PRediction Of Geospace Radiation Environment and Solar wind parameterS

Work Package 3
Forecast of the evolution of geomagnetic indices

Deliverable 3.5
AE Forecast Models

Magnus Wik, Peter Wintoft, Juri Katkalov
February 5, 2018

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 637302.

Document Change Record

Issue	Date	Author	Details
v1	June 30, 2017	M. Wik	Initial draft
v2	February 5, 2018	M. Wik	Second version

Contents

Summary

The overall aim of WP 3 concerns improvement and new development of models based on data driven modelling, such as NN and NARMAX. Existing models for *Dst* and *Kp* will be analysed and verified with the aim of finding weaknesses and to suggest improvements. Solar wind and geomagnetic indices shall also be analysed in order to develop models for the identification of features, such as (but not limited to) shocks, sudden commencements, and substorms. Such categorisation will aid the model development and verification, and can also serve as alternative approach to models providing numerical input-output mapping. In addition to the development of *Dst* and *Kp* models new models will be developed to forecast AE. The models will be implemented for real-time operation at IRF and data and plots will be provided on a web server.

This deliverable, within task 3.5, concerns development of new forecast models for the AE indices. The models are compared against previous models and analysed considering, e.g. input parameters, network topology, lead time, and time delays. All three indices, AE, AL and AU were used. The predicted AE index will be used as input to the IMPTAM model. This report is a summary of the development.

Acronyms

ACE	Advanced Composition Explorer
AE	Auroral Electrojet
CME	Coronal Mass Ejections
DOY	Day of year
IMPTAM	Inner Magnetosphere Particle Transport and Acceleration Model
IRF	Institutet for rymdfysik (Swedish Institute of Space Physics)
ML	Machine Learning
NARMAX	Nonlinear AutoRegressive Aoving Average model with eXogenous inputs
NARX	Nonlinear AutoRegressive eXogenous Model
NN	Neural Network

1 Introduction

A few models exist for the prediction of the AE indices, as described in the model overview in deliverable D3.1.

The sections below describe the new development of the AE, $AU$and AL models. For this purpose the data collected and described in D3.2 have been used. The evaluation of the models, in D3.3, indicates weaknesses and provides a starting point for the further development. The work here is compared to the paper by Gleisner (1997).

2 Development Tools

For this task, we used the Python Ecosystem for Machine Learning (ML). One of the reason for using Python for machine learning is because it is a general purpose programming language that can be used both for research and in production. This simplifies the transition from development to production.

SciPy is an add-on of Python libraries for mathematics, science and engineering. It consists of NumPy, Matplotlib and Pandas (https://www.scipy.org). The Scikit-learn library is useful for developing and practice machine learning in Python. It is built upon and requires the SciPy ecosystem. The focus of the library is machine learning algorithms for classification, regression, clustering and more. It also provides tools for related tasks such as evaluating models, tuning parameters and pre-processing data.

For developing the forecast models we used Keras and Theano, where Keras is a minimalist Python library for deep learning that can run on top of Theano (https://keras.io). Theano is an open source project released under the BSD license, and was developed by the University of Montreal.

As Keras was the main ML library used in this task, we summarize the construction of a Keras model. The main type of model is a sequence of layers called a Sequential which is a linear stack of layers. First a model is defined, using a stack of configured layers. The model is then compiled, where the loss function and optimizer are specified. The model is trained and optimised on a sample of data. Finally, the model is verified on test data and ready for predictions on new data.

3 Data

For the model development, we focused on temporal resolutions of 5 and 30 minutes, where the 5 minute models are for real-time forecasts based on measurements at L1, whereas the 30 minute models can be used in connection with Sun-L1 models. The 30 minute models can be updated to other temporal resolutions, eg. 1 hour, if needed. The data were resampled using the mean, minimum and maximum for the given time interval. Missing data where removed before training the models, and no interpolation of datagaps were performed.

3.1 Solar Wind at L1

The solar wind data used for training the models are the solar wind density, n, speed, V, IMF vectors B_z, and B_y and the magnitude B, where B is the magnitude of the three-component vector magnetic field $\mathbf{B}$, using the ACE level 2 data. We did not include the B_x component since this parameter is unlikely to improve the forecasts, as shown in other studies (e.g. Gleisner (1997)) The number of 1-minute datagaps per day, for each parameter, is shown in Figure 1. It is clear from the figure, that the missing data for the density, limits the number of available data samples.

In Table 1 we list the count, data coverage, mean, min and max for the solar wind parameters used in training the models. Again, the solar wind density, have a data coverage of only 61%. This is partly due to plasma instrument outage during proton events. To capture daily and seasonal variation, we also use the sine and cosine of UT and day of year. The correlations between the solar wind parameters are small, with the highest (anti) correlation between B_x and B_y (-0.38) and between n and V (-0.28).

Table 1: Descriptive statistics for the solar wind parameters.

Inputs	B_z	B_y	B_x	n	V
Count	$9.3 \cdot 10^6$	$9.3 \cdot 10^6$	$9.3 \cdot 10^6$	$5.8 \cdot 10^6$	$8.7 \cdot 10^6$
Coverage	98%	98%	98%	61%	91%
Mean	$7.8 \cdot 10^{-3}$	$5.9 \cdot 10^{-2}$	$5.6 \cdot 10^{-2}$	5.8	$4.3 \cdot 10^2$
Min	$-7.8 \cdot 10^1$	$-5.6 \cdot 10^1$	$-5.3 \cdot 10^1$	$3.4 \cdot 10^{-2}$	$2.2 \cdot 10^2$
Max	$6.9 \cdot 10^1$	$5.7 \cdot 10^1$	$4.2 \cdot 10^1$	$1.9 \cdot 10^2$	$1.3 \cdot 10^3$

Next we describe the construction of the data sets for training, validation and testing. All data is initially stored locally in a database at IRF Lund. The solar wind data and the AE indices for the period 1998 to 2015 are imported and added to a Python Dataframe, consisting of data values with time stamps. The solar wind parameter B is derived and added to the dataframe. We also added the AE indices for a model study (see Section 7.2). The solar wind data are propagated to the location of the magnetopause, and a shift (additional lead time) is also added up to 30 minutes. The data are resampled to 5 or 30 minutes, and we introduce time delays up to 120 minutes, to capture the dynamics (memory). For the AE indices we resample the original 1-minute data to 5 or 30 minutes.

The distribution, for the years 1998 to 2015, of the solar wind parameters B_z, B_y, n and speed, V are plotted in Figure 2. Only B_z is close to having a normal distribution, whereas the parameter B_y have a bimodal distribution, which is related to the Parker spiral. The solar wind speed, V, have basically two fundamental states, the slow and fast solar wind. The average solar wind reach around 400 km/s, and the fast up to around 700 km/s. In addition to these two states are CMEs with velocities up to around 2-3000 km/s. So the distribution of V is actually a sum of several distributions.

Next, in Figure 3, are the Box- and whisker plots for the same parameters. Boxplots summarize the distribution of each attribute, drawing a line for the median (middle value) and a box around the 25th and 75th percentiles (the middle 50 of the data). The whiskers give an idea of the spread of the data, and the dots outside of the whiskers show possible candidate outlier (or perhaps rare or extreme) values (values that are 1.5 times greater than the size of spread of the middle 50% of the data). As can be seen, the distribution for any parameter, vary from one year to the other. It is therefore not suitable to pick any random year to be included into the training, validation and test sets. This is a multidimensional problem, which is difficult to tackle. Our approach is to cover data for the whole period into all data sets, to try and capture as much as possible of the variance in the data. The selection of data for training is described in Section 5.2.

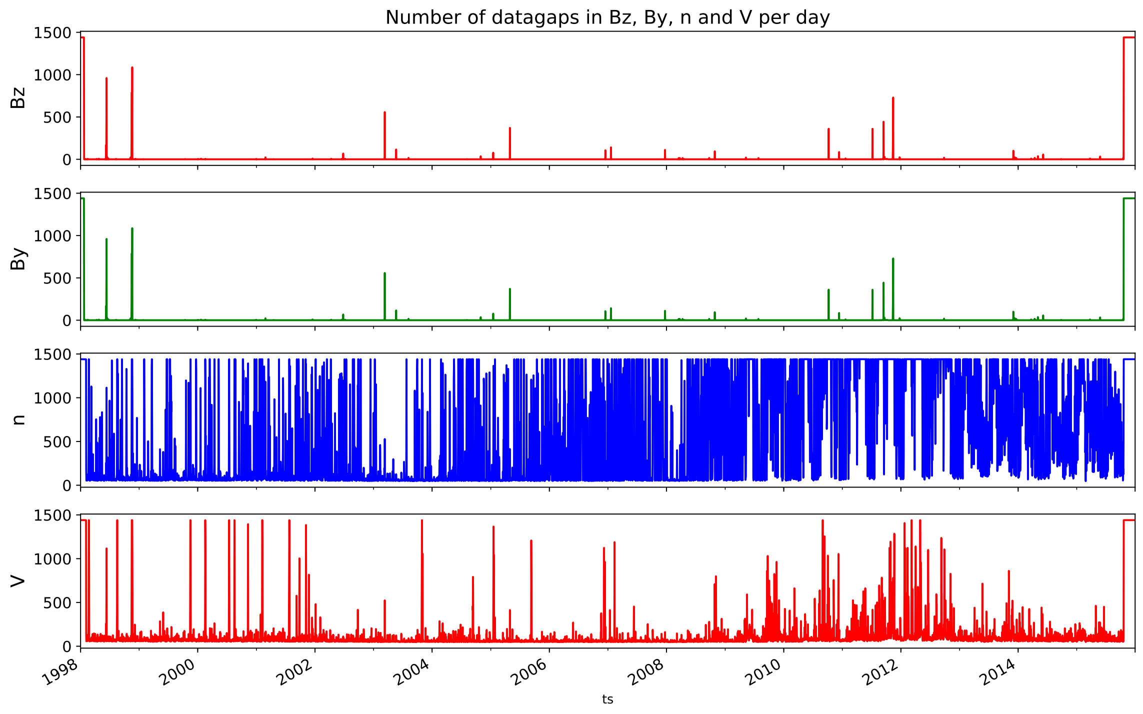

Figure 1: Number of datagaps in IMF B_z, B_y and solar wind density, n and speed, V measured by the ACE spacecraft, for the period 1998 to 2015.

