Report on work in Vienna

on the common verification package

Edit HÁGEL

hagel.e@met.hu

Hungarian Meteorological Service

Work was supervised by:

Yong Wang and Alexander Kann

From 16 July 2006 until 29 July 2006

Table of Contents

1. Introduction

LAMEPS related research activities are going on in many ALADIN countries. In order to enable the comparison of the results, it was decided that a common visualization and verification package is needed. During this two weeks stay work was concentrating on the verification part of this package.

In the first place we had to agree on some basic issues. These were the following:

1.1 General questions:

Question	**Answer**
Software or programming language to be used	1. For verification: FORTRAN 90 + some graphical tool e.g. gnuplot 2. For visualization: Metview
Question of „TRUTH“	1. In case of surface fields: SYNOP 2. In case of upper air fields: analysis (e.g. ECMWF or ARPEGE)
Data format of forecasts	GRIB I (but we may switch to GRIB II in the future)
Data format of analyis	GRIB I (but we may switch to GRIB II in the future)
Data format of SYNOP observations	ASCII

1.2 Questions connected with verification:

Question	**Answer**
Surface parameters to be verified	2 meter temperature, 10 meter wind, 6 and 24 hours accumulated total precipitation, 2 meter relative humidity or 2 meter dewpoint temperature, mean sea level pressure
Upper level parameters to be verified	temperature, wind, geopotential and relative humidity on 850 hPa, geopotential on 500 hPa
Matching forecasts and observations	When using observations, verification should be performed at the observation location.

	Forecasted values should be interpolated to the observation point for every variable, except for precipitation and 2 meter dewpoint temperature. In these two cases it is preferable to use the value of the nearest grid point.
Verification methods	• RMSE and BIAS of the ensemble mean, the control forecast and the individual members • Standard deviation of the ensemble members • Talagrand diagram and Percentage of outliers • Continuous Rank Probability Score • Brier score, Brier Skill score, Reliability diagram and Resolution • ROC diagram and the area under the ROC • Relative value score

1.3 Questions connected with visualization:

Question	**Answer**
What kind of products	• ensemble mean • spread • spaghetti diagram • probability maps • stamp charts • meteograms

During this two weeks stay work was concentrated on verification using **analysis** as „truth“. In the following the source code and the associated shell scripts will be described. Comments placed in the source code and in the scripts should also be read and considered.

2. The verification package: the FORTRAN codes

The verification package was written in FORTRAN 90, with some associated shell scripts which are mainly used to control the whole verification procedure. The basic parameters can be set in a namelist and a Makefile is needed to compile the FORTRAN routines. The visualization of the scores is performed by using gnuplot.

.

2.1 The namelist

A namelist (called **namelist**) is used to set the basic variables like size of the domain, directories, number of ensemble members, parameters to be verified and so on. In the following a detailed overview will be given about the different sections and parameters of the namelist file.

Namelist variable	Possible values	Description
RECORD_01:		
type_data	***GRIB*** or ***ASCII***	Type of the „truth“. In case of analysis, it must be set to ***GRIB***, in case of synop observations, it must be set to ***ASCII***
xdim	Integer number	Number of grid points in x
ydim	Integer number	Number of grid points in y
RECORD_02:		
num_ensemble	Integer number	Number of ensemble members
days_statis	Integer number	Not used yet
num_var_statis	Integer number	Number of variables
code_parameter_ANA	Integer numbers	Grib code numbers of parameters to be verified in the analysis file
code_parameter_FOR	Integer numbers	Grib code numbers of parameters to be verified in the forecast file
calc_windspeed	0 or 1	Switcg to calculate windspeed or not, 0: no, 1: yes (not used yet)
wind_code_ANA	Integer numbers	Code number of u and v in analysis file(not used yet)
wind_code_FOR	Integer numbers	Code number of u and v in forecast file(not used yet)
num_level_statis	Integer number	Number of vertical levels
level_ver_statis	Integer numbers	We perform the verification on these pressure levels
time_step_statis	Integer number	Number of time steps to verify

