In proceedings of IEEE International Conference on Neural Networks 1995, Perth, Australia, Nov. 27–Dec. 1, 1995.

Training and Evaluation of Neural Networks for Multi-Variate Time Series Processing

Torben L. Fog*, Jan Larsen and Lars Kai Hansen
CONNECT, Electronics Institute B349
Technical University of Denmark
DK-2800 Lyngby, Denmark
Emails: fog,jlarsen,lkhansen@ei.dtu.dk

ABSTRACT

We study training and generalization for multi-variate time series processing. It is suggested to used a quasi maximum likelihood approach rather than the standard sum of squared errors, thus taking dependencies among the errors of the individual time series into account. This may lead to improved generalization performance. Further, we extend the Optimal Brain Damage pruning technique to the multi-variate case. A key ingredient is an algebraic expression for the generalization ability of a multi-variate model. The variability of the suggested techniques are successfully demonstrated in a multi-variate scenario involving the prediction of the cylinder pressure in a marine engine.

1. Introduction

Multi-variate nonlinear time series processing models are of interest in many neural network application areas. The application which motivates the present study is the supervision and fault diagnosis of marine engines. Time-series from sensors mounted on the engine are used as inputs to a signal processing device which is able to deliver feature time-series from which it is possible to monitor the condition of the engine and take any action required.

In this paper we study the training and generalization properties of a feed-forward neural network implementation of a multi-input multi-output signal processing model. The standard cost function for multiple output networks is the sum of mean square errors on the individual outputs; however, we suggest instead to use a quasi maximum a posteriori (QMAP) approach (see e.g., [12], [16]) in which the dependency among the errors plays a significant role. The dependency among errors cts as a source of prior knowledge which assists the training and is generally expected to improve generalization performance. This is due to the fact that the prior knowledge constraints the variation of the parameters in the network; thereby reducing the so-called *model variance* [4]. On the other hand, the dependence among the errors is typically not known in advance; hence, it must be estimated from data. This implies that even though the model variance is decreased the *model bias* is increasing, thus facing the ubiquitous bias/variance dilemma [4]. It is important to stress that the technique is useful for *nonlinear* parametrizations only: it is possible to show that the estimated weights of a simple linear network are invariant when including error dependencies.

The paper provides numerical results on prediction cylinder pressure in a marine engine. The suggested QMAP approach gives significantly better predictions than using the standard sum of mean square errors. Further we extend pruning techniques – as reported in [14], [15] for uni-variate series – to cope with multi-variate series which provide more parsimonious models in the numerical example.

2. Training Multi-Output Neural Networks

2.1. System and Model

Define a multi-variate input and output series by $\boldsymbol{x}(k) = [x_1(k), \cdots, x_p(k)]^\top$ and $\boldsymbol{y}(k) = [y_1(k), \cdots, y_q(k)]^\top$, respectively. For instance if considering a single input signal, the input vector is often chosen as a tapped delay line: $\boldsymbol{x}(k) = [x(k), x(k-1), \cdots, x(k-p+1)]^\top$. Suppose the existence of a transfer vector function (or mapping)

*Permanent address MAN B&W Diesel A/S, Teglholmsgade 41, DK-2450 Copenhagen SV.

$\boldsymbol{g}(\cdot)$ such that

$$\boldsymbol{y}(k) = \boldsymbol{g}(\boldsymbol{x}(k)) + \varepsilon(k) \tag{1}$$

where $\varepsilon(k)$ is a q-dimensional noise signal independent of $\boldsymbol{x}(k)$ with zero mean vector and covariance matrix $\boldsymbol{\Sigma}$. Next formulate a neural network model of Eq. (1), given by:

$$\boldsymbol{y}(k) = \boldsymbol{f}(\boldsymbol{x}(k); \boldsymbol{w}) + \boldsymbol{e}(k) \tag{2}$$

where $\boldsymbol{f}(\boldsymbol{x}(k); \boldsymbol{w})$ denotes the mapping of the neural network with $\boldsymbol{w}$ being the vector of network parameters (weights and thresholds).