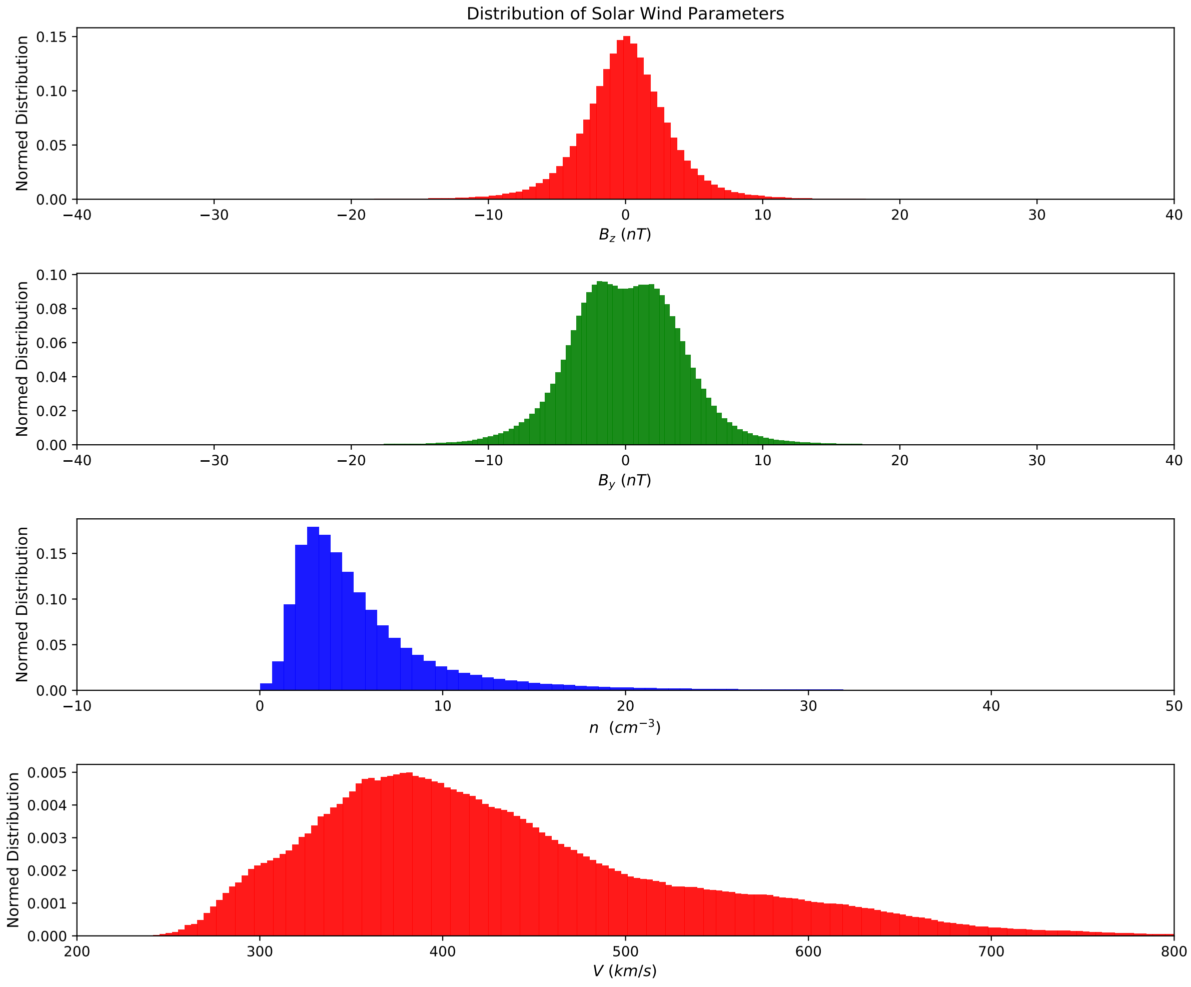

Figure 2: Distribution of solar wind parameters B_z, B_y, n and speed, V measured by the ACE spacecraft, during 1998 to 2015.

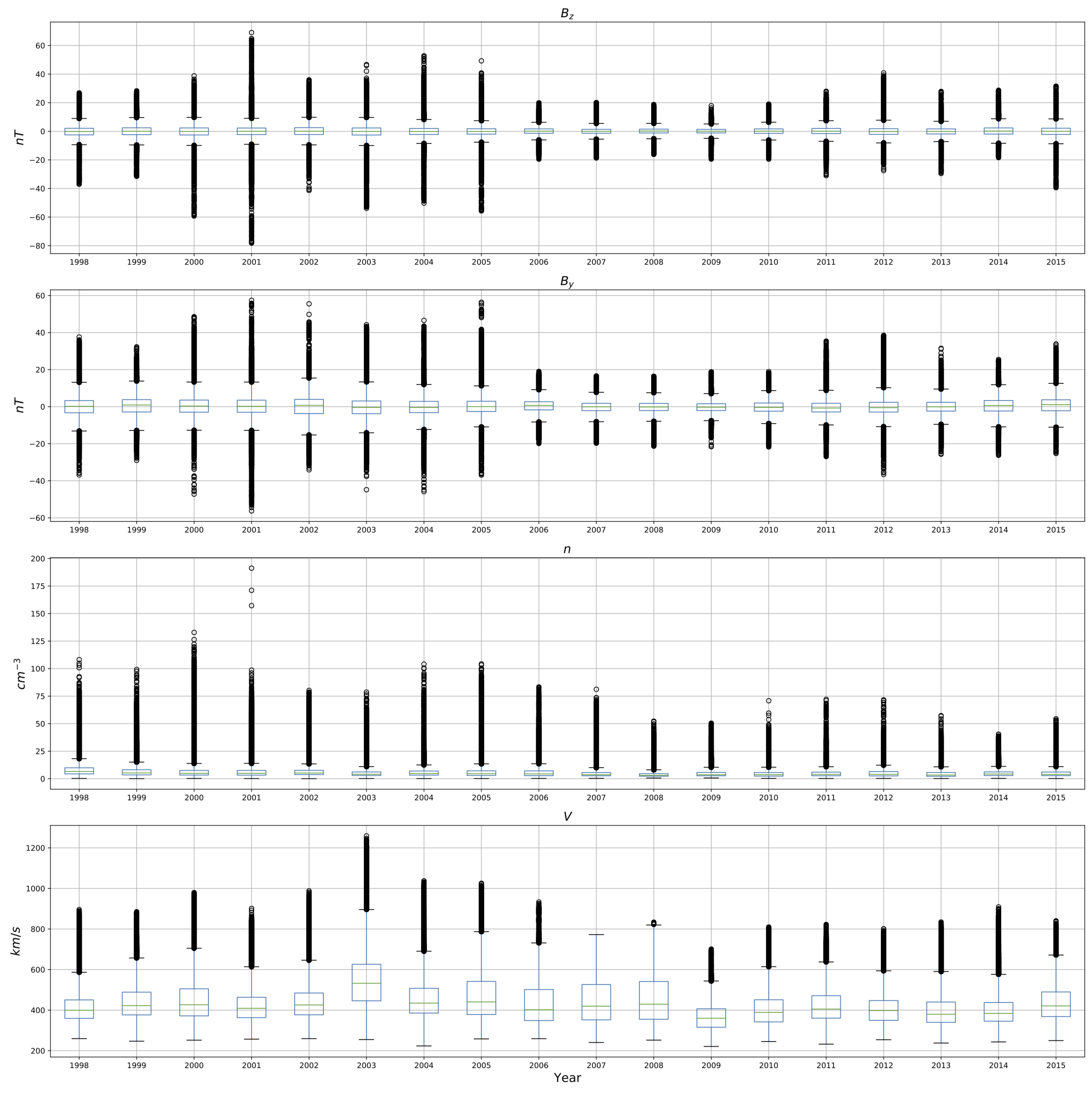

Figure 3: Box and Whisker plots of the solar wind parameters B_z, B_y, n and speed, V measured by the ACE spacecraft, during 1998 to 2015. The boxes, represent the middle 50% of the data, and the Whiskers represent the percentage of data outside the middle 50%. Data points beyond whiskers on the two sides are outliers.

3.2 AE indices

We use the three indices AE, AL and AU as target data for training the models. The AE is the difference between AU and AL. Therefore, models predicting AU and AL separately can be combined to also provide forecasts of AE. However, we also make a comparison against a model targeted to AE. The AE indices are described in detail in the book by Mayaud (1980).

A scatter plot matrix, for a selected event, for the indices is shown in Figure 4. All three indices are clearly skewed, with the highest correlation between the AE index and the AL index. Descriptive statistics for the indices are listed in Table 2. The correlation between them are listed in Table 3. Due to the high (negative) correlation between AE and AL it should be possible to use only AL and AU to forecast AE. This is verified in Section 8.

Table 2: Descriptive statistics for the AE indices.

Inputs	AL	AU	AE
Count	$9.2 \cdot 10^6$	$9.2 \cdot 10^6$	$9.2 \cdot 10^6$
Mean	$-1.1 \cdot 10^2$	$6.9 \cdot 10^1$	$1.8 \cdot 10^2$
Min	$-4.1 \cdot 10^3$	$-9.7 \cdot 10^2$	1.0
Max	$7.9 \cdot 10^1$	$2.1 \cdot 10^3$	$4.2 \cdot 10^3$

Table 3: Correlation between the AE indices

Index	AL	AU	AE
AL	1.00	-0.65	-0.96
AU	-0.65	1.00	0.83
AE	-0.96	0.83	1.00

4 Time Series Analysis

In this section we analyse the data prior to the pre-processing. First we examine the data with respect to pronounced seasonal and universal time (UT) variations in the indices. We also calculate the auto-correlation, which is a useful measure when creating the data sets.

4.1 Seasonal and UT variations

A contour plot of the average AL and AU as function of UT and month is shown in Figure 5. For the AL index there is a minimum in the summer time, whereas for the AU index there is a maximum. This is also shown in Figure 6, and in Figure 7.

