Adapting neural networks for modelling structural behavior in geodetic deformation monitoring

John Bosco Miima and Wolfgang Niemeier

Summary
Artificial neural networks are adapted for use in modelling geodetic deformations. Using temperature, pressure, humidity, water-level variations, and traffic volume as input signals (forces) and respective point component displacements (deformations) as output signals, the »input-output« behavior of points on a deforming bridge has been modelled. The results indicate an acceptable relational representation of the deformation process of the points. It is shown that, based on the determined models, the deformation behavior of the bridge from 0 to 180 days may be predicted reasonably well.

Zusammenfassung
Mit Hilfe von Künstlichen Neuronalen Netzen lassen sich Bauwerksdeformationen aus geodätischen Überwachungsmessungen modellieren. Für die Modellierung des Deformationsverhaltens eines Brückenbauwerks wurden die meteorologischen Randbedingungen (Lufttemperatur, Luftdruck, rel. Luftfeuchte), der Wasserstand des überquerten Flusses und die Verkehrsauflast am Bauwerk als Systemeingang definiert. Der Systemausgang ist die gleichzeitige geometrische Bauwerksverformung in den drei Koordinatenkomponenten. Die Werte der Gewichtsmatrix des Künstlichen Neuronalen Netzes stellen eine Repräsentation der wirkenden Systembeziehungen des Deformationsprozesses dar. Es lässt sich zeigen, dass ein derart bestimmtes Modell in der Lage ist, das Bauwerksverhalten aus den Umgebungsparametern bis zu 180 Tagen glaubhaft zu prädizieren.

1 Introduction

It has long been a problem to geodesists to find efficient solutions to approximate functions that define geodetic deformations, especially when dealing with continuously monitored processes. Most solutions are derived in the time domain as nowadays measurements are obtained on line as either continuous or discrete time sequences. A deforming object can be considered as a dynamic system (Pfeufer 1990, Welsch 1996, Heunecke and Pelzer 1998) whereby, forces acting on the object (both internal and external loads) are regarded as input signals that lead to geometrical changes e. g. displacements and distortions as output signals. Both inputs and outputs may be obtained as time sequences.

A mathematical function to define a real-world dynamic system, e. g. as exemplified by deformation processes in geodesy, can be very complex and its exact form is usually never known such that in practice, modelling such a process must be based upon some chosen model of known functions. In the recent years, an abundance of authors has outlined the use of different models for describing deformation processes in geodesy: Felgendreher (1982) outlines the use of Correlation analysis, Spectral analysis, Fourier analysis, and Convolution integral as tools for modelling dynamic processes while Pfeufer (1990) highlights the use of the *Volterra* functional series to model dynamic systems in geodesy. The use of both artificial neural networks and fuzzy logic for system description and identification is presented in Heine (1999). Chrzanowski and Yong (1996) give a similarity transformation and piecewise approximation method for identifying deformation in space and time domain.

It is however generally agreed upon that there exists no single method that can satisfactorily describe structural deformation as its underlying processes are normally too complex to be expressed by one simple expression. The present study is concerned with modelling geodetic deformation processes as dynamic systems by adopting the use of neural networks. Artificial neural networks are inspired by biological systems in which large numbers of neurons, which individually function rather slowly and imperfectly, collectively perform extraordinarily complex computations that even the fastest computers may not match. This field is among the most rapidly developing scientific research today and is interdisciplinary in nature. Its applications include speech and image recognition, linear and non linear optimization, and automatic control.

2 Structural deformation as a dynamic system

As stated above, a dynamic system is characterized by input forces or signals; these include all possible forces perceived as acting on an object causing it to deform i. e. changes in the nature and geometry of the structure. Resulting deformations are then considered as output signals as shown in Fig. 1. For the study presented herein, data sets from an ongoing bridge deformation monitoring project (Niemeier et al. 2000) at the institute of geodesy and photogrammetry TU-Braunschweig are used. It is assumed that meteorological factors pressure, humidity and temperature coupled with variations in the water levels of the river carrying the bridge and traffic volume are the main forces acting on the bridge resulting in its deformation measurable geodetically as displacements as depicted in the figure below.

