Northern Hemisphere (20°N - 90°N).*

3.2 The Brier Score and the Ranked Probability Score

The Brier score (*BS*) is similar to the *RMSE*, except that it compares probabilities instead of forecasted and observed values. The *BS* measures the difference between the forecasted probability of an event (f) and its occurrence (o, expressed as 0 or 1, if the event has occurred or not). Defining an event by the exceedance of a threshold τ of an observable X, the Brier score is defined as:

$$BS_\tau = \frac{1}{N}\sum_{i=1}^{N}(f_{i,\tau} - o_{i,\tau})^2 \,, \qquad 3.4$$

where $i = 1,...,N$ counts the sample. As with RMSE, the lower the Brier score the "better". It was shown by Murphy (1973) that the Brier score can be decomposed into three factors: reliability, resolution and uncertainty.

Summing up the Brier score for all possible event thresholds τ of the observable X leads to the **Ranked Probability Score** (*RPS*):

$$RPS \;=\; \frac{1}{\mathrm{T}-1}\sum_{\tau=1}^{\mathrm{T}} BS_\tau \;=\; \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\mathrm{T}-1}\sum_{\tau=1}^{\mathrm{T}}\left(f_{i,\tau} - o_{i,\tau}\right)^2 . \qquad 3.5$$

Assume that the observable *X* is binned in categories $l = 1,...,NC$, where a category embraces a range of values from the observable *X*. This is in contrast to the role of the threshold τ which splits all values of *X* in two parts, the one that is below and the other that is above the threshold. The binning of X should allow for all observations and forecasts to be associated to one of the categories. Then the exceedance probabilities p_τ for a threshold τ can be calculated by adding up the probabilities p_l of all categories greater than τ. That is:

$$p_\tau = \sum_{l>\tau}^{NC} p_l = 1 - \sum_{l=1}^{\tau} p_l = 1 - CDF_\tau \qquad 3.6$$

CDF_τ is the cumulative distribution function at τ. If the observable is continuous, CDF_τ can be generated from a probability distribution (or probability density function, PDF) by determining the area under the distribution from -∞ up to the threshold τ. Figure 3.3 shows an example of a probability distribution where the value of CDF_τ for a threshold τ is indicated by the grey area (p).

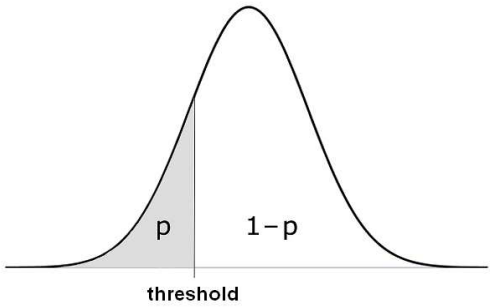

Figure 3.3: *Example of a probability distribution showing the value p of the cumulative distribution function (CDF) for a given threshold.*

Using the definition of the cumulative distribution function the *RPS* can also be written as:

$$\overline{RPS} \;=\; \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\mathrm{T}-1}\sum_{\tau=1}^{\mathrm{T}}\left(CDF_{i,\tau}^{f} - CDF_{i,\tau}^{o}\right)^2 . \qquad 3.7$$

The *RPS* measures the difference between the cumulative distribution functions of the forecast and the observation for a given sample $i = 1,...,N$ of forecast/observation pairs (the overbar denotes the averaging over the data sample). In the continuous version of the ranked probability score (CRPS, see e.g. Hersbach, 2000) the sum over the thresholds τ is substituted by the integral over the entire forecasted probability distribution:

$$\overline{CRPS} \;=\; \frac{1}{N}\sum_{i=1}^{N}\int_{-\infty}^{\infty}\left(CDF_i^{\,f}(x) - CDF_i^{\,o}(x)\right)^2 dx \qquad \mathbf{3.8}$$

For an ensemble with M members the calculation of the C*RPS* is illustrated in Figure 3.4.

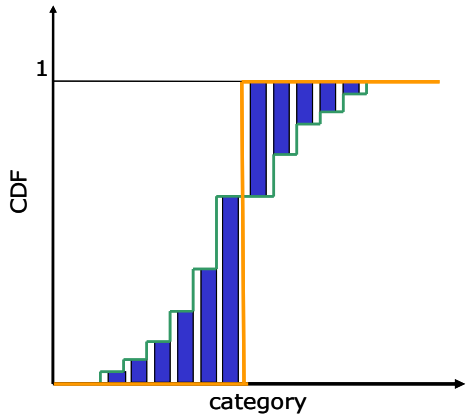

Figure 3.4: *Calculation of the continuous ranked probability score (CRPS) for an ensemble of M members. Green is the cumulative distribution function (CDF) of the ensemble forecast and orange is the CDF of the observation. Blue are the differences between both CDF's for each category.*

The cumulative distribution is then given by an M-step piecewise constant function. Depending on the position of the verifying analysis the CDF^o of the observation is 0, 1 or partly 0 and partly 1 over the constant pieces. Thus the Integral in 2.7 reduces to

$$CRPS = \sum_{m=0}^{M} \alpha_m \left(CDF_m^f\right)^2 + \beta_m \left(1 - CDF_m^f\right)^2 , \qquad \textbf{3.9}$$

where α is the part of the constant piece for which $CDF^o = 0$ and β is the part for which $CDF^o = 1$.