2.2. Quasi Maximum A Posteriori Estimation

Since maximum likelihood estimation ensures asymptotic efficient estimates of the weights – i.e., with minimal weight covariance equivalent to the Cramer-Rao bound – we will adopt this framework for training multiple output networks. However, we do not know if the model is complete, i.e., if there exist a true weight vector $\boldsymbol{w}^\circ$ such that $\boldsymbol{g}(\boldsymbol{x}) \equiv \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{w}^\circ)$ $\forall \boldsymbol{x}$, so we refer to this scheme as quasi maximum likelihood, cf. [12, Ch. 2.3 & 12.5], [16]. Suppose that a training set of N input-output pairs, $\mathcal{T} = \{(\boldsymbol{x}(k), \boldsymbol{y}(k))\}_{k=1}^{N}$, has been collected. Next define the negative sample quasi log-likelihood:

$$\begin{aligned} -\log p_{\boldsymbol{y}(k)}&(\boldsymbol{y}(k) \,|\, \boldsymbol{x}(k), \boldsymbol{w}) \\ &= -\log p_{\boldsymbol{y}(k)}(f(\boldsymbol{x}(k); \boldsymbol{w}) + \boldsymbol{e}(k) \,|\, \boldsymbol{x}(k), \boldsymbol{w}) \\ &\equiv -\log p_{\boldsymbol{e}(k)}(\boldsymbol{e}(k) \,|\, \boldsymbol{x}(k), \boldsymbol{w}), \end{aligned} \tag{3}$$

and suppose that the errors $\boldsymbol{e}(k)$ are i.i.d. to obtain the negative quasi log-likelihood:

$$L(\boldsymbol{w}) = \sum_{k=1}^{N} -\log p_{\boldsymbol{e}}(\boldsymbol{e}(k) \,|\, \boldsymbol{x}(k), \boldsymbol{w}). \tag{4}$$

In order to assist training of the network and for ensuring proper generalization performance, we employ a maximum a posteriori approach by adding a negative log-prior $-\log p(\boldsymbol{w}) = R(\boldsymbol{w})$ to yield the negative quasi log-posterior cost function $P(\boldsymbol{w}) = L(\boldsymbol{w}) + R(\boldsymbol{w})$. Assume that the errors $\boldsymbol{e}(k)$ are Gaussian distributed with zero mean and covariance matrix $\boldsymbol{E}$. Apart from a constant and a trivial factor of $2/N$, the negative log-posterior is given by [12, Ch. 11.3][1]:

$$\begin{aligned} &\widetilde{P}(\boldsymbol{w}, \boldsymbol{E}) \triangleq \widetilde{L}(\boldsymbol{w}, \boldsymbol{E}) + \widetilde{R}(\boldsymbol{w}) = \\ &\log\det(\boldsymbol{E}) + \frac{1}{N} \sum_{k=1}^{N} \boldsymbol{e}^\top(k; \boldsymbol{w}) \boldsymbol{E}^{-1} \boldsymbol{e}(k; \boldsymbol{w}) + \widetilde{R}(\boldsymbol{w}) \end{aligned} \tag{5}$$

where $\widetilde{R}(\boldsymbol{w}) = R(\boldsymbol{w})/N$. Both the weights $\boldsymbol{w}$ and the error covariance matrix $\boldsymbol{E}$ are unknown and must be estimated from data, i.e., $(\widehat{\boldsymbol{w}}, \widehat{\boldsymbol{E}}) = \arg\min_{\boldsymbol{w},\boldsymbol{E}} P(\boldsymbol{w}, \boldsymbol{E})$. By examining the so-called concentrated log-posterior, it is possible to show that:

$$\widehat{\boldsymbol{E}} = \frac{1}{N} \sum_{k=1}^{N} \boldsymbol{e}(k; \widehat{\boldsymbol{w}}) \boldsymbol{e}^\top(k; \widehat{\boldsymbol{w}}), \tag{6}$$

$$\widehat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} C(\boldsymbol{w} | \boldsymbol{\Lambda} = \widehat{\boldsymbol{E}}) \tag{7}$$

where the regularized cost function is defined by the sum of the *generalized mean square error* and the regularization term:

$$\begin{aligned} C(\boldsymbol{w}|\boldsymbol{\Lambda}) &= C(\boldsymbol{w}) = S(\boldsymbol{w}) + \widetilde{R}(\boldsymbol{w}) \\ &= \frac{1}{N} \sum_{k=1}^{N} \boldsymbol{e}^\top(k; \boldsymbol{w}) \boldsymbol{\Lambda}^{-1} \boldsymbol{e}(k; \boldsymbol{w}) + \widetilde{R}(\boldsymbol{w}) \end{aligned} \tag{8}$$

with $\boldsymbol{\Lambda}$ defining a positive definite, symmetric weighting matrix. The literature suggest a number of methods for carrying out this estimation, we will adopt the so-called *iterated generalized least square* (IGLS) procedure [17] which is carried out as follows:

