OPTIMIZING NEURAL NETWORK ARCHITECTURES USING GENERALIZATION ERROR ESTIMATORS

J.Larsen

This paper addresses optimization of neural network architectures. It is suggested to optimize the architecture by selecting the model with minimal estimated averaged generalization error.

We consider a least–squares (LS) criterion for estimating neural network models, i.e., the associated model weights are estimated by minimizing the LS criterion. The quality of a particular estimated model is measured by the average generalization error. This is defined as the expected squared prediction error on a novel input–output sample averaged over all possible training sets.

An essential part of the suggested architecture optimization scheme is to calculate an estimate of the average generalization error. We suggest to use the *GEN*–estimator [9, 10] which allows for dealing with nonlinear, incomplete models; i.e., models which are not capable of modeling the underlying nonlinear relationship perfectly. In most neural network applications it is impossible to suggest a perfect model, and consequently the ability to handle incomplete models is urgent.

A concise derivation of the *GEN*–estimator is provided, and its qualities is demonstrated by comparative numerical studies.

1. INTRODUCTION

Selection of proper model architectures from a finite training set is a fundamental issue for application of neural network models. In connection with multi–layer feed–forward networks the model architecture is determined by the number of layers, the number of units within the layers, and the connectivity among adjoining layers. The objective is to design architectures with high quality which — in this work — is expressed in terms of the generalization error defined as the expected squared prediction error [1]; that is, the mean square error on a novel random sample which is independent on the samples used for training of the network.

The paper addresses the possibility of optimizing the model architecture by using generalization error *estimates*. Contemplate a set of candidate architectures; then for each architecture estimate the generalization error, and select the architecture with minimal estimated generalization error.

[1] In the literature also known as the prediction risk (e.g., [11]) or the integrated mean square error.

Consider the following data generating nonlinear system:

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

where the scalar output, $y(k)$, (k is the discrete time index) is generated as the sum of a nonlinear mapping, $g(\cdot)$, of the input vector $\boldsymbol{x}(k)$ and an additive noise $\varepsilon(k)$.

Assumption 1. *The input $\boldsymbol{x}(k)$ and the inherent noise $\varepsilon(k)$ are assumed to be strictly stationary sequences. Furthermore, $E\{