time_steps	Integer numbers	Time steps to verify
RECORD_03:		
path_input_ANA	String	Input directory of analysis files
path_input_FOR	String	Input directory of forecast files
path_output	String	Output directory (not used yet)
RECORD_04:		
CalcBIAR_RMSE	0 or 1	Switch to calculate BIAS and RMSE, 0 : no, 1 : yes
CaclSpread	0 or 1	Switch to calculate spread, 0 : no, 1 : yes
CalcTalagrand	0 or 1	Switch to calculate Talagrand, 0 : no, 1 : yes
CalcROC	0 or 1	Switch to calculate ROC, 0 : no, 1 : yes
CalcReliability	0 or 1	Switch to calculate Reliability, 0 : no, 1 : yes
CalcBrierScore	0 or 1	Switch to calculate Brier score, 0 : no, 1 : yes
RECORD_05:		
NumThresholdWind	Integer number	Number of thresholds for wind
ThresholdWind	Integer numbers	Thresholds for wind for ROC, Reliability, Brier score

If a new variable is added to the namelist, it must be defined in subroutine **Rd_namelist.f90** and in module **Module_Definevars.f90**.

2.2 The main program: LAEF_Verification.f90

This is the main program of the verification package. It performs the following things for all the time steps of a forecast run which were set in namelist:

1. Reading the namelist through a call to subroutine **Rd_namelist**.

2. Reading the ensemble members through a call to subroutine **Rd_ensMember**.
3. Calculating the ensemble mean.
4. Reading the „truth“, verifiying analysis or observations. Up to now, program only works with analysis. Analysis is also read through a call to subroutine **Rd_ensMember**.
5. Rescaling of some fields. E.g. according to ECMWF grib coding, relative humidity is stored between 0 and 100 (in %), while other coding standards store it between 0 and 1.
6. Call the subroutines calculating the different scores only if their switch was set to 1 in namelist. Possible calls are:

   - ✔ Calc_BIAS_RMSE
   - ✔ Calc_Spread
   - ✔ Calc_Talagrand
   - ✔ Calc_ROC
   - ✔ Calc_Reliability
   - ✔ Calc_BrierScore

2.3 The routine Calc_BIAS_RMSE.f90

This subroutine performs BIAS and RMSE calculations for all ensemble members and also for the ensemble mean. Scores are saved to a file in a formatted way.

BIAS and RMSE are calculated in every grid point for every parameter at every level. At the end the average value of every field is calculated and results are saved.

Printout is done in the following way and format (example is for BIAS, the same stands for RMSE):

Palce in output file	**Description of variable**	**Format**	**Type**
1st column	name of score	A	string
2nd column	level	I5.4	integer
3rd column	parameter	I4.3	integer
4th column	timestep	I4.3	integer
5th column	BIAS of the ensemble	f12.6	real

	mean		
6th column and onwards	BIAS of the ensemble members	NEPS × f12.6	real (NEPS times)

Results are saved to a file called fort.10 which will be ranemed later by the controlling shell script.

2.4 The routine Calc_Spread.f90

This subroutine performs standard deviation calculations for the ensemble members. Scores are saved to a file in a formatted way.

Standard deviation is calculated around the ensemble mean in every grid point for every parameter at every level. At the end the average value of every field is calculated and results are saved.

Printout is done in the following way and format:

Palce in output file	Description of variable	Format	Type
1st column	name of score	A	string
2nd column	level	I5.4	integer
3rd column	parameter	I4.3	integer
4th column	timestep	I4.3	integer
5th column	Standard deviation of the ensemble (spread)	f12.6	real

Results are saved to a file called fort.11 which will be ranemed later by the controlling shell script.

2.5 The routine Calc_Talagrand.f90

This subroutine calculates the input for Talagrand diagrams and Percentage of outliers. The calculations are done in the following way:

- ✔ As a first step, ensemble members have to be sorted in an ascending order.
- ✔ The next step is to place the analysis/observation to the correct bin determined by the sorted ensemble members.
- ✔ Finally one has to count the number of analysis in the different bins and save the results.

Printout is done in the following way and format:

Palce in output file	Description of variable	Format	Type
1st column	name of score	A	string
2nd column	level	I5.4	integer
3rd column	parameter	I4.3	integer
4th column	timestep	I4.3	integer
5th column and onwards	Number of elements in the given bin	(NEPS + 1) × I10	integer (NEPS+1 times)
Last column	Sum of all events (i.e. number of grid pointts)	I10	integer

Results are saved to a file called fort.12 which will be ranemed later by the controlling shell script.

