# Stay Report

To: Zentralanstalt für Meteorologie und Geodynamik — ZAMG, Vienna, Austria

Period: 2nd February – 3rd March, 2018

Topic: Work on analog-based post-processing method

Supervisors: Mag. Alexander Kann and Irene Schicker, PhD

***Introduction***

I stayed at the Zentralanstalt für Meteorologie und Geodynamik (ZAMG) for four weeks during which I was working on the point-based analog-based post-processing method applied to a NWP model output. Main aim of the stay at ZAMG was to investigate the usability of the analogs method, a more statistics-based methodology of creating ensemble forecasts, with different datasets available at ZAMG.

Therefore, several steps were needed:

- rewrite the original codes in Python
- make the algorithms computationally efficient
- test the method using at least one deterministic model

The analog-based method uses historical data within the specified analog training period for which both the deterministic NWP (starting model) and the verifying observation are available. The analog-based method uses one consistent grid-point. The datasets (deterministic NWP) used are AROME (1/1/2015-31/08/2017) and corresponding OBS (1/1/2015-31/10/2017). The best-matching historical forecasts to the current prediction (analogs) may originate in any past date within the training period. The quality of the analog (the ''difference'') is evaluated by the pre-defined metric (more details in the previous stay report). Analogs are found independently for every forecast time and location, narrowing the search around particular time of a day by a time window. The verifying observations of the best-matching analogs are the members of the analog ensemble (AnEn). The $AN_t$ forecast is a (weighted) mean of $N$-sized AnEn for a (future) time t.

### *Scripts and optimization*

Building the database for the training period is computationally the most demanding part of the entire analog-based scheme. Due to the lack of RAM available, I was not able to load the entire database during my previous stay. During this stay I was able to modify and optimize pre-existing script `IOP-loading-data.py`. The database can be loaded directly from the (historical) model outputs (for all the locations), but not for every analog forecast. For instance, it can be updated once every month, once every few months or even once a year. By using modules `load_AROME` and `load_OBS` from `IOP.py` the data is loaded from original files. The variable names are:

- `statnr` – unique station number
- `idate` – initialization date
- `itime` – initialization time
- `fhour` – forecast hour (lead time)
- `rrr` - precipitation
- `tl` – 2-m temperature
- `ff` – wind speed
- `dd` – wind direction (deg)
- `rf` – relative humidity
- `pred` – red. pressure
- `ff_obs` – wind speed observations

The observations are joined to corresponding forecasts. The data are modified as needed (measurement units adjusted, missing data replaced with NaN etc.), then saved in the database called `MyData2015_6.db`. So, this database contains all the 2015-2016 data for the analog training. This is the sql database, created by using `sqlite3` module.

Additionally, in the `IOP-loading-data.py` script, the stations (not) missing more than 50% of data are listed. Then, the list of the stations containing more than 50% data is copied in the separate module (`included_stations` in `IOP.py`). This way the method will save some computational power by only dealing with the stations that have more than a year of training (267 stations).

I was developing the basic scripts for analog method application (the forecasting of the mean of the ensemble: AN forecasting) during my previous stay. The algorithms were written so that they would use pre-prepared database and “current” AROME NWP. This is done in the `IOP-analogs.py` script. This scripts than uses the `anen` module from `IOP.py`. Due to the lack of the computational power, I was not able to provide any further optimization or meaningful testing during my last visit. During this stay I tested algorithms and concluded that they are too slow. For instance, three function calls lasted for almost 500 seconds (Figure 1). That means one forecast at 3 locations needed 8 minutes to finish. Obviously, the algorithms needed further optimizations such as reducing the number of global variables, avoiding loops, using the matrix instead of dataframes as much as possible, etc. This was done and the execution time was reduced from 157 to 3 seconds per call (1 forecast for 1 station, up to +48 h forecast time). In other words, one analog-based forecast (up to 48 hours lead time) for 267 stations can now be executed in less than 14 minutes instead in more than 11 hours before the optimization process. Of course, the algorithm can and will be optimized even further in the future.

```
a)      178617672 function calls (177588042 primitive calls) in 480.410 seconds

 Ordered by: cumulative time

 ncalls  tottime  percall  cumtime  percall filename:lineno(function)
      1    0.018    0.018  480.426  480.426 IOP-analogs.py:1(<module>)
      3    2.106    0.702  469.966  156.655 IOP.py:60(anen)
  65997    1.629    0.000  238.317    0.004 ops.py:827(wrapper)
  65997    0.293    0.000  220.104    0.003 ops.py:773(na_op)
  65700    0.256    0.000  218.695    0.003 ops.py:751(_comp_method_OBJECT_ARRAY)
  65700  218.104    0.003  218.104    0.003 {pandas._libs.lib.scalar_compare}
  19860    0.256    0.000  174.560    0.009 indexing.py:187(__setitem__)
```

```
b)      50081692 function calls (49383865 primitive calls) in 111.510 seconds

 Ordered by: cumulative time

  ncalls  tottime  percall  cumtime  percall filename:lineno(function)
       1    0.013    0.013  111.525  111.525 IOP-analogs.py:1(<module>)
       3    1.480    0.493  105.837   35.279 IOP2.py:60(anen)
79374/68271    0.542    0.000   29.689    0.000 indexing.py:1358(__getitem__)
63819/54906    0.189    0.000   28.092    0.001 indexing.py:856(_getitem_tuple)
63897/54984    0.783    0.000   27.147    0.000 indexing.py:963(_getitem_lowerdim)
    6867    0.182    0.000   25.549    0.004 ops.py:827(wrapper)
   19860    0.222    0.000   24.297    0.001 indexing.py:187(__setitem__)
   61623    0.380    0.000   23.844    0.000 indexing.py:1572(_getitem_axis)
    6867    0.035    0.000   23.541    0.003 ops.py:773(na_op)
```

```
--- 291.634227037 seconds ---
c)      157541493 function calls (153223194 primitive calls) in 291.617 seconds

 Ordered by: cumulative time

 ncalls  tottime  percall  cumtime  percall filename:lineno(function)
      1    0.012    0.012  291.635  291.635 IOP-analogs.py:1(<module>)
     62    0.007    0.000  193.013    3.113 IOP2.py:60(anen)
    124    0.137    0.001  191.821    1.547 groupby.py:656(apply)
    124    0.007    0.000  191.664    1.546 groupby.py:723(_python_apply_general)
    124    0.168    0.001  176.621    1.424 groupby.py:1854(apply)
    124    0.002    0.000  154.144    1.243 groupby.py:4600(fast_apply)
```

Figure 1. The time needed for analog-based forecast execution before, during and after optimization process: a) 157 seconds, b) 35 seconds and c) 3 seconds per station per forecast (up to 48 h lead time).

