Analysis of inclination measurement by means of artificial neural networks – A comparison of static and dynamic networks

L. Alkaiem, F. Keller, H. Sternberg
Department of Geomatics Engineering,
HafenCity University Hamburg, Ueberseeallee 16, D-20457 Hamburg, Germany

Abstract. The monitoring of buildings and the modeling of occurring deformations are main duties in the Engineering Geodesy. In this article a new procedure for describing (modeling) and forecasting deformations is presented and used, which based on the artificial neural network technology. This procedure has been proved effective especially for non-linear models and complex sensor systems as well as for sensor combinations.
In this article the inclinations are examined in the Geomatics metrology laboratory of the HafenCity University Hamburg. The data therefore have been recorded at two positions with the help of two inclinations sensors: one on a measurement pillar and the second on the inner wall. The building is located directly on the Elbe river thus it is considered that the inclinations are under the influence of two essential factors: tide of the Elbe river and temperature changes, respectively. Two models are discussed here: a static – and a dynamic model. The first mentioned one considers only the current conditions of the input and output value, whereas the second model also considers the previous state in addition to the current conditions by including the delays between influences and inclination reactions, detected through the cross-correlation. The addressed models are compared and the approaches to define the network structure are discussed.

Keywords. Deformations measurements, inclination sensor, artificial neural networks, dynamic model, static model

1 Introduction

The use of artificial neural networks (ANN) for deformation tasks has been moved into scientific interest during the last years, especially in the field of Geodesy and Geomatics. Here, Heine was one of the first who used ANN for a dynamic modeling of a landslide,J.-B.Miima used this methodology for system identification of a bridge structure and Böhm for the modeling of the deformation of a lock [Heine 1999, Miima 2002 and Böhm 2006]. For system identification [Neuner, 2012] gives a theoretical approach with ANN.
The strength of ANN, which can also be used for monitoring tasks, lies in their ability to model and to reproduce complex and physical not easy to describe connections through learning examples. Beyond these examples, they provide, by further event inputs, correctly predicted output. This ability of generalization is of prime importance for prediction tasks [Heunecke 2013].
However, with this method you will achieve good results only when certain criteria of the network structure and the learning parameters have been logically chosen (such as number of hidden neurons, learning algorithm, network topology respectively the connection type of the neurons etc.).
Because of their wide variety and their high flexibility, ANN can be met in different application fields. However, to achieve the best possible results with this method, it is necessary to understand the approach of ANN profoundly as well as the features and the optimization opportunities they offer. One question for this study was: How far can static models acquire and model temporal variations or delays in the learning data?

2 Theoretical principles

Artificial neural networks are information processing systems, which imitate the functioning of the human brain. They consist of a large number of simple, parallel working units, the so called neurons, which are spread over several layers in the network. They send each other information via given connections. These connections are weighted and due to the change of weight they will be adapted in a way that the network is able to provide the best solution for a problem at hand. This learning process is character-

peer JISDM 2016 Vienna reviewed

ized by the iterative modification of the variable weights, which store the knowledge of the network [Kruse, 2012].

A neural network has at least one input and output layer. Many applications need more covered layers to mark the transmission behavior between input and output value. Here one speaks of a behavior-model, which does not record physical correlations between input and output value but it learns by means of certain learning algorithms and of training examples being present in the deformation analysis in form of observation pair.

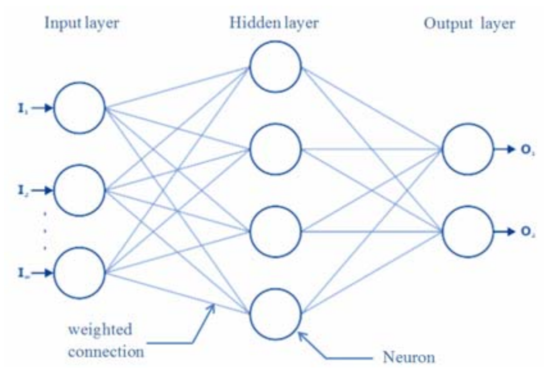


Fig. 1 The form of an artificial neural network. Typical schematic representation

ANN are characterized essentially by their abilities to learn and to generalize, their robustness against disorder and lack of data, their fault tolerance and their high performance based on parallel processing [Strecker 1997]. During the training period the network learns the present learning examples by adapting their parameter (connection weights and bias) to the examples. Here, an already defined learning algorithm indicates the exact method how the neuronal net carries out this adaption (Rey, 2010 p26). For this purpose an error function was defined, which determines a total error term F to all kinds of weight combinations. The strategy of the training is to search and to determine the minimum of the error function. Here, the learning algorithms use the gradient of the error surface, which indicates the direction of the steepest descent. This method is called ‘Method of Steepest Descent’. In this procedure the weight changes in the network take place on the basis of the errors or deviations between target data output and the calculated output of the output neurons [Haykin, 1999]. If the network has one or more hidden layers, then there is no longer the possibility to compare target- and actual output, because the neurons of the hidden layers have are no target output. In this case, the backpropagation process is used. It is also possible to determine the weight changes to the hidden neurons this way (Kruse, 2012). Despite of these attractive characteristics, however, the risk of overfitting exists in cases where ANN only store the data instead of learning from them. This means, the ability of the network to generalize has been compromised and the network is only able to reproduce the same random sample of the training but will fail with new data sets (in simple words, it is the case where: an ANN just “remembers” but “does not learn”) [Rey, 2010].