1. Start with an arbitrary matrix[2] for $\boldsymbol{\Lambda}_{(0)}$, e.g., $\boldsymbol{\Lambda}_{(0)} = \boldsymbol{I}$, and choose initial weights $\boldsymbol{w}_{(0)}$. Set the iteration number, $i = 1$.
2. Compute the weight estimate $\widehat{\boldsymbol{w}}_{(i)}$ using $\boldsymbol{\Lambda}_{(i)}$ e.g., by employing the damped Newton method[3] (see below).
3. Compute the estimated error covariance matrix

$$\boldsymbol{\Lambda}_{(i)} = \frac{1}{N} \sum_{k=1}^{N} \boldsymbol{e}(k; \boldsymbol{w}_{(i)}) \boldsymbol{e}(k; \boldsymbol{w}_{(i)})^\top \tag{9}$$

4. Check if a specified stop criterion is met[4]. Otherwise, increment the iteration number, $i \leftarrow i + 1$, and go to step 2.

It is possible to show that the IGLS procedure converges under fairly mild conditions [12, Ch. 12.5]. Moreover, when neglecting the regularization term, the method delivers strongly consistent ($N \to \infty$)

[1] Here the errors dependence on the weights is written out explicitly.

[2] As suggested in [2] for general nonlinear models: One could perform an ensemble of q neural networks, then train them individually, and use the estimated error covariance matrix as an initial guess.

[3] A standard optimization procedure for the sum of squared error cost can also be used for a neural net with linear output neurons after a simple transform. Suppose that $\boldsymbol{\Lambda}_{(i)}^{-1} = \boldsymbol{Q}^\top \boldsymbol{Q}$. Then optimize a network using the transformed output series $\widetilde{\boldsymbol{y}} = \boldsymbol{Q}\boldsymbol{y}$. Finally, if the predicted transformed outputs are obtained by multiplying the hidden activation by the weight matrix $\widetilde{\boldsymbol{W}}$, the untransformed outputs are obtained by using the weight matrix $\boldsymbol{Q}\widetilde{\boldsymbol{W}}$ in the last layer.

[4] Note that $S(\boldsymbol{w}) \to q$, and $\boldsymbol{\Lambda}_{(i)} \to \widehat{\boldsymbol{E}}$ as $i \to \infty$.

estimators of $\boldsymbol{w}^*$ and $\boldsymbol{E}$. Here $\boldsymbol{w}^*$ denotes the optimal weights: $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} E\{S(\boldsymbol{w})\}$, with $E\{\cdot\}$ denoting the expectation w.r.t. the true input-output distribution.

The damped Newton method in step 2 of the IGLS procedure is given by

$$\boldsymbol{w}_{(i+1)} = \boldsymbol{w}_{(i)} - \mu_{(i)}\boldsymbol{J}^{-1}(\boldsymbol{w}_{(i)})\boldsymbol{\nabla}(\boldsymbol{w}_{(i)}) \tag{10}$$

where $\mu_{(i)}$ is the step-size or learning parameter. This scheme involves the calculation of the Hessian matrix $\boldsymbol{J}(\boldsymbol{w}) = \partial^2 C(\boldsymbol{w})/\partial\boldsymbol{w}\partial\boldsymbol{w}^\top$ and the gradient vector $\boldsymbol{\nabla}(\boldsymbol{w}) = \partial C(\boldsymbol{w})/\partial\boldsymbol{w}$. If we use the so-called Gauss-Newton approximation for the Hessian[5] – which ensures positive semidefiniteness – they read:

$$\boldsymbol{J}(\boldsymbol{w})=\frac{2}{N}\sum_{k=1}^{N}\boldsymbol{\Psi}(k;\boldsymbol{w})\boldsymbol{\Lambda}^{-1}\boldsymbol{\Psi}^\top(k;\boldsymbol{w}) + \frac{\partial^2\widetilde{R}(\boldsymbol{w})}{\partial\boldsymbol{w}\partial\boldsymbol{w}^\top}, \tag{11}$$

$$\boldsymbol{\nabla}(\boldsymbol{w})=\frac{-2}{N}\sum_{k=1}^{N}\boldsymbol{\Psi}(k;\boldsymbol{w})\boldsymbol{\Lambda}^{-1}\boldsymbol{e}(k;\boldsymbol{w})+\frac{\partial\widetilde{R}(\boldsymbol{w})}{\partial\boldsymbol{w}} \tag{12}$$

where $\boldsymbol{\Psi}$ is the derivate matrix of the network function, $\boldsymbol{\Psi}(k;\boldsymbol{w}) = \partial\boldsymbol{f}^\top(\boldsymbol{x}(k);\boldsymbol{w})/\partial\boldsymbol{w}$.

