Comparison of Levenberg-Marquardt and Extended Kalman Filter based Parameter Estimation of Artificial Neural Networks in Modelling Deformation Processes

S. Horvath, H. Neuner
Department of Geodesy and Geoinformation,
TU Wien, Gußhausstraße 27-29, 1040 Vienna, Austria

Abstract. In monitoring various deformation models are well-established. Considering deformations as a dynamic reaction of the object structure to influencing parameters describes the monitoring task in its entirety. However, if the structure and the response characteristics of the object are unknown or too complex to be modelled, a behavioural approach is formulated.
Estimating parameters of an unknown input-output dependency from observed data considering available a priori information is the aim of learning. In general, a learning algorithm consists of a set of approximating functions and an optimisation method. In this contribution the well-known Kalman filter is applied as an optimisation method in combination with artificial neural networks as a set of approximating functions. This approach will be compared with the prevailing optimisation method of Levenberg-Marquardt.
It is shown that the Levenberg-Marquardt is a particular case of the extended Kalman filter, which results under simplifying assumptions. Morover, the covariances included in the Kalman filter can be chosen reasonable and exhibit the same functionality as e.g. the step size in optimisation. The results obtained by the two optimisation methods are compared at a theoretical level as well as based on synthetic data.

Keywords. Learning, Extended Kalman filter, Levenberg-Marquardt, ANN, System identification

1 Introduction

Determining models from data gain importance in various scientific fields due to simple data generation by sensors with high discretisation rates and high computation capacity. The advantage of such learning methods is that many influencing parameters and their dependencies can be considered in simple mathematical models. Learning from data is a powerful tool, which has been successfully shown in fields of classification as well as in regression. The two main steps in learning theory are estimating the model parameters first; based on them the prediction of the behaviour is accomplished. That approach matches well with monitoring tasks, in which the behaviour of an observed object shall be predicted, e.g. to warn if a non-reversible deformation will take place or to assess the structure behaviour under certain loading conditions.

"A learning algorithm is defined by the selection of a set of approximating functions and an optimisation method." (Cherkassky, Mulier, 1998, p. 131) The applied approximating functions are artificial neural networks, which strongly depend on the optimisation method. As a result this paper concentrates on the parameter estimation and presents the Kalman filter as an appropriate optimisation method.
First, the basic principles of the used methods will be explained, focusing on optimisation. In the third part the equivalence of the well-known Levenberg-Marquardt algorithm and the extended Kalman filter (EKF) will be argued. Moreover, the potential of the EKF will be stressed. On the basis of a simulated deflection of a cantilever due to variation in temperature the approaches will be evaluated.

2 Learning Theory

Learning applies the induction principle, i.e. a model will be derived on basis of true facts or measurements, but contrary to the physical world in a high dimensional space. However, the computed empirical model conforms to the general model only in case of an infinite set of test data. This is of importance for the interpretation of the

generalisation capability and consequently for the predictive accuracy reached.
A priori information is crucial to formulate and constrain a problem. Before the estimation, the model structure of a problem needs to be determined. Therefore, the amount of data is essential; it has to increase with model complexity. Otherwise, the data can not represent the high dimensionality as concluded in the model. This is known as the curse of dimensionality.

In general, a model between related input $\underset{T\times L}{\mathbf{X}} = (\mathbf{x}(t)) = (x_1(t), x_2(t), \cdots x_L(t))$ and output measures $\underset{T\times L}{\mathbf{Y}} = (\mathbf{y}(t)) = (y_1(t), \mathrm{y}_2(t), \cdots y_L(t))$ with $t = 1 \cdots T$ shall be constructed, whereby T represents the number of training samples. The function f includes k free parameters or weights $\mathbf{w} = (w_k)$, which need to be estimated in order to map the relation between the measures $\mathbf{x}(t)$ and $\mathrm{y}_n(t)$ as given in Formula (1). Herein as well as in the following formulas, single elements of the output measure vector $\mathbf{y}$ are referred. This conforms to the usual approach in artificial neural networks (ANN) training (see also section 2.1).