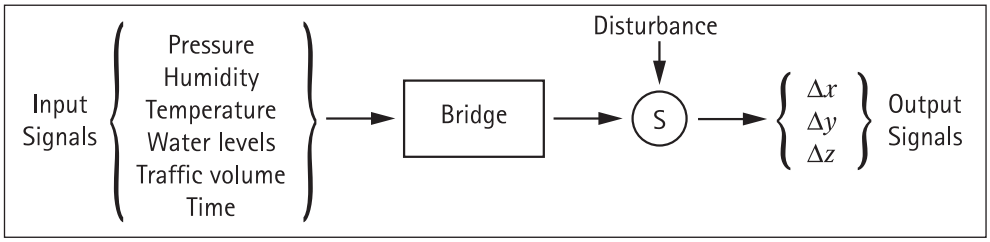

Abb. 1: Schematic representation of the bridge as a system

A distinction is made between three main classes of models that may be used to model the behavior of such a system; *parametric* or *white-box*, *grey* and *non-parametric* or *black-box* models (Welsch and Heunecke 2000). White-box models are used in modelling well understood processes, which involves derivation of mathematical equations describing the process based on first principles of physics. Modelling unknown parameters based on a model structure chosen *a priori* or partially motivated by physical analysis falls under the grey-box class while black-box models outline experimental modelling to identify parameters (even though the name suggests the absence of such parameters!) that intrinsically define the functional relationship between inputs and outputs for a given process. Artificial neural networks are considered black-box models as they can map any input domain to any given output domain. They therefore lead themselves useful for modelling the deformation behavior of objects given the corresponding input and output signals. The adaptation of neural networks for dynamic modelling is discussed in the following sections.

3 Artificial Neural Networks (ANNs)

Biological systems perform extraordinarily complex computations in the real world without recourse to explicit quantitative operations. Organisms are able to learn a task gradually over time, this learning property being a reflection of the ability of large ensembles of neurons to learn through exposure to external stimuli and to generalize across related instances of a given signal. Such properties of biological neural systems make them attractive as a model for computational methods designed to process complex data in a flexible manner, that is relatively independent of the task defined.

Artificial Neural Networks are simplified simulation models of the biological nervous system. Just like the biological neuron represents the elementary unit of a biological nervous system, an artificial neuron, Fig. 2, represents the basic unit of an artificial neural network. The function of artificial neurons is similar to that of real neurons: they integrate input from other neurons and communicate the integrated signal to a decision making center. This is summarized mathematically as

$$y_i = f(a) = \frac{1}{1+\exp(-\beta a_i)} \qquad (3\text{-}1)$$

with

$$a_i = \sum_1^n w_{ij} x_j + b_i \qquad (3\text{-}2)$$

where y_i is the activity output of neuron i, a_i is the weighted sum of the neuron i from the input of the last neurons, x_j is the input from the neuron j, w_{ij} is the weight between two neurons i and j, n is the number of input units. b_i is the bias, which is an external parameter of the neuron i and constant β is threshold value which shifts the activation function $f(a)$ along the x axis. An activation function is a nonlinear function that, when applied to the net input of a neuron, determines the output of that neuron. There exists a wide range of functions used as activation functions which include the *sigmoidal* (used above), *hyperbolic tangent*, *Heaviside*, and *piecewise linear* functions among others (Nilson 1965).

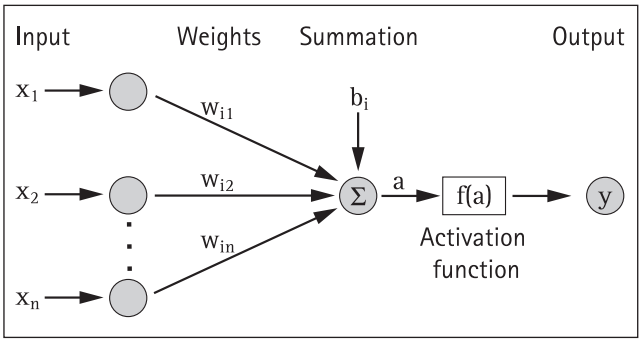

Abb. 2: Single artificial neuron

3.1 Multilayer networks

A neural network typically consists of many simple neurons operating in parallel. The neurons are connected through synaptic links, weights, and are grouped in layers. Three different layer types can be distinguished: input layer (the one to which external stimuli are applied to), output layer (the layer that outputs results), and one or more hidden layers (intermediate computational layers between input and output).

A class of neural networks, where the input feeds forward through the network layers to the output, is referred to as the multilayer feedforward network. This kind of network is known to be capable of learning complex input-output mappings (Cybenko 1989). That is, given a set of inputs and the desired outputs or targets, an ad-

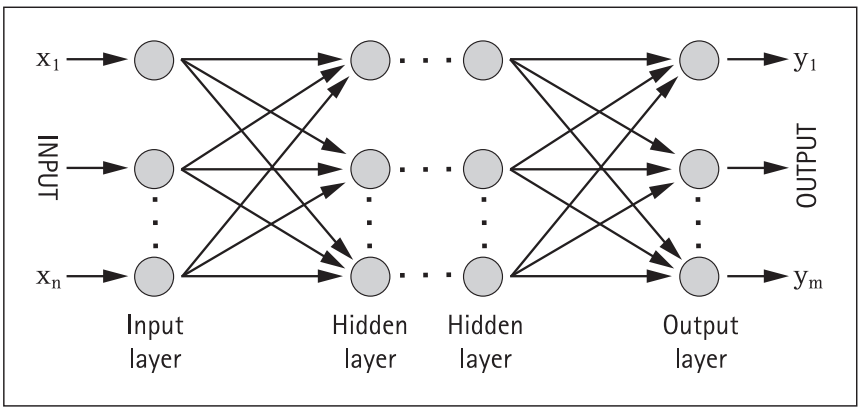

Abb. 3: A multilayer feedforward neural network

equately chosen neural network can emulate the mechanism which produced the data set through learning.