At this point the `IOP-analogs.py` script loads the current AROME NWP and data from previously built database. Even though the testing is performed for two testing months (62 forecasts), the ''current'' AROME NWP is loaded one by one – still left in a loop. The reason is that this way the algorithm can easily be adjusted to work operationally.

Everything besides loading from training sql database and writing forecasts to another database (`Testing_Jan_2017.db` or `Testing_Jul_2017.db`) is done in the `anen` module from `IOP.py`. The module seeks for 20 most similar analogs, sorted by difference (similarity). This number can be changed. However, in my previous experience I reached a conclusion that approximately 15 members is optimal for forecasting a common event, or even smaller ensemble if forecasting a rare event. Since the members are sorted, it is possible to choose an appropriate number of members up to 20, while there is probably no need to use more anyway. As shown in previous works (1, 2), it is enough to use 3-time-steps time window (1 time step before/after lead time for which analog-based forecast is made). In other words, widening the time frame does not lead to further improvement of results. For this reason, the width is fixed and not used as a variable in this module. All the available predictor variables are used (5 predictors: `rrr, tl, ff, dd, rf, pred`). The difference metrics between different predictor variables needs to be comparable, so the weights for each predictor variable are set as 1 divided by standard deviation (normalization). Since the wind direction is a circular variable, the standard deviation is calculated differently (i.e. the standard deviation uses difference between two angles calculated as: `180-abs(abs(a1-a2)-180)`). The difference metrics is calculated by applying `mymetrics` function to `groupby` object (grouped by initialization date – one by one forecast). To save some memory used for calculation the difference metrics is stored at `training.loc[:,'tl']`. Then, all is left is to sort the observations (`ff_obs`) corresponding to the training forecasts by the difference metrics and choose the 20 closest ones. The analog members are numbered `ff1` – `ff20` and saved in dataframe, together with ALARO `ff` forecast.

The example for this analog-based forecast is plotted by using `IOP-plotting-example.py` script and shown at Figure 2. The exact station and date is chosen and the corresponding forecasts and observations loaded from database. The 10 analog ensemble members are used and the spread of the ensemble is shown via boxplots (with outliers – circles). The red line represents the forecasting the ensemble mean (AN forecast). The results are compared to corresponding observations (green line).

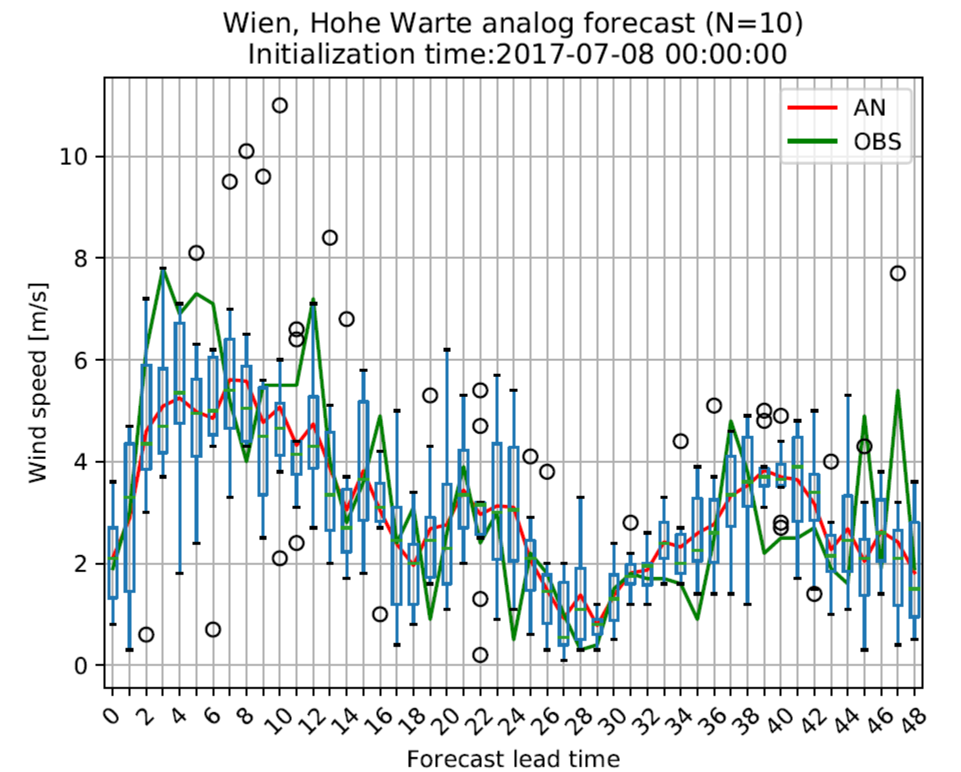


Figure 2. The example of the analog-based forecast for Hohe Warte station initiated at 2017/07/08 (up to 48-h forecast lead time). The ensemble is consisted of 10 members. The spread of the ensemble is represented by boxplots, where circles represent the outliers. The red line represents AN – forecasting the mean of the ensemble. The results are compared to observations (green line).

***Results***

We agreed to test and verify the algorithm for the two months selected – one winter and one summer month. We chose January and June 2017. The analog method used the same training period (2015-2016) and the same setup for both. The AN forecast is the mean of 15 analog ensemble members. Since the analog and AROME forecasts were already saved in a database, all that was left to do was to join them together with corresponding verifying observations. This was done in `IOP-preplot-merge.py` script. The mean of the analog ensemble was calculated and called `an`. All the data needed for the verification procedure was saved to `Results_Jan_2017.db` and `Results_Jul_2017.db` databases.