2.6 The routine Calc_ROC.f90

This subroutine calculates the input for generating ROC diagrams. The calculations are done in the following way:

- ✔ First step is the definition of probability intervals. The frequent practice is to use 10 % for probability intervals. However the number of bins should be consistent with the number of ensemble members. Here we use probability intervals the size of 1 / imem, where imem goes from 1 to NEPS.
- ✔ Second step is to calculate the probability for every variable, in every level at every grid point and for all thresholds.
- ✔ Once it is done, the scores **Hit**, **False alarm**, **Miss** and **Zero forecast** can be calculated.

Printout is done in the following way and format:

Palce in output file	Description of variable	Format	Type
1st column	name of score	A	string
2nd column	level	I5.4	integer
3rd column	parameter	I4.3	integer
4th column	timestep	I4.3	integer
5th column	threshold	I6.4	integer
6th column	probability	f8.4	real
7th column	Hit	I10	integer
8th column	False alarm	I10	integer
9th column	Miss	I10	integer
10th column	Zero forecast	I10	integer
11th column	Sum of Hit, False alarm, Miss and Zero	I10	integer

Results are saved to a file called fort.13 which will be ranemed later by the controlling shell script.

2.7 The routine Calc_Reliability.f90

This subroutine calculates the input for generating Reliability diagrams. The calculations are done in the following way:

- ✔ First step is the definition of probability intervals. The frequent practice is to use 10 % for probability intervals. However the number of bins should be consistent with the number of ensemble members. Here we use probability intervals the size of 1 / imem, where imem goes from 1 to NEPS.
- ✔ Second step is to calculate the probability for every variable, in every level at every grid point and for all thresholds.
- ✔ For every probability interval, the number of occurance has to be calculated, the number of times when the event was forecasted and occured in reality.
- ✔ Once it is done, the results can be saved.

Printout is done in the following way and format:

Palce in output file	Description of variable	Format	Type
1st column	name of score	A	string
2nd column	level	I5.4	integer
3rd column	parameter	I4.3	integer
4th column	timestep	I4.3	integer
5th column	threshold	I6.4	integer
6th column	probability	f8.4	real
7th column	Number of forecasts in the given bin	f14.6	real
8th column	Number of observations (or analysis) in the same bin	f14.6	real
9th column	Forecast frequency	f14.6	real
10th column	Observation (or analysis) frequency	f14.6	real

Results are saved to a file called fort.14 which will be ranemed later by the controlling shell script.

2.8 The routine Calc_BrierScore.f90

This subroutine calculates the Brier score. The calculations are done in the following way:

- ✔ First step is the calculation of probability for every variable, in every level at every grid point and for all thresholds.
- ✔ For every grid point it is also checked whether the event occured in reality or not. This is saved in a variable the folloing way: NbAnal = 1 if the event occured, NbAnal = 0 if not.
- ✔ Once this was all done, we can calculate the Brier score for every variable, in every level at every grid point and for all thresholds. The average value over all grid points is calculated and scores are saved.

Printout is done in the following way and format:

Palce in output file	Description of variable	Format	Type
1st column	name of score	A	string
2nd column	level	I5.4	integer
3rd column	parameter	I4.3	integer
4th column	timestep	I4.3	integer
5th column	threshold	I6.4	integer
6th column	probability	f8.4	real
7th column	Brier score	f14.6	real

Results are saved to a file called fort.15 which will be ranemed later by the controlling shell script.

2.9 The routine Rd_namelist.f90

This subroutine reads the namelist file and also prints out the parameters. If we want to add new variables to namelist, we have to define them here and also in **Module_Definevars.f90**.

2.10 The routine Rd_ensMember.f90

This subroutine reads the GRIB files, both the ensemble members and the verifying analysis (if type_data was set to GRIB in namelist).

2.11 The module Module_Definevars.f90

In this module the namelist variables are defined and the routine Rd_namelist is included. If we want to add new variables to namelist, we have to define them here and also in **Rd_namelsit.f90**.

2.12 The Makefile

For compilation a Makefile is needed. Here you have to list all the routines you want to compile. Because we are reading GRIB files, the library **emos** is also needed!