3.2 How learning is realized

Let the output of such a network be denoted by $y(t)$, where t denotes time. The input of the network be denoted by $x(t)$ and all the weights w and thresholds β of the network be ordered as a vector Θ. Assuming we are given a training set $d(t)$, $x(t)$; $t = 1, 2, \ldots, M$, where $d(t)$ is the desired output at time t and M the number of outputs of the neural network. The task of a learning algorithm is to estimate the unknown parameters in the network architecture, using the training set. Assuming that the architecture is fixed, i. e. the number of hidden layers and neurons. The only unknowns in the network are then weights (parameters) interconnecting the different neurons in different layers.

To perform this, we need to define an error criterion which describes the »closeness« between the network output $y(t)$ and the desired output $d(t)$. The quadratic error or cost function is commonly used:

$$J(\Theta) = \frac{1}{M}\sum_{i=1}^{M}[d(t) - y(t)]^2 \qquad (3\text{-}3)$$

where Θ as stated above is an n_Θ dimensional vector for the weights in the network. The estimated value of Θ can be obtained by minimizing the error criterion (3-3) with respect to the parameter Θ. Hence we need to evaluate the differential $\frac{\partial J}{\partial \Theta}$. The minimum value can be obtained by setting this differential to zero and solving it analytically. In practise, the minimum value of Θ is obtained numerically by using a simple first order gradient algorithm as is implemented in the back propagation learning algorithm (Rumelhart et al. 1986). The learning algorithm can be formulated as the value of the parameters which minimizes this cost function. This value of parameter can be obtained by expanding the cost function to a first order approximation as follows:

$$J(\Theta + \Delta\Theta) \approx J(\Theta) + (\Delta\Theta)^T J'(\Theta) \qquad (3\text{-}4)$$

where $\Delta\Theta$ denotes a perturbation from the nominal value Θ and J' is the differential of $J\Theta$ with respect to Θ. This perturbation must be small, otherwise, the approximation does not hold. If we let

$$\Delta\Theta = -\eta J'(\Theta) \qquad (3\text{-}5)$$

where $\eta \geq 0$, then, we have

$$J(\Theta + \Delta\Theta) \approx J(\Theta) - \eta(J'(\Theta))^T J'(\Theta) \qquad (3\text{-}6)$$

which is non-increasing, as $\eta \geq 0$. η is referred to as the learning parameter. Then, parameter vector Θ can be updated as follows:

$$\Theta_k = \alpha\Theta_{k-1} - \eta f \frac{\partial J}{\partial \Theta} \mid \Theta = \Theta_{k-1} \qquad (3\text{-}7)$$

where Θ_k denotes the estimated value of Θ at the k-th iteration. α is an arbitrary positive constant, usually chosen between 0.1 and 0.8 called momentum term, which ensures a smooth change in Θ to help speed up convergence (Hertz et al. 1991). Many other learning algorithms have been developed over time, either to solve the learning problem in multilayer feedforward networks e. g. the *Gauss-Newton* and the *Levenberg-Marquardt* algorithms or to propose different neural structures from the multilayer feedforward network e. g. *Radial basis functions* and *Bolzmann* networks. More details can be found in Bishop (1995).

3.3 Approximating structural behavior with neural networks

A detailed study on the deformation of the Fallersleber Tor bridge in Braunschweig has been carried out with a view to explaining its deformation (Niemeier et al. 2000). A multilayer feedforward network using the backpropagation algorithm was used to model the bridge behavior as a dynamic input-output system. The underlying processing may be summarized in four major steps as follows:

1. Data collection and preprocessing,
2. Identification of the network architecture,

3. Parameters estimation using training data subset and validation using second data subset,
4. Prediction of the future behavior.

Step 1

In the above mentioned project, forces assumed to be acting on the bridge i. e. pressure, humidity, temperature, river water level variations, and traffic volume are measured thrice a day at 5 a. m., 11 a. m. and 11 p. m. concurrent with terrestrial measurements to 180 points attached on the bridge abutments. The geodetic measurements are adjusted and filtered to yield point components to an accuracy of 0.75 mm. Time sequences resulting from the 5 a. m. measurements have been used in this study, since these measurements are taken at a time of least activity on the bridge thus providing data sets devoid of direct external influences. Measurements are continuously undertaken since June 1999. However, the data sets used in this study cover a $2\,^1/_2$ year period (between June 1999 and January 2002) resulting in 924 epochs of observations for inputs (forces) and outputs (point positional changes derived from the terrestrial measurements) which have been effectively used to determine an input-output relationship of the structure over this time.