One possibility to adjust the parameters in learning theory is to apply the error correction in Equation (2). Therefore, the computed output of (1) is compared with the given output $\mathrm{y}_n(t)$. If the model f is selected reasonably and the constraints for generalisation are met, the true relation will be approximated by minimising the computed squared error, which is known as the empirical risk minimisation. Thus, it ends in an iterative optimisation problem.

$$\hat{y}_n(t) = f_n(\mathbf{w}, x_1(t), x_2(t), \cdots) \quad (1)$$

$$e_n(t) = y_n(t) - \hat{y}_n(t) \quad (2)$$

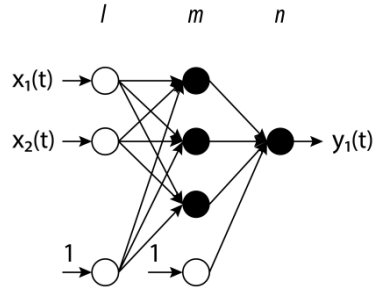

Fig. 1 Artificial neural network

2.1 Artificial Neural Networks

Artificial neural networks are a flexible type of approximating functions to describe linear and nonlinear dependencies between measures. A simple multilayer structure of an ANN is presented in Figure 1. It consists of an input layer (l) with two nodes, a hidden layer (m) with three nodes and an output layer (n) with one output node. The more hidden layers or nodes in the hidden layers, the more complex and higher dimensional the function becomes. The connections between the layers (e.g. l and m) represent the adjustable weights w_{ml}. In Figure 1 the biases are pictured as additional input nodes for each layer due to their treatment as a fixed input of 1. The bias itself corresponds to the weight w_{0l}. It effects an affine transformation to the subsequent node.
In Equation (3) the mathematical description of the ANN from Figure 1 is shown. Different activation functions serve φ as basis functions in ANNs, which are motivated in neurobiology. For further reading see Heunecke et al. (2013) or Haykin (1999).

$$\begin{aligned}\hat{y}_n(t) &= f_n(\mathbf{w}, x_1(t), x_2(t), \cdots) \\ &= \varphi^{(n)}\left(\sum_m w_{nm}\varphi^{(m)}\left(\sum_l w_{ml}x_l(t)\right)\right) \\ &= \varphi^{(n)}\left(\sum_m w_{nm}y_m(t)\right)\end{aligned} \quad (3)$$

Equation (3) describes the forward step of the learning process by ANNs. For a given input $x_1(t)$ and actual weights $\mathbf{w}$ the appropriate output $\hat{y}_n(t)$ is computed. The weight adjustment will be computed by the extended error correction rule $\varepsilon(t)$ in Formula (4-7). The so called total error energy is the squared error summed up over all nodes of the output layer. The error at the output nodes is computed directly. However, the hidden nodes are not directly accessible, but they share responsibility for any error made at the output of the ANN as well. The not trivial backpropagation of the error to the hidden or the input nodes can be done by using the chain rule of differentiation as presented in (5) and (6). The backward step passes the error signal through the network and recursively computes the local gradients δ_j for each node. The weight correction e.g. ∂w_{ml} is accomplished by multiplying

a learning parameter η, the local gradient δ_m and the input signal of the appropriate node x_1 (7).

$$\varepsilon(t) = \frac{1}{2}\sum_n e_n^2(t) \tag{4}$$

$$\frac{\partial \varepsilon(t)}{\partial w_{nm}} = -e_n \varphi^{(n)\prime} y_m = -\delta_n y_m \tag{5}$$

$$\begin{aligned}\frac{\partial \varepsilon(t)}{\partial w_{ml}} &= -\sum_n \left(e_n \varphi^{(n)\prime}\right) w_{nm} \varphi^{(\mathrm{m})\prime} x_l \\ &= -\varphi^{(\mathrm{m})\prime} \sum_n \left(\delta_n w_{nm}\right) x_l = -\delta_m x_l\end{aligned} \tag{6}$$

$$\Delta w_{ml} = \eta \cdot \delta_m \cdot x_l \tag{7}$$

The implementation can be carried out by online or batch mode. The distinction between them is that either just one (*t)* or few to all sets of labeled input-output data (t_{batch}) cause an updating of the weights. An online implementation usually requires the labeled pairs to be presented in a random order. The advantage of this implementation is to run less likely into a local minimum. However, the batch version leads to more accurate estimates.

2.2 Nonlinear Optimisation

The aim of optimisation in general is to find global extreme values. If searched for the minimum or the maximum depends on the formulation of the target function. As stated by the error correction rule, the minimisation of the error is the answer of the problem at hand. However, the minimum found strongly depends on the starting point, the initial guess. Brute-force techniques are often used to overcome this problem - like repeating the procedure with different starting points (Cherkassky, Mulier, 1998).