Considering neural networks specifically, related work can be found in [3] considering a *fixed* architecture, while [5] suggested a scheme with fixed $\boldsymbol{\Lambda} = \boldsymbol{I}$.

3. Generalization and Network Optimization

The generalization performance is expressed by the generalization error – or prediction risk – which for multiple output systems can take several forms.

Define the *generalization error matrix* as the expected outer product of the error on a test sample $(\boldsymbol{x};\boldsymbol{y})$ independent on those in the training set, i.e.,

$$\begin{aligned}\boldsymbol{G}(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}}) &= E\left\{e(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}})e^\top(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}})\right\}\\ &= \int e(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}})e^\top(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}})\cdot p_\circ(\boldsymbol{x},\boldsymbol{y})\,d\boldsymbol{x}d\boldsymbol{y}\end{aligned} \tag{13}$$

where $p_\circ(\cdot)$ denotes the true input-output density. One may not be interested in accessing the full generalization error matrix. Thus a common choice would be instead to define the *generalization error*, G, as the expected negative log-likelihood on a test sample (see Eq. (5)),

$$\begin{aligned}&G(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}}) = E\left\{\widetilde{L}\left(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}}\right)\right\} =\\ &\log\det(\widehat{\boldsymbol{E}})+\int e^\top(\widehat{\boldsymbol{w}})\widehat{\boldsymbol{E}}^{-1}e(\widehat{\boldsymbol{w}})\cdot p_\circ(\boldsymbol{x},\boldsymbol{y})\,d\boldsymbol{x}d\boldsymbol{y}\end{aligned} \tag{14}$$

since the likelihood is used for assessing the quality of a model on the training data. The generalization error depends on the actual training set through the parameters of the model, viz. the estimated weights $\widehat{\boldsymbol{w}}$ and the estimated covariance matrix $\widehat{\boldsymbol{E}}$. Thus a common generalization performance measure is to consider the *average* of $G(\boldsymbol{w},\widehat{\boldsymbol{E}})$ over all training sets of size N, $\Gamma = E_{\mathcal{T}}\{G(\boldsymbol{w},\widehat{\boldsymbol{E}})\}$.

As a natural extension of the work reported in [14], [15] for uni-variate series, we optimize the network by using Optimal Brain Damage [8] and use an algebraic estimate of the average generalization error as a stopping criterion for pruning. In line with [6], [10], [11] we employ asymptotic expansions $N \to \infty$ of the expected negative log-likelihood. Let $\boldsymbol{\theta} = (\boldsymbol{w}, \mathrm{vec}(\boldsymbol{E}))$[6], be the vector of all unknown parameters. It is possible to show that Γ to $o(1/N)$ is estimated by[7]

$$\begin{aligned}\widehat{\Gamma} &= \widetilde{L}\left(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}}\right) + \frac{q(q-1)}{N} + \frac{2m_{\mathrm{eff}}}{N}\\ &= \log\det(\widehat{\boldsymbol{E}}) + q + \frac{q(q-1)}{N} + \frac{2m_{\mathrm{eff}}}{N}\end{aligned} \tag{15}$$

where $m_{\mathrm{eff}} = \mathrm{tr}(\boldsymbol{J}^{-1}(\widehat{\boldsymbol{w}})\boldsymbol{H}(\widehat{\boldsymbol{w}}))$ with $\boldsymbol{H} = \boldsymbol{J} - \partial^2\widetilde{R}/\partial\boldsymbol{w}\partial\boldsymbol{w}^\top$ denoting the Hessian matrix of the un-regularized cost function.

Suppose that a set of N_c cross-validation (test) samples are available, then estimates of the generalization error matrix Eq. (13) as well as the generalization error in Eq. (14) become:

$$\widehat{\boldsymbol{G}}(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}}) = \frac{1}{N_c}\sum_{k=1}^{N_c} e(k;\widehat{\boldsymbol{w}})e^\top(k;\widehat{\boldsymbol{w}}), \tag{16}$$

$$\widehat{G}(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}}) = \log\det(\widehat{\boldsymbol{E}}) + \mathrm{tr}\left(\widehat{\boldsymbol{G}}(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}})\widehat{\boldsymbol{E}}^{-1}\right). \tag{17}$$

In the limit $N_c \to \infty$, $\widehat{G}(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}}) \to G(\widehat{\boldsymbol{w}},\widehat{\boldsymbol{E}})$ under mild ergodicity conditions; however, for a finite set the estimate is afflicted with error. In order to compare competing models[8] it is possible to carry out a statistical test of significant difference in performance. We use the technique suggested in [7], which incorporates the fact that the models are validated on the *same* cross-validation set. Assume that the generalization error of the two competing models are given by[9] $\widehat{G}_1 = N_c^{-1}\sum_{k=1}^{N_c}\ell_1(k)$,

[5] This involves the negligence of terms which are the product of errors e_j and second order derivative of the outputs w.r.t. the weights, $\partial^2 f_{j'}/\partial w_\ell \partial w_{\ell'}$. The terms are negligible when the weights are close to the minimum, see further [9], [12].