3. The verification package: the shell scripts

Some basic shell scripts have been written to control the verification process and also to visualize the scores. In the following these shell scripts will be discussed in detail.

3.1 The script RunVerification.job

This is the main script which controls the whole verification process. As a first step we have to set the dates and the increments between the two successive model runs. This is done through the following variables:

Variable	Example	description
StartDate	20060201	starting date
StartHour	18	UTC of analysis at starting date
EndDate	20060226	finnishing date
EndHour	18	UTC of analysis at finnishing date
DataInterval	24	interval between two model runs (in hours)

The next step is to set the input and output directories. This is done through the following variables:

Variable	description
AnalDir	Directory of the anals we want to use for the verification
Forecastdir	Directory of the forecasts
InputDirAnal	input data will by copied here (anal)
InputDirForecast	input data will be copied here (forecast)
SourceDir	Directory of the FORTRAN routines and shell scripts
SaveDir	scores will be saved here

We also need a tool to loop on dates, this is performed by using the C program **dateAddHour**. Once we have set everything, starts the loop on the dates between **StartDate** and **EndDate**. In this loop we perform the following things:

- ✔ Copy the appropriate files to **InputDirAnal** and **InputDirForecast**.
- ✔ Run the verification program **LAEF_Verification.exe**.
- ✔ Save the scores to **SaveDir**.

The following files will be saved to **SaveDir**:

- ✔ BIAS_RMSE_yyyymmddrhh.dat
- ✔ SPREAD_yyyymmddrhh.dat
- ✔ TALAGRAND_yyyymmddrhh.dat
- ✔ CONTINGENCY_yyyymmddrhh.dat
- ✔ RELIABILITY_yyyymmddrhh.dat
- ✔ BRIERSCORE_yyyymmddrhh.dat

Where yyyymmdd stands for the date of the forecast run and hh stands for the hour of the analysis. Doing so we will have one file / forecast run / score.

3.2 The script RunPlotScores.job

The general framework of this script is very similiar to the one of **RunVerification.job**. As a first step we have to set the dates in the same manner. Then comes the setting of the directories. This is done through the following variables:

Variable	**description**
SourceDir	Directory of the FORTRAN routines and shell scripts
ScoreDir	Input directory, the scores of the individual model runs will be taken from here
PlotDir	Plots will be saved here

We also need a tool to loop on dates, this is performed by using the C program **dateAddHour**. Once we have set everything, we perform the following things:

- ✔ Calculate the average values for the time period between **StartDate** and **EndDate** (using **awk**) and save these to temporary data files which will be used for visualization and will be deleted at the end of the script.
- ✔ Plot the different scores using simple **gnuplot** scripts:

    - ✔ Plot_BIAS_RMSE_SPREAD.scr
    - ✔ Plot_Outliers.scr
    - ✔ Plot_BrierScore.scr
    - ✔ Plot_Talagrand.scr

At the end of the script the temporary data files used for plotting are deleted.

4. Problems to be solved

There are several things which need to be considered and refined in the source code and also in the shell scripts.

- ✗ One such a thing is the question of wind speed. Wind speed is not stored directly in the GRIB files, instead we have u and v components. For the sake of simplicity I would suggest to compute wind speed before everything else (as a preprocessing) and write it to the GRIB files. This would make the verification much easier and would keep the code more simple. The problem with this solution is that wind speed does not have a grib code number, so we should agree on one.
- ✗ It can happen that we do not want to verify every variable on every level. We should tell it to the program somehow through namelist, otherwise we have to write into the code (as it is done at the moment).
- ✗ The question of thresholds in case of ROC, Reliability and Brier score. Again, we may not want to define thresholds for every parameter. It is not obvious how to define the thresholds. For wind speed it is easy, we can define e.g. 5, 10, 15, 20 m/s or any other values. But for temperature it is not so straightforward. The threshold 0 Celsius does not have the same meaning in the Alps as in Greece. Therefore it is desirable to use values like „5 Celsius higher than average“. Doing so, we will need to create climate data, e.g. from ERA40. (At the moment thresholds are only defined for wind speed.)
- ✗ Problem of plotting. In default of time, scripts to plot ROC and Reliability could not be written. This should be done in the near future. Existing plotting programs should be refined aswell.
- ✗ Also, interface to use synop data should be written. The subroutines calculating the different scores may not be changed too much.