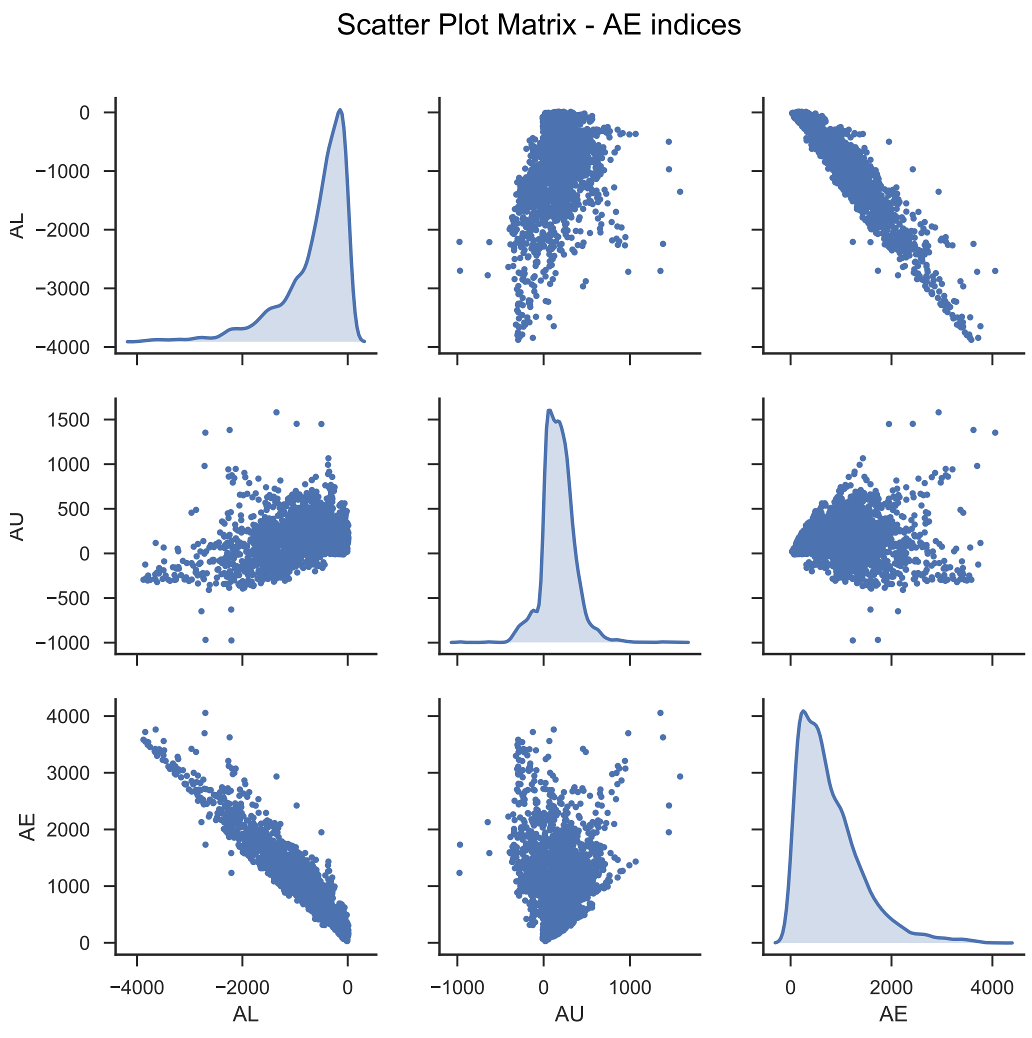

Figure 4: Scatter plot matrix of the AE indices, for October 28-30, 2003.

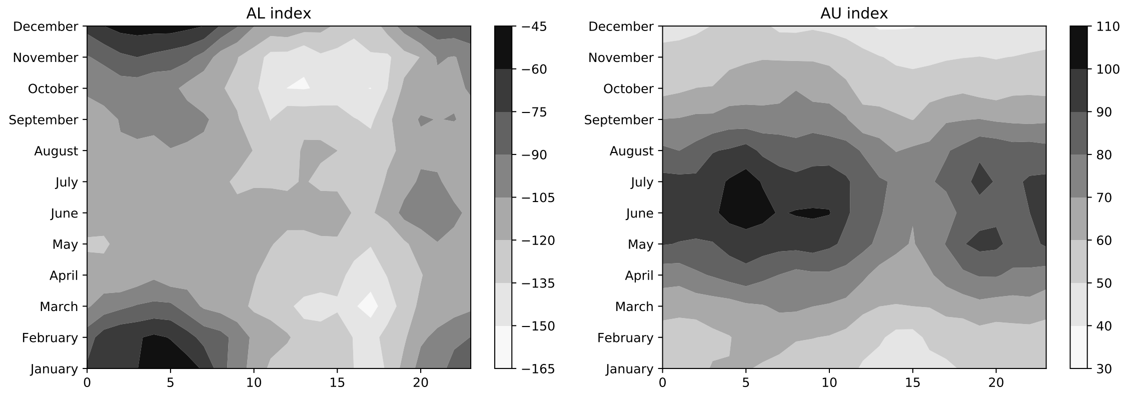

Figure 5: Seasonal and daily (UT) variations of the AL and AU index

Similarly we can also spot a UT variation in Figure 5, that changes during the year, for both AL and AU. A UT variation is also seen, for $AE > 800$ nT (Figure 8). During Winter, the occurrence frequency is highest around 15 UT, and at around 17 UT during the equinoxes. During summer these peaks have vanished. The results here are in accordance with those in Ahn & Moon (2003).

These results indicate that we need to take the seasonal and UT variation into consideration for the models. The UT and seasonal variation is fully described in a continuous way by adding sine(UT), cosine(UT), sine(DOY) and cosine(DOY) to the network inputs. It is also shown in Section 7.2, that indeed the performance is higher when including these additional inputs.

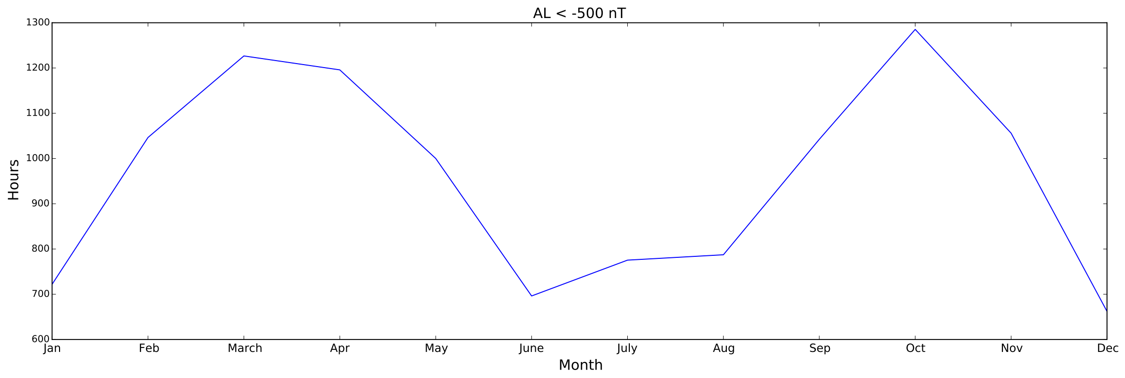

Figure 6: Seasonal variation in the number of hours per month for which $AL < -500$ nT

4.2 Auto-correlation

When training a neural network, it is important that we separate the training, validation and test set by at least the autocorrelation length of the input parameters, as was described

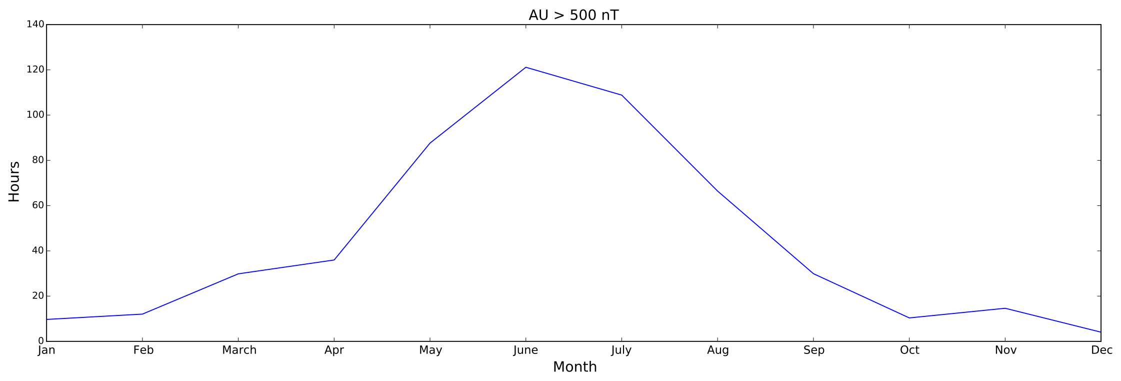

Figure 7: Seasonal variation in the number of hours per month for which $AU > 500\ nT$

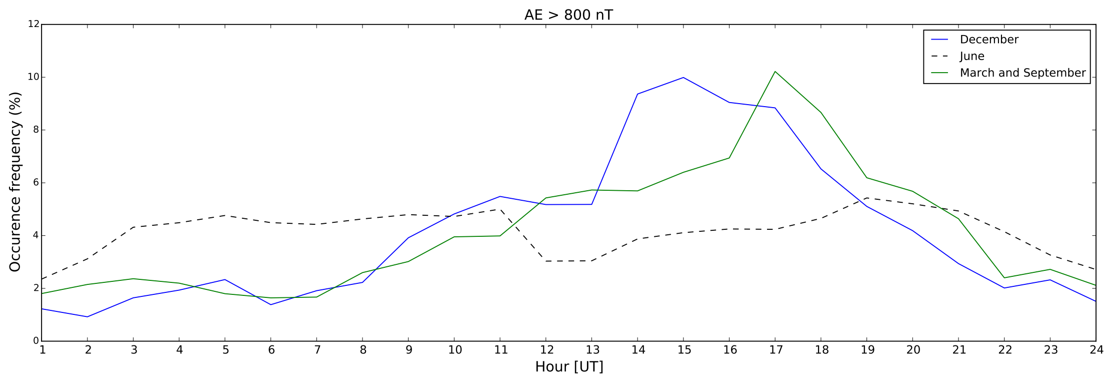

Figure 8: Occurence frequency for $AE > 800\ nT$ during a Winter month, a Summer month and during the Equinoxes.

in Vassiliadis et al. (1995). The autocorrelation for B_z during two days in 2003, the Halloween event, is plotted in Figure 9. We note that in this case, the autocorrelation drops to zero after about 2-3 hours at most. This have implications when selecting the rows of input data. Each row in our training data consists of up to 120 minutes back in time. It is therefore not possible to randomly split any rows into the data sets, training, validation and test sets. Otherwise there will be an overlap in data between the sets, and they are no longer independent sets. We decided to use yearly data, with few and minimal overlaps between the data splits.