Tab. 1: Data subsets

Sample	Training set	Validation set	Testing set
1	779	100	45
2	724	100	100
3	644	100	180

Due to logistic problems in presenting results for every point, only one generalized vector representing each component for either of the central parts of the eastern and western abutments is presented. The vector was determined by getting the average time sequence for the respective components over the time period, a concept similar to the *stacking* process in geophysics. The averaged time sequence for each of the two abutments was taken and separated into three subsets for training, validation and testing the ANN. With a view to determining whether there would be any limitation to the window of prediction, the same data sets were split into three samples as shown in Tab. 1. Sample 1 gives the results of predicting 45 epochs or days into the future while samples 2 and 3 offer us the chance to undertake predictions for 100 and 180 days, respectively.

Step 2

While the number of input and output neurons is determined by the number of input and output forces respectively, finding an optimal number of hidden layers and their neurons is not a clear cut process. One has to strike a reasonable compromise between having layers and neurons enough to learn and represent a given system while at the same time ensuring that they are not so many to overburden computational resources during the learning process. A few authors e. g. Widrow et al. (1987) and Cybenko (1989) suggest possible ways of archiving this task. However, for the work presented here, the number of hidden layers and neurons was determined based on intuition and a trial-and-error procedure. The resulting network architecture constituted an input layer with five neurons (one for each input force), three hidden layers with eight neurons each, and an output layer of three neurons for the signals of changes in the three computed coordinate components Δx, Δy, and Δz respectively. The modelling for the two abutments was then done separately using similar inputs (acting forces) and the respective computed coordinate components in two networks of the same architecture, as described above.

Step 3

The software NeuroSolutions (NeuroDimensons 2000) was used in this study. This software offers a unique capability of icon-based construction of neural network simulations. The hardware consisted of a normal IBM compatible PC with a 700 MHz processor running on Microsoft windows NT. The calculation time for one learning run was about two minutes. The general learning process was achieved by feeding the input and output data sets into the network and manually searching for the heuristically optimal network parameters, i. e. learning rate η, iterations k and the momentum term α in (3-7) by trials until the learning was successful. A network that is too complex may fit noise, not just the signal, leading to overfitting. Overfitting is especially dangerous because it can easily lead to predictions that are far beyond the range of the training data, which leads to poor generalization.

To ensure optimal learning and to minimize overtraining, a combination of criteria was used to terminate the back propagation learning process. First, the process was set to stop when the error criterion function in (3-3) crossed a threshold of 0.01 mm^2 between the expected and learned vectors. Secondly, a validation data set (see Tab. 1) drawn from the same population as the training data but not used in training is also fed in the network. Like the training set, its error reduces at the start of training. However, continuing the training past a certain point causes the validation set error to increase. This is a precursor for possible overtraining; it is thus advisable to terminate the training process just when the validation set error bottoms out. The example in Fig. 4 shows results from one of the runs used in the study, a typical behavior

Tab. 2: Average training errors

	Sample 1	Sample 2	Sample 3
Δx	0.016	0.019	0.010
Δy	0.015	0.025	0.036
Δz	0.018	0.020	0.028