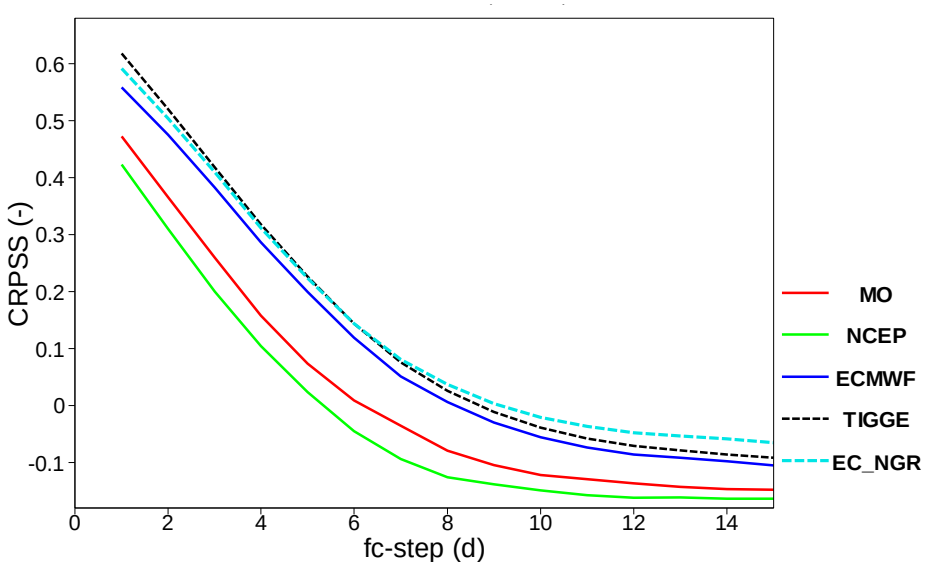

Figure 3.5: *Typical example of the visualisation of the results of the evaluation suite, here comparing the performance of 10m wind speed forecasts from different global ensemble prediction systems (red solid line: UK MetOffice, green solid line: US National Center for Environment Prediction, blue solid line: ECMWF operational EPS, black dashed line: TIGGE multi-model system combining MO, NCEP and ECMWF, cyan dashed line: experimental version of ECMWF calibrated forecast system). The Continuous Ranked Probability Skill Score (CRPSS) is calculated for forecasts started in the period DJF 2008/09 and averaged over all grid points in the Northern Hemisphere (20°N - 90°N).*

Figure 3.5 shows an example of Continuous Ranked Probability Skill Scores (CRPSS), where the benchmark for the calculation of the skill scores (see definition in section 2.4) is climatology. Three different global ensemble forecast systems are compared with a combined forecast of all three systems having equal weights and the calibrated ECMWF-EPS using Non-homogenous Gaussian regression (NGR, see Gneiting et al., 2005). For a comparison of calibration methods see e.g. Wilks (2006).

3.3 Expected Ignorance

Assume that the forecasted and observed probabilities *f* and *o* are not related to a threshold but to a category *l* of the observable *X*. A measure of forecast skill, which compares the forecasted probabilities f_l for all event categories $l = 1,...,NC$ of an observable X with those of the observation o_l, is the expected Ignorance (see Roulston & Smith, 2002), defined as:

$$E[IGN] = -\sum_{l=1}^{NC} o_l \log_2(f_l). \qquad 3.10$$

The logarithm of the forecasted probability for outcome l, which is called ignorance $IGN_l = -\log_2 f_l$, determines the number of bits necessary to encode the outcome of an event l in an optimal data compression scheme. If the probability f_l is low, much information is needed to encode the outcome. The expected Ignorance specifies the part of the encoding which matches to the observed probability o_l. Thus, it is a measure of the difference between the probability distributions. It has a minimum, if and only if $f_l = o_l$.

3.4 Rank Histogram

Another way of analysing the probability forecast of an ensemble system is to construct a rank histogram, also called Talagrand diagram (see e.g. Hamill 2001). Rank histograms are usually generated for ensemble systems with a limited number of members. In the case of continuous probability forecast (e.g. kernel dressed ensemble forecasts) the Probability Integral Transform (PIT) can be used (see McSharry et al., 2009).

If the probability forecast of an ensemble is well calibrated, the observation is equally likely to lie between any two ordered adjacent members, including the cases when the observation will be outside the ensemble range on either side of the distribution. Then the rank histogram should be flat with the same number of verifications in each interval. Especially due to the limited size of the ensemble, the observation may lay outside the ensemble range. For a system with 51 members this will happen 2/51 (~4%) of the time.

Figure 3.6 shows examples of a rank histogram for the 10m wind speed forecast at different lead times. The rank histogram distribution is U–shaped due to the over-representation of cases when the verification falls outside the ensemble and under-representation when it falls in the ensemble centre. If the U-shape degenerates into a J-shape, the system has a bias for this parameter.