3 Measurement setup

In the HCU measurement laboratory, it was noticed that the measurement pillars are not stable. Because the stability of the laboratory is an essential requirement to perform all exercises and works of the Geomatics as well as for its reliability, the stability of the laboratory pillars were investigated, given that they are under the simultaneous influence of two essential factors: tide of the Elbe river and temperature. The result of this examination was that the two investigated pillars have indeed deformations (inclinations).

Two inclination sensors of the type Leica Nivel 20 were used for the monitoring. These sensors are able to dissolve a horizontal inclination of 0.001 mrad (mm/m). The accuracy of the sensors is about 0.010 mrad (mm/m) (sigma 95%).

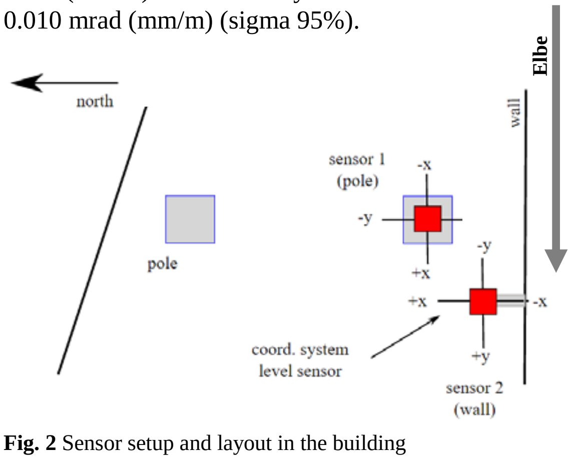


Fig. 2 Sensor setup and layout in the building

The two sensors were fixed in the laboratory. One sensor was placed on a stable mounting fixed on the wall of the measurement laboratory. According to the blueprints, this wall is the supporting part of the northern building and extends up to the foundation. The second sensor is fixed on one of the measurement pillars (see Figure 2). The stability of the measurement pillars and the wall was examined with this measurement setup.

4 Provision of the learning data (data basis)

The presumably inherent biggest factors should be the water level of the Elbe river (tide) and the sun (temperature). Besides the sensor data of the Nivel 20 that was represented in the form of inclinations in the respective directions X and Y, the hourly values of temperature for Hamburg from the DWD (Deutscher Wetter Dienst) and the level data for the level St.Pauli from the HPA (Hamburg Port Authority) were used and outliers were removed manually. As common zero-point of all measurement data, the 01.05.2015 0:00 was chosen. The first aim was to synchronize the present data. The common time period, for which we have available synchronous data for temperature, level, Nivel_1 and Nivel_2, lies between 302 and 491 hours. The synchronized time series of the inclination in the X- and Y-direction for both sensors and of the two influence variables are represented in Figure 3. They result from an equidistantly scanning for the duration of an hour, comprise 189 data sets and cover a time period of about 8 days.

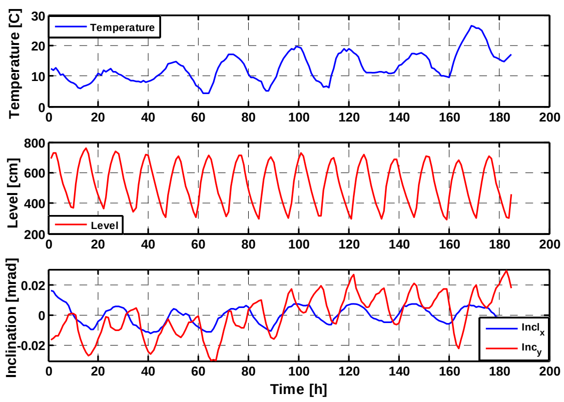


Fig. 3 The synchronized time series of temperature, level measurement and inclination, in X- (red) and Y-(blue) directions

5 ANN for inclination modeling

This model is a network with two input neurons (according to the influence variables temperature and level) and two output neurons (according to the inclination of each sensor in the X- and Y-direction). However, the topology varies depending on whether the static model is used or the dynamic.

5.1 The static model

This model considers only the current state of the input variables and assumes thus that the deformation reacts to the influence variables without hesitation. In this case, the network is a feed-forward-network with two input and two output neurons. It remains to find the number of hidden neurons, which are most suitable for this problem. The precise network structure will be discussed in more detail later on. The neurons of the hidden layer have a sigmoid tanh-activation function, while a linear activation function is used for output neurons. As long as reference data respective targets are present, this is supervised learning.