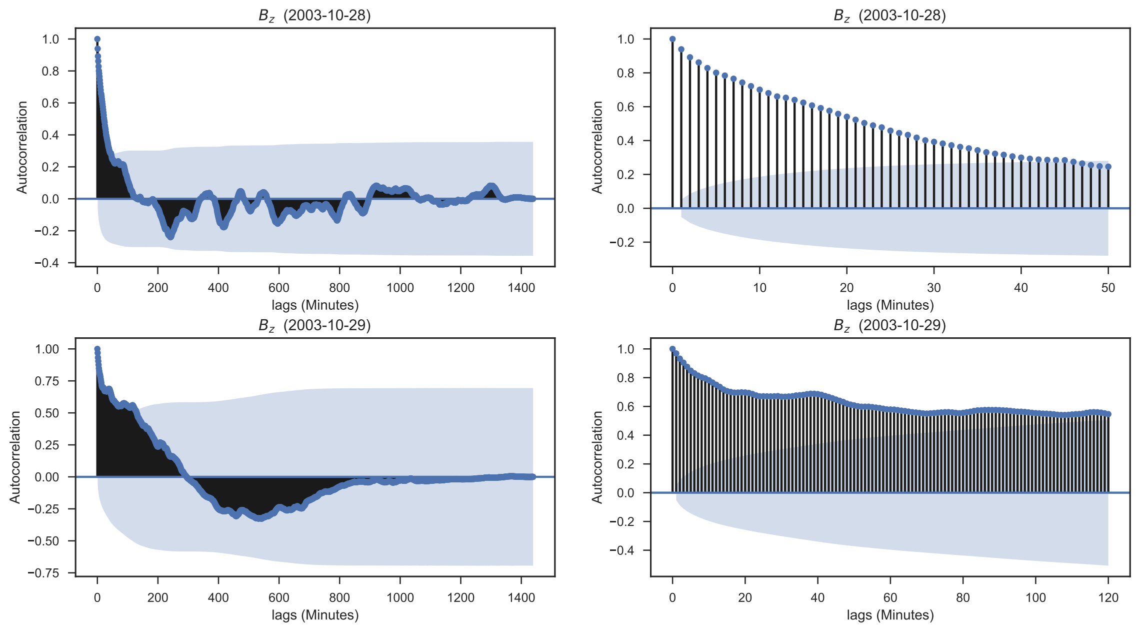

Figure 9: The autocorrelation for the B_z component during October 28 and 29, 2003. The figures on the left show the lags, in minutes, for the whole day and the figures on the right show lags up to 50 and 120 minutes. Confidence intervals are drawn as a blue cone. By default, this is set to a 95% confidence interval, suggesting that correlation values outside of this cone are very likely a correlation.

5 Data Pre-processing

Before applying any machine learning algorithms, the data needs to be pre-processed. This is because ML algorithms make certain assumptions about the data. Pre-processing involves e.g. data transforms, rescaling, normalisation and standardisation.

5.1 Standardise and Transform Data

Most machine learning algorithms perform better if all data have a consistent scale or distribution. Normalisation is one way, but it requires knowledge (or estimates) about the minimum and maximum observable values. This we obviously don't know. Another way is to standardise the data which involves rescaling the distribution of values so that the mean of observed values is 0 and the standard deviation is 1. During testing however, we found no significant difference between these two rescaling methods.

However, even after rescaling the data, most of the parameters have skewed distributions. It is sometimes suggested that such distributions should be transformed to be close to normal before training. It is obvious that none of the distributions will be exactly normally distributed after a transform, but perhaps close enough. But the transformed data will have 0 mean and 1 standard deviation. A common transform is the Box-Cox transform, which we can use, with the caveat that the values need to be positive (Box et al. (1964)). We will however, leave this for later studies.

5.2 Algorithm Evaluation

The purpose of algorithm evaluation is to know how well the model performs on unseen data. There are however, several useful estimates of performance for machine learning algorithms. The simplest method (data split) is to use different training and validation sets, which is fast and suitable for large data sets, as in this case. Another approach is k-fold Cross validation (CV). The data is then split into k parts (or folds) usually with k=10, and the algorithm is then trained on k-1 folds and the last fold is the validation set. This is repeated k times, providing a more reliable estimate of the model. For time series, however, this is not straight forward, and there are alternative methods to simple k-fold CV. We will leave CV for time series to future studies.

The data split method may produce lower bias for large datasets, as we have here, where we can assume that both data sets are representative of the underlying problem and the data sets have similar distributions. However, in practice there will always be differences as is also the case here. A low bias may also indicate an algorithm that is too simple, which should not be the case here.

The downside of this approach is that it can have a high variance and lower accuracy. This means that any differences in the training and test dataset can result in meaningful differences in the estimate of accuracy. So there is a tradeoff between minimising the errors due to bias and variance, where the errors will consist of both. By improving a model, or algorithm, and adding more data, the bias and variance will approach zero, although it will never reach zero, due to irreducible errors.

Since we use the data split method, our original data, spanning 18 years from 1998 to 2015, were split into two parts together with a third part for testing the final model. The purpose of the validation set is to avoid overfitting when training, and we evaluate the model using the test set. In Table 4, we list our training, validation and test sets. This choice is based on the analysis in Section 3.1.

Table 4: Data selected for training, validation and testing.

Data set Years	**Train**	**Val**	**Test**
1998	x	-	-
1999	x	-	-
2000	-	x	-
2001	-	-	x
2002	x	-	-
2003	x	-	-
2004	-	x	-
2005	-	-	x
2006	x	-	-
2007	x	-	-
2008	-	x	-
2009	-	-	x
2010	x	-	-
2011	x	-	-
2012	-	x	-
2013	-	-	x
2014	x	-	-
2015	x	-	-

6 Neural Networks

The power of neural networks come from their ability to learn the non-linear and complex relationships from the training data and map it to the output variable, or variables.

6.1 Network topology and configuration

A NN is basically multiple neurons arranged into a network. Each neuron is a simple computational unit that have weighted input signals and produce an output signal using an activation function. The neurons are distributed in layers, where the architecture of the neurons is called the network topology. The layers are divided into input (or visible), hidden and output layers. The visible layer takes the inputs from the dataset. In this case, the output layer have a single neuron.

The weights are often initialized to small random values, and it is desirable to keep the weights in the network small. The weighted inputs are summed and passed through an activation function, or transfer function. Typically, nonlinear activation functions are used, except for linear functions in the output layer. Here we use the hyperbolic tangent (tanh) activation function in the hidden layers. Typically the weights are updated using the stochastic gradient descent algorithm, but here we use the more efficient Adam optimizer, as described in Kingma & Ba (2014).

6.2 Network Training

Once the NN have been configured, it needs to be trained using the dataset. One row of data is exposed to the network at a time as input. The input data is then propagated through the network layers, a so called forward pass, to finally produce an output value. The output value is compared with a target value, in this case e.g. an AE value, and an error is calculated. For the Back-propagation algorithm, the error is then propagated back through the network, and the weights are updated one layer at a time, according to the amount that they contributed to the error. This process is repeated for all the training data examples.

One round of updating the network is called an epoch. Typically, a network is trained for hundreds or even thousands of epochs, depending on the algorithm, network topology and training set size. The weights are not updated after each example, but rather after a batch of examples. This is called batch learning, which here is chosen to be between 20 and 100 examples. The amount that weights are updated is controlled by a configuration parameter called the learning rate, typically 0.1 or 0.01 or smaller. The update equation is usually complemented with additional configuration terms for faster training. During training, the network is exposed also to a validation set to avoid overfitting.

After training, predictions are made on test data in order to estimate the skill of the model on unseen data. The final model, ready for operational use, consists of the network topology, network weights, and parameters for transforming and standardising the data. Predictions are then made by providing new input, in real-time, as a forward-pass generating an output as a predicted value. The final AE models will be implemented in WP3.6.

7 Model studies

In this section we show the results from several model studies. The purpose of these studies were to get insight into the importance of e.g. network configurations, forecast lead time and input parameters, and how these will affect performance. For each study, a total of 10 models were used. The model with the smallest validation error (MSE) was then selected, but here the actual error values are not so important, but rather the difference between the models. In all models, we used the solar wind variables B_z, B_y, B, n and speed, V together with sine and cosine of UT and year.

7.1 Evaluate a deep and wide network

In this study we used multiple models, with combinations of 1-3 hidden layers and 2-20 nodes per hidden layer, in total 30 models. Each configuration, of layer and nodes, was trained 10 times. The number of nodes are the same for all hidden layers. The network performance as function of layers and hidden nodes is plotted in Figure 10 for AU. The results indicate that an optimal network should use 1-2 hidden layers and at least 12 nodes per hidden layer.

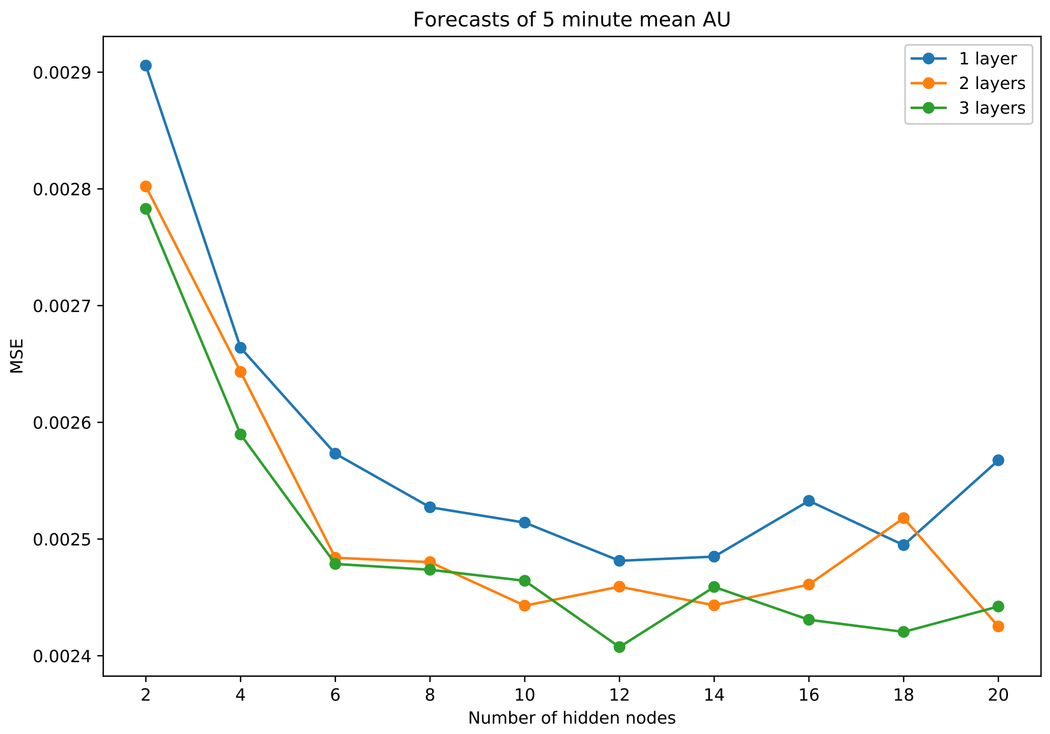

Figure 10: The mean square error as a function of network topology, with combinations of 1-3 hidden layers and 2-20 nodes per hidden layer.