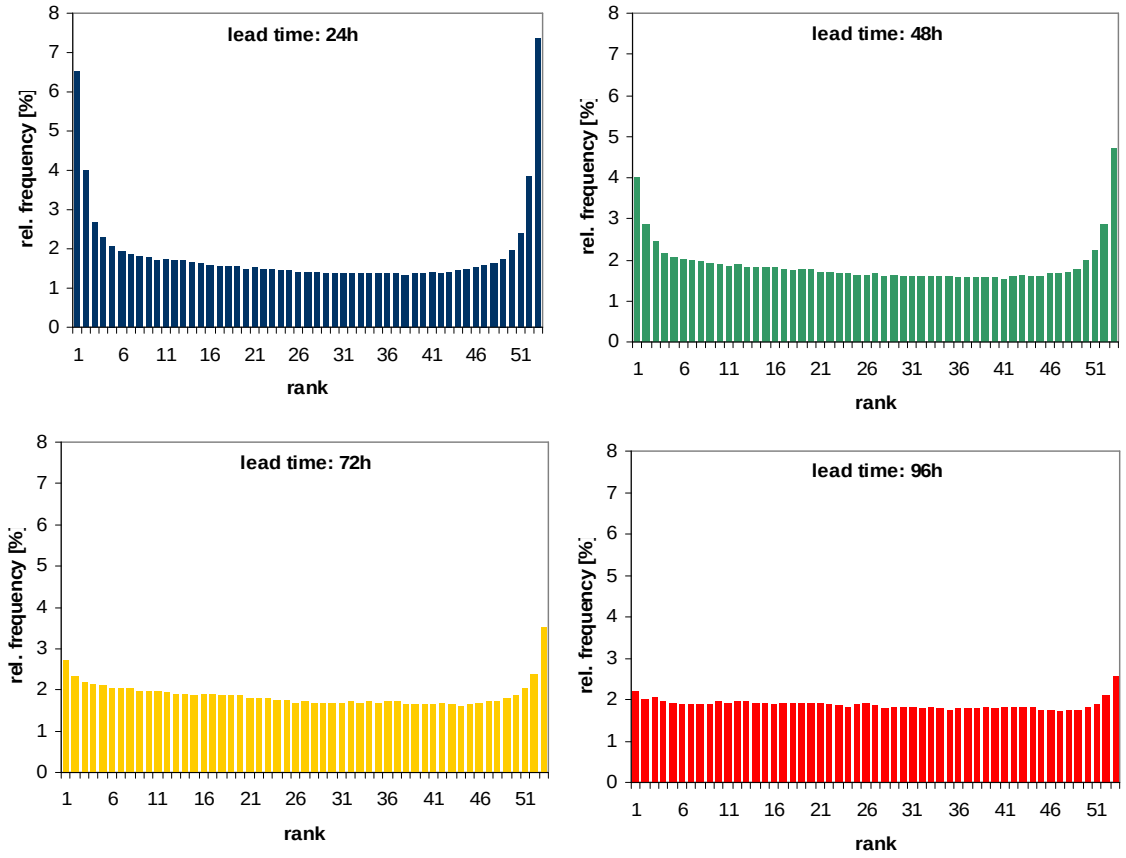

Figure 3.6: *Examples of rank histograms for 10m wind speed over Europe. The rank of the analysis in a 52 member combined prediction system is determined at each grid point of a 0.25°x0.25° reduced Gaussian grid for the period December 2008 to February 2009 (DJF). It can be seen that the U-shape for shorter lead times (indicating to less spread of the ensemble) changes to a more or less flat rank histogram for longer lead times at approximately 1/52=1.9% relative frequency. The analysis exceeds the maximum predicted value of the ensemble (rank 53) more often than its minimum value (rank 1).*

3.5 Methods for dichotomous (yes/no) forecasts

The error measures discussed in section 2.3 are based on the difference between the observed and the forecasted value. But none of the scores separate categories of events. For example, when predicting a cut-off event, (i.e. wind speeds greater than the cut-off wind speed of the turbines) it is not necessary to know exactly what the wind speed will be. It is sufficient to know, if the wind speed exceeds the cut-off speed or not. To assess the forecast skill under these conditions the forecast is simplified to a yes/no statement (categorical forecast). Similarly, the observation is assigned to one of the two categories: event observed/not observed.

Let h denote “hits”, i.e. all correct yes-forecasts - the event is predicted to occur and it does occur, f “false alarms”, i.e. all incorrect yes-forecasts, m “misses” (all incorrect no-forecasts) and z “zeros” all correct no-forecasts. Assume altogether N forecasts of this type with h + f + m + z = N. A perfect forecast sample is when f and m are zero. Table 1 summarizes these numbers in a so called contingency table. Many verification scores can be computed from these numbers. We will review only those with relevance for wind and wind power forecast evaluation (see WWRP/WGNE for more details).

Table 1: *A forecast/verification table*

Data	observed	not observed	Total
forecasted	*Hits (h)*	*False alarms (f)*	forecasted yes
not forecasted	*Misses (m)*	*Correct zeros (z)*	forecasted no
Total	observed yes	observed no	N

Frequency Bias (Bias score): FBIAS = ($h + f$) / ($h + m$)
compares the frequency of forecasted events to the frequency of observed events. **Range:** 0 to infinity. **Perfect score:** 1. Indicates whether the forecast system has a tendency to under-forecast (FBIAS < 1) or over-forecast (FBIAS > 1) events.

Probability of Detection (Hit Rate): POD = h / ($h + m$)
is the fraction of observed events that were correctly forecast. **Range:** 0 to 1. **Perfect score:** 1. It ignores the false alarms and can therefore be artificially improved by issuing more "yes" forecasts to increase the number of hits (over-forecast).

False Alarm Ratio: FAR = f / ($h + f$)
gives the fraction of the forecasted “yes” events that were false alarms. **Range:** 0 to 1. **Perfect score:** 0. Sensitive to false alarms, but ignores misses.

Probability of False Detection (False Alarm Rate): POFD = f / ($z + f$)
is the fraction of false alarms given the event did not occur (relative to observed “no” events). **Range:** 0 to 1. **Perfect score:** 0. It can be artificially improved by issuing fewer "yes" forecasts to reduce the number of false alarms.

Threat score (Critical Success Index): TS = h / ($h + m + f$)
measures the fraction of the observed and forecasted “yes” events that were correctly predicted - ignoring correct negatives. **Range:** 0 to 1, 0 indicates no skill. **Perfect score:** 1. TS is only concerned with forecasts that count. Depends on climatological frequency of events (poorer scores for rarer events) since some hits can occur purely due to random chance (-> ETS).

Equitable Threat score: ETS = ($h - h_{ran}$) / ($h + m + f - h_{ran}$)
where h_{ran} = ($h + m$)($h + f$)/N are the hits due to random chance. **Range:** -1/3 to 1, 0 indicates no skill. **Perfect score:** 1. It corrects the TS for hits associated with random chance.

Heidke skill score: HSS = (h + z - hz_{ran}) / ($N - hz_{ran}$)
where hz_{ran} = [$(h+m)(h+f) + (z+m)(z+f)$] / N are the expected correct forecasts due to random chance. **Range:** -∞ to 1, 0 indicates no skill. **Perfect score:** 1. Measures the fraction of correct forecasts after eliminating those forecasts which would be correct due to random chance.

True skill statistic: TSS = POD - POFD
measures how well did the forecast separate the "yes" events from the "no" events. **Range:** -1 to 1, 0 indicates no skill. **Perfect score:** 1. The TSS does not depend on the climatological frequency of the event, but for rare events the TSS is weighted towards the POD term, because then most forecasts will be correct negatives and the second term (POFD) is close to zero.

Extreme Dependency Score (EDS): EDS = 2 log(($h + m$)/N)/(log(h/N)) - 1
compares the fraction of the observed events with the fraction of the correctly forecasted events (see Stephenson et al., 2008). **Range:** -1 to 1. **Perfect score:** 1. It has been suggested as an alternative to more common contingency table scores, since the EDS does not tend to zero for rare events.

3.6 Decision making from probabilistic forecast systems

In contrast to a deterministic forecast system, which predicts an event (e.g. wind speed > 25m/s) by a yes/no decision, a probabilistic forecast system assigns a probability *p* between 0 and 1 to the event. Nevertheless, users can generate dichotomous (yes/no) forecasts (warnings) from the ensemble, if they specify a threshold τ_p for the forecasted probability. The probabilistic forecast system must then predict the event with at least τ_p to generate a warning.

To analyse the skill of such warnings, contingency tables (see Table 1) can be generated for every threshold of the forecasted probability τ_p of an event. A "hit" is then a forecast that predicts the event (e.g. wind speed > 25m/s) with a probability greater than τ_p. For each contingency table the scores listed in the previous section were calculated and plotted against τ_p. The ECMWF evaluation suite provides the contingency tables.