The optimisation is an iterative procedure. In each iteration *i* the parameters $w_{k,i}$ will be updated as given in (8). To increase the convergence speed to an optimum, updating parameters like step size γ_i and search direction should be considered. In this contribution mainly the step-size, equivalent to the before mentioned learning rate η, will be treated.

$$w_{k,i+1} = w_{k,\mathrm{i}} + \gamma_i \Delta w_{k,i} \tag{8}$$

2.2.1 *General methods for local nonlinear optimisation*

If the approximating function is nonlinear in parameters correspondingly, nonlinear or second order optimisation methods are needed. These consider the first and the second derivative of the Taylor series expansion of an error function $\boldsymbol{\varepsilon}(\mathbf{w}_{i+1})$ as shown in (9). Minimising the second order Taylor expansion, leads to the Newton method (10), which represents the standard nonlinear optimisation method. Thereby, the parameters will be updated on basis of the gradient $\nabla\boldsymbol{\varepsilon}$ of the error function - the linear part - and the Hessian matrix $\mathbf{H}$ including the quadratic term of the error function.

$$\boldsymbol{\varepsilon}(\mathbf{w}_{i+1}) = \boldsymbol{\varepsilon}(\mathbf{w}_i) + \nabla\boldsymbol{\varepsilon}^T \Delta\mathbf{w} + \frac{1}{2}\Delta\mathbf{w}^T \mathbf{H} \Delta\mathbf{w} \tag{9}$$

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \mathbf{H}^{-1}\nabla\boldsymbol{\varepsilon} \tag{10}$$

In comparison, the backpropagation procedure given in (5-7) is a linear optimisation method, which is based only on the first derivative of the Taylor series expansion. This optimisation method is known as steepest descent or gradient descent.

First order methods are reliable due to their permanent progress towards a local minimum, but offer a slow convergence behaviour.

The quadratic approximation by second order methods lead to faster convergence. However, the knowledge of local curvature provided by the Hessian matrix $\mathbf{H}$ is useful only close to a minimum. In regions far away the algorithm can lead to divergence. Moreover, the computation of the second partial derivatives is computationally expensive.

Due to that reason methods approximating the second derivative have been developed. These approaches are called Quasi-Newton methods. The most popular method of this class is the Gauss-Newton, used to solve a least-squares problem. Therefore, the error function $\boldsymbol{\varepsilon}(\mathbf{w}_{i+1})$ will be linearised at the current iterate $\mathbf{w}_i$ (11). The linear part is represented by the Jacobi matrix $\mathbf{J}$, the partial derivatives of the vector-valued functional (see (14)). Subsequent, this linearised function will be minimised, as presented in Equation (12).

$$\frac{1}{2}\left\|\boldsymbol{\varepsilon}(\mathbf{w}_{i+1})\right\|^2 = \frac{1}{2}\left\|\boldsymbol{\varepsilon}(\mathbf{w}_i) + \mathbf{J}(\mathbf{w}_i)\Delta\mathbf{w}\right\|^2 \tag{11}$$

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \left(\mathbf{J}^T\mathbf{J}\right)^{-1}\mathbf{J}^T\mathbf{e} \tag{12}$$

The Gauss-Newton method transforms the nonlinear to a linear optimisiation problem. However, if the deviations from the true Hessian Matrix are large, the approximation is not accurate and will lead to slow convergence or divergence.

For the stated reasons the Levenberg-Marquardt (LM) algorithm is a good pratical alternative. It extends the Gauss-Newton approach by a term $\lambda\mathbf{I}$ as presented in (13). This term decides if the algorithm behaves more like the Gauss-Newton method (small λ) or like steepest descent (increased λ). The main virtue is that the advantages of both methods are combined, which are fast and reliable convergence.
The optimisation of ANNs by LM or any other standard second order method is usually applied in a batch mode. The structure of the used Jacobi matrix is shown in (14). The $t_{batch}\cdot n$-rows of the matrix indicate a batch implementation, where t_{batch} stands for the time points of data processed in one batch. To conform with the updating equation (13), the Jacobi matrix is build up of the partial derivatives of the error function $e_n(t)$ and not of the total error as done in backpropagation.