As already mentioned the learning and the training of ANN presents an iterative process of weight change. To improve the results of this process, the following approaches are possible:

- New initialization of the network parameters or multiple trainings.
- Successive increasing of the number of hidden neurons. This concern is of prime importance, when defining the network structure, and will be explained later in more detail.
- To train the network with different learning algorithms: In the process of this work, the network was trained with two learning functions - Levenberg-Marquardt (LM) and Backpropagation (BP). It should be mentioned at this point, that due to its very good convergence behavior, LM provides in principle the best results for fitting problems (non-linear regression) with a smaller ANN architecture and training data quantity.

The three above mentioned approaches were considered for training the network until the network has achieved the best possible generalization. The early-stopping-approach is used to generalize the network. The available learning data in the network is divided into three sets: training data, validation data and test data. The training data set is used to train the neural network through weight adaption. The errors of the output of the validation data set in relation to the training data set are recorded during the training phase and diminish at the beginning of the training. When the neural network tends to overfit, this error increases and the weights are determined on the basis of the minimum error. The test data is not part of the training process - it is only used to compare different models [Peters 2012]. The data was divided like this: 70% training data, 20% validation data and 10% test data. There are no hard rules for data division. It depends on the complexity of the problem and on the amount and nature of the learning data (much or less noise…). However, there is no clear connection between the data division and the network performance. But 20% for validation data is recommended in certain literature. The rest should be share like 70 % for training and 30 % for testing [Shahin, 2004]. So, 20 % for validation data presents a compromise between the representative nature of the learning data and the deficit in the training data. In the case of small number of learning data, like in our case, the training data should not be subtracted very much, this is why we have chosen 70 % for training data. The motivation here is to validate (test) the model with another data set which differs from the data set used to train or estimate the network. The most important indicator for the quality of the network generalization is its mean square error (mse) (in Matlab performance).

5.1.1 *Cross-validation as approach to determine network structure*

To determine the suitable network structure, it is necessary to define the number of hidden neurons. This quantity has major influence on the performance potential of the network and should be chosen therefore very carefully [Bishop, 2005]. An increase in the number of neurons in the hidden layer raises the flexibility of the neural network. But this needs more calculation and it is more vulnerable to overfitting, unless different neural networks are trained with various, successively increased, numbers of hidden neurons. The network structure is defined by the number of neurons which leads to the smallest value with regard to the validation data. In this context it is essential, to carry out two test series with two different learning algorithms whereby in every test the amount of hidden neurons were increased gradually. (3, 5, 10, 15, 20 and 30 hidden neurons).

A test with Levenberg Marquardt training function [LM]:

In general, (LM) is the fastest learning function because of their very good convergence behavior and provides the best results for fitting problems (nonlinear regression) in case of a small architecture of the artificial neural networks. This method is particularly effective to avoid the problem of skipping over the global minimum because the curve of the error surface, represented by Hessian-matrix, is considered beyond the gradient [Kisi, 2005].

A test with backpropagation-trainings function [BP] with momentum term (also called scaled conjugated gradient backpropagation):

This method extends the normal backpropagation process for a so called momentum term, which is added to the current gradient. This allows the weight chance depending on the error minimization of the prior steps. This way, the process converges faster [Moller, 1993].

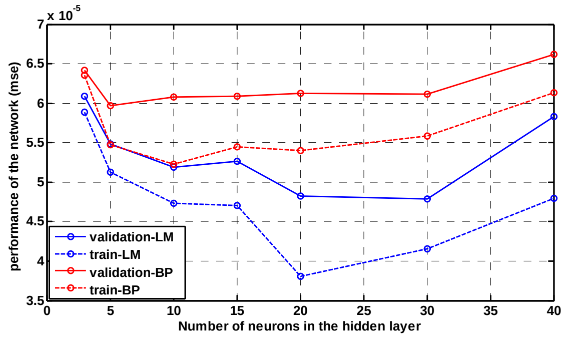


Fig. 4 Determining the number of hidden neurons according to the cross-validation method

Figure 4 shows that the successively increased number of hidden neurons over 10 have a negative impact on the network performance (mse) when the network is trained with BP-algorithm.

While training with LM-algorithm, when the numbers of hidden neurons increases up to 20, the mean square deviation (performance), which is calculated with the training data set, diminishes. From a quantity of 20 hidden neurons an up however, it increases again. The performance curve of the validation data achieves the minimum at 20 or 30 hidden neurons. To keep the complexity of the network within limits and in order to minimize the risk of overfitting, the network with 20 hidden neurons and

LM-learning algorithm was finally chosen. Figure 5 shows the structure of the selected network.