[6] Here $\mathrm{vec}(\cdot)$ is the reshaping operator which takes a matrix as the argument and delivers a column vector.

[7] In the calculations leading to this result, it is utilized that the Hessian matrix of the negative log-likelihood w.r.t. $\boldsymbol{\theta}$ is block-diagonal in $\boldsymbol{w}$ and $\mathrm{vec}(\boldsymbol{E})$, respectively. This is a consequence of the fact that no regularization is used on the $\boldsymbol{E}$ parameters. Further, we assume that the model is quasi complete.

[8] Consider e.g., models estimated with the QMAP approach rather than using the standard sum of squared errors (the IGLS procedure with fixed $\boldsymbol{\Lambda} = \boldsymbol{I}$).

[9] E.g., $\ell(k) = N_c^{-1}\log\det(\widehat{\boldsymbol{E}}) + e(k;\widehat{\boldsymbol{w}})^\top\widehat{\boldsymbol{E}}^{-1}e(k;\widehat{\boldsymbol{w}})$

$\widehat{G}_2 = N_c^{-1} \sum_{k=1}^{N_c} \ell_2(k)$. The relevant test statistics is given by: $T = \Delta/\mathrm{std}(\Delta)$ where $\Delta = \widehat{G}_2 - \widehat{G}_1$ and $\mathrm{std}^2(\Delta) = N_c^{-1}(N_c - 1)^{-1} \sum_{k=1}^{N_c} (\ell_2(k) - \ell_1(k) - \Delta)^2$. It is tacitly assumed that $\ell_1(k)$, $\ell_2(k)$ are i.i.d. processes in time. According to the central limit theorem T is approximately standard Gaussian distributed (zero mean, unity variance) for large N_c. For instance, the critical region associated with the hypothesis $G_2 \leq G_1$ on a α significance level is given by: $T < u_\alpha$ where where u_α is the α fractile of the standard Gaussian distribution. For $\alpha = 5\%$, we have $u_{1-\alpha/2} = 1.96$.

4. Numerical Experiments

In this section we evaluate the proposed methods on real world marine engine data. The prediction of cylinder pressure is important for engine combustion analysis or fault diagnosis and nonlinear methods for this problem are of significant current interest [13].

Under ordinary engine operation it is difficult to measure the cylinder pressure directly. The aim is consequently to measure or predict the cylinder pressure indirectly from measuring, say, strains in the cylinder. This is done by mounting a strain-gauge at the cylinder. The data set consists of simultaneous measurements of the cylinder pressure and cylinder strain sampled at 218.75Hz. The data sets are given in arbitrary units and studentisized, i.e., with zero mean and unity variance. Fig. 1 show three periods of the recorded cylinder pressure and strain time series, denoted $cp(k)$ and $sg(k)$ respectively.

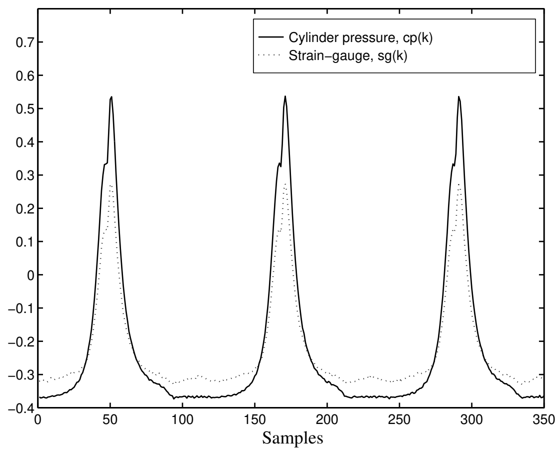

Fig. 1: Recorded cylinder pressure $cp(k)$ and strain-gauge $sg(k)$ time series from a marine engine.

4.1. Testing The QMAP Approach

A simple modelling scenario is to perform a simultaneous one step ahead prediction of the cylinder pressure and strain from a lag space of strain data. Under standard engine operation only the strain-gauge signal will be available. The prediction accuracy of this signal may provide a tool for detecting non-stationarities in the time series and for a later model consistency check. The diagnosis of the engine is done from the predicted cylinder pressure.