Figure 3.7 shows an example for 84h forecasts of hourly mean wind speeds in approximately 105m height over Europe for the 2008 DJF period compared to the analysis at the same height on a 0.25°x0.25° grid. Here the event is defined as the occurrence of wind speeds between 4 and 12m/s, which are related to an increased level of wind power forecast uncertainty, because of the non-linear power curve transformation. Moreover, these wind speeds might be related to ramping events.

Figure 3.7 illustrates the different aspects of deterministic and probabilistic forecast systems. In addition to the scores from the EPS some selected scores for the deterministic high resolution model of ECMWF (IFS) are plotted on the y-axis. It can be seen that probability thresholds can be found where the best EPS scores exceeds the deterministic scores.

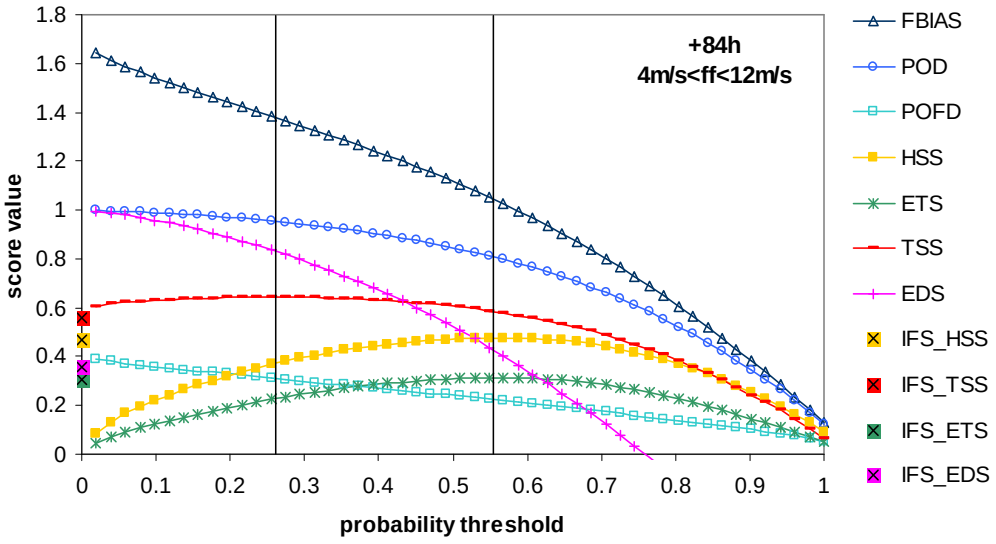

Figure 3.7: *Contingency table scores for hourly wind speeds in approximately 105m height between 4 and 12 m/s over Europe for the DJF period 2008 plotted against the probability threshold. For every probability threshold a contingency table is generated. A correct forecast (hit) was issued, if the probability of wind speeds between 4 and 12m/s exceeds the probability threshold. The squares with the crosses mark the score values for the deterministic IFS forecast. The black vertical lines are two possible user defined thresholds for decision making. The right line marks a threshold of 55% with maximum HSS and the left line a probability threshold of 27% with maximum TSS.*

It is also obvious from Figure 3.7 that the EPS provides much more flexibility for making user decisions. For example the vertical black lines mark two different probability thresholds a user can choose to issue a warning. The right threshold is chosen to have a maximum Heidke skill score (*HSS*), what means a maximum fraction of correct forecasts (hits and correct negatives). Figure 3.8 shows, that this is related to approximately the same number of false alarms and misses. But if the costs of a missed event are greater than the costs of a false alarm, it would be more appropriate to choose the left threshold, because the user can accept to over-forecast the event (*FBIAS*>1). As it can be seen from Figure 3.8, the left threshold is associated to a much smaller number of misses but a greater number of false alarms. How much more misses a user can accept depends on the difference between the costs of misses and false alarms, the cost-loss ratio.

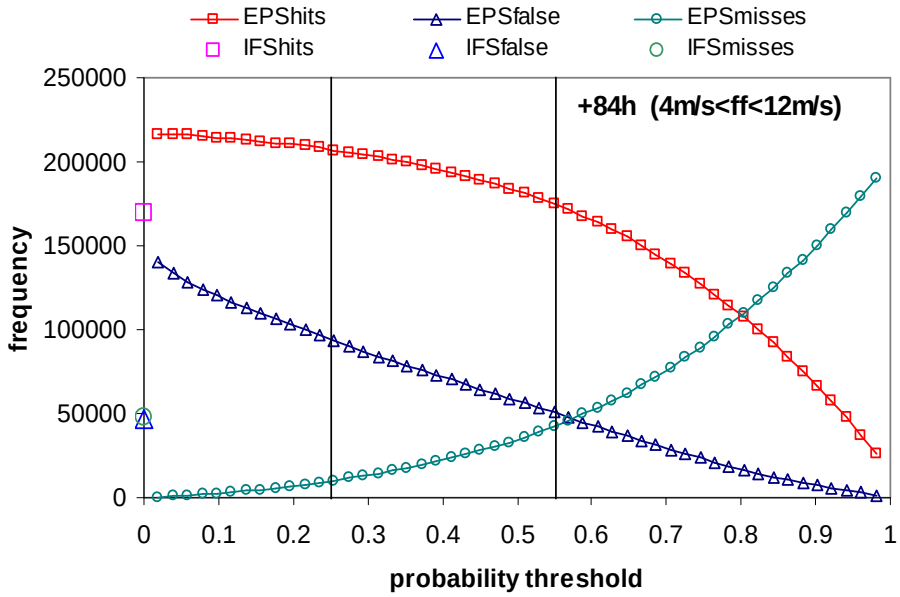

Figure 3.8: *same data as in Figure 3.7, but showing the basic items hits misses and false alarms of the contingency tables.*

3.7 Reliability diagram

A forecast is reliable, if the predicted probabilities are equal to the observed frequencies of an event. But the reliability is not sufficient to characterize the quality of a probabilistic forecast. For example, a forecast based on climatology is perfectly reliable, but has no skill.