7.2 Parameter studies

The forecast errors of 5 minute mean AU for varying input parameters, are shown in Figure 11. In this study we also added the index itself to the inputs, with the same time stamps as for the solar wind parameters. With only B_z as input we obviously get the largest error. However, if we use data from ACE, it might be useful to have a model based on only B_z and maybe B_z and V due to the larger uncertainties with the solar wind density n. When adding n and V the error drops, as expected, and some more when we also add B. When adding also B_y we do not see any improvements. A major drop in MSE occurs when we also add time and the index itself as inputs. This is an indication that the forecast lead time is shorter than the autocorrelation length of the index, which means that past values of the index can partly be used to forecast the index, if we use data from e.g. ACE.

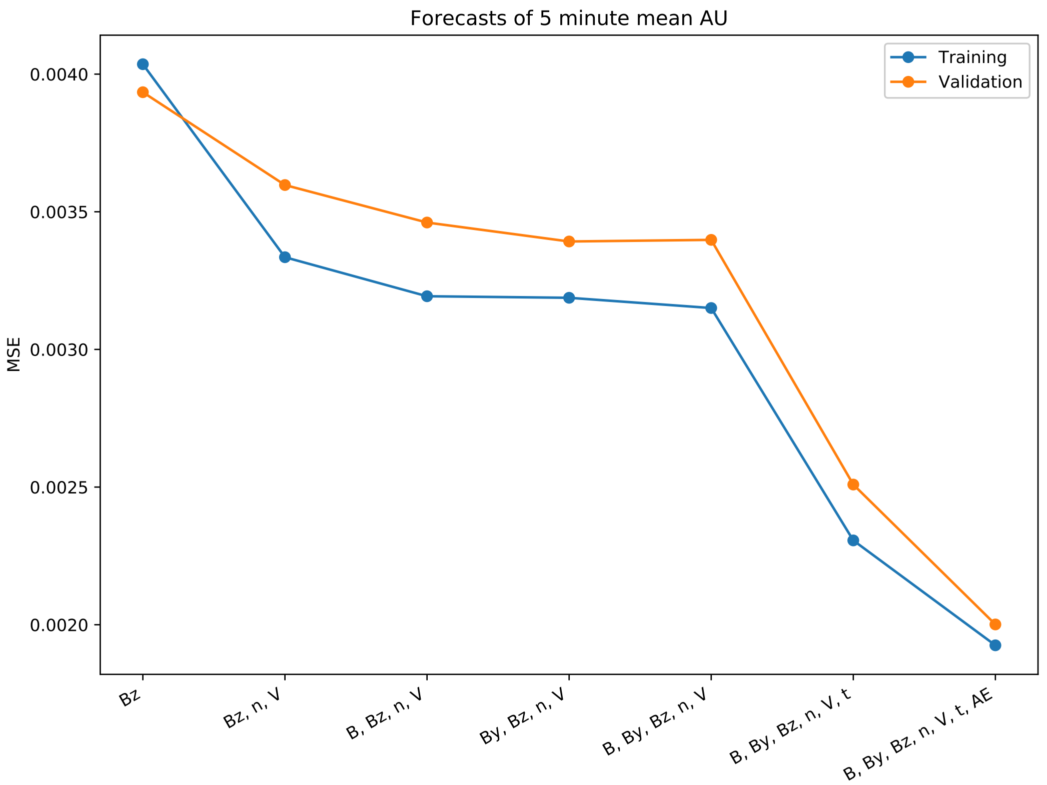

Figure 11: Forecast errors of 5 minute mean AU for different input parameters.

7.3 Forecast lead time

Here we examined how the errors change when we increase the forecast lead time. By lead time we here mean the actual lead time added after the propagation time. As seen in

Figure 12, the error increases slowly up to 30 minutes, which might suggest that a forecast lead time above 30 minutes are less useful. However, a more careful study is needed.

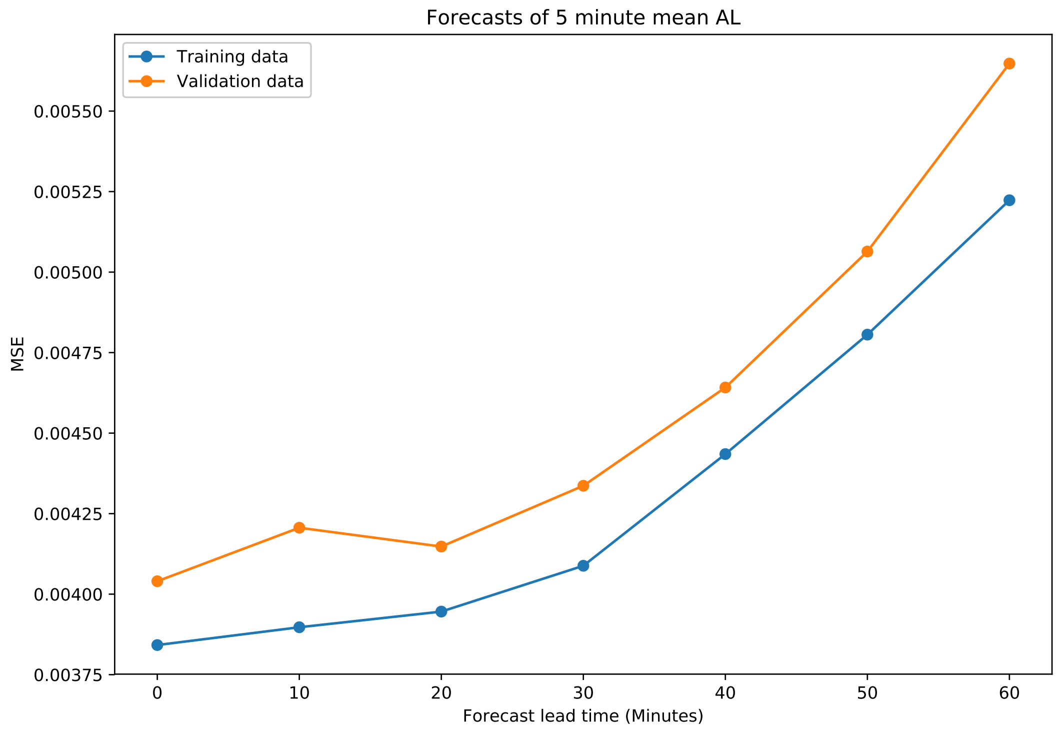

Figure 12: Forecasts of 5 minute mean AL for different forecast lead time.

7.4 Hidden nodes vs time delays

In the next study we investigated how the forecast error change with different combinations of time delays and number of hidden nodes. Earlier we found that at least 12 hidden nodes should be sufficient, which is similar here as well. It is interesting to note that the difference in MSE, between any two nearby models, is almost halved when we increase the time delays. There seem to be a limit in performance at about 100 minutes in time delays, which indicate that the magnetospheric system memory saturates at a time delay of 100 minutes. The same conclusion was made by Gleisner (1997).

7.5 Parameters vs time delays

This study is a combination of two other studies. In Figure 14 we plot the model error against models with different input parameters and time delays (solar wind history). Again, we see that time delays up to about 100 minutes is sufficient and that, especially, the sine and cosine of UT and year (t) are important inputs.

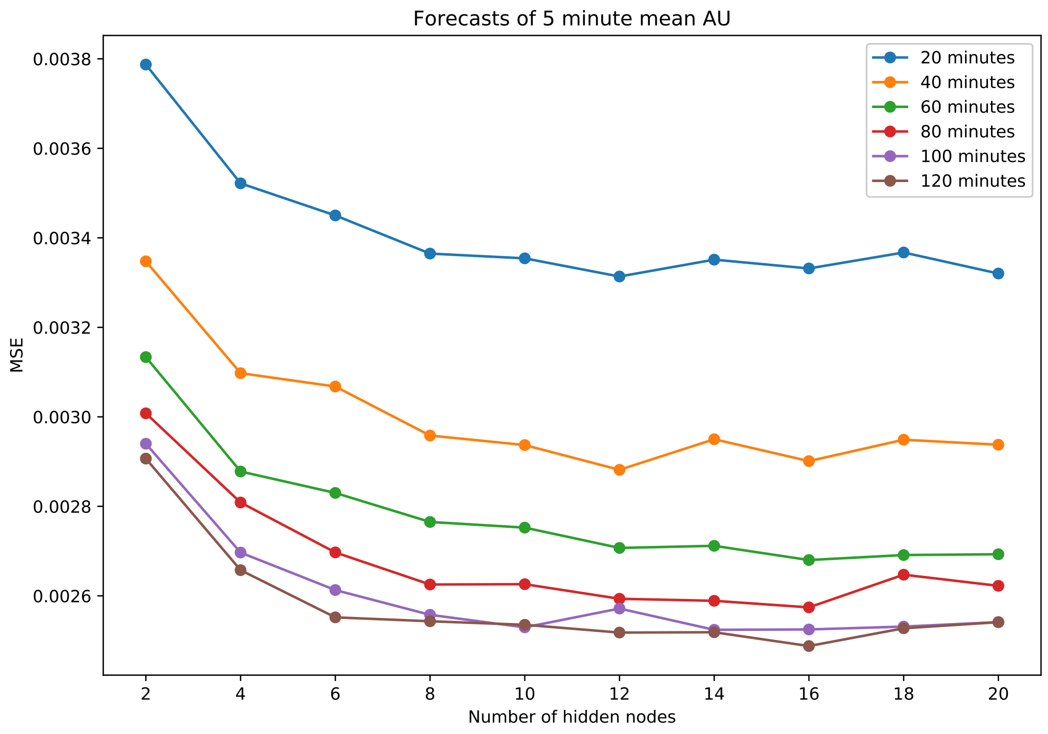

Figure 13: Forecasts of 5 minute mean AU for different number of hidden nodes and time delays.