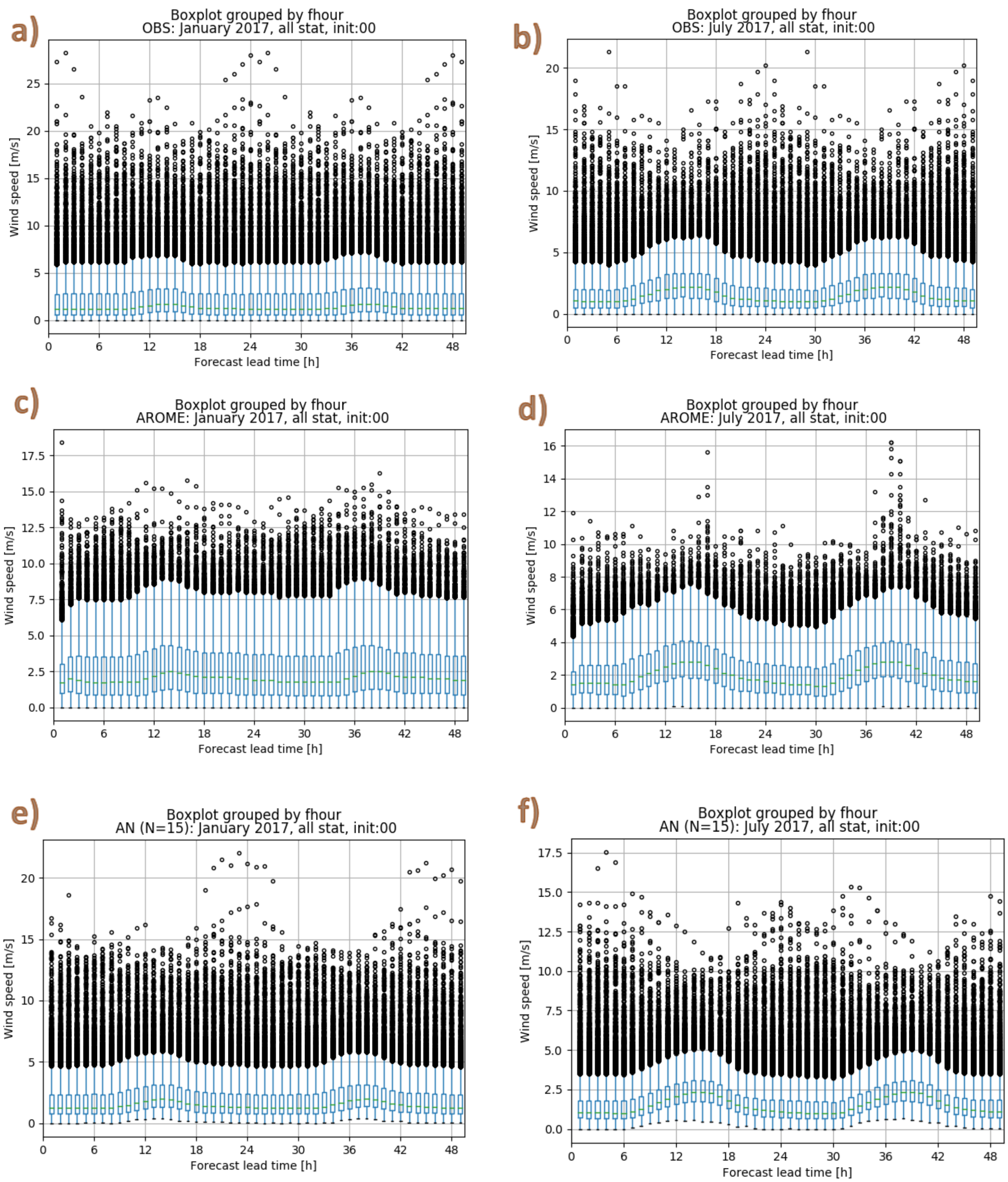


Figure 3. Boxplots of the wind speed observations for January (a) and July (b) 2017, in comparison to the AROME forecasts (c, d) and the AN forecast (e, f).

The verification procedure consisted of two separate parts – time (script: `IOP-plotting-verification.py`) and spatial (script: `IOP-plotting-verification_maps.py`) analysis. Besides the winter-summer comparison, the analog AN forecast is compared with the AROME forecast as well. The distribution for the wind speed observations and the forecasts was described by histograms and boxplots.

It seems that (observed) wind speed, as well as its diurnal cycle is stronger in July than in January (Figure 3). Both AROME and AN forecast exhibit the same distributional features. However, the wind speed median and the interquartile range (IQR) seem to be overestimated by the AROME forecasts. The AN forecasting shows somewhat smaller value for wind speed median and smaller IQR, which seems closer to observations.