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \left(\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I}\right)^{-1}\mathbf{J}^T\mathbf{e} \tag{13}$$

$$\left\{\mathbf{J}(\mathbf{w})\right\}_{t_{batch}\cdot n,k} = \frac{\partial e_n(t)}{\partial w_k} \tag{14}$$

2.2.2 *Extended Kalman filter*

The Kalman filter (KF) is an optimal recursive data processing algorithm (Maybeck, 1979). It has been developed for state space representation of linear dynamic systems and has been treated in different scientific areas, see Gelb (1974) and Heunecke et al. (2013). It allows for optimal parameter estimation in case of linear systems corrupted by Gaussian white noise. In case of nonlinear systems the equations of the KF need to be linearised about a nominal point. This procedure is known as the extended Kalman filter (EKF).
In this contribution the EKF will be used as an optimisation method and therefore the two equations have been adapted according to Singhal, Wu (1989), who proposed using the EKF for training multilayer neural networks.

$$\begin{aligned}\bar{\mathbf{w}}_{i+1} &= \mathbf{I}\hat{\mathbf{w}}_i + \mathbf{p}\\ \mathbf{p} &\sim \mathrm{N}(0, \boldsymbol{\Sigma}_{\mathbf{pp}})\end{aligned} \tag{15}$$

$$\begin{aligned}y_n(t) - \mathrm{o}_n &= f_n\left(\hat{\mathbf{w}}_{i+1}, x_1(t), x_2(t), \cdots\right)\\ \mathbf{o} &\sim N(0, \boldsymbol{\Sigma}_{\mathbf{yy}})\end{aligned} \tag{16}$$

Due to a static system (13) the optimisation mainly relies on the observation equation in (14), which consists of the approximating function f_n and the observation noise o_n, similar to (1).
The observation or measurement noise $\mathbf{o}$ is assumed to be additive, white and Gaussian, with zero mean and with covariance matrix $\boldsymbol{\Sigma}_{\mathbf{yy}}$. Due to the consideration of the σ_0^2 a pass over to cofactor matrices e.g. $\mathbf{Q}_{\mathbf{yy}}$ is possible.
The updating of the weights $\mathbf{w}_i$ is done according to the EKF algorithm and is presented in Equation (18). Therefore, first the Kalman gain $\mathbf{K}_{i+1}$ needs to be computed (17), which considers the influence of observations on the state estimation. The predicted approximate error cofactor matrix $\mathbf{Q}_{\bar{w}\bar{w},i+1}$ consists of the adjusted approximate error cofactor matrix of the previous iteration step and the process cofactor matrix, as shown by: $\mathbf{Q}_{\bar{w}\bar{w},i+1} = \mathbf{I}\mathbf{Q}_{\hat{w}\hat{w},i}\mathbf{I}^T + \mathbf{Q}_{pp,i+1}$.
It is also estimated recursively and results in the adjusted approximate error cofactor matrix $\mathbf{Q}_{\hat{w}\hat{w},i+1}$ according to Equation (19). Due to the reason that no information about the initialisation of the weights is given, it shall be initialised by the identity matrix multiplied by a factor $\mathbf{Q}_{\bar{w}\bar{w},0} = \rho^{-1}\mathbf{I}$ (ρ can vary between 10^{-1} and 10^{-3} (Haykin, 2001, p. 32).
As will be shown in the next sections the process noise $\mathbf{Q}_{pp}$ has a crucial contribution to the stability of the estimation process. According to Puskorius, Feldkamp (1991) the approximate error cofactor matrix $\mathbf{Q}_{\bar{w}\bar{w}}$ can become singular, lose the necessary property of nonnegative definitness, or vanish completely. The integration of $\mathbf{Q}_{pp}$ in the estimation process avoids these effects. It is recommended to define $\mathbf{Q}_{pp}$ as a diagonal matrix including small and nonnegative components in the range of 10^{-6} to 10^{-2} (Haykin, 2001).
Thus, two parameter values have to be chosen for the EKF estimation, the cofactor matrix of the measurement noise $\mathbf{Q}_{yy}$ and the process noise $\mathbf{Q}_{pp}$.

$$\mathbf{K}_{i+1} = \mathbf{Q}_{\bar{w}\bar{w},i+1}\mathbf{A}_{i+1}^T\left(\mathbf{Q}_{yy,i+1} + \mathbf{A}_{i+1}\mathbf{Q}_{\bar{w}\bar{w},i+1}\mathbf{A}_{i+1}^T\right)^{-1} \tag{17}$$

$$\hat{\mathbf{w}}_{i+1} = \overline{\mathbf{w}}_{i+1} + \mathbf{K}_{i+1}\mathbf{e}_{i+1} \tag{18}$$

$$\mathbf{Q}_{\hat{w}\hat{w},i+1} = \mathbf{Q}_{\overline{w}\overline{w},i+1} - \mathbf{K}_{i+1}\mathbf{A}_{i+1}\mathbf{Q}_{\overline{w}\overline{w},i+1} \tag{19}$$

$$\left\{a_{n,k}\right\}_{i+1} = \frac{\partial \hat{y}_{n,i+1}(t)}{\partial \overline{w}_{k,i+1}} \tag{20}$$