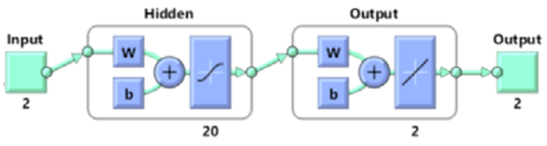


Fig. 5 Structure of the chosen network

5.1.2 *Selection criteria for the most appropriate network*

Every knot in the graphic of Figure 4 arises from multiple trainings of the network. Here, attempts were made to get a most similar course of the performance curves of training-, validation- and test-data. When all three curves are similarly formed, this means that the network responds similarly to learning data as well as to the validation- and test-data [Beale, 2014]. The probability of overfitting is thus smaller but not excluded. Figure 6 shows the performance curves of the chosen network with 20 hidden neurons and LM-learning algorithm.

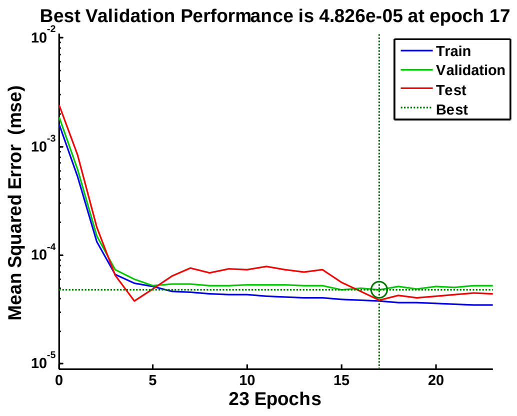


Fig. 6 Performance-plot of the chosen network

This performance plot shows no noticeable problems. The validation- and the test curves do not indicate overfitting. The training curve diminishes more than the validation curve and this way, represents that, the performance of the trained network with learning data is better than with the data not involved in the learning process.

A further possibility to evaluate the generalization of a network is a regression analysis between the network output and the respective targets, which typically is represented by a regression factor (correlation coefficient R). With a perfect network performance R should be one (1.0) but in reality this is not the case. Figure 7 shows the regression plot with R=0.7.

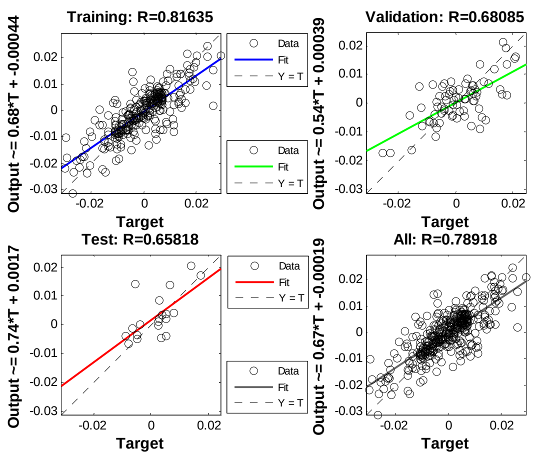


Fig. 7 Regression plots of the chosen network

More information for evaluating the quality of the trained network is available from the error-histogram which shows the distribution of the residuals between targets and network output. This histogram is able to indicate outliers. In this case, it is shown that most errors lie between -0.001 and 0.001. However, there is a learning point with an error of 0.023. Based on the present outliers, it makes sense to check if the quality of the learning data is bad or if these outliers are just different from the rest data.

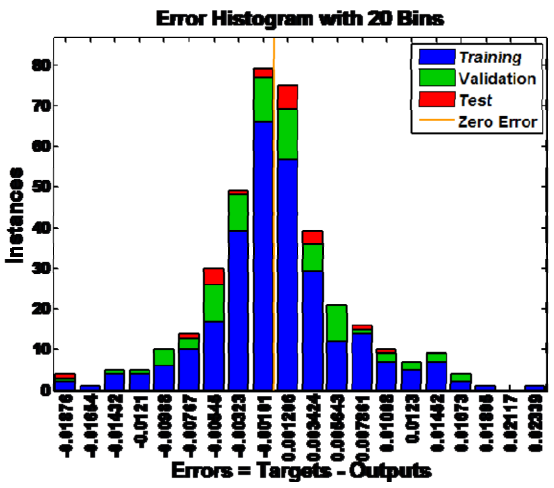


Fig. 8: Error histogram of the chosen network

5.1.3 *Results of the static model*

The results of modeling are presented in Figure 9 in the form of residuals between, the observed output values and the output values calculated in the trained network.