We used a two-layer feed forward neural network with $p = 10$ input units, $h = 8$ hidden sigmoid units and $q = 2$ linear output units, i.e., the network function can be written as:

$$f_l(\boldsymbol{x}(k); \boldsymbol{w}) = \sum_{j=1}^{h} w_{jl}^H \tanh\left(\sum_{i=1}^{p} w_{ij}^I x_i(k) + w_{i0}^I\right) + w_{0l}^H \quad (18)$$

where $\boldsymbol{w} = [\boldsymbol{w}^I, \boldsymbol{w}^H]$ is the network weight vector consisting of 106 weights. The input is a lag space of strain-gauge data $\boldsymbol{x}(k) = [sg(k), sg(k-1), \cdots, sg(k-p+1)]^\top$. The first output is the prediction of the cylinder pressure, $\widehat{y}_1(k) = cp(k+1)$, and the second output the prediction of the strain $\widehat{y}_2(k) = sg(k+1)$.

The network is trained with a set of $N = 250$ examples by using the IGLS procedure and the weight are updated by the damped Gauss-Newton method[10]. For comparison we did a similar training with $\boldsymbol{\Lambda}$ *fixed* at the identity matrix, corresponding to the standard sum of mean square errors. The regularization is a simple weight decay, i.e., $\widetilde{R}(\boldsymbol{w}) = \kappa^I |\boldsymbol{w}^I|^2/N + \kappa^H |\boldsymbol{w}^H|^2/N$, and optimized on an independent test set to yield optimal performance. When using IGLS we used $\kappa_{\mathrm{adap}}^I = \kappa_{\mathrm{adap}}^H = 0.1$, and for fixed $\boldsymbol{\Lambda}$, $\kappa_{\mathrm{fix}}^I = 5 \cdot 10^{-3}$, $\kappa_{\mathrm{fix}}^H = 2.5 \cdot 10^{-4}$.

The evolution of the training posterior Eq. (5) and the generalization error Eq. (17) on a cross-validation set consisting of $N_c = 10,000$ examples are depicted in Fig. 2. The figure indicates that updating $\boldsymbol{\Lambda}$ matrix significantly improves generalization performance. The generalization error matrices when using the IGLS procedure (adaptive error covariance matrix) and fixed error covariance matrix on the cross-validation set are given by:

$$\widehat{\boldsymbol{G}}_{\mathrm{adap}} = \begin{bmatrix} 4.07 & 2.66 \\ 2.66 & 5.53 \end{bmatrix} \cdot 10^{-4} \quad (19)$$

$$\widehat{\boldsymbol{G}}_{\mathrm{fix}} = \begin{bmatrix} 7.36 & 3.12 \\ 3.12 & 5.71 \end{bmatrix} \cdot 10^{-4} \quad (20)$$

A significant error correlation, around 0.6, is noticed. Moreover, also in the individual error variances we obtain a significant reduction when updating the $\boldsymbol{\Lambda}$ matrix.

[10] That is, the damped Newton method with the Gauss-Newton approximation for the Hessian. In order to get reasonable initial weights for the Gauss-Newton iteration we first did 15 pseudo Gauss-Newton iterations in which the off-diagonal elements of the Hessian matrix are left out. In each iteration the step-size $\mu_{(i)}$ is adapted (by successive bisections) in order to enforce the cost function to decrease.