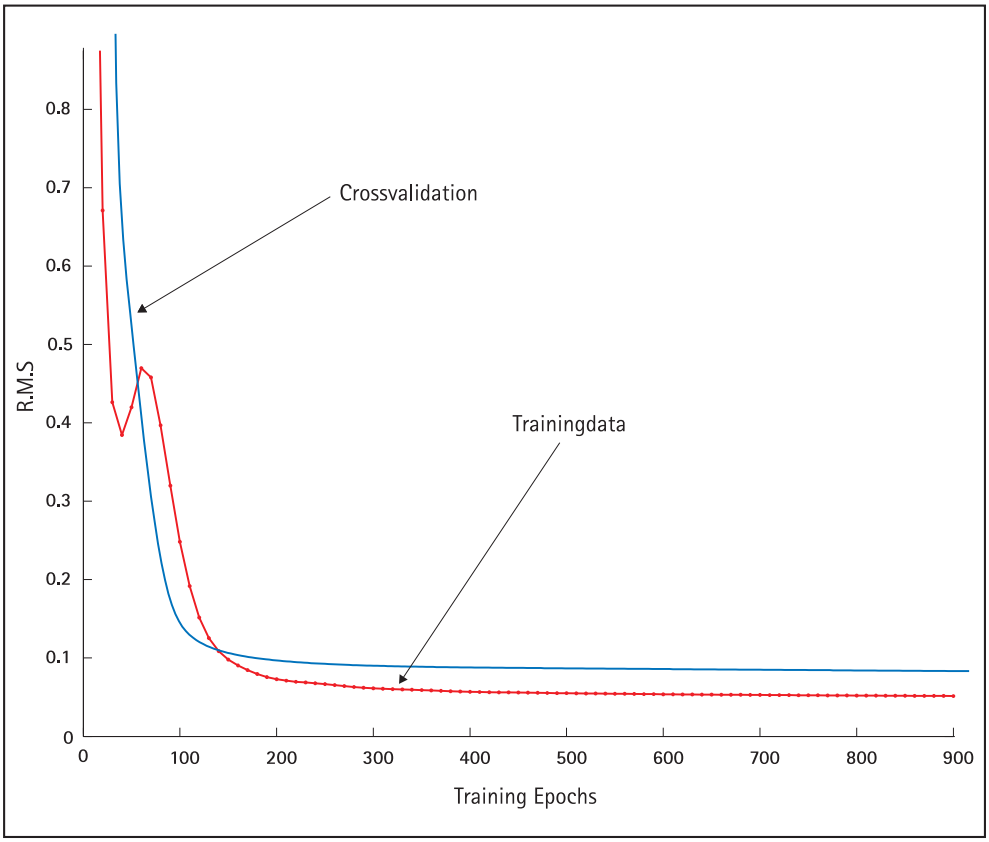

Abb. 4: Error development of a training and validation set in one of the runs

of the training and validation errors during a training process while Tab. 2 gives averages of the errors of training data sets during all the training epochs. Finally, the training was set to stop after a certain fixed number of iterations.

Step 4

After successful training, the resulting final weights Θ for each signal were then obtained as an intrinsic representation of the mapping function between the inputs and outputs for the respective abutments. The network structure was then frozen and its output to new signals determined as a measure of applicability for predicting the structural behavior based on the »knowledge« acquired from past events. To validate the modelling process, a residual analysis of the modelling and prediction errors was performed. Residual analysis comprised of the computation of

a) the *error mean*

$$\mu = \frac{1}{n}\sum_{t=1}^{n}(d(t)-y(t)), \tag{3-8}$$

and

b) the *coefficient of determination*

$$r^2 = 1-\frac{\sum_{t=1}^{n}(d(t)-y(t))^2}{\sum_{t=1}^{n}(d(t)-\bar{d}(t))^2} \tag{3-9}$$

as quantitative means of evaluating the modelling and prediction processes (Chatfield 1975), where $d(t)$, $y(t)$ denote the desired and modelled or predicted values, respectively, while $\bar{d}(t)$ denotes the mean of the desired data. The error mean μ is a measure of just how close the modelled values are to the desired valued with a value close to zero as most desirable. The coefficient of determination r^2, which is a function of the mean squared error normalized by the variance of the desired data, is the most important parameter for evaluating modelling and prediction accuracy. For a perfect model or predictor, the coefficient of prediction should be 1.

4 Sample results

Results for the western abutment are presented in Fig. 5, 6, and 7. The figures are combinations of the three data sets outlined in Tab. 1 above: training, validation, and prediction sets. To ensure clarity of presentation, only the last 300 epochs of the data sets have been plotted, see Miima (2002) for complete plots. Error means and coefficients of determination for the training and prediction results are summarized in Tab. 3. These together with the graphics should give a broader picture of the capabilities of ANNs.

Fig. 5 shows the results of a network trained using 779 epochs, with the resulting weights subsequently used in making a 45-day prediction of the behavior of the

Tab. 3: Summary of computed error means and coefficients of determination – western abutment. The subscript t represents computed values for the training set while subscript p denotes those computed for the predicted signal.

	Sample 1			Sample 2			Sample 3		
	Δx	Δy	Δz	Δx	Δy	Δz	Δx	Δy	Δz
μ_t	0.001	0.006	0.009	-0.001	-0.003	0.001	-0.001	0.001	0.002
μ_p	0.007	-0.021	-0.013	0.015	-0.035	-0.006	0.011	0.004	-0.006
r_t^2	0.888	0.978	0.960	0.891	0.978	0.959	0.855	0.973	0.961
r_p^2	0.957	0.959	0.979	0.916	0.903	0.983	0.892	0.972	0.966

western abutment. Both the computed error mean μ and coefficient of determination r^2 indicate a good fitting between the modelled and observed signals. The predicted signal also fits the expected signal reasonably.

Fig. 6 depicts results from a neural network trained using 724 epochs enabling the determined weights to be used in predicting a 100 days window period in the behavior of the bridge's western abutment. Fig. 7 shows results obtained from a network trained using 644 epochs, thus enabling a prediction for 180 epochs which equate to almost six months in time. The computed μ for the three samples lies between 0.001 to –0.035, with the highest value corresponding to the Δy component in sample 2, which on the other hand has an r^2 of 0.978 as proof to the fact that the modelled signal fits the expected signal well. In general, the values of r^2 lie between 0.855 to 0.983 which are pretty close to the optimal value of 1, meaning that the prediction was very efficient.

Fig. 5 to 7 show that well designed and trained multiple feed-forward neural networks are capable of modelling input-output relationships quite accurately. The successful application of neural networks in modelling data sets from the Fallersleber Tor bridge shows how this model encapsulates the dynamic behavior of the structure. This clearly demonstrates that the model can be used to predict the behavior of the bridge successfully from anywhere between 0 to 180 days for inputs that lie within the range of the training ensemble. It is however difficult to tell what the prediction would look like for a case where data sets do not lie within the bounds of the training set.