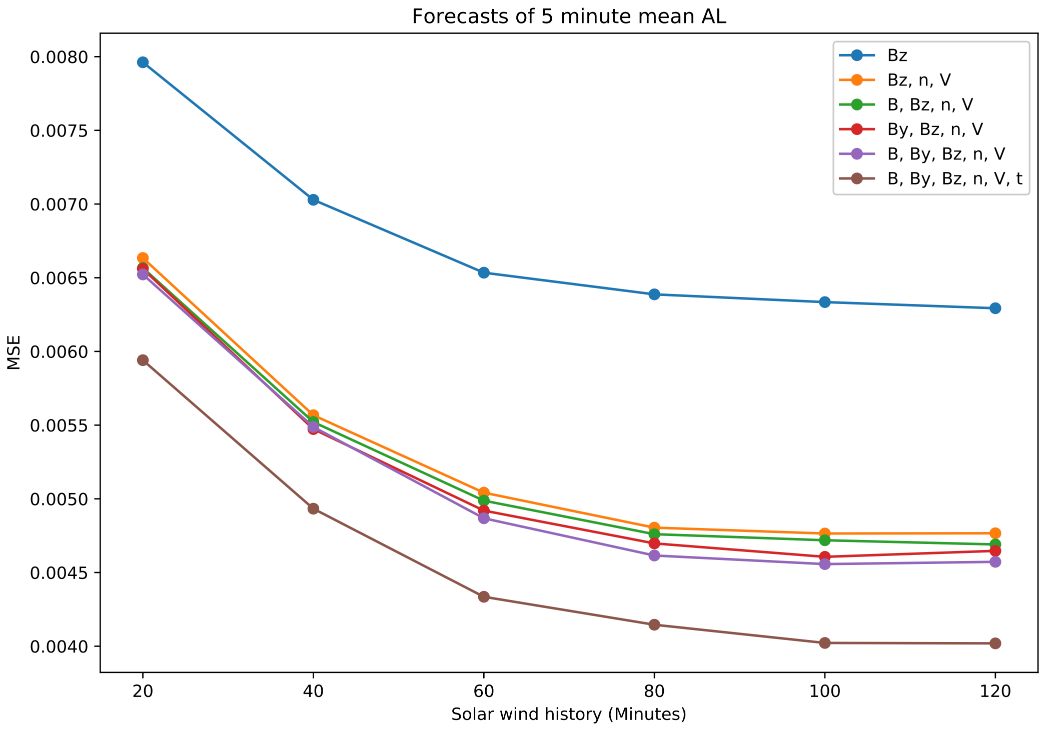

Figure 14: Forecasts of 5 minute mean AL for different forecast lead time.

7.6 Learning curves

Learning curve theory is based on a graph that compares the performance of a model on training and validation data using a varying number of training instances. When examining the training and validation curves, the performance for both should reach a limit or plateau. Then there will be a fixed gap between the two errors. The purpose of learning curves is to understand how many input parameters are needed as well as the amount of data.

Typically there are three types of learning curves: high bias, high variance and the ideal learning curve. Each of these actually consists of two curves, one for training and the other for validation. With a high bias, the error is high and similar for both curves, with a high variance, the error is lower but with a large gap between them, and finally, for an ideal learning curve, the errors are similar, but lower compared to those for a high bias. In principle, this means that with a high bias it should help to add more input parameters, whereas for a high variance, getting more data is likely to help. The best model is achieved when it generalises well to new data, which occurs when the training and validation curves converges.

In Figure 15 we have plotted three learning curves, each for training and validation, with increasing number of input parameters. In the first (blue), the errors are high, which is obvious since were are only using B_z as input. Adding more parameters (in red and green), the errors decrease as expected, and the gap between the training and learning curves, also decreases. In the last learning curve (green), the errors seem to almost converge. This might indicate that a lot more data is needed. Of course, in this test we only used about 2 months of data. Later we will examine what happens if we increase the length of the training set to years. The problem is however, that it takes a very long time to run these tests.

8 Forecast models: 5 minute AE, AL and AU

These models are based on the work by Gleisner (1997). They also used 5 minute averaged AE data. With the solar wind variables B_z, B_y, n and V as inputs with time delays up to 100 minutes, they were able to reach a correlation of 0.87. Our best model reach a correlation of 0.883. When we combine the forecast models for AU and AL to forecast AE we get similar results with a correlation of 0.881. However, it should be noted that it is not straightforward to compare model results when the datasets are different, which is the case here. Also, they did not use the cosine and sine of UT and day of Year. As indicated in Section 7.2, this contributed to a slightly higher correlation. The results are listed in Table 5, for all three indices as well as for the difference between AU and AL.

Table 5: Linear correlation for all 5 minute models

AE	AU - AL	AL	AU
0.883	0.881	0.836	0.836

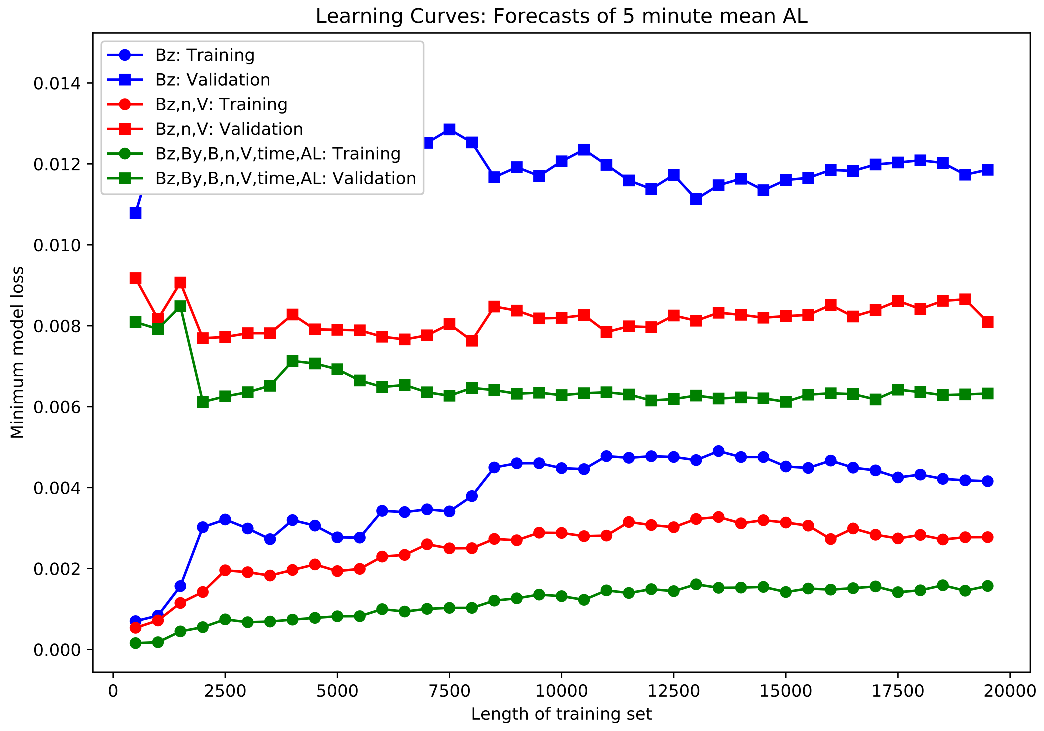

Figure 15: Forecasts of 5 minute mean AL for different forecast lead time.

For each index, we trained the network 20 times. The models were evaluated for the training, validation and test sets. We also included all data for 1998-2015 in the evaluation. As an example, the results for the AU index from these runs are plotted in Figure 16. We then picked the model with the lowest validation error, which is model 14. This model also gave the lowest test error. We note that the errors, between the training, validation and test sets are all well separated, although the differences are small. This might indicate, that the distribution in the sets are different and that we should look into cross-validation in the future.

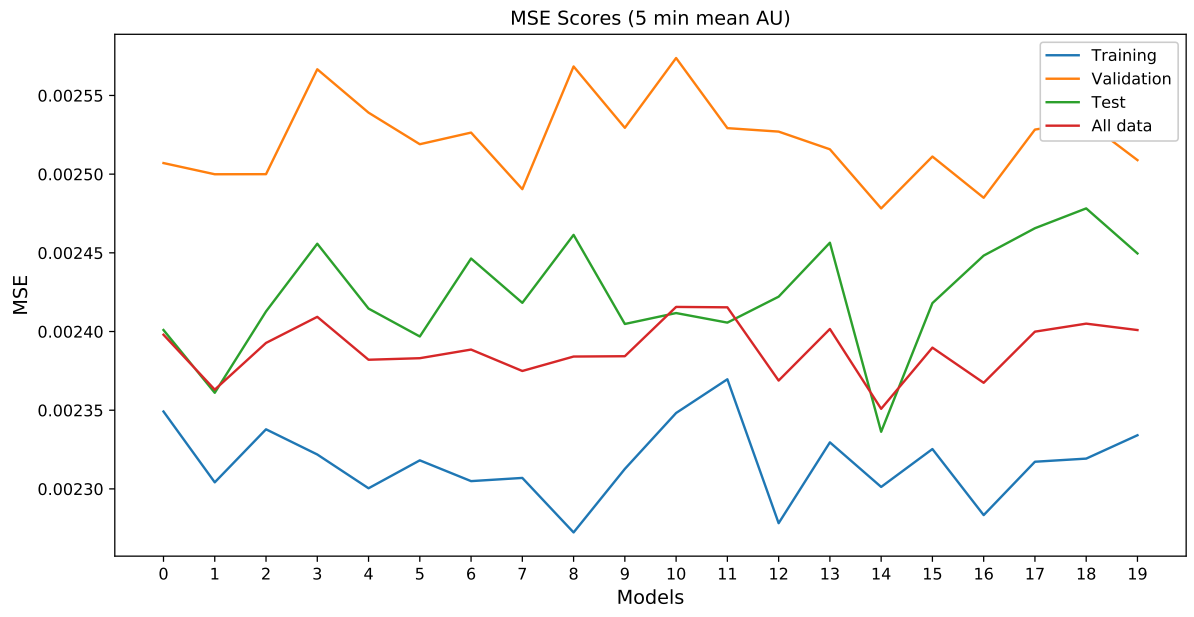

Figure 16: MSE scores for the 5 minute averaged AU forecasts. Model 14 provides the lowest validation error and was selected as the model for testing.

The performance of the models are illustrated, in Figure 17, with two examples from the test set. The first example is from end of March, and beginning of April, in 2001. The second example is from April 4-6, in 2001. The solar wind parameters are propagated and 5 minute averaged.

The first storm is stronger, with B_z reaching down to about -50 nT, and density up to about 80 cm^{-3}. The model follows the observed, but misses a few peaks. In the second example, the model underestimates and overestimates until after around noon, on April 5, when the forecasted values agree with the 5 minute AE quite well.