The reliability diagram (see e.g. Atger 2004) plots the observed frequency of the event against the forecasted probability, where the range of forecasted probabilities is divided into bins (for example, 0-5%, 5-15%, 15-25%, etc.). The diagonal line indicates perfect reliability (the average observed frequency is equal to the predicted probability for each category), and the horizontal line represents the climatological frequency. Sometimes sample sizes are plotted either as a histogram, or as numbers next to the data points. An example of a reliability diagram can be found in Atger et al. (2009). McSharry et al. (2009) give more detailed guidance on how to use and interpret the reliability diagram.

3.8 The ROC-diagram

Using a set of increasing probability thresholds (for example, 0.05, 0.15, 0.25, etc.) to make the yes/no decision and plotting the Hit Rates (POD) vs. the False Alarm Rates (POFD) generates the two-dimensional so called Relative Operating Characteristics or ROC-diagram. A point in the ROC diagram for a given probability threshold is defined by the POFD value on the x-axis and the POD value on the y-axis.

The upper left corner of the ROC-diagram represents a perfect forecast system where there are no false alarms and only hits. The closer the point is to this upper left corner the higher the skill. The lower left corner, where both hit and false alarm rate are zero, represents a system which never warns of an event. The upper right corner represents a system where the event never occurs. In reality a non-perfect system will have its values on a long convex curve pointing to the upper-left corner (the “ROC curve”). The area under the ROC curve is often used as a measure of forecast skill.

The ROC curve enables a comparison between a probabilistic and a deterministic forecast system. If the deterministic value lies above the ROC curve, the deterministic system is more skilful than the probabilistic. However, in terms of utility, greater advantages might be gained from the probabilistic information. Only probabilistic forecast systems enable the minimisation of a user specific cost-function, as it is shown in section 3.6. Therefore, it takes very good deterministic forecasts to be more useful than probabilistic ones. But every good deterministic forecast can be added to a probabilistic forecast system making the combined system more useful than any of its components!

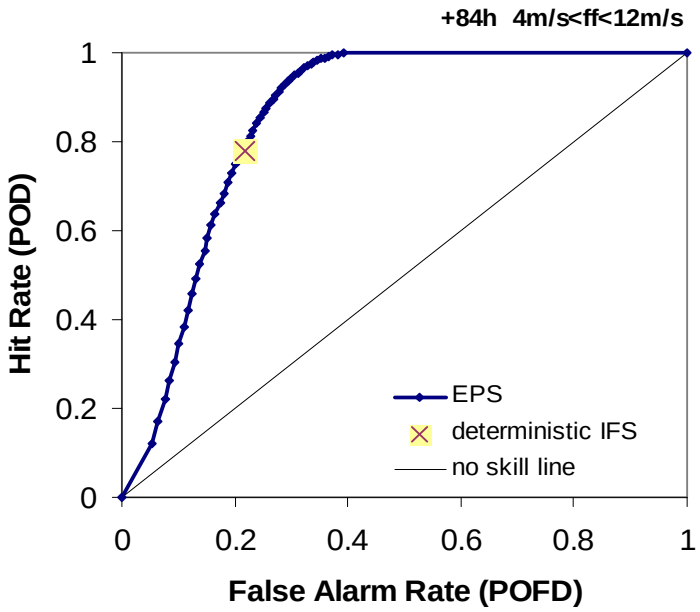

Figure 3.9: *Example of a ROC-diagram. The data is the same as described in the caption of Figure 3.7.*

4. Conclusions

The ECMWF evaluation suite, which has been described in this document, will be used to develop an optimized probabilistic forecast system for wind power prediction. We have described the verification methodology provided by the suite and developed additional components that will be added to the operational suite during the SafeWind project.

Together with the SafeWind deliverable Dp-6.2 “Methodology for the evaluation of probabilistic forecasts” (McSharry et al., 2009) this report provides an overview of the verification of wind and wind power forecasts. The focus of both documents is on the evaluation of probabilistic forecasts, because only these forecasts enable the minimisation of a user specific cost-function. Moreover, every deterministic system or stochastic process that produces good point forecasts can be added to a probabilistic forecast system making the combined system more useful than any of its components.

We also pointed out (see also McSharry et al. 2009) how probabilistic forecasts can be used for decision making. Probabilistic forecasts provide full flexibility to practitioners. The ensemble mean can be used as a best first guess point forecast having a smaller mean error than any of its members. The

uncertainty information shows that alternative scenarios exist, but also what is not likely to occur - which is sometimes good to know.

When forecasting extreme events like cut-off's, ramping or periods of extreme variability in the power output of a wind farm, probabilistic forecasts are of particular value. As it is described in section 3.6 probability thresholds can be used to optimize user specific cost-loss functions which mainly determine the costs of a missed event and a false alarm. This cost-loss function can be kept very simple. Only the ratio between the costs of a missed event and a false alarm is needed. If this ratio is known, a probability threshold can be determined that provides an optimum cost-loss benefit. This threshold may also depend on the lead time, e.g. using relatively low thresholds for longer lead times to minimize the number of missed events but using higher thresholds for shorter lead times to reduce the number of false alarms

5. References

Atger F., Baldwin, M., Brill, K., Brooks, H., Brown, B., Casati, B., Damrath, U., Ebert, B., Ghelli, A., Göber, M., Jenkner, J., Jolliffe, I., Nurmi, P., Stephenson, D., Wilson, C., Wilson, L., 2009: WWRP/WGNE Joint Working Group on Forecast Verification - Issues, Methods and FAQ, http://cawcr.gov.au/bmrc/wefor/staff/eee/verif/verif_web_page.html

Atger, F., 2004: Estimation of the reliability of ensemble-based probabilistic forecasts, Quarterly Journal of the Royal Meteorological Society, Volume 130, Issue 597, 627-646

Bougeault, P., 2003: The WGNE survey of verification methods for numerical prediction of weather elements and severe weather events. Meteo France, Toulouse, France.

Casati, B., Wilson, L.J., Stephenson, D.B., Nurmi, P., Ghelli, A., Pocernich, M., Damrath, U., Ebert, E.E., Brown, B.G. and S. Mason, 2008: Review, forecast verification: current status and future directions. Meteorol. Appl. 15, 3-18.

Giebel, G., ….. , Denhard, M. 2009: Report on the status of actual wind power forecasting technology, Technical Report, EU project SafeWind, Deliverable Report Dp-1.5.