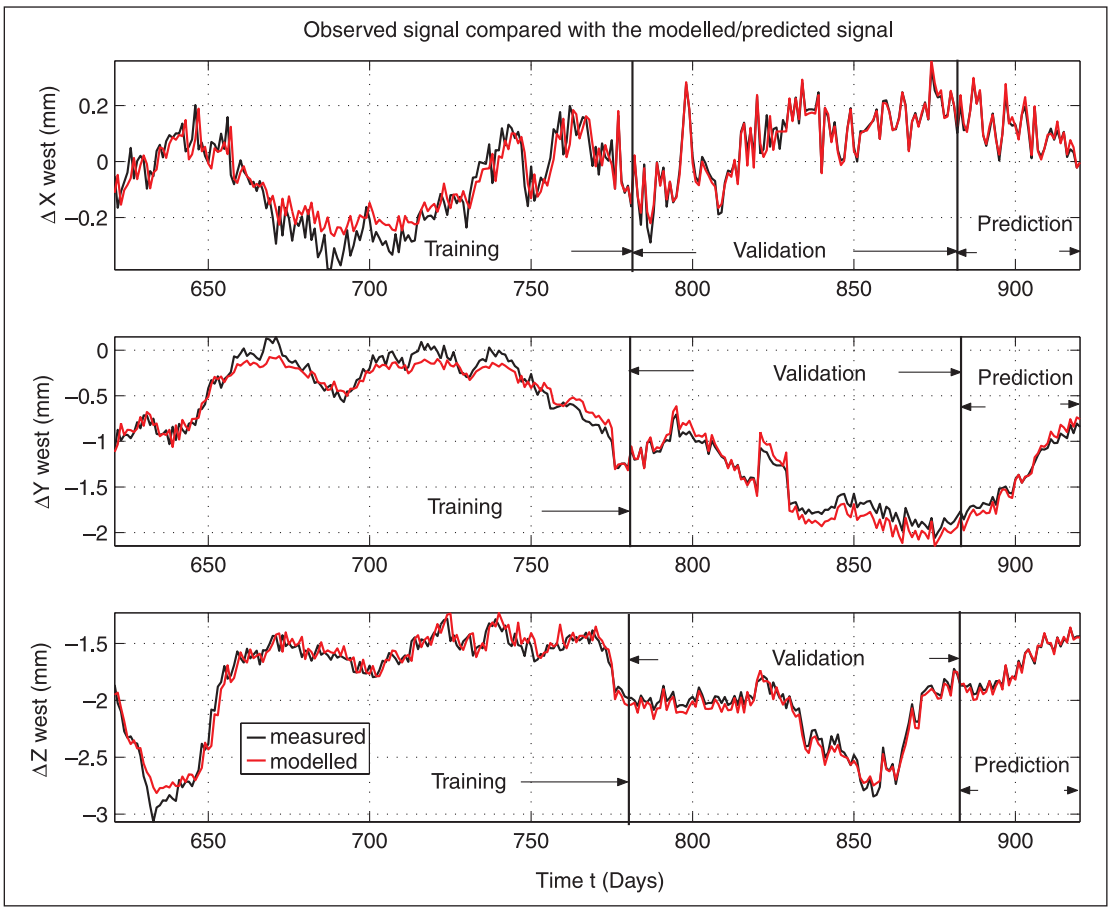

Abb. 5: Comparison between observed and modelled signals for sample 1

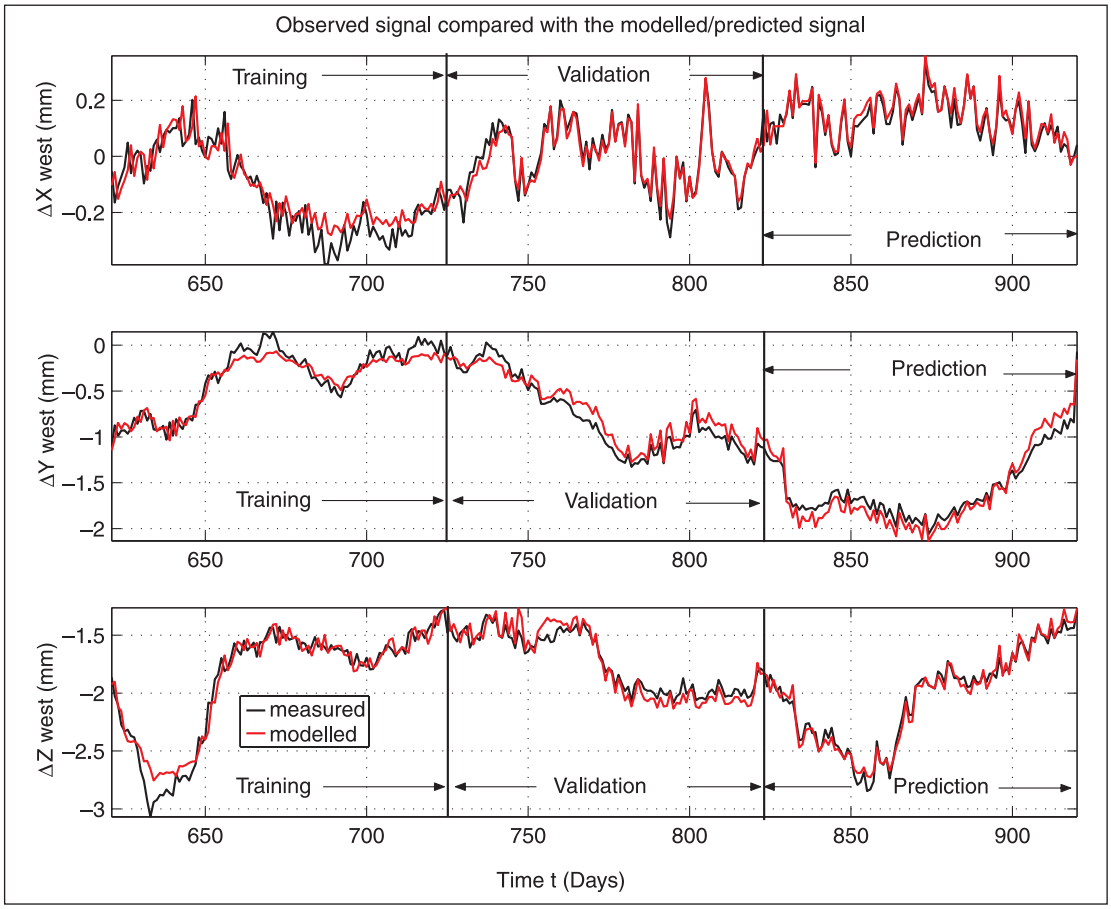

Abb. 6: Comparison between observed and modelled signals for sample 2

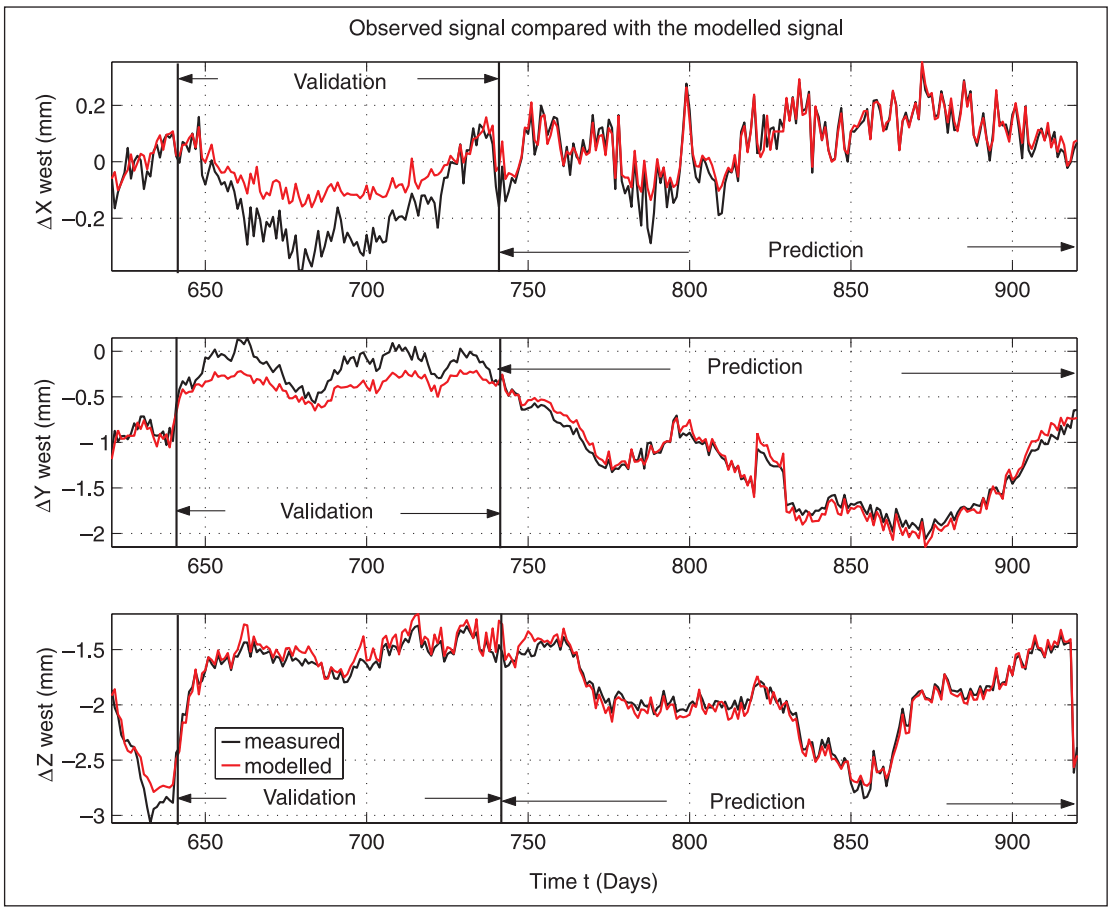

Abb. 7: Comparison between observed and modelled signals for sample 3

A comparison of the three windows of prediction (45-, 100-, 180-day) in this study does not show any significant difference in the results obtained. All three networks seem to perform well enough to be used in the description prediction of structural deformation. This does however leave one open question that would be of interest for the purposes of determining optimal results within a short time. That is, the need to know the optimal length of epochs that may be used in the computations such that unnecessary data sets are disbanded while at the same time ensuring that the results of the computation are plausible. It can also be observed that in some cases the coefficients of computation for the prediction data are greater than those of the training data sets. This is not typical for most prediction methods in mathematics, the authors are however investigating further to determine the possible explanation for this and whether, among others, the setting up of different networks for each component would make any difference in results obtained.

5 Concluding remarks

The adaptation of neural networks to model deformation offers geodesists a good alternative of describing structural deformation. The determined parameters, weights in this instance, intrinsically describe the mapping between the inputs and outputs but cannot be used in any other way as representing a typical mathematical function for deformation process. Neural networks can be viewed as a black-box tool capable of modelling nonlinear input-output relationships.