9 Forecast models: 30 minute AE, AL and AU

The 5 minute AE forecast models are useful for real-time solar wind data. But when coupled to solar wind models from the Sun it might be better to use a more robust

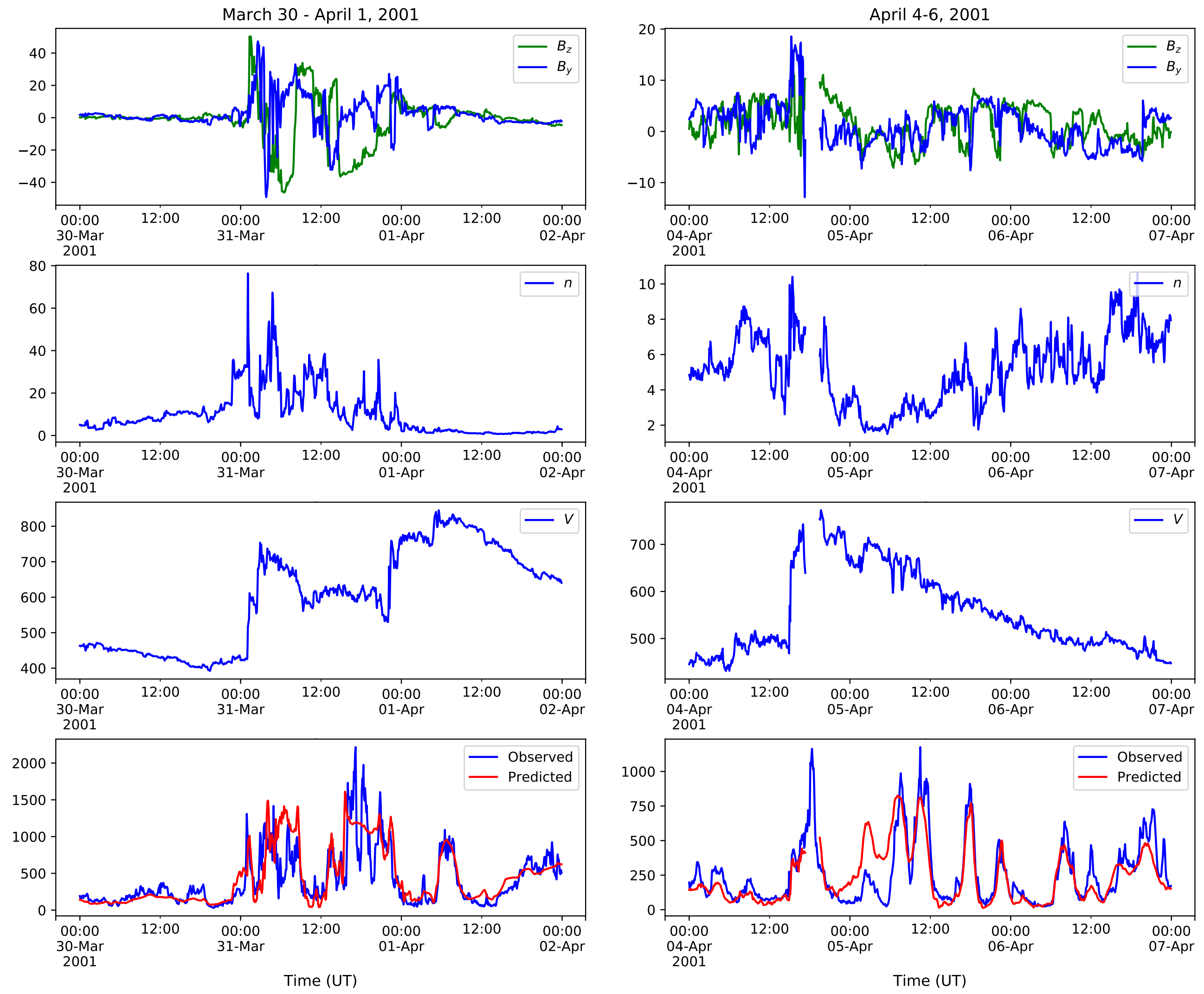

Figure 17: Forecast of 5 minute averaged AE during two events in 2001. The top three panels show the 5 minute averaged solar wind magnetic field (B_z and B_y), density (n) and speed (V). The bottom panel show observed (blue) and forecast (red) AE index.

model, with e.g 30 minute, or even higher, resampled data, to try and achieve a better performance. For that reason we also developed 30 minute AE forecast models. The results are shown in Table 6.

Both the solar wind parameters and the indices were resampled using the mean, minimum or maximum value for each 30 minute interval. Using the Mean method, all parameters, both solar wind data, as well as the indices, were averaged. In the Max and Min method, all solar wind parameters were resampled using the maximum values, except B_z, which used the minimum values.

Table 6: Linear correlation for all 30 minute AEmodels

Method	AE	AL	AU
Mean	0.90	0.87	0.86
Max	0.86	-	0.84
Min	-	0.83	-

Scatter plots of the 30 minute mean AE is seen in Figure 18, for the training, validation and test sets as well as for all data. The green line indicates a perfect fit, and the orange line is the calculated linear regression.

The performance of the models are illustrated, in Figure 19, with the same two examples from the previous section. The results are similar, but the 30 minute model is slightly more accurate, at least after around noon, on April 5. It is interesting to note, that both the 5 minute, as well as the 30 minute model overestimates AE at around midnight on April 5.

10 Forecast Verification: 5 minute models

For our models we use the flat delay propagation, with lead times between 20 to 110 minutes, with a median value of about 60 minutes. We use this value for the persistence model. The persistence model is a baseline model to compare our models with. It is a very simple model, that takes the observed value, of the index, at timestamp t, and use this as the forecast for timestamp $t + 60$ minutes.

Forecast verification is described in detail in Jolliffe & Stephenson (2012). The forecasts are verified using the following measures.

- Bias (or mean error)
- Mean absolute error (MAE)
- Root mean square error (RMSE)
- Linear correlation (Corr)
- Mean square error skill score, 1-MSE$_{model}$/MSE$_{persistence}$

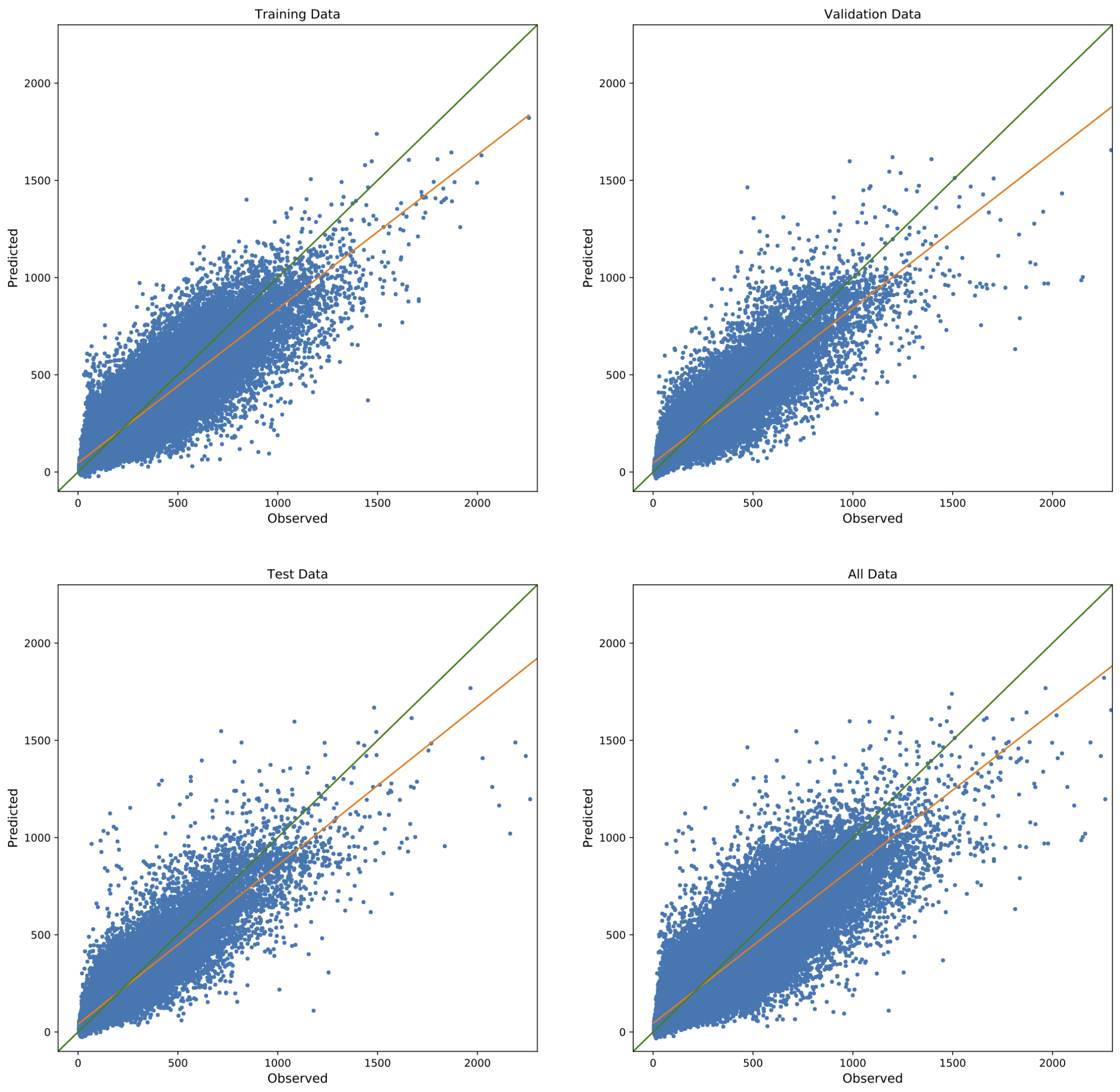

Figure 18: Scatter plots of 30 minute mean AE forecasts. The green line indicates a perfect correlation, and the orange the calculated linear regression.