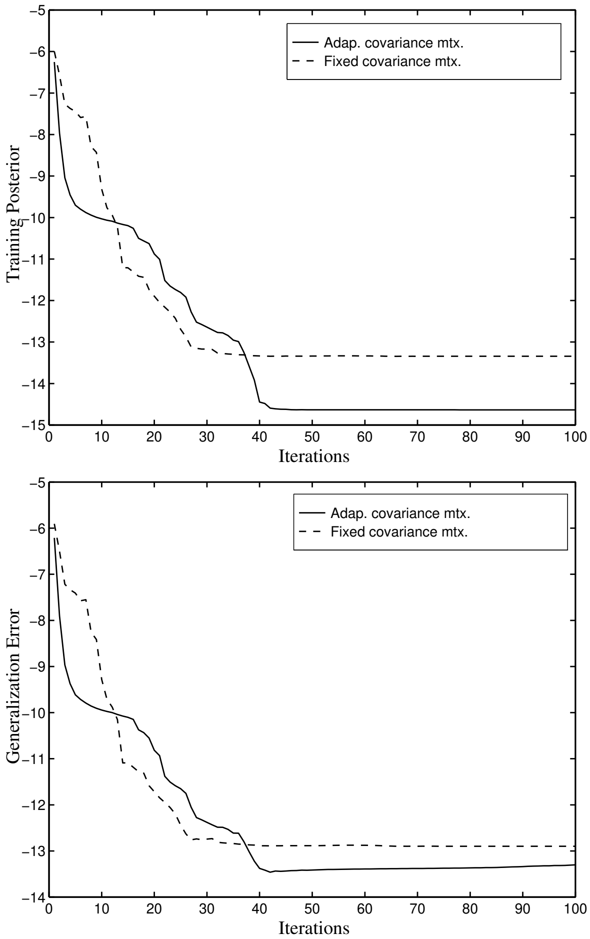

Fig. 2: Upper panel: The training posterior $\widetilde{P}(\boldsymbol{w}, \boldsymbol{\Lambda})$ as a function of number of iterations. A pseudo Gauss-Newton method is used for the first 15 iterations. The training curves for both methods seem to converge reliably. Moreover, updating the error covariance matrix increases the speed of convergence. Lower panel: The estimated generalization error $\widehat{G}$ during training. Certainly updating the covariance matrix gives better generalization on a 5% significance level, thus suggesting that the weights are more focused on their optimal values. Note that overtraining occurs, suggesting that the network is too large, and probably the weight decay is somewhat small. For comparison, a *linear* network with the same input lag space resulted in $\widehat{G} = -12.31$ which is significantly higher than using a neural network.

4.2. Network Optimization

The network trained with the IGLS procedure was pruned using optimal brain damage, and the optimal network was chosen as the one with minimal estimated average generalization error, cf. Eq. (15). In these simulation we used $\kappa^I_{\text{adap}} = \kappa^H_{\text{adap}} = 1.43 \cdot 10^{-2}$. In Fig. 3 the evolution of network pruning is depicted. Pruning reduced the number of weights from 106 to 83 and slightly improved generalization performance. The optimized network is shown in Fig. 4

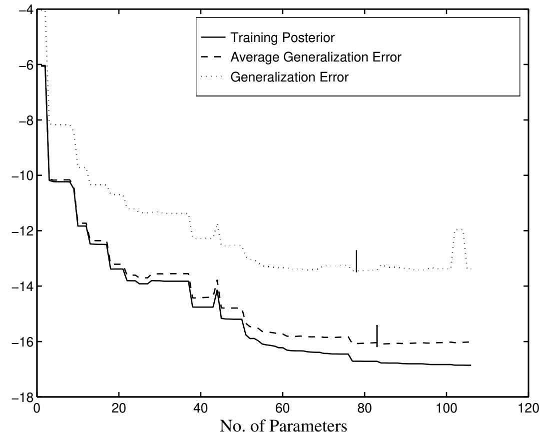

Fig. 3: The evolution of the training posterior $\widetilde{P}$, the estimated generalization error $\widehat{G}$ from the cross-validation set, and the estimated average generalization error $\widehat{\Gamma}$. The optimal network is the one with minimal $\widehat{\Gamma}$ as indicated by a vertical line. For comparison, the minimal $\widehat{G}$ is also depicted by a vertical line.

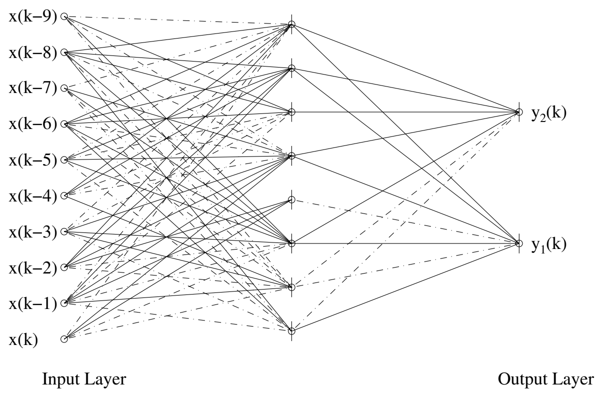

Fig. 4: Optimized neural network architecture. The input is a lag space of strain-gauge data, $y_1(k) = cp(k+1)$ is the cylinder pressure, and $y_2(k) = sg(k+1)$ is the strain-gauge. Solid lines indicate positive weights, while dashed lines corresponds to negative weights. The vertical lines indicate thresholds.

5. Conclusion

This papers focused on training and evaluation of multi-output neural network signal processing models. We adopted a quasi maximum a posteriori framework for parameter estimation. In particular, when the errors are Gaussian distributed, we suggested to use the iterative generalized least squares (IGLS) procedure [17]. The cost function is a sum of a regularization term and the generalized mean square error, in which the errors are weighted according to a matrix $\boldsymbol{\Lambda}$. When the weighting matrix equals the identity matrix, one gets the usual sum of mean square errors cost. In the IGLS procedure $\boldsymbol{\Lambda}$ is updated iteratively according to the error covariance matrix. The benefit of using this scheme is a significant improvement of generalization performance which is substantiated numerically.