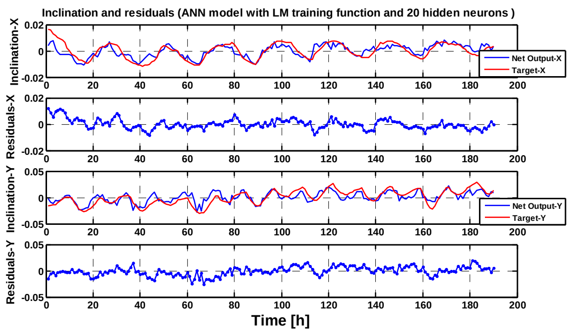


Fig. 9 Static model: Inclination of the sensors and residuals

The graphic analysis of the results allows the statement that the static modeling of the deformation model, to be processed with the help of neural networks within this work, do not provide reliable results. The outcome can be confirmed through the calculation of the standard deviation of the residuals. In the X- and Y-direction these are 0.0037, 0.0082 mrad, respectively. This can probably be attributed to the fact that the deformation or inclination is affected by other influencing variables. In addition, it is possible that the nature of the learning data, which contains no unique (or very similar) relationships between input and output variables, can lead to confusion within the network during the training. However, the most important reason is a possible response delay of the sensor inclination, an issue that should be noted about the cross-correlation of the input and output variables. In this case an upgrade to a dynamic model is required, in which this delay is taken into account.

5.2 The dynamic model

The basic idea of the dynamic modeling is that the state of the deformation at the time (t) is depending on the previous states of the input variables (temperature and level) and (possibly), of the output variables itself.

To investigate the dependence between the external influences (i.e. tide and temperature) on the position of the measuring pillar in the geodetic laboratory, the cross-correlation between the respective influence values and the respective axes of the sensor was calculated. The results are presented in Figure 10 and Figure 11.

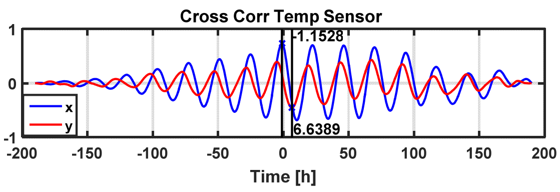


Fig. 10 Cross-correlation between temperature and sensor

The measurement pillar responds clearly to the temperature change. With a correlation coefficient of 0.73 a clear correlation can be observed in X. On the other hand, with a correlation coefficient of 0.39, a far more moderate one can be observed in Y. The reaction starts about one hour and ten minutes later. Presumably, the storage of the pillar and the elastomer beneath are responsible. This responds to the temperature change and these changes contributes to deformations.

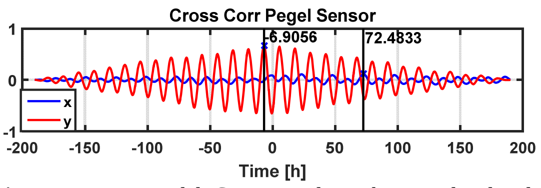


Fig. 11 Dynamic model: Cross-correlation between level and sensor

A clear correlation can be detected transversely to the flow direction of the river. The correlation is on the wall -0.75 and on the pillar 0.67 (Figure 11). The time-shift is approximately 7 hours. The inclination is of maximum gradient after 7 hours of level peak.

This paper deals with the modeling of the deformation, so that out of known measurements of influencing variables (temperature and level), the corresponding inclinations of the sensors should emerge (prediction). Therefore, the dynamic model is different from the static model, both in number and type of input values. That means that the input of the network are no longer only T(t) and P(t), but also T(t-1)… T(t-k1) and P(t-1)…P(t-k2), whereas the parameters k1, k2, represent the response delay of the system on the influencing factors. Thus, the parameters represent the temporal shift (offset) of the inclination towards the temperature and level. In this case the network has the following structure:

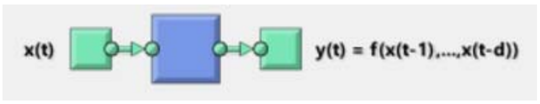


Fig. 12 Dynamic network: General structure

The used scenario of the dynamic model in this work can be described as follows:

Input **Output**

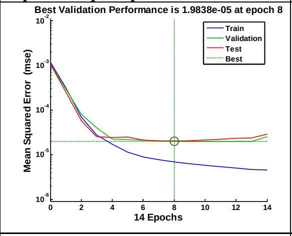
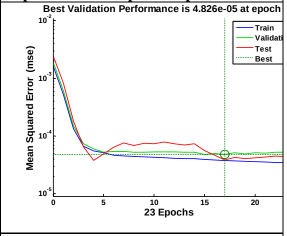
$$\{T(t-k), P(t-k), T(t), P(t)\} \longrightarrow \{N(t)\}, \quad k - 1{:}d, \quad d - \text{delay}$$

Based on the reaction delay of the outputs on the input variables (observed in the correlation investigation procedure), it is possible to assign the values 1-7 to the factor k. Should the model be described in more detail, two factors of delay for each (temperature and level) ought to be considered then. To simplify matters, only the largest delay, specifically the 7 hours (delay towards level), was taken.

The procedure for determining the network structure or number of hidden neurons, the learning algorithm etc. is exactly the same as the one followed for the static model. The procedure is not repeated here. Instead, the same configuration, with the network that was chosen for the static model, should be used for comparison purposes, hence with 20 hidden neurons and LM-learning algorithm.

Normally the response delay has to be eliminated before working with ANNs [Heunecke 2013, chapter 14.2 and 14.4], for instance with a linear model and time series analysis. The residuals from the linear model can then be argument for ANNs.