The derivative matrix $\mathbf{A}$ is computed via backpropagation (see section 2.1) with the subtle distinction that the approximating function $\hat{y}_n$ and not the error e_n will be differentiated with respect to the weights. There is a separate backpropagation for each output node $\hat{y}_n$. Thus, the matrix exhibits the structure as presented in (20). The difference between EKF and to the gradient approach is that the second order approximation $\mathbf{A}\mathbf{A}^T$ is applied additionally in the first case and therefore a faster convergence will be reached. In fact, no additional linearisation takes place due to the utilisation of the EKF.

3 Comparison of LM and EKF

In this section, the equivalence of Levenberg-Marquardt and the EKF based estimation is shown. Moreover, the role of the additional measures included in the Kalman filter will be discussed. Due to the consideration of stochastic information, the heuristic of forcing the total error to become smaller by applying a varying step-size, can be replaced by a stochastic meaningful test.

For the comparison of the Levenberg-Marquardt estimation with the EKF based estimation the second summand on the right side of (13) will be expanded as follows:

$$\begin{aligned}
&\left(\underset{K\times N}{\boldsymbol{J}^T}\ \underset{N\times K}{\boldsymbol{J}} + \lambda\ \underset{K\times K}{\boldsymbol{I}}\right)^{-1}\ \underset{K\times N}{\boldsymbol{J}^T}\ \underset{N\times 1}{\boldsymbol{e}} \\
&= \left(\underset{K\times N}{\boldsymbol{J}^T}\ \underset{N\times K}{\boldsymbol{J}} + \lambda\ \underset{K\times K}{\boldsymbol{I}}\right)^{-1}\ \underset{K\times N}{\boldsymbol{J}^T}\left(\underset{N\times K}{\boldsymbol{J}}\ \underset{K\times N}{\boldsymbol{J}^T} + \lambda\ \underset{N\times N}{\boldsymbol{I}}\right)\left(\underset{N\times K}{\boldsymbol{J}}\ \underset{K\times N}{\boldsymbol{J}^T} + \lambda\ \underset{N\times N}{\boldsymbol{I}}\right)^{-1}\ \underset{N\times 1}{\boldsymbol{e}} \\
&= \left(\underset{K\times N}{\boldsymbol{J}^T}\ \underset{N\times K}{\boldsymbol{J}} + \lambda\ \underset{K\times K}{\boldsymbol{I}}\right)^{-1}\left(\underset{K\times N}{\boldsymbol{J}^T}\ \underset{N\times K}{\boldsymbol{J}} + \lambda\ \underset{K\times K}{\boldsymbol{I}}\right)\underset{K\times N}{\boldsymbol{J}^T}\left(\underset{N\times K}{\boldsymbol{J}}\ \underset{K\times N}{\boldsymbol{J}^T} + \lambda\ \underset{N\times N}{\boldsymbol{I}}\right)^{-1}\ \underset{N\times 1}{\boldsymbol{e}} \\
&= \underset{K\times N}{\boldsymbol{J}^T}\left(\underset{N\times K}{\boldsymbol{J}}\ \underset{K\times N}{\boldsymbol{J}^T} + \lambda\ \underset{N\times N}{\boldsymbol{I}}\right)^{-1}\ \underset{N\times 1}{\boldsymbol{e}}
\end{aligned} \tag{21}$$

Thus (13) becomes:

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \mathbf{J}^T\left(\mathbf{J}\mathbf{J}^T + \lambda\mathbf{I}\right)^{-1}\mathbf{e} \tag{22}$$

The EKF weight updating (17-18) matches Equation (22), if $\mathbf{Q}_{\overline{w}\overline{w},i} = \mathbf{I}$, $\mathbf{Q}_{yy,i} = \lambda\mathbf{I}$ and no updating of the approximate error cofactor according to (19) is accomplished. Moreover, the transition from $\mathbf{A}$ to $\mathbf{J}$ leads to the change of the algebraic sign. The similarity of the techniques considering (21) is obvious.

This indicates that in the typical training of neural networks it is not accounted for stochastic information of the output data nor for the propagation of algebraic correlations between the output measures in one epoch to subsequent ones. Due to the included measurement noise in the EKF algorithm, the error made by observing an object expected to deform can be considered. The question arises if quantifiable measurement noise effects the parameter estimation of an ANN. This will be evaluated in the next section.