5. Some example of plots

The plots were made using the above described verification package. As „truth“ ARPEGE analysis was used. Due to some technical problems we were not able to retrieve the data from ECMWF's MARS database. At ECMWF they promised they would try to solve the problem soon. Verification was performed with a time step of 6 hours from +00 to +48.

The verified experiment is breeding performed with the ALADIN model for the time period 01 February 2006 – 26 February 2006.

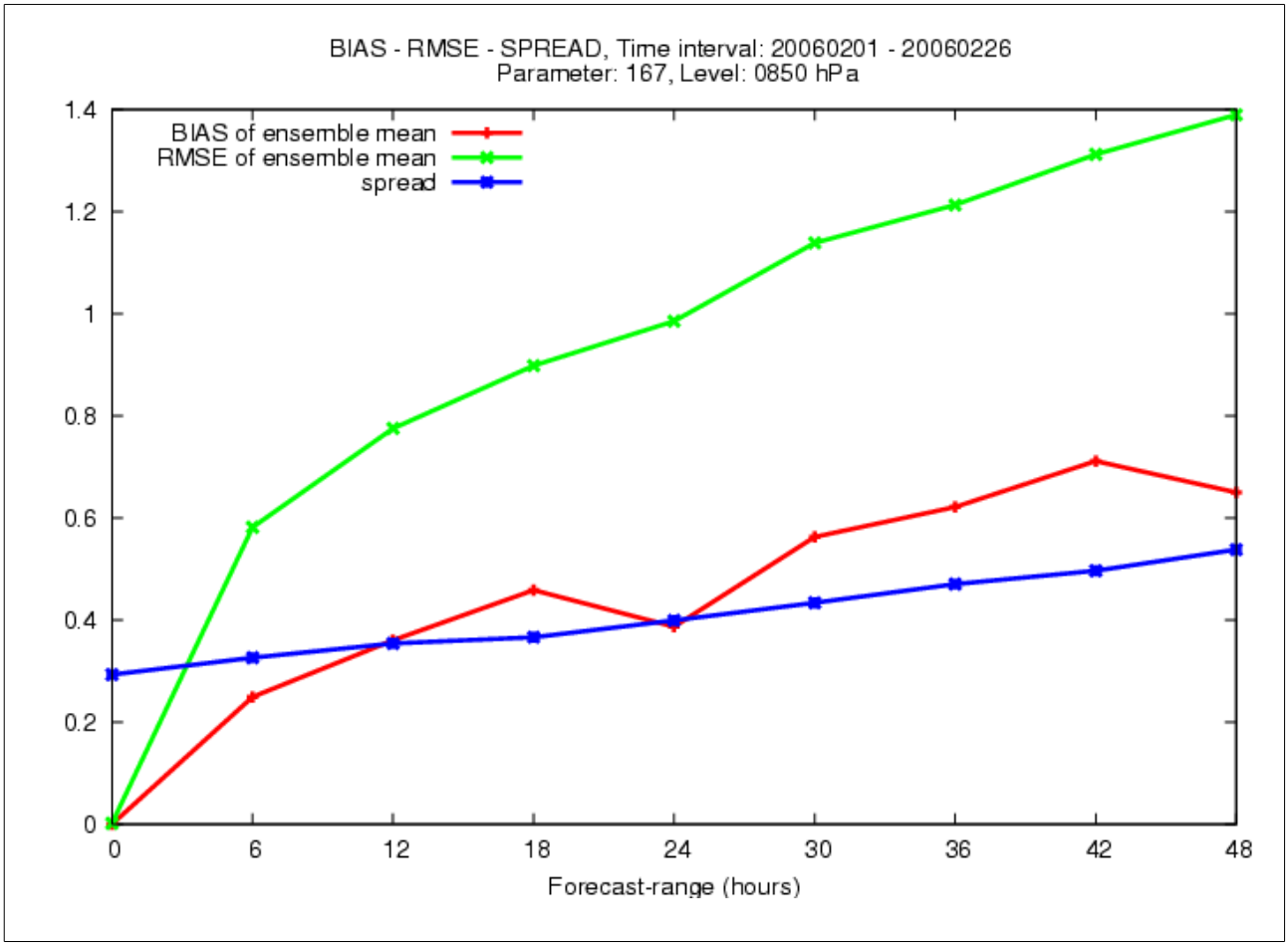

***Figure 1**: The plot of BIAS (red curve) and RMSE (green curve) of the ensemble mean for 850 hPa temperature. Standard deviation of the ensemble members (spread, blue curve) is also plotted.*