In the following, the results of the two models are compared:

Dynamic model **Spatial-temporal prediction**	**Static model** **Spatial-time independent prediction**
	
Validation-performance: 1.98 e-5 Training-performance: 0.97	Validation-performance: 4.8 e-5 Training-performance: 3.81 e-05

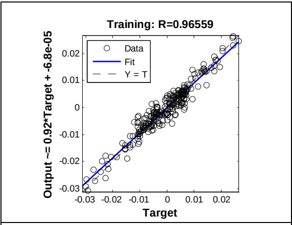
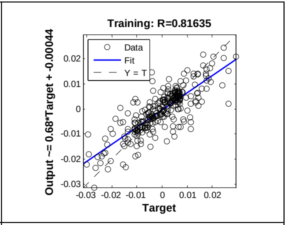
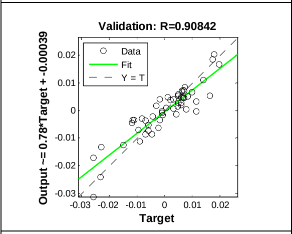
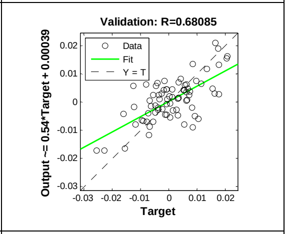
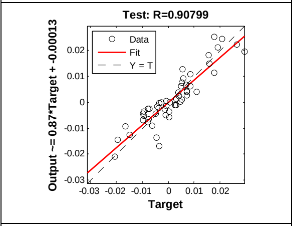
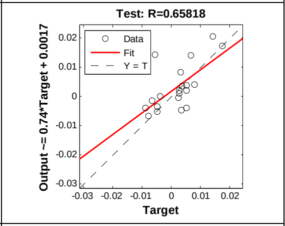
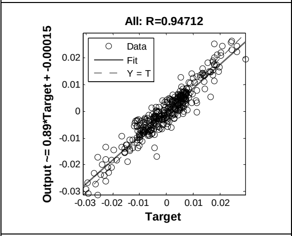
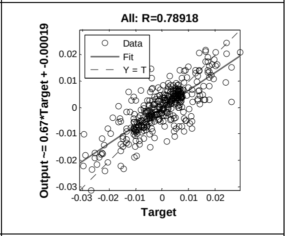
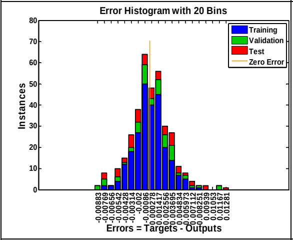
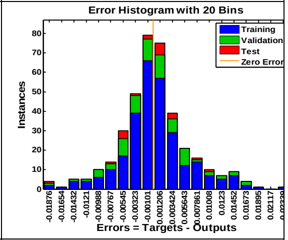
	
	
	
	
Very good regression of ca. 0.94 for the entire data set. Good regression for validation- and test data of ca. 0.9 each.	Poor regression of 0.78 for the entire data set. Very poor regression in validation- and test data of 0.68 and 0.65.
	
Coverage of the residuals between targets and network output= [-0.00883 _ 0.01281] i.e. lower deviation p.s.both graphics are standardized, so that the relative difference can be recognized.	Coverage of the residuals between targets and netork output= [-0.01876 _ 0.0339]
Standard deviation of the residuals: Direction X= 0.0026 Direction Y= 0.0038 Particularly in the Y-direction the dynamic model indicates considerable improvement. The standard deviation in this case is less than half compared with the static model.	Standard deviation of the residuals: Direction X= 0.0037 Direction Y= 0.0082

Fig. 13 Comparison of the two models

The results of the dynamic modeling are presented in Figure 14, in form of residuals between the observed output values and the output values calculated in the trained network. The residuals derived from the dynamic model are obviously smaller than those from the static model, presented in Figure 9. This corresponds to the grown expectation caused by the analysis of performance plot, regression plot and error histogram.

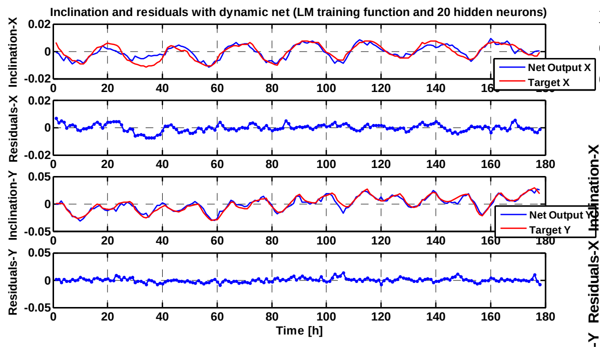
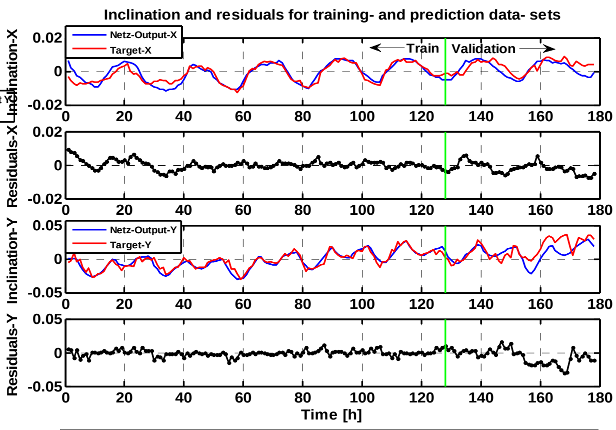