In practice, the training of a neural network using the Levenberg-Marquardt algorithm is performed with a varying λ. By this variation, it is possible to control the region where the assumption of the model's linearity used in the Levenberg-Marquardt algorithm is admissible. Hagan et al. (2014) propose to start with a small value ranging from 0.01 to 1. If the total error ε_i of the actual estimation step decreases with respect to the total error obtained in the previous estimation step ε_{i-1}, then λ is decreased by a factor of 10. This increases the region $1/\lambda$ where the linearity assumption holds. On the other hand, if the total error increases in the actual step, then λ is increased by a factor of 10. This step is repeated until a decrease of the total error is obtained in the actual step (see Bishop, 1995, p. 292).

A similar variation of the region, where the model linearity assumption holds, can be obtained in the Kalman filter based estimation procedure. Before the final update of the state vector in each iteration or epoch *i*, a compatibility test is performed. Thereby, the consistency of the functional and the stochastic model is proven. The test value is given by:

$$T_i = \frac{\mathbf{e}_i^T\mathbf{Q}_{dd,i}^{-1}\mathbf{e}_i}{\sigma_o^2} \tag{23}$$

Where:

$\mathbf{e}_i$ – error vector, also called innovation in the Kalman filter context;

$\mathbf{Q}_{dd,i}$ – cofactor matrix of the innovations, given by: $\mathbf{Q}_{dd,i} = \sigma_{yy,i}^2\tilde{\mathbf{Q}}_{yy,i} + \mathbf{A}_i\mathbf{Q}_{\overline{w}\overline{w},i}\mathbf{A}_i^T$.

In case of a consistent stochastic and functional model the test value follows a central χ^2-distribution with a probability of error α and $n_{y,i}$ degree of freedom, where $n_{y,i}$ is the number of

observations in epoch *i* or updating step t_{batch}. If the test value T_i does not lie in the confidence region $\left[\chi^2_{n_{y,i},\alpha/2}, \chi^2_{n_{y,i},1-\alpha/2}\right]$, calculated from the quantiles of the χ^2-distribution, the stochastic and the functional model need to be adapted before finalising the update of the state vector. For a large amount of geodetic problems, the functional model is well known, such that the stochastic model is primarily adjusted. In case of neural networks the model selection task is of major importance and quite challenging with regard to the generalisation capability, see Cherkassky, Mulier (1998) or Neuner (2012). However, we assume an adequate chosen functional model and focus on the adaption of the stochastic model, if the compatibility test fails. When the test value exceeds the upper limit of the confidence interval one firstly seeks for outliers in the observation set of the current epoch by applying an individual test for each innovation value. An alternative approach to eliminating observations can be a common down-weighting of the observations. This corresponds to the adaption of the process noise - the general approach in Kalman filtering. Similar, when the test value falls below the lower limit of the confidence interval, a global up-weighting of the observations is performed. Both situations correspond to a change of the a priori variance σ_{yy}^2.

This is a similar approach to the adaption of λ in LM in case of equal weighted and uncorrelated observations. Beyond the formal identity, the nature and the interpretation of the two measures λ and σ_{yy}^2 is distinct. λ is a functional parameter that controls the regularisation. σ_{yy}^2 is a stochastic parameter. The main task in LM based training is to find appropriate values for λ, which are not restricted to a certain domain. The factors for the variation of λ are heuristics and need to be set problem dependent. In contrast, $\sigma_{yy,0}^2$ is derived from the properties of the used sensors and the variation factors of this measure are controlled by the quantiles of the χ^2-distribution. Thus, it can be concluded that the setting of the training parameters is more accessible and sound in case of the Kalman filter-based training.

To better understand the role of the process noise in the neural network training, it is necessary to analyse Equation (17). We focus on the Kalman gain matrix $\mathbf{K}_{i+1}$ because it controls the change of the state from one epoch to another by weighting the innovations contribution to the updated state estimate (see (18)). For the sake of clarity the Kalman gain $\mathbf{K}_{i+1}$ is derived for the one-dimensional case with a direct observed state.

$$\mathbf{K}_{i+1} = \frac{1}{1+\left(\frac{\sigma_{yy,i+1}}{\sigma_{\bar{w}\bar{w},i+1}}\right)^2} = \frac{1}{1+\left(\frac{\sigma_{yy,i+1}^2}{\sigma_{\hat{w}\hat{w},i}^2 + \sigma_{\mathrm{pp},i+1}^2}\right)} \quad (24)$$

Equation (24) presents the interdependency between measurement and process noise. The higher the process noise $\sigma_{\mathrm{pp,i+1}}^2$, the higher the gain and implicitly the weight for the innovation in the updating equation (18). The extent to which this interdependency influences the actual estimation is driven by the variance of the state estimated in the previous epoch $\sigma_{\hat{w}\hat{w},i}^2$. This plays an important role in the first epochs of the Kalman filter, where the state is initialised with high variances. At this stage the innovation is practically completely included in the actual estimation. During the filter progress the estimation uncertainty of the state decreases and the interplay between process and measurement noise becomes more important, crucially influencing the convergence behaviour of the filter and the quality of the estimated solution.