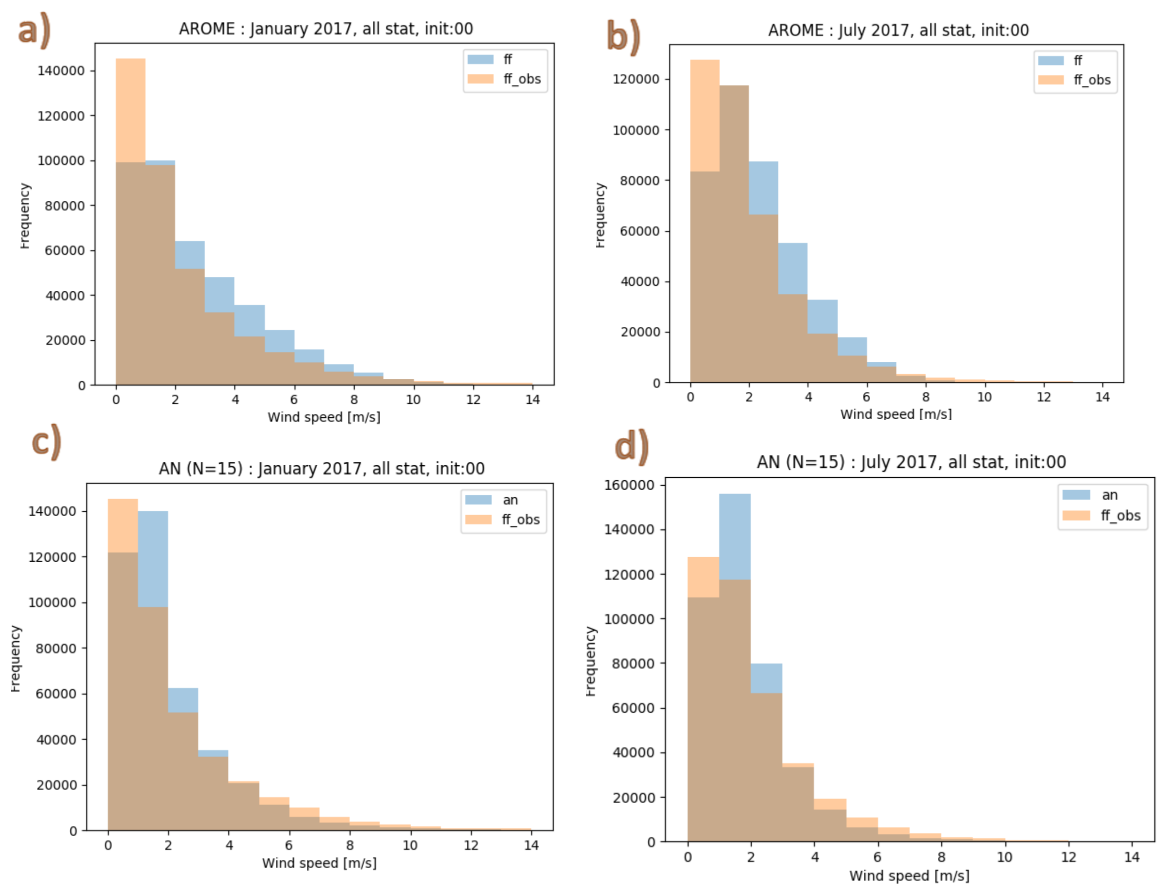


Figure 4. Histograms of the AROME (upper) and the AN (down) wind speed forecasts for January (left) and July (right) 2017, in comparison to the observations.

The AROME forecast seems to under-predict low and high wind speeds, while over-predicting medium range (approximately 2-8 m/s) for both January and July (Figure 4). The AN forecasts exhibits distribution more similar to observations for medium range. However, the over-forecasting wind speeds 1-2 m/s and under-forecasting high wind speeds (i.e. higher than 5 m/s) can be spotted as well.

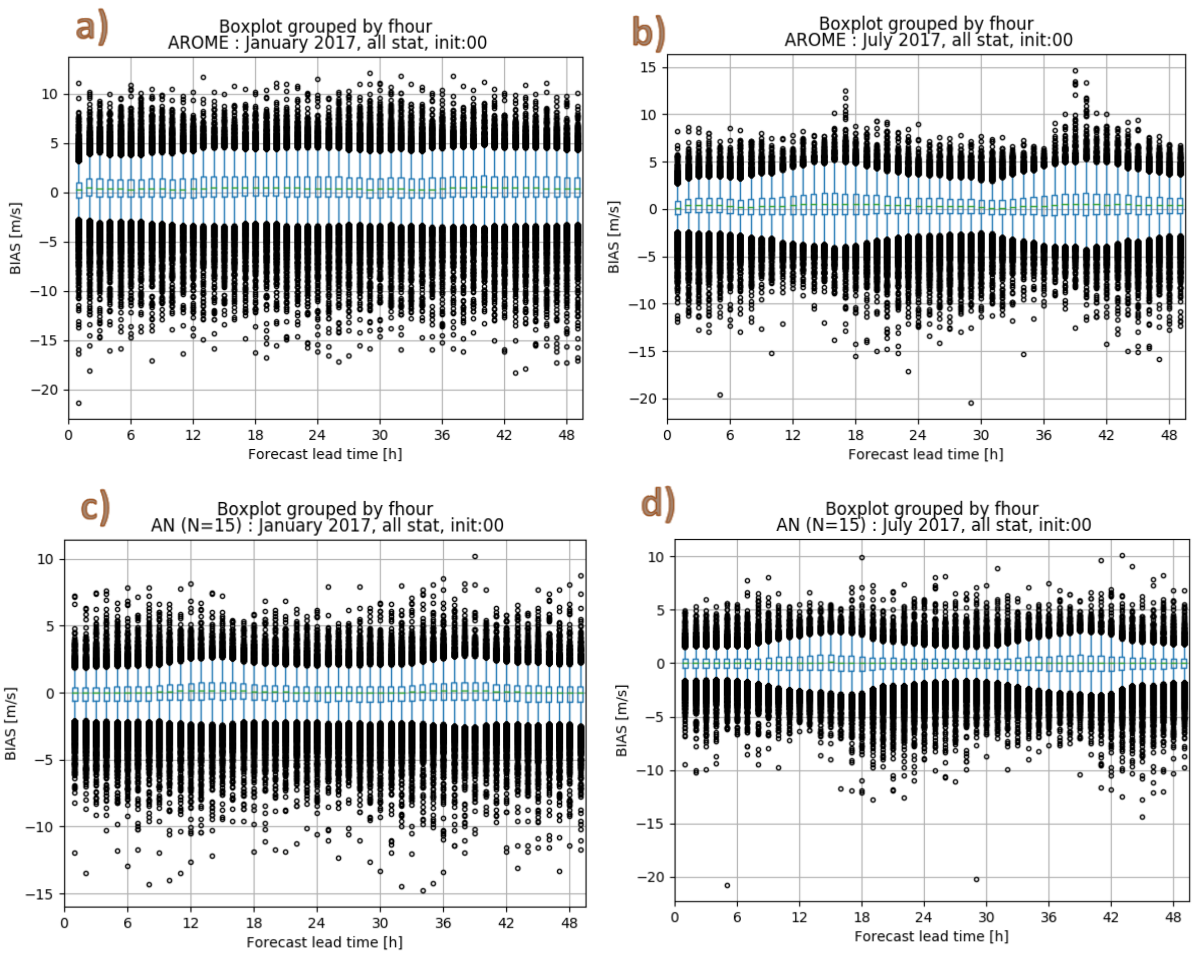


Figure 5. Boxplot of the bias: the AROME (upper) and the AN (down) wind speed forecasts for January (left) and July (right) 2017.