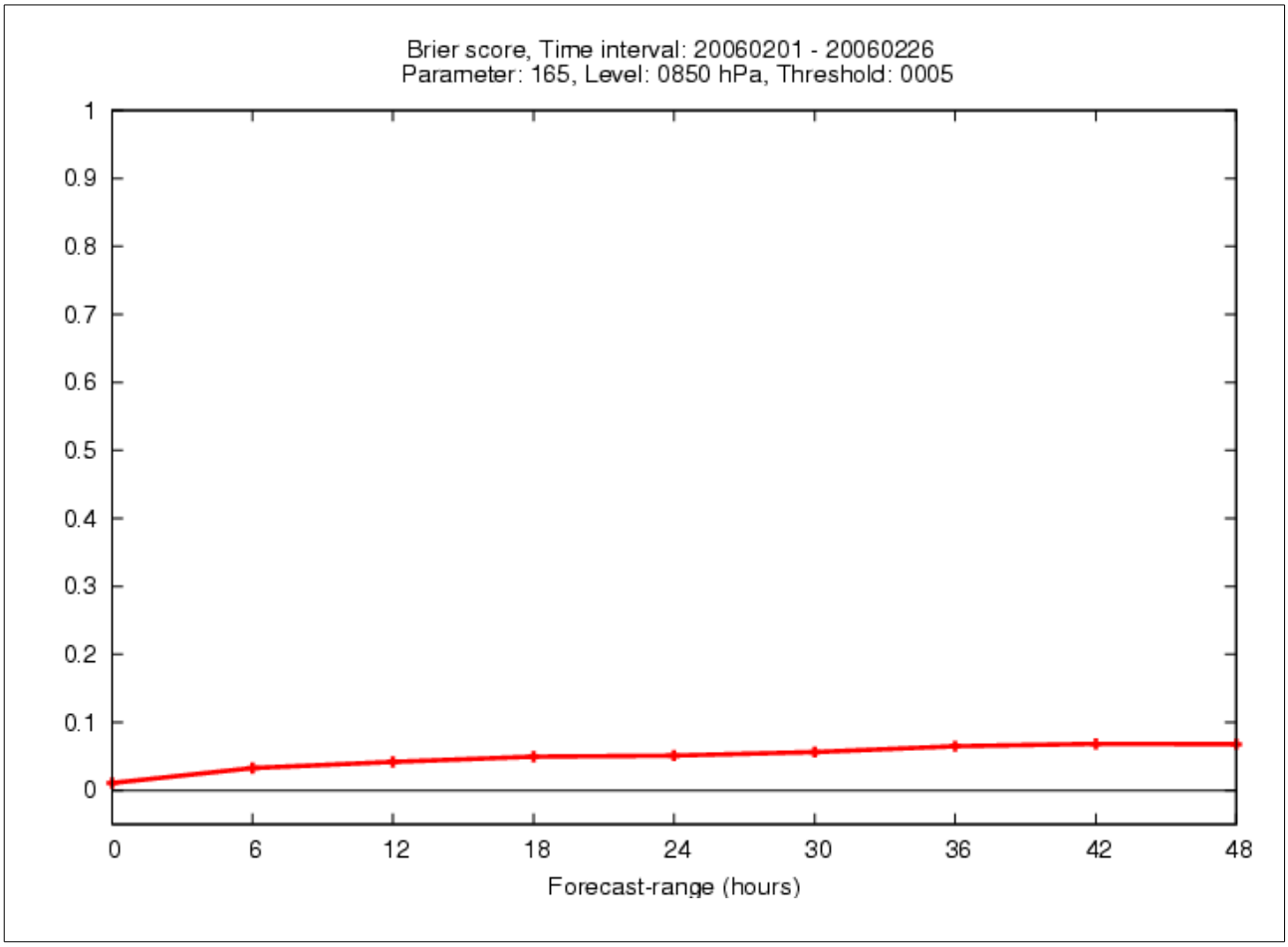

***Figure 2**: Brier score of the u component of the 850 hPa wind. (The horizontal black line is representing the zero Brier score, which means perfect forecast.)*

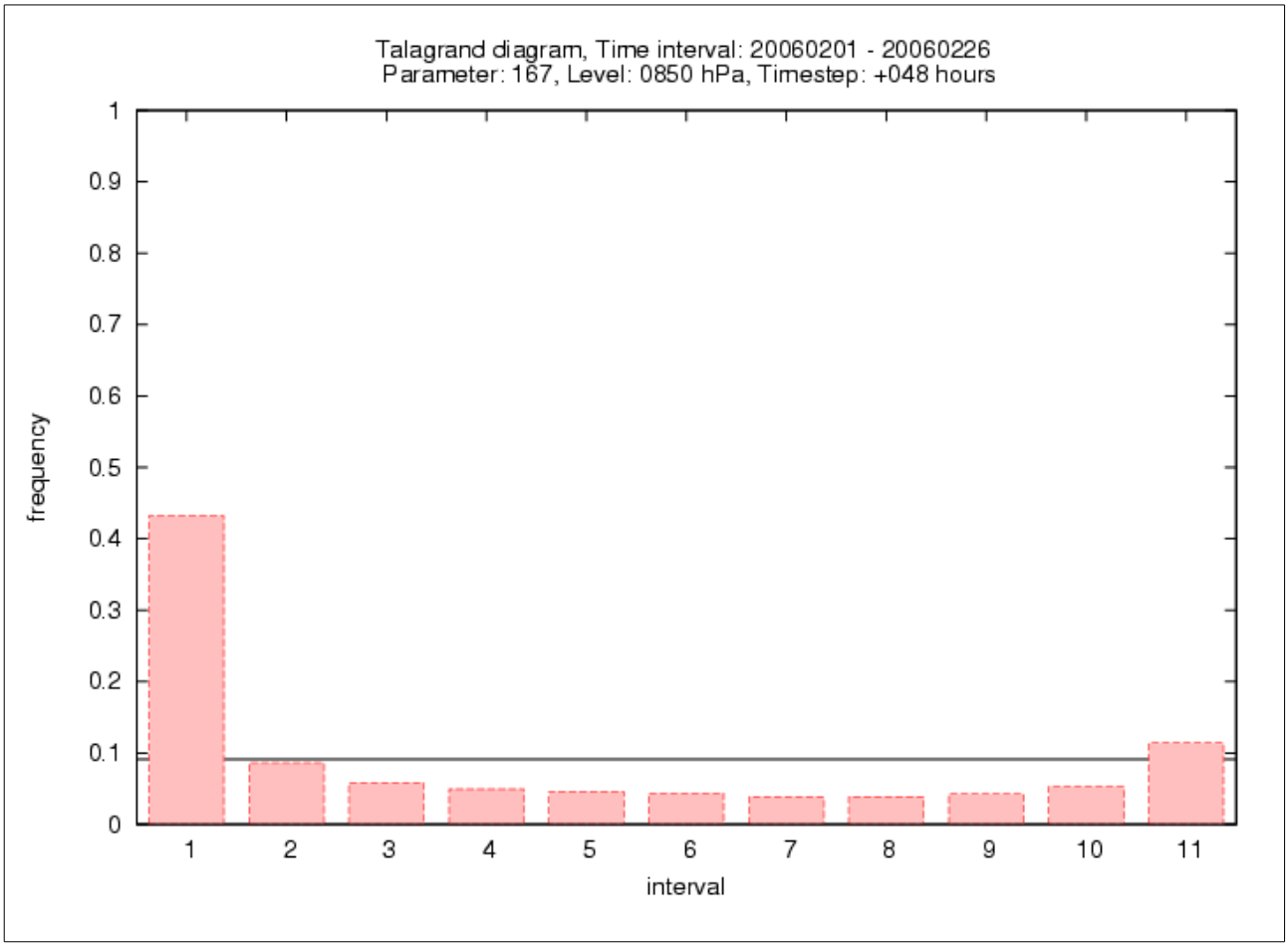

***Figure 3**: Talagrand diagram of 850 hPa temperature for forecast-range +48 hours. (The horizontal black line is representing the expected value, 1 / (number of ensemble members+1).)*