For analogy reasons between Kalman filter and Levenberg-Marquardt training, we decided in the above sections to adjust the measurement noise σ_{yy}^2 during the iterations. Therefore, the interdependency between measurement and process noise is driven by changing the measurement noise and keeping the process noise constant. However, to highlight the impact of the process noise, in the next section two solutions of the same problem are calculated including and excluding it in the estimation. Its contribution to the acceleration of the training process and obtaining superior solutions compared to those found without process noise will be shown in Figure 4. This conclusion is in accordance to the research of Puskorius, Feldkamp (1991) which however used a different strategy to account for the process noise.

As a result of the Kalman filter training, besides the estimates for the weights, corresponding uncertainty values are achieved too (see (19)). This allows to check the statistical significance of the weights based on a t-test. Thus, the structure of the neural network can be optimised. Compared to other

methods like the saliency of weights, where the contribution of a network connection to the total error is qualitatively assessed relative to the contribution of the other connections (see Bishop (1995); Neuner (2012)), the quantiles of the t-distribution provide objective and definite limit values for this task in case of Kalman filter training. The exploit of this advantage will be content of future research.

4 Evaluation

To compare the Levenberg-Marquardt with the extended Kalman filter optimisation, a deflection of a cantilever influenced by a temperature difference between its two sides has been simulated. According to Figure 2, a cantilever of width $h = 3m$ and length $l = 28m$ has been assumed. The considered temperatures T_1 and T_2 as well as the simulated deflection on the end of the cantilever d are presented in Figure 3. Normal distributed noise with $\sigma_{yy}^{\ 2} = 0.5mm$ has been added to the computed deflection and will be considered in the Kalman filter. The input of the neural network is the temperature difference $(T_1 - T_2)$ and the output is the deflection including noise. On basis of the cross-validation technique (Neuner, 2012) the model structure of the ANN has been identified. It consists of one hidden layer with three nodes (1-3-1).

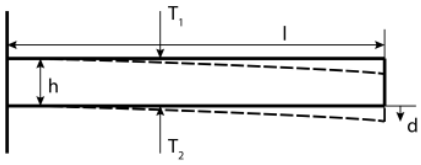

Fig. 2 Scatch of the assumed cantilever

The training of the neural network has been accomplished with LM and the EKF under the assumption of $\mathbf{Q}_{\overline{ww},i} = \mathbf{I}$, $\mathbf{Q}_{yy,i} = \lambda\mathbf{I}$ to show the equivalence (see also Section 3). Both methods estimate ident weights and thus an ident total error and standard deviation over all training samples, as depicted in the upper plot of Figure 4. However, the same variation of λ needs to be applied.

To experience the influence of the covariances, first only the approximate error cofactor matrix $\mathbf{Q}_{\hat{w}\hat{w},0}$ is initialised. The influence of the updating of the covariance is shown by the red graph in Figure 4. A further step is to add the stochastic information of the observations $\mathbf{Q}_{yy}$. The scaling of the noise is thereby of importance. The result is represented by the yellow graph in Figure 4.