The bias is smaller in July than in January for both AROME and AN forecasts (Figure 5). Both forecasts seem to have a bit more negative outliers (under-prediction of wind speed). Some diurnal variations can be spotted, especially in July. The bias seems the highest in the summer afternoon for both forecasts. It is smaller for AN than for AROME: the IQR is smaller and the outliers seem closer to zero value.

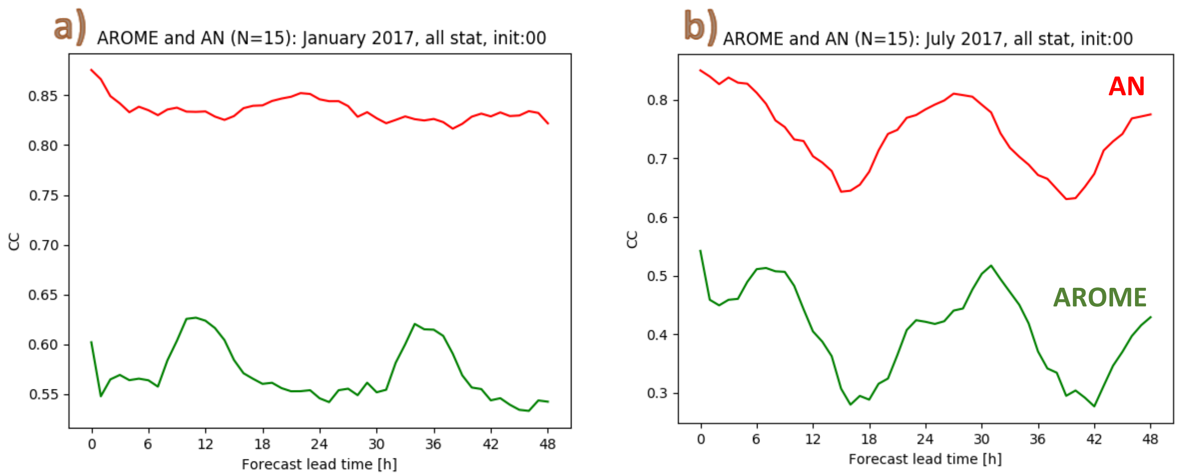


Figure 6. The correlation coefficient for the AROME (green line) and the AN (red line) wind speed forecasts for January (left) and July (right) 2017.

If the AN CC results are compared with the AROME results, it can be seen that there is a great improvement achieved via post-processing (Figure 6). The values are somewhat higher for January for both forecast, and the daily cycle of CC is less evident. The daily CC cycle is almost non-existent for the AN in January. As expected, the values decrease with forecast lead time for all the forecast tested.

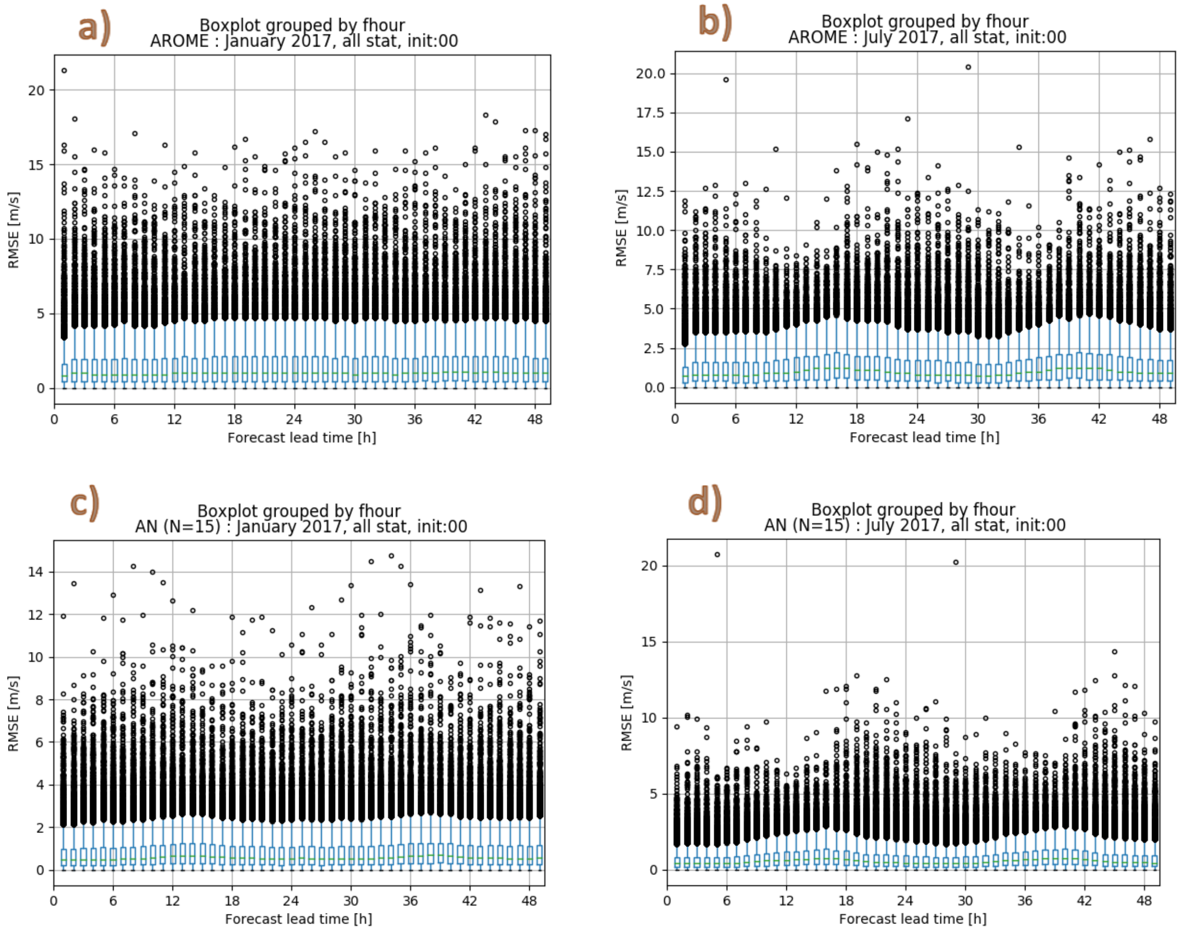


Figure 7. Boxplot of the RMSE: the AROME (upper) and the AN (down) wind speed forecasts for January (left) and July (right) 2017.