Gneiting, T., Raftery, A. E., Westveld, A. H. and Goldman, T. (2005): Calibrated probabilistic forecasting using ensemble model output statistics and minimum CRPS estimation. Monthly Weather Review, 133, 1098-1118.

Hamill, T. M., 2001: Interpretation of rank histograms for verifying ensemble forecasts. Mon. Wea. Rev., 129, 550-560.

Hersbach, H., 2000: Decomposition of the Continuous Ranked Probability Score for Ensemble Prediction Systems. Weather and Forecasting, vol. 15, issue 5, pp. 559–570

Leutbecher, M. and T. N. Palmer, 2008: Ensemble forecasting. J. Comp. Phys. 227, 3515-3539.

McSharry, P. E., Pinson, P. and Girard, R., 2009: Methodology for the evaluation of probabilistic forecasts, Technical Report, EU project SafeWind, Deliverable Report Dp-6.2.

Murphy, A.H., 1973: A new vector partition of the probability score. J. Appl. Meteor., 12, 595-600.

Murphy, A.H., 1993: What is a good forecast? An essay on the nature of goodness in weather forecasting. Weather and Forecasting, 8, 281-293.

Roulston, M.S. and L.A. Smith, 2002: Evaluating probabilistic forecasts using information theory. Mon. Wea. Rev., 130, 1653-1660.

Stephenson, D.B., Casati, B., Ferro, C.A.T. and C.A. Wilson, 2008: The extreme dependency score: a non-vanishing measure for forecasts of rare events. Meteorological Applications, vol. 15, issue 1, pp. 41-50.

Wilks, D. S., 2006: Comparison of ensemble-MOS methods in the Lorenz '96 setting. Meteorol. Appl. 13, 243–256.

Appendix A: Structure of evaluation suite

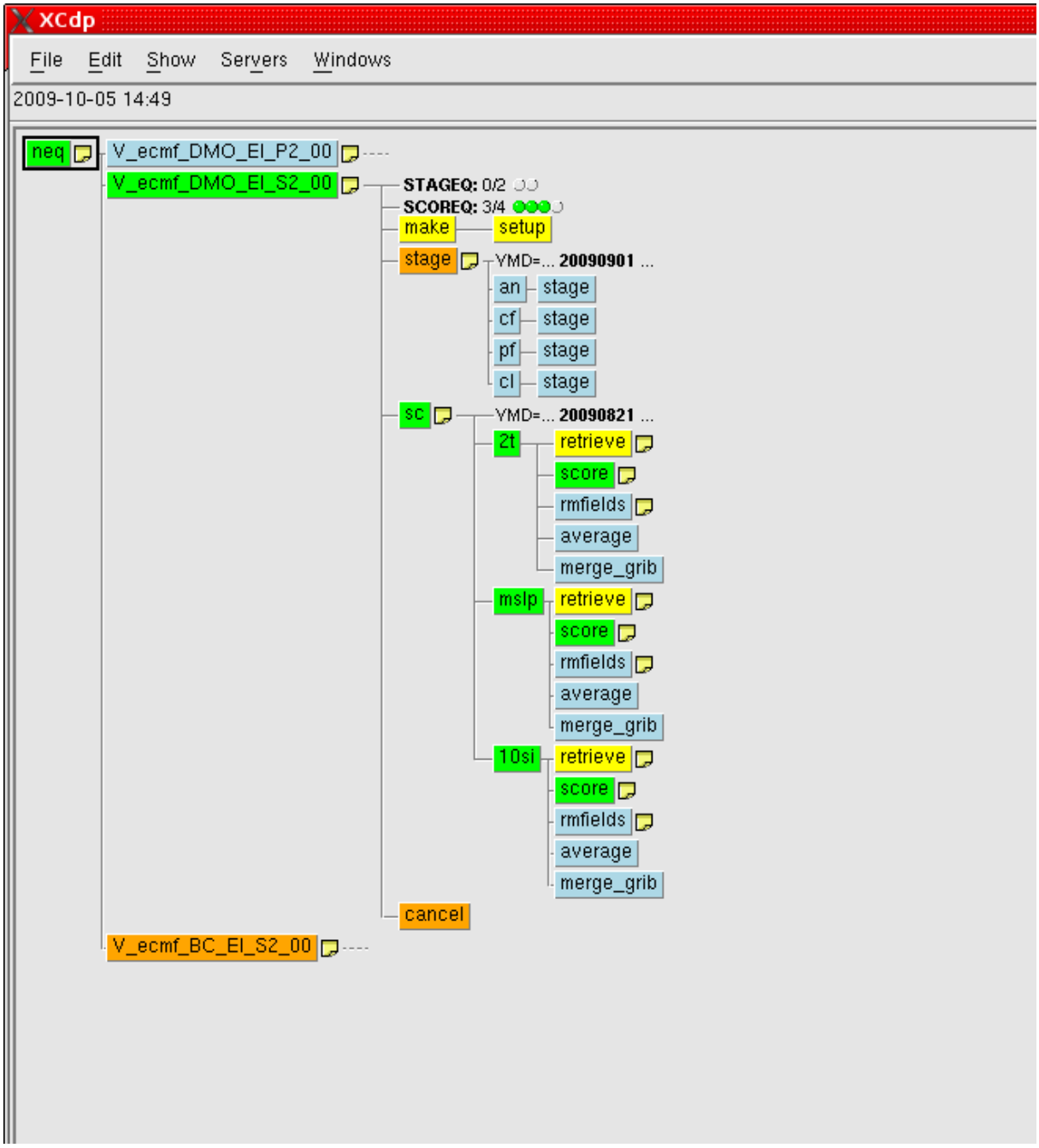

Figure A1: *Screenshot of the evaluation suite running under SMS/XCDP. Individual boxes represent individual tasks, with the colours of the boxes indicating the status of the tasks (green: active, yellow: completed, light blue: queued, orange: suspended).*