Fig. 14 Dynamic model: Inclinations of the sensor and residuals

5.3. Prediction

To check the validity of the trained network in the dynamic model for future determination of the inclination of the sensor due to the present values of temperature and level (i.e. to check the prediction ability of the network), the learning data were divided into two sub-sets: approximately 70% of the learning data should be used to train the network (or to optimize the model), and the rest will be used for independent validation (or for the network verification). Here is to differentiate between this verification part of 30% and the validation data-set, which is used to stop the learning of the network. Figure 15 shows the results in two parts: on the left, starting on the green line, the results are based on training data; on the right, the results are based on the remaining data whose corresponding inclination is to be determined.

In relation to the residuals of the respective parts, it is clear that the network performance concerning the verification data-set is significantly worse than the one concerning the training data-set. With regard to the relating performance plot of the trained network, which is represented on the above part in Figure 16, this result is expected. For the blue curve, representing the training performance, it drops extremely compared to the green curve (validation) and the red curve (test). This means, when new data is added to the network, it is expected that there won't be results of the same quality as in the case of the learning data. This is confirmed by the regression plot on the beneath part in Figure 16. Here, it is illustrated that the regression factor for the training data is significantly better than for the test- and validation data, as well.



TRAIN	VALIDATION
Std-X= 0.0022	Std-X= 0.0033
Std-Y= 0.0044	Std-Y= 0.0108

Fig. 15 Dynamic model: Inclination and Residuals for training and prediction datasets

This may lead to the interpretation that further influences or long-period changes and effects have not been included in the modeling. The used dynamic model includes as input values, for example, only information about the influence values (temperature and level) and no information about the state of the object respectively the inclination of the sensors, which could reflect this long-period effect.

But on the other hand, they are more difficult to train since they have to learn sequential, temporal varying patterns. Another important result of this study is that the use of the same gradient-based learning algorithm to train a static and a dynamic network, with exactly the same learning parameters, can result into completely different performances. This can be explained by the fact that the calculation of the gradient is much more complex for dynamic networks, because the network-output (at a specific time-moment) is not only depending on the input of this time but also, on the progress of the sequence input. Furthermore, for networks with feedback connection, there is also dependence on the progress of the outputs.

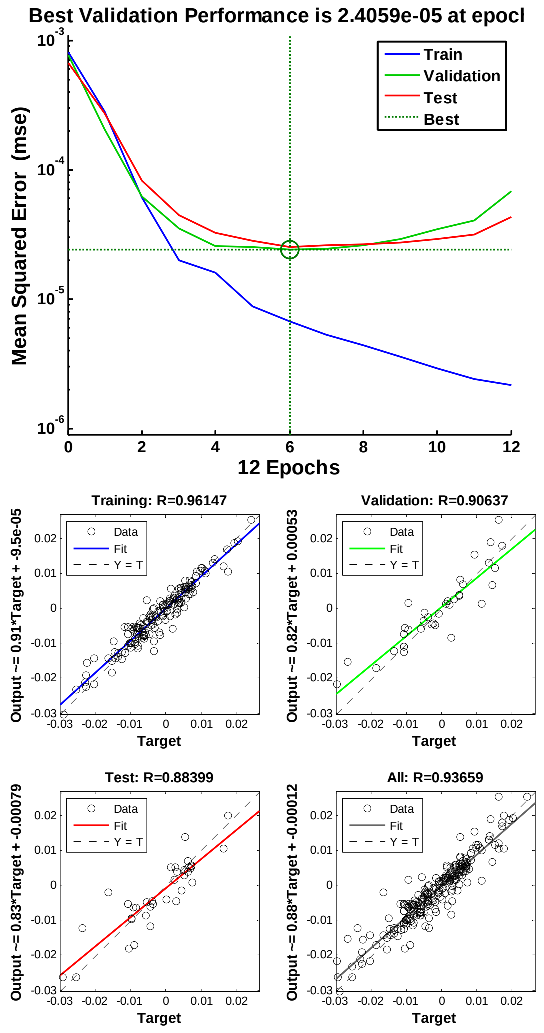


Fig. 16 Above part: performance plot of the trained network; Beneath part: regression plot