The forecast error is generally higher in January than in July, for both forecasts (Figure 7). In the winter month the root-mean-square-error (RMSE) is quite uniformly distributed (no diurnal cycle). In the summer some diurnal cycle of the error can be noticed, similarly to bias results. However, the RMSE increase with lead time is almost unnoticeable. The AN RMSE is lower than AROME RMSE, with smaller range (i.e. there is no outliers larger than 15 m/s in the January).

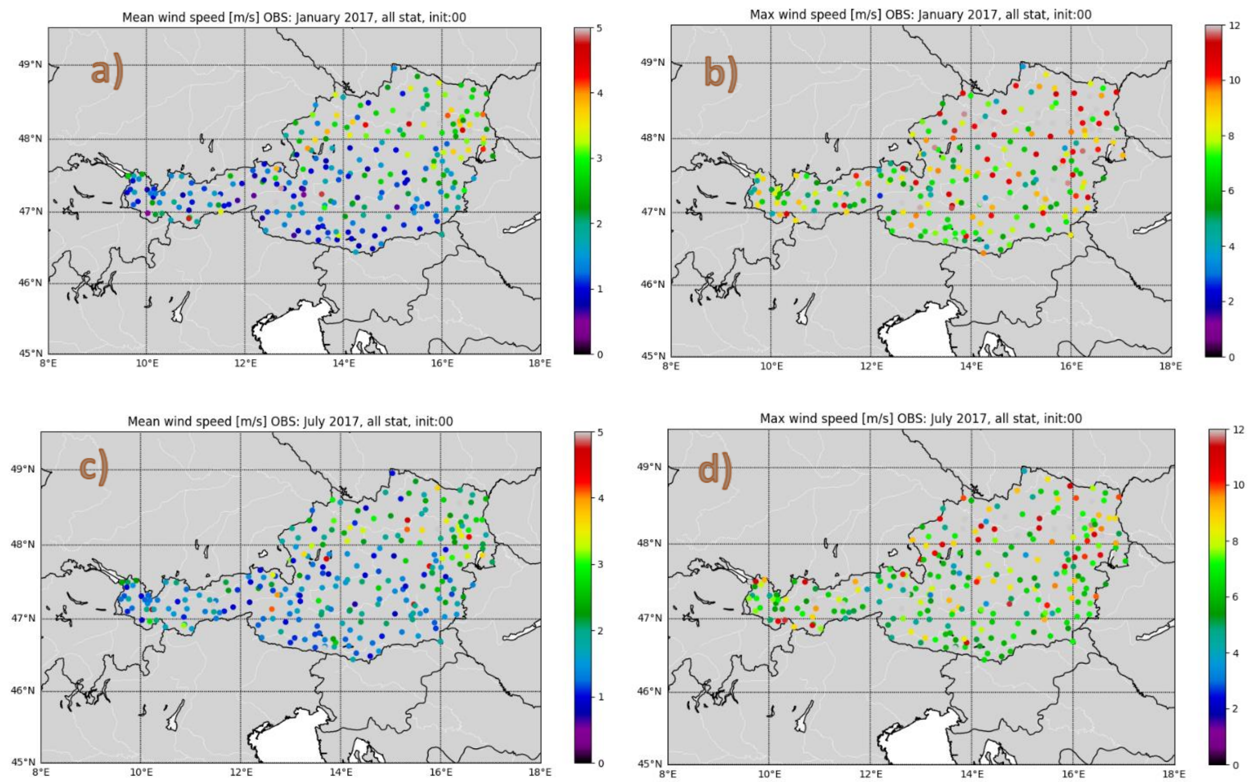


Figure 8. The spatial distribution of the monthly mean (a, c) and maximum (b, d) of the observed wind speed in the January (upper) and July (down), 2017.

The wind speed was moderate for both January and July at majority of the stations (Figure 8). The mean and the maximum monthly wind speed increases towards north-eastern part (Pannonian plate) for both January and July. The values seem to be slightly higher in January than July.