Appendix B: Script to launch the evaluation suite

Program SafeWind_verify.ksh

```
#!/bin/ksh
#
# launch scoring of EPS forecasts (ECMWF)
#
set -ex
#
# 1. User settings, that define scoring
#
LDLIST=0       # flag whether to use a list of dates or dates from D1 to D2
               # 0: use dates between D1 and D2
D1=20090601    # first date
D2=20091231    # last date
               # 1: use a list of dates (DLIST)
DLIST="2009060112 2009060312 2009060512"
#
DELTA=1        # step in days
if [[ $LDLIST -eq 1 ]]; then
  HH=""
  suite_suffix=""
else
  HH=00
  suite_suffix="_$HH"
fi
#
LAST_DATE=2009083100       # Date, at which averaging of scores is launched
#
VSTREAM=SAFEWIND_REF       # main verification stream identifier
EPSVERIPY_VERSION='1.6.0'  # EPSVERIPY version identifier
#
# perturbed forecasts
#
originEPS=ecmf; expverEPS=prod; streamEPS=enfo; classEPS=ti; NPF=50
#
# analysis
#
originANA=erai; expverANA=0001; streamANA=da; classANA=ei
#
# high-resolution forecast
#
expverHR=0001; streamHR=da; classHR=od
#
SCORECONTROL=true          # flag whether to score control forecast
SCOREHIGHRES=false         # flag whether to score high-resolution forecast
SCOREMM=false              # flag whether to score multi-model forecast
#
SAVE_SCORES_MONTHLY=false  # flag whether to save scores on monthly basis
#
PARAMS="Z/T/U"             # parameters to be scored
LTYPES="PL/PL/PL"          # level-types to be scored
LEVELS="500/850/850"       # levels to be scored
#
GRID=2.5 ; export GRID     # grid used for verification
#
DEBIAS=False               # True | False   : debias data
FILTMODE=none              # spfilt | none  : spectral filtering of data
if [[ $FILTMODE = 'spfilt' ]]; then
   BANDS="UF T8to21"
elif [[ $FILTMODE = 'none' ]]; then
   BANDS="UF"
else
  exit 1
fi
#
PERTURB_FC=false           # perturb forecasts to account for analysis error
SCALE_ANERR_STDEV=1.3      # scale of analysis error
#
#
WRITESPERRGRIB=True        # True | False write grib fields with error and spread
WRITESCOREFIELDS=True      # True | False write grib fields with CRPS ...
#
WSHOST=swarm               # workstation host to be used for running the suite
#
# write score configuration definition file
#
cat > $TMPDIR/score_conf_suite.def <<EOFSC
[data]
format=grib
grid=${GRID},${GRID}
resol=63
levelist=:env:
param=:env:
levtype=:env:
```

```
[score]
areas=n.hem,n.hem.mid,n.subtropics,s.hem,s.hem.mid,s.subtropics,tropics,tropics.x,europe,none
bin_event_thresholds=-0.5,0.,0.5
bin_event_thr_type=stdev,stdev,stdev
bin_event_operators=<,>,>
bin_event_anomalies=true,true,true
write_fields=spread,score
use_coslat_weights=true
[time]
steps=24,to,360,by,24
[control forecast]
class=$classEPS
origin=$originEPS
expver=prod
score=$SCORECONTROL
[ensemble]
class=$classEPS
origin=$originEPS
expver=prod
npf=$NPF
include_control_in_ens=$SCORECONTROL
multi_model_ens=$SCOREMM
[high-resolution forecast]
score=$SCOREHIGHRES
[analysis]
class=$classANA
expver=$expverANA
stream=$streamANA
EOFSC
#
# 2. General auxiliary variables, usually untouched by standard user
#
filtmode1=$(echo $FILTMODE | cut -c 1 )
if [[ $DEBIAS = 'True' ]]; then
  debias1=bc
else
  debias1=''
fi
SUITE=V_${VSTREAM}${suite_suffix}
export SUITE
SDIR=$SCRATCH/Ev/$SUITE; export SDIR
DATADIR=$SCRATCH/Ev/$SUITE; export DATADIR
ONLINEDIR=/gpfs1/eps_verify/neq/data; export ONLINEDIR
SCRIPTSDIR=$HOME/EPSverify/
SRCDIR=$SCRIPTSDIR/src
#
# 3. Submission procedure, should not be touched
#
# a: tidy up, set up constant files
#
if [[ -d $SDIR ]]; then
  chmod -R u+w $SDIR
  find $SDIR -type f -exec rm {} \;
else
  mkdir -p -m 755 $SDIR
fi
cp     $SCRIPTSDIR/def/*.def     $SDIR
cp -R  $SCRIPTSDIR/sms/          $SDIR
mkdir -p $SDIR/sms/include
cp     $SCRIPTSDIR/scripts/*.ksh $SDIR/sms/include
cp     $SCRIPTSDIR/scripts/*.h   $SDIR/sms/include
cp     $SCRIPTSDIR/sms_include/* $SDIR/sms/include
cp -R  $SCRIPTSDIR/py/           $SDIR
cp $TMPDIR/score_conf_suite.def  $SDIR/py/suite
#
chmod -R u+w $SDIR
#
cat > $SDIR/sms/include/config.ev.h <<EOFCONFIG
EPSVERIPY_VERSION=${EPSVERIPY_VERSION}
WRITESCOREFIELDS=${WRITESCOREFIELDS}
WRITESPERRGRIB=${WRITESPERRGRIB}
SAVE_SCORES_MONTHLY=${SAVE_SCORES_MONTHLY}
PERTURB_FC=${PERTURB_FC}
SCALE_ANERR_STDEV=${SCALE_ANERR_STDEV}
SCOREHIGHRES=${SCOREHIGHRES}
SCORECONTROL=${SCORECONTROL}
SCOREMM=${SCOREMM}
#
EXPVEREPS=${expverEPS}
CLASSEPS=${classEPS}
STREAMEPS=${streamEPS}
ORIGINEPS=${originEPS}
EXPVERANA=${expverANA}
ORIGINANA=${originANA}
EOFCONFIG
#
# b: build variable files
```

```
#
cd $SDIR
#
cat >scoreexp.variable.def<<EOFX
set VSTREAM $VSTREAM
set LDLIST $LDLIST
set DLIST "$DLIST"
set BANDS "$BANDS"
set PARAMS "$PARAMS"
set LTYPES "$LTYPES"
set LEVELS "$LEVELS"
set FILTMODE $FILTMODE
set D1 $D1
set D2 $D2
set DELTA $DELTA
set HH \"$HH\"
set SUITE $SUITE
set SDIR $SDIR
set DATADIR $DATADIR
set ONLINEDIR $ONLINEDIR
set GRID $GRID
set WSHOST $WSHOST
set LOGDIR $SCRATCH/Ev
set USER   $USER
set SRCDIR $SRCDIR
set DEBIAS  $DEBIAS
set LAST_DATE ${LAST_DATE}
EOFX
#
cat scoreexp.variable.def epscores.def > $SUITE.def
#
# c: submit to SMS
#
cdp <<EOF
login $USER $USER 1
play $SUITE.def
suspend $SUITE
suspend $SUITE/cancel
begin $SUITE
exit
EOF
#
```