6. Conclusion and Outlook

In this article, the static- and the dynamic modeling of inclinations through ANN-modeling have been presented. The performance of the two tested models has been compared and the more efficient model, namely the dynamic model, was examined for the ability of the prediction. In this process, no new data was provided, but the available learning data was divided in two sets: a training-set and a verifying-set respectively (validation part). Examinations within the scoop of this contribution have shown that dynamic neural networks normally are more efficient and capable in comparison with the static neural networks because they have more parameters for adaption due to temporal series inserted into the model. Consequently, they are more flexible and, because of their increased degree of complexity, they are more realistic in modeling the context between network's input and output. The investigation of the performance of static and dynamic neuronal networks for delay-models reveals that the dynamic networks provide significant better results. Although the results of the static network aren't poor, the adaptivity of the dynamic network is superior in such cases. Therefore, it is recommended to define in advance which connections exist between input- and output value respectively if there are temporal correlations.

References

Beale, M.; Hagan M.; Demuth H. (2014). Neural Network Toolbox™ User's Guide, R2014a, http://de.mathworks.com/ (last time accessed: 12/04/2015)

Bishop, CM. (2005). Neural networks for pattern recognition. Department of Computer Science and Applied Mathematics Aston University Birmingham, Oxford Press 2005, UK

Böhm, S.; Kutterer, H. (2006). Modeling the Deformations of a Lock by Means of Neuro-Fuzzy Techniques, Shaping the Change XXIII FIG Congress Munich, Germany, October 8-13, 2006.

Haykin, S. (1999). Neural Networks: A Comprehensive Foundation. 2nd Edition. Prentice Hall. 1999, ISBN 0-13-908385-5.

Heine, K. (1999): Beschreibung von Deformationsprozessen durch VOLTERRA- und FUZZY-Modelle sowie Neuronale Netze. DGK-C, 516. ISBN 3-7696-9554-2. Handbuch Ingenieurgeodäsie, 2. Auflage (Wichman; ISBN 978-3-87907-467-9.

Heunecke, O. ; Kuhlmann, H. ; Welsch, W. ; Eichhorn, A. ; Neuner, H. (2013). Auswertung geodätischer Überwachungsmessungen, Handbuch Ingenieurgeodäsie, 2. Auflage (Wichman; ISBN 978-3-87907-467-9, E-Book: ISBN 978-3-87907-562-1. S 410-485

Kisi, O.; Uncuoghlu, E. (2005): Comparison of three back-propagation training algorithms for two case studies," Indian Journal of Engineering & Materials Sciences, Vol. 12, October 2005, page 434-442

Kruse, R.; Borgelt, Ch.; Klawonn, F.; Moewes, Ch.; Ruß, G.; Steinbrecher, M. (2012). Computational Intelligence:

Eine methodische Einführung in Künstliche Neuronale Netze, Evolutionäre Algorithmen, Fuzzy-Systeme und Bayes-Netze. Wiesbaden 2012. ISBN: 978-3-8348-8299-8

Neuner, H. (2012). Model selection for system identification by means of artificial neural networks, In: Journal of Applied Geodesy. Volume 6, Issue 3-4, pages 117–124, ISSN (Online) 1862-9024, ISSN (Print) 1862-9016, November 2012.

Miima, J.-B. (2002): Artificial Neural Networks and Fuzzy Logic Techniques for the Reconstruction of Structural Deformations. TU Braunschweig, 18. Handbuch Ingenieurgeodäsie, 2. Auflage (Wichman; ISBN 978-3-87907-467-9).

Moller, M.F. (1993): A scaled conjugate gradient algorithm for fast supervised learning, Neural Networks.

Peters, T. (2012). Ableitung einer Beziehung zwischen der Radarreflektivität, der Niederschlagsrate und weiteren aus Radardaten abgeleiteten Parametern unter Verwendung von Methoden der multivariaten Statistik. Wissenschaftliche Berichte des Instituts für Meteorologie und Klimaforschung des Karlsruher Instituts für Technologie Band 49, ISBN 978-3-86644-323-5].

Rey, GD.; Wender, KF. (2010). Neuronale Netze Eine Einführung in die Grundlagen, Anwendungen und Datenauswertung, 2.; vollständig überarbeitete und erweiterte Auflage, Verlag Hans Huber, Hogrefe AG, Bern 2010.

Shahin, M., Maier, H., Jaksa, M. (2004). "Data Division for Developing Neural Networks Applied to Geotechnical Engineering."J. Comput. Civ. Eng., 10.1061/(ASCE) 0887-3801(2004)18:2(105), 105-114.

Strecker, S. (1997). Künstliche Neuronale Netze – Aufbau und Funktionsweise, in: Arbeitspapiere WI, Nr. 10/1997, Hrsg.: Lehrstuhl für Allg. BWL und Wirtschaftsinformatik, Johannes Gutenberg-Universität: Mainz 1997.