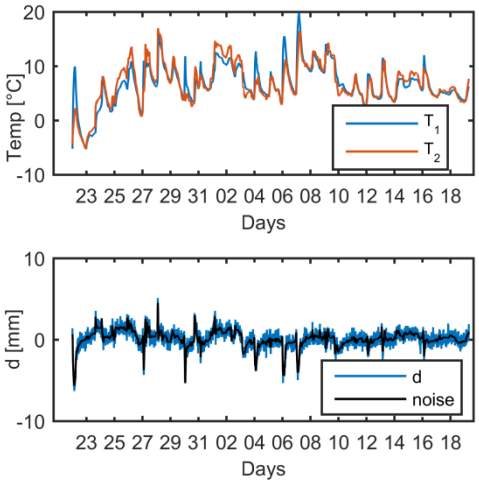

Fig. 3 Temperature difference causing a simulated deflection of a cantilever

The EKF has also been implemented with the compatibility test. Therefore, the first 10 iterations have been computed with the standard approach in learning - the adaption of λ - to reach reasonably stable initial values of the weights. After this initialisation period, only the compatibility test for each batch (t_{batch}) consisting of 50 training samples has been accomplished. A significance level of 95% was chosen. By means of the magenta graph in Figure 4 the initialisation is recognisable. The adaption of $\sigma_{yy}^{\ 2}$ - to fulfil the null hypothesis - leads to a specific level, thus it appears in Figure 4. In some cases this level can not be held as indicated in the averaged error graph in Figure 4. Likely, a sudden change of the weights lead to this effect. Finally, the adaption of the process noise $\sigma_{pp}^{\ 2}$ in the course of the EKF estimation with compatibility test has been applied and leads to an improvement of the error graph (Figure 4, blue graph).

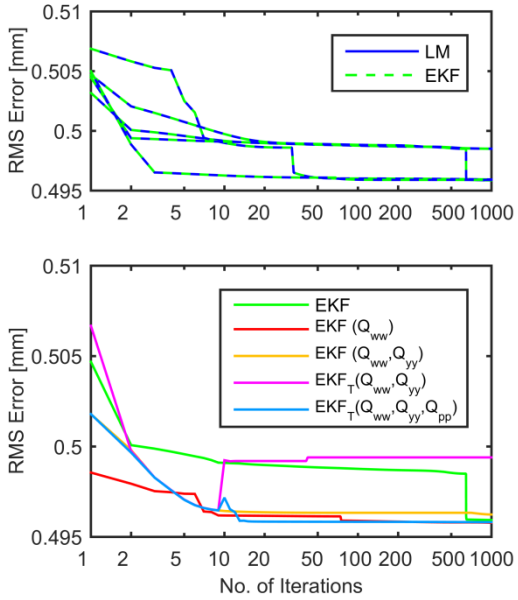

Fig. 4 Output error during training; The upper plot shows the comparison LM and EKF; the lower one the different versions of the EKF.

5 Conclusion

This contribution focuses on the optimisation part of the learning task. The objective is to show, that the Kalman filtering in comparison to the prevailing Levenberg-Marquardt method is an appropriate tool and to assess the influence caused by the stochastic information included in the EKF.

It is shown that the Levenberg-Marquardt is a particular case of the extended Kalman filter, which results under simplifying assumptions.

Compared to the problem specific setting of λ the EKF enables a reasonable selection of the measurement noise $\sigma_{yy}^{\ 2}$, though the functionality of the parameters is the same. The second additional parameter of the EKF, the process noise $\sigma_{pp}^{\ 2}$, accelerates convergence behaviour and leads to a more accurate estimation.

Future research will treat the additional value of the covariances of the parameters with regard to the model selection task and the structure of ANN by means of significance test of the weights.

References

Bishop C. M. (1995). Neural networks for pattern recognition. Clarendon Press.

Cherkassky V., F. Mulier (1998). Learning from data: concepts, theory, and methods. Wiley.

Gelb A. (1974). Applied optimal estimation. MIT press.

Hagan M. T., H. B. Demuth, M. H. Beale, O. De Jesus (2014). Neural Network Design. 2nd Edition.

Haykin S. S. (1999) Neural Networks - A Comprehensive Foundation. 2nd edition. Prentice-Hall.

Haykin S. S. (2001). Kalman filtering and neural networks. Wiley.

Heunecke O., H. Kuhlmann, W. Welsch, A. Eichhorn, H. Neuner (2013). Handbuch Ingenieurgeodaesie: Auswertung geodaetischer Ueberwachungsmessungen. 2., neu bearbeitete und erweiterte Auflage., Wichmann.

Maybeck P. S. (1979). Stochastic models, estimation and control. Vol. 1. Academic Press.

Neuner H. (2012). Model selection for system identification by means of artificial neural networks. Journal of Applied Geodesy. Vol. 6, Issue 3-4. pp. 117-124.

Puskorius G. V., L. A. Feldkamp (1991). Decoupled extended Kalman filter training of feedforward layered networks. In: *International Joint Conference on Neural Networks*. Vol. 1, pp. 771-777.

Singhal S., L. Wu (1989). Training multilayer perceptrons with the extended Kalman algorithm. In: D. Touretzky, Ed. *Advances in Neural Information Processing Systems 1*. San Mateo, CA: Morgan Kauffman, pp. 133-140