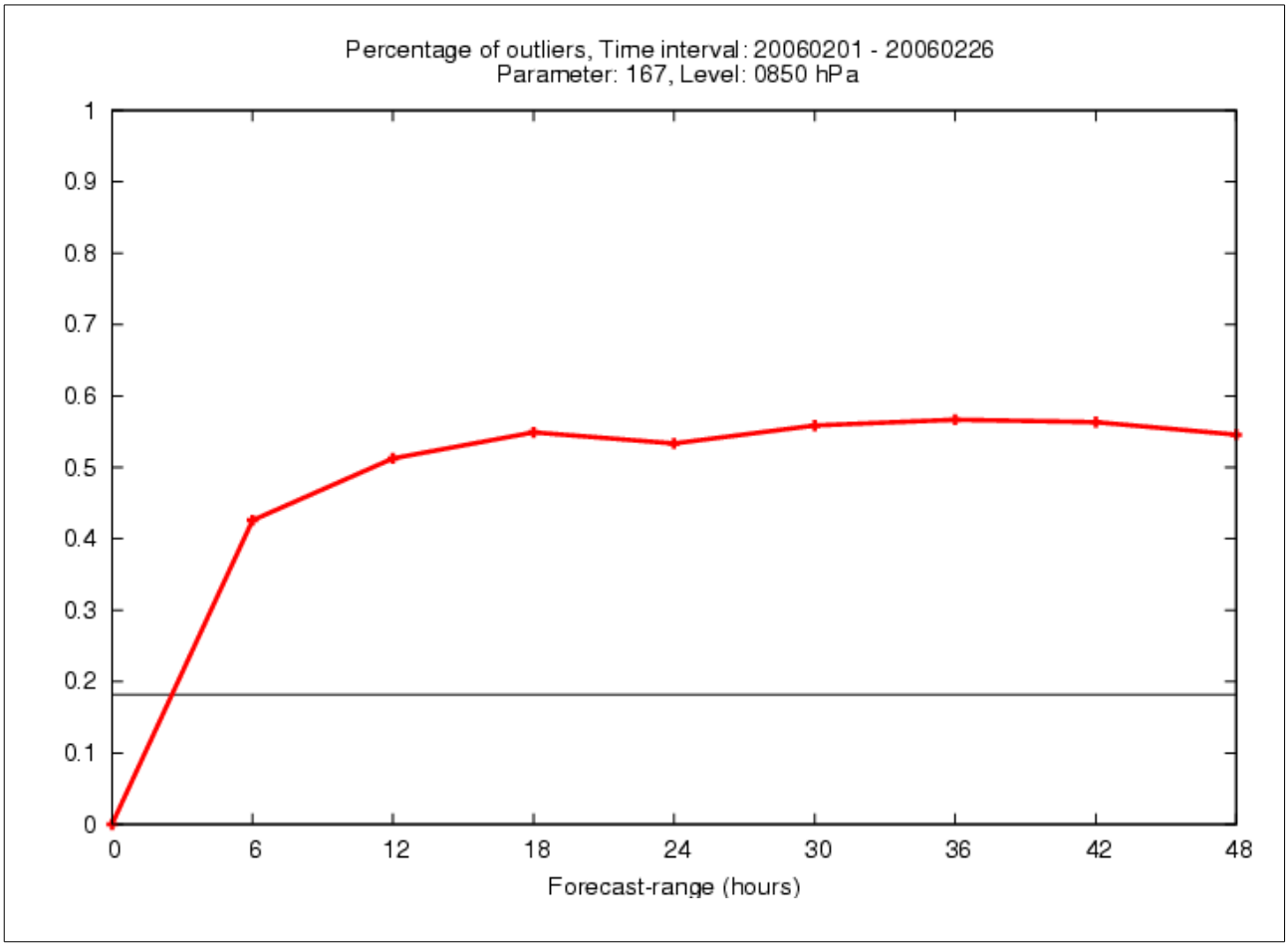

***Figure 4**: Percantage of outliers for 850 hPa temperature. (The horizontal black line is representing the expected value 2 / (number of ensemble members+1).)*