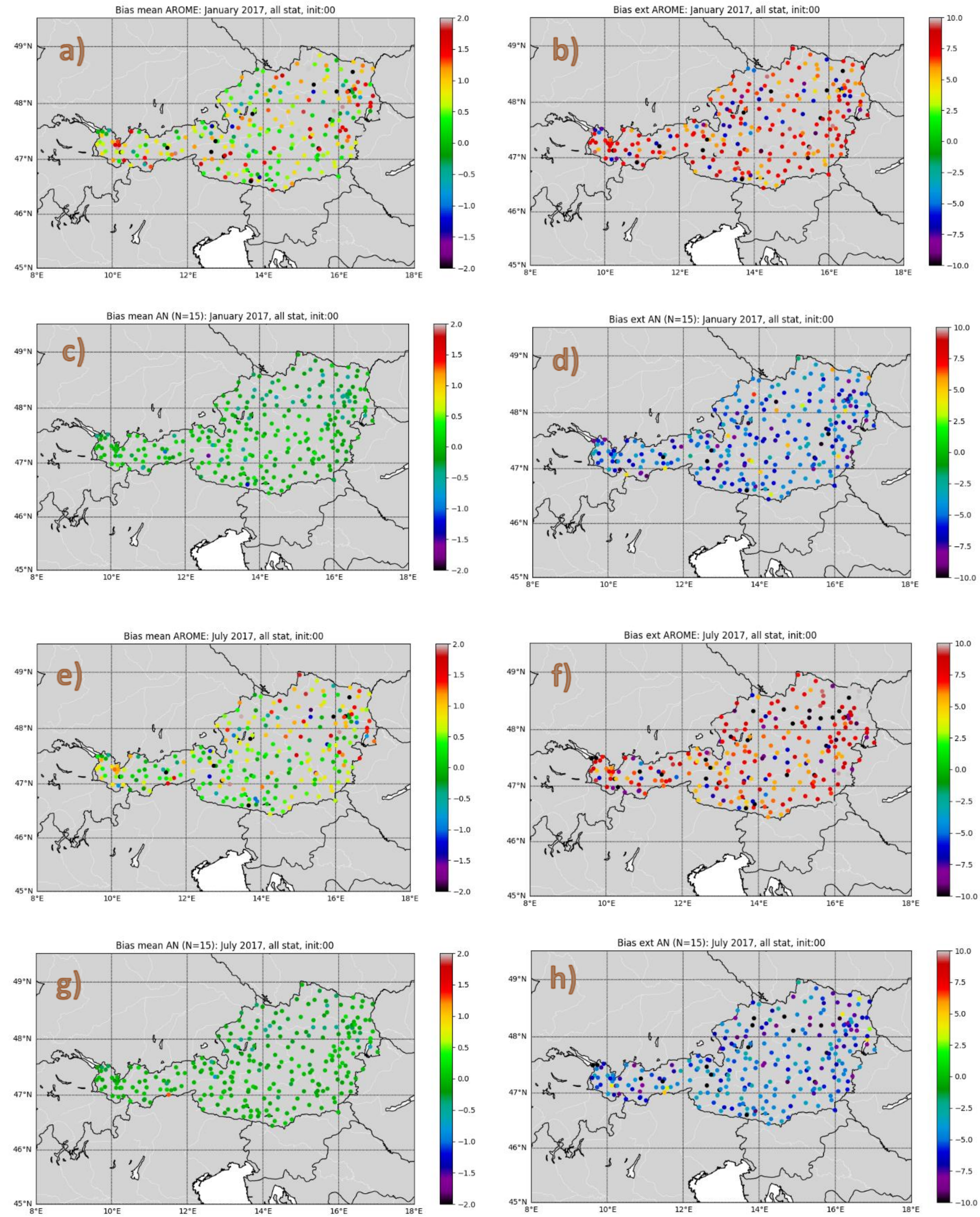


Figure 9. The spatial distribution of the monthly mean (left) and the most extreme bias (right) for AROME (a, b, e, f) and AN (c, d, g, h) forecast in the January (a-d) and July (e-h), 2017.

For the AROME forecast, the bias seems to be slightly positive on average at majority of the stations, especially in both January. (Figure 9). In both winter and summer months, there is a positive bias in the northeast area (Pannonian plain) and near Warth for the AROME forecast. The extreme bias value are positive at the majority of locations, especially during January. The extreme bias values seem to be more randomly distributed around zero value for summer month, even though the Pannonian plain still seems to be positively biased. Even though the negative extreme bias values for the AROME are present at less locations, they are more biased (absolute value is i.e. below - 15 m/s, compared to maximum 10 m/s for positive bias – not shown). The AN mean bias is smaller than for the AROME forecast. The AN at the locations at Pannonian plain are slightly negatively biased on average, while at the Alps the bias is positive in January. The AN forecast in July is slightly negatively biased on average. Also, the extreme bias seems to be negative on the most locations, during both January and June. There are only a few locations exhibiting positive extreme bias for AN forecast and they can be considered randomly distributed.

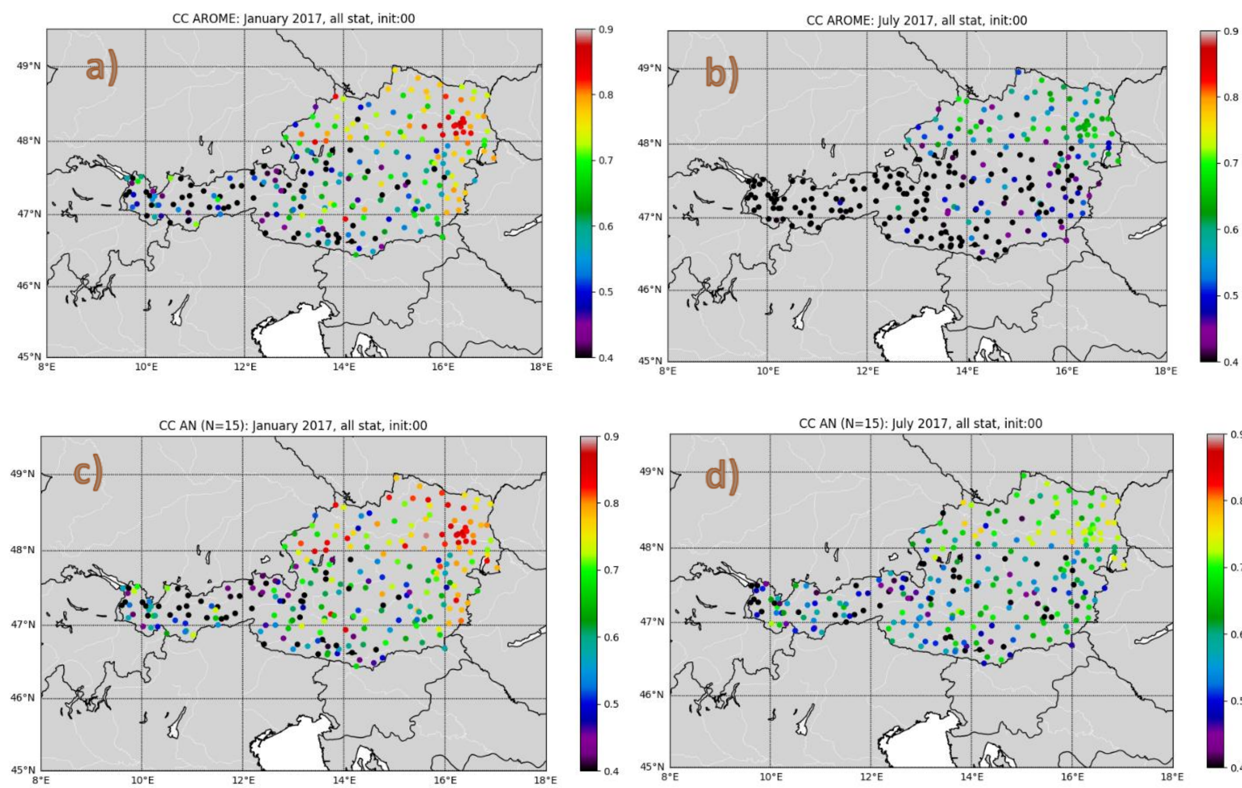


Figure 10. The spatial distribution of the monthly mean correlation coefficient for AROME (a, b) and AN (c, d) forecast in the January (left) and July (right), 2017.