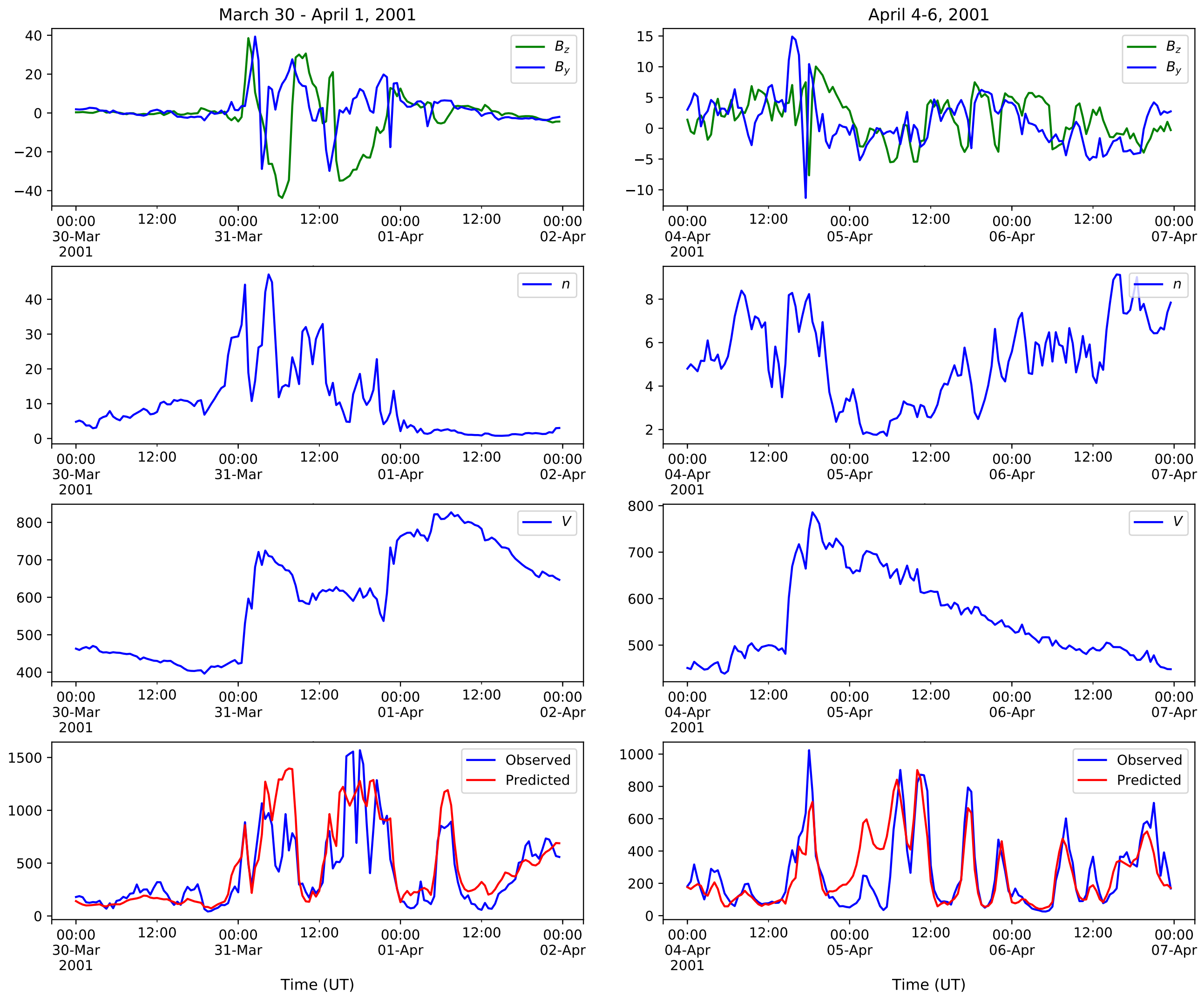

Figure 19: Forecast of 30 minute averaged AE during two events in 2001. The top three panels show the 30 minute averaged solar wind magnetic field (B_z and B_y), density (n) and speed (V). The bottom panel show observed (blue) and forecast (red) AE index.

- Prediction efficiency, 1-MSE$_{model}$/MSE$_{observation}$

The verification results, for all three indices, are listed in tables 7, 8 and 9. The results are listed for training, validation and test sets separately. We also calculate the measures for all data for the neural network model and for the persistence model. We achieve the highest correlation 0.88 for AE and MSESS of 0.6. This is clearly better than for the persistence model.

Table 7: Verification table for the AE index.

	Bias	**MAE**	**RMSE**	**Corr**	**MSESS**	**PE**	**Max**	**Min**
Model (train)	-2.09	69.79	107.22	0.88	0.62	0.78	2987.2	2.8
Model (val)	-1.81	69.87	108.58	0.87	0.59	0.76	3260.0	2.6
Model (test)	-0.44	64.59	103.35	0.88	0.58	0.78	3407.2	2.6
Model (all)	-1.66	68.66	106.70	0.88	0.60	0.78	3407.2	2.6
Per (all)	-0.01	103.48	169.34	0.72	0.00	0.43	3407.2	2.6

Table 8: Verification table for the AL index.

	Bias	**MAE**	**RMSE**	**Corr**	**MSESS**	**PE**	**Max**	**Min**
Model (train)	2.29	56.90	90.39	0.84	0.61	0.71	35.2	-2894.8
Model (val)	1.69	57.34	93.48	0.83	0.58	0.69	32.2	-3330.0
Model (test)	0.83	52.48	87.66	0.84	0.57	0.70	41.4	-3747.4
Model (all)	1.82	56.03	90.54	0.84	0.59	0.70	41.4	-3747.4
Per (all)	0.01	82.65	141.49	0.64	0.00	0.28	41.4	-3747.4

Table 9: Verification table for the AU index.

	Bias	**MAE**	**RMSE**	**Corr**	**MSESS**	**PE**	**Max**	**Min**
Model (train)	0.59	28.61	43.02	0.84	0.47	0.71	1389.4	-404.2
Model (val)	0.75	29.12	44.64	0.82	0.41	0.67	1106.2	-442.2
Model (test)	1.98	27.56	43.35	0.84	0.41	0.70	1011.0	-352.0
Model (all)	0.93	28.50	43.48	0.84	0.45	0.70	1389.4	-442.2
Per (all)	-0.00	35.95	58.40	0.73	0.00	0.46	1389.4	-442.2

11 Discussion and Conclusions

In this work we have developed, in total, 9 forecast models for the three geomagnetic indices AE, AL and AU. The models were developed using feed-forward neural networks using the error back-propagation algorithm. The networks were trained using 10 years of data, spanning 18 years in total, and tested against 4 years of data.

We performed model studies, with various inputs and different network topology, to find key features and network configuration, for best performance. Based on these studies we can summarise the findings:

- Networks consisting of 2 hidden layers, and 12 or more hidden nodes, results in best performance, even though the differences are small.
- With input parameters n, V, B_z, B_y, B, sine(UT), cosine(Year), sine(UT) and cosine(Year) we achieve the lowest errors.
- For the 5 minute models, we achieve the highest correlation 0.88 for the AE index and MSESS of 0.6.
- When we increase the lead time from 0 to 60 minutes, the error increases slowly up to 20 minutes, and after that the increase is faster. This suggest that a forecast lead time above 30 minutes are less useful.
- Adding the sine and cosine of UT and day of year, lower the error significantly. This is also indicated in Section 4, where we showed that the AE indices have both a UT and a seasonal dependence.
- The 30 minute forecast models, using averaged data, resulted in better performance compared to the 5 minute models, which was expected.
- The 30 minute mean show a better performance compared to the models using minimum and maximum values.

Based on the results from the model studies we developed three 5 minute models for AE, AL and AU indices. In the future we plan to develop AE forecast models using recurrent networks. This would then be similar to the work by Gleisner (2001), where they used Elman recurrent networks. We did train using the LSTM (long-short term memory) network, and the results where on par with the results here, at least for the 5 minute data. However, we did not run the model with missing values, but in future studies we will train the LSTM model both with missing values and using a masking layer, to exclude missing values from the calculations. Additionally, NARX and other methods and algorithms should be considered.

As pointed out earlier, all input parameters, except the time parameters, are skewed. This might be less optimal when training, although the network should be able to map the inputs to the output even with skewed data. Still, some claim that normal distributed data is the best choice for training neural networks. So far we rescaled the data to be in the range -1 to 1, but the data will then still be skewed. In future studies we will examine various data transforms, such as e.g. Box-Cox, and examine the performance.

In this study we used the train/validation/test split. Another approach, is then to use cross-validation, where the dataset is split into k-folds (e.g. k = 10). The algorithm is then trained on k -1 folds with one held back and tested on the held back fold. This is repeated 10 times, shifting the folds, using 10 different test sets. This approach assumes Independent Identically Distributed (i.i.d.) data. Normally, however, this is not suitable for time series. Instead, for time series, one can use a time series cross validator, or hv-cross validation. We will apply these techniques in later studies.

Although we have developed forecast models for e.g. the peak AE values, within 30 minutes, it is fairly simple to retrain using another temporal resolution if needed. Another approach is to forecast the actual substorm onsets. This is however a classification problem, so careful selection of the substorm onsets are required. We have already implemented a simple substorm onset algorithm, which in principle could be developed further and used to create a substorm onsets dataset. Another possibility, is to use Pi2 pulsations to determine the substorm onsets, using e.g. the Wp index (Nosé et al. 2012). Using Keras, we could train the network, this time as a classification problem, to forecast the substorm onsets.

One also have to remember that the indices themselves are constructed artificially, thereby introducing properties of the time series that are non-physical. As an example, the AE indices are supposed to be global indices, but it is clear from the previous analysis that they have both an UT and a seasonal dependence. In a strict sense, it may not be suitable to use the AE at all, as described in Kamide & Rostoker (2004), but rather to use only AU and AL.

The models will be further updated and tested in WP5 and finally implemented in WP7. The implementation will be done according to the same procedure as for Kp and Dst.

References

Ahn, B.-H. & Moon, G.-H. (2003), 'Seasonal and universal time variations of the au, al and dst indices', *Journal of the Korean Astronomical Society* **36**, S93–S99.

Box, G. E. P., Box, G. E. P. & Cox, D. R. (1964), 'An analysis of transformations', *Journal of the Royal Statistical Society. Series B* pp. 211–252.

Gleisner, H. (1997), 'Response of the auroral electrojets to the solar wind modeled with neural networks', *Journal of Geophysical Research* **102**(A7), 14269–14278.

Gleisner, H. (2001), 'Auroral electrojet predictions with dynamic neural networks', *Journal of Geophysical Research* **106**(A11), 24541–24549.

Jolliffe, I. T. & Stephenson, D. B. (2012), *Forecast verification: a practitioner's guide in atmospheric science*, John Wiley & Sons.

Kamide, Y. & Rostoker, G. (2004), 'What is the physical meaning of the ae index?', *Eos, Transactions American Geophysical Union* **85**(19), 188–192.

Kingma, D. P. & Ba, J. (2014), 'Adam: A method for stochastic optimization'.

Mayaud, P. (1980), *Derivation, Meaning, and Use of Geomagnetic Indices*, Geophysical monograph, American Geophysical Union.

Nosé, M., Iyemori, T., Wang, L., Hitchman, A., Matzka, J., Feller, M., Egdorf, S., Gilder, S., Kumasaka, N., Koga, K., Matsumoto, H., Koshiishi, H., Cifuentes-Nava, G., Curto, J. J., Segarra, A. & Çelik, C. (2012), 'Wp index: A new substorm index derived from high-resolution geomagnetic field data at low latitude', *Space Weather* **10**(8).

Vassiliadis, D., Klimas, A. J., Baker, D. N. & Roberts, D. A. (1995), 'A description of the solar wind-magnetosphere coupling based on nonlinear filters', *Journal of Geophysical Research: Space Physics* **100**(A3), 3495–3512.