Appendix C: Script to display scores versus lead time

Program plot_scores.py

```
#
import sys, os, re
import copy
import numpy, math
from MetPy import *
from MagPy import *
from EnsVeriPy import *
#
# ================================================
# Define common settings for all experiments
# ================================================
#
# list of fields to be plotted
#
fields=['2t_sfc','mslp_sfc','10si_sfc']
lev='S'
#
# list of binary events to be plotted
#
my_events=[BinEvent(thr_value= 0,thr_type='stdev',op='>'),
           BinEvent(thr_value= 0.5,thr_type='stdev',op='>'),
           BinEvent(thr_value= -0.5,thr_type='stdev',op='<')]
#
# list of scores or tuples of scores to be plotted
#
my_scores=[('rmse_em','spread_em')
           ,'ContinuousRankedProbabilitySkillScore'
           ,'ContinuousRankedProbabilityScore'
           ,'ContinuousRankedProbabilityScoreGaussianClimate'
           ,'RankedProbabilitySkillScoreQuan10'
           ,'RankedProbabilityScoreQuan10'
           ,'RankedProbabilityScoreQuan10GaussianClimate'
           ,'BrierSkillScore'
           ,'BrierScore'
           ,'BrierScoreGaussianClimate'
           ,'IgnoranceSkillScore'
           ,'IgnoranceScore'
           ,'ROCarea'
          ]
#
# additional definitions for curve, i.e. a tuple of dictionaries containing
# MAGICS GRAPH Plotting Parameters
#
my_curve_tuple=( dict( ),
                 dict( graph_line_style ='dash'),
                 dict(     graph_symbol     ='ON',          graph_symbol_height     =     0.4,
graph_symbol_marker_index=3),
                 dict(     graph_symbol     ='ON',          graph_symbol_height     =     0.4,
graph_symbol_marker_index=4),
                 )
#
# list of areas to plot
#
my_areas=['n.hem','n.hem.mid','n.subtropics','s.hem','s.hem.mid','s.subtropics','tropics','tro
pics.x','europe',None]
#
my_class='ti'
my_stream='enfo'
my_expver='prod'
ext='DMO'
my_vstream=ext+'_EI_'+lev+'2'
date_range='2009060100to083100'
ndates='92'
hh='00'
#
for fid in fields:
    #
    # create ScorePlot instance
    #
    p=ScorePlot( area=my_areas, event=my_events,
                 scores=my_scores, curve_tuple=my_curve_tuple)
    #
    ps_disk = '/gpfs1/eps_verify/neq/ps/'
    ps_filename = ps_disk+'sc_'+date_range+'N'+ndates+'_'+fid+'.ps'
    magics.psetc('ps_file_name',ps_filename)
    #
    # ================================================
    # Define verification data and curve properties
    # ================================================
    rqall={'fid': fid, 'class':my_class,
           'stream':my_stream, 'vstream':my_vstream,
           'expver':my_expver, 'ndates':ndates,
```

```
        'date_range':date_range, 'hh':hh,
        }
p.append_curve( label='NCEP',
                request=dict(origin='kwbc', **rqall),
                curve=dict(graph_line_colour = 'RGB(0.00,1.00,0.00)',
                           graph_line_style  = 'solid',
                           graph_line_thickness= 8),
                )
p.append_curve( label='MO'+ext,
                request=dict(origin='egrr', **rqall),
                curve=dict(graph_line_colour = 'RGB(1.00,0.00,0.00)',
                           graph_line_style  = 'solid',
                           graph_line_thickness= 8),
                )
p.append_curve( label='ECMWF'+ext,
                request=dict(origin='ecmf', **rqall),
                curve=dict(graph_line_colour = 'RGB(0.00,0.00,1.00)',
                           graph_line_style  = 'solid',
                           graph_line_thickness= 8),
                )
# ================================================
# Plot
# ================================================
p.execute()
magics.pclose()
magics.popen()
```

Appendix D: Script to display global scores

Program plot_fields.py

```
#
import sys, os, re
import copy
import numpy, math
from MetPy import *
from MagPy import *
from EnsVeriPy import *
#
# list of origins to be plotted
#
origins=['ecmf','egrr','kwbc']
#
# list of verification analysis to be plotted
#
anas=['m','a']
#
# list of fields to be plotted
#
fields=['2t_sfc',’10si_sfc’,'gh500hPa','t850hPa','u850hPa']
#
# list of diagnostics to be plotted
#
files=['crps',
       'crps_gclim',
       'crps_qclim',
       'err_em',
       'mae_em',
       'rmse_em',
       ]
#
# other settings
#
my_class='ti'
expver='prod'
date_range='2009060100to083100'
ndates='92'
mean=False
var_contour=False
offset=0.
fac=1.
unit='-'
area='Global'
cont_libname='crps_contour'
ps_dir='/gpfs1/eps_verify/neq/ps/'
for origin in origins:
    origin_org=origin
    for ana in anas:
        for field in fields:
            for file in files:
                p=ScoreMapPlot()
                vstream=''
                if field=='2t_sfc' or field==’10si_sfc’:
                    vstream=vstream+'S'
                else:
                    vstream=vstream+'P’
                request={'fid': field, 'scoreids':[file],
                         'class':my_class, 'origin':origin,
                         'expver':expver,'vstream':vstream,
                         'ana':ana, 'date_range':date_range,
                         'ndates':ndates,'mean':mean,
                         'var_contour':var_contour,'offset':offset,'
                         'fac':fac,'unit':unit,
                         'area':area, 'cont_libname':cont_libname
                        }
                #
                # setup of data retrieval etc
                #
                ps_filename = ps_dir+date_range+'_'+field+'_'+file+'.ps'
                magics.psetc('ps_file_name',ps_filename)
                p.setup(request=request)
                #
                p.execute()
                magics.pclose()
                magics.popen()
```