It is very important to observe that results from the use of neural networks are highly dependent on the choices of both inputs, outputs, and architecture of the network to be used, as they are capable of learning anything! They also have the disadvantage that there is no single similar solution to any given input-output data set as the determined parameters depend on various settings, undertaken during the learning process, that rely purely on personal human judgment. However, neural networks can provide an alternative to the methods of describing deformation process in geodesy, especially in continuous structural monitoring where there is no *a priori* knowledge of the underlying deformation processes. They thus may serve to complement existing methodology for modelling deformation processes.

References

Bishop C. M.: Neural networks for pattern recognition. Oxford University Press, 1995.

Chrzanowski A., Yong C. I.: Identification of deformation models in space and time domain. Survey Review 33(263): 518–528, 1996.

Cybenko G.: Approximations by superpositions of a sigmoidal function. Mathematics of Control, Signals and Systems 2: 303–314, 1989.

Felgendreher N.: Zu Modellierungsproblemen bei dynamischen Systemen ZfV 3: 125–129, 1982.

Chatfield C.: The analysis of time series: an introduction. Chapman and Hall, 1975.

Heine K.: Beschreibung von Deformationsprozessen durch Volltera- und Fuzzy-Modelle sowie Neuronale Netze. Dissertation. Deutsche Geodätische Kommission, Reihe C, Heft Nr. 516, 1999.

Hertz J., Krogh A., Palmer R. G.: Introduction to the theory of neural computation. Addison-Wesley, 1991.