Furthermore, in line with earlier work on univariate time series [14], [15], we have implemented a pruning procedure for optimization of the network architecture. We used Optimal Brain Damage for pruning and suggested to use an algebraic estimate of the generalization performance choosing the optimal network.

Acknowledgments

This research was supported by the Danish Natural Science and Technical Research Councils through the Computational Neural Network Center (CONNECT). TLF furthermore acknowledge the Danish Academy of Technical Sciences, and JL the Radio Parts Foundation for financial support.

References

[1] T.W. Anderson: *An Introduction to Multivariate Statistical Analysis*, New York, New York: John Wiley & Sons, 1984.

[2] J.J. Beauchamp & R.G. Cornell: "Simultaneous Nonlinear Estimation," *Technometrics*, vol. 16, pp. 147–192, 1966.

[3] S. Chen, S.A. Billings & P.M. Grant: "Non-Linear System Identification Using Neural Networks," in *International Journal of Control*, vol. 51, no. 6, pp. 1191–1214, 1990.

[4] S. Geman, E. Bienenstock & R. Doursat, "Neural Networks and the Bias/Variance Dilemma," *Neural Computation*, vol. 4, pp. 1–58, 1992.

[5] S. Kollias & D. Anastassiou: "An Adaptive Least Squares Algorithm for the Efficient Training of Artificial Neural Networks," *IEEE Transactions on Circuits and Systems*, vol. 36, no. 8, pp. 1092–1101, Aug. 1989.

[6] J. Larsen & L.K. Hansen: "Generalization Performance of Regularized Neural Network Models," in J. Vlontzos, J.-N. Hwang & E. Wilson (eds.), *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing IV*, Piscataway, New Jersey: IEEE, pp. 42–51, 1994.

[7] J. Larsen & L.K. Hansen: "Empirical Generalization Assessment of Neural Network Models," to appear in *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing V*, Piscataway, New Jersey: IEEE, 1995.

[8] Y. Le Cun, J.S. Denker & S.A. Solla: "Optimal Brain Damage," in D.S. Touretzky (ed.) *Advances in Neural Information Processing Systems 2, Proceedings of the 1989 Conference*, San Mateo, California: Morgan Kaufmann Publishers, 1990, pp. 598–605.

[9] L. Ljung: *System Identification: Theory for the User*, Englewood Cliffs, New Jersey: Prentice-Hall, 1987.

[10] J. Moody: "The Effective Number of Parameters: An Analysis of Generalization and Regularization in Nonlinear Learning Systems," in J.E. Moody, S.J. Hanson, R.P. Lippmann (eds.) *Advances in Neural Information Processing Systems 4, Proceedings of the 1991 Conference*, San Mateo, California: Morgan Kaufmann Publishers, 1992, pp. 847–854.

[11] N. Murata, S. Yoshizawaand & S. Amari: "Network Information Criterion — Determining the Number of Hidden Units for an Artificial Neural Network Model," *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 865–872, Nov. 1994.

[12] G.A.F. Seber & C.J. Wild: *Nonlinear Regression*, New York, New York: John Wiley & Sons, 1989.

[13] Y. Shiao & J.J. Moskwa: "Cylinder Pressure and Combustion Heat Release Estimation for SI Engine Diagnostics Using Nonlinear Sliding Observers," *IEEE Transactions on Control Systems Technology*, vol. 3, no. 1, 70–78, March 1995,

[14] C. Svarer, L.K. Hansen, and J. Larsen: "On Design and Evaluation of Tapped Delay Line Networks", In *Proceedings of the 1993 IEEE International Conference on Neural Networks*, San Francisco, vol. 1, 46–51, 1993.

[15] C. Svarer, L. K. Hansen, J. Larsen & C. E. Rasmussen: "Designer Networks for Time Series Processing," in C. A. Kamm, G. M. Kuhn, B. Yoon, R. Chellappa & S. Y. Kung (eds.), *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing 3*, Piscataway, New Jersey: IEEE, 1993, pp. 78–87.

[16] H. White: "Learning in Artificial Neural Networks: A Statistical Perspective," *Neural Computation*, vol. 1, pp. 425–464, 1989.

[17] A. Zellner: "An Efficient Method of Estimating Seemingly Unrelated Regressions and Tests for Aggregation Bias," *Journal of American Statistical Association*, vol. 57, pp. 348–368, 1962.