The correlation coefficient (CC) seems to reduce its value from northeast area towards west and south-west of Austria (Figure 10). Also, the values are higher for the January than the July. This is regardless of the exact forecast and time of a year. Therefore, it could probably be concluded that the wind speed is less predictable towards west and during winters. Both forecasts have very low values in the Alps. The CC values as low as shown can suggest very unpredictable month, but also a potential error made in forecasting, loading the data or analysis. However, there is an evident improvement achieved with post-processing for January and especially July.

The RMSE values seem to be slightly higher during January than in July for both forecasts (Figure 11). The values for the monthly mean and maximum RMSE are higher for the AROME than for the AN forecasts in both cases. The error is seem to be larger in the Panonian plane and in the western part of Austria for the AROME forecast (especially for July), while there is a here is no obvious spatial distribution of error for the AN forecasts.

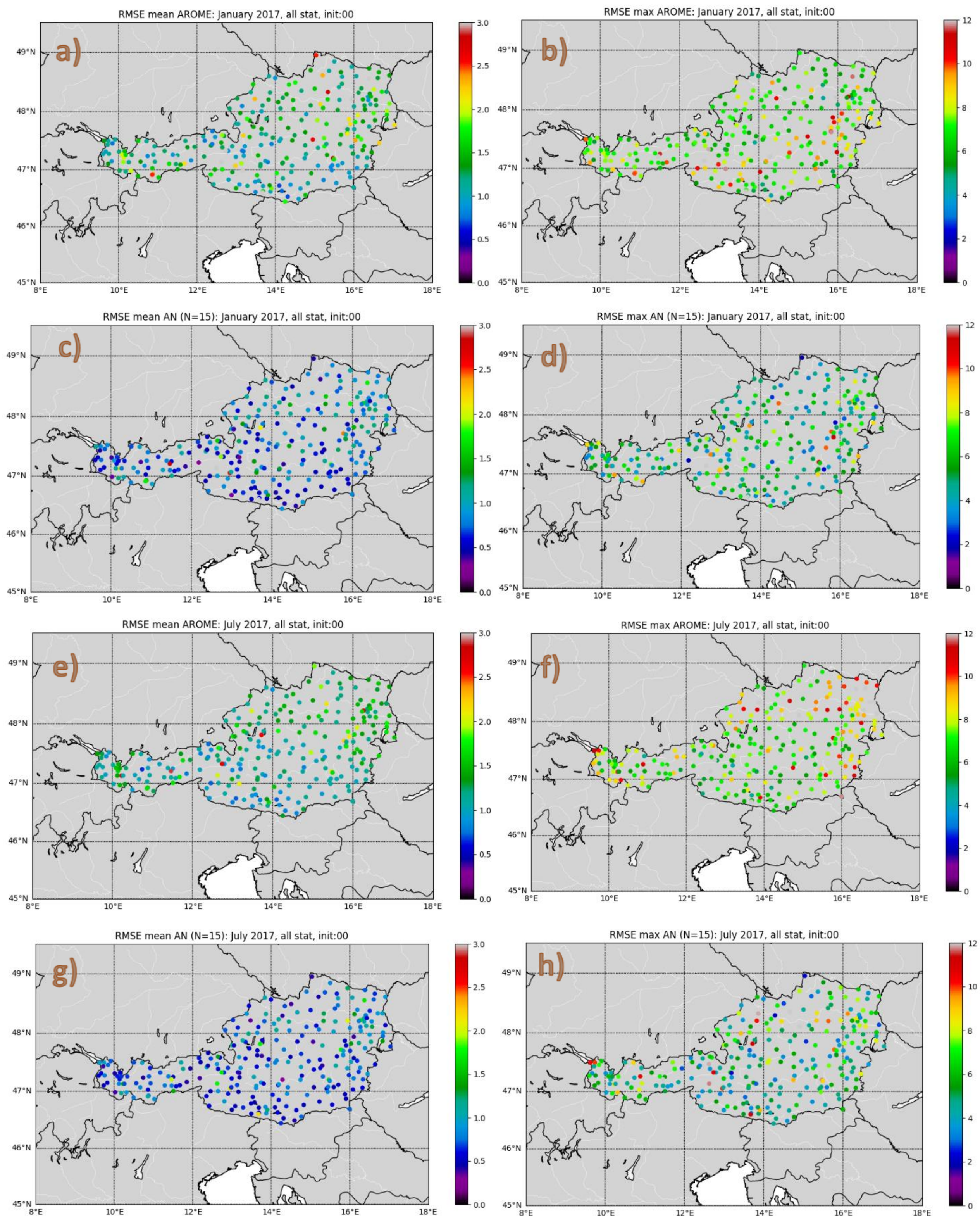


Figure 11. The spatial distribution of the monthly mean (left) and the most extreme RMSE (right) for AROME (a, b, e, f) and AN (c, d, g, h) forecast in the January (a-d) and July (e-h), 2017.

### *Conclusion and future plans*

During this stay it is shown that the analog-based approach compared to the AROME forecasts has:

- The distribution closer to the observed distribution
- Smaller bias
- Higher correlation coefficient to the observations
- Lower error (measured by root-mean-square-error).

These very satisfactory results suggest that this methodology is applicable to Austrian data, encouraging the continuation of this work. Next steps should be:

- Develop and test the probabilistic output for the AROME data
- Use the ECMWF model for both deterministic and probabilistic approach
- Use at least two deterministic models as input, "poor-man ensemble"
- Use the LAEF model as input and compare the results

### *References*

[1] Delle Monache, L., Nipen, T., Liu, Y., Roux, G., Stull, R., 2011: Kalman filter and analog schemes to post-process numerical weather predictions. Monthly Weather Review 139, 3554-3570.

[2] Delle Monache, L., Eckel, T., Rife, D., Nagarajan, B., 2013: Probabilistic weather prediction with an analog ensemble. Monthly Weather Review 141, 3498-3516.