Heunecke O., Pelzer H.: A new Terminology for Deformation Analysis Models based on System Theory. IAG Symposium on Geodesy for Geotechnical and Structural Engineering in Eisenstadt, 20.–22. April, 285–292, 1998.

Miima J. B.: Artificial Neural Networks and Fuzzy Logic Techniques for the Reconstruction of structural Deformations. Dissertation. Geodätische Schriftenreihe der Technischen Universität Braunschweig Nr. 18, 2002.

Niemeier W., Kraus B., Miima J. B., Flebbe H.: Bestimmung von 3D-Verformungen einer Brücke mit motorisierten Tachymetern. Messen in der Geotechnik 2000, Mitteilung des Instituts für Grundbau und Bodenmechanik, Heft Nr. 62: 235–246, 2000.

Nilson N. J.: Learning machines. McGraw-Hill, 1965.

NeuroDimensions: Neurosolutions version 3, NeuroDimension, Inc. Gainesville, Florida, 2000.

Pfeufer A., Werner H.: Dynamische Betrachtung von Deformationsprozessen – ein noch junges Aufgabengebiet der Ingenieurvermessung mit Zukunft. ZfV 12: 527–533, 1990.

Rumelhart D. E., Hinton G. E., Williams R. J.: Parallel Distribution Processing: Foundations. MIT Press Cambridge, MA., 1986.

Welsch W., Heunecke O.: Zur Systematisierung der Auswertemodelle geodätischer Überwachungsmessungen. ZfV 11: 361–368, 2000.

Welsch W.: Geodetic Analysis of Dynamic Processes: Classification and Terminology. 8th FIG International Symposium on Deformation Measurements, Hong Kong, 1996.

Widrow B., Winter R. G., Baxter R. A.: Learning Phenomena in Layered Neural Networks. Proceedings of the First IEEE International Conference on Neural Networks, San Diego, 1987.

Authors' address

Dr.-Ing. John Bosco Miima
Professor Dr.-Ing. habil. Wolfgang Niemeier
Institut für Geodäsie und Photogrammetrie
Technische Universität Braunschweig
Gaußstraße 22
38106 Braunschweig
Tel: +49-531-7482.
j-b.miima@tu-bs.de
w.niemeier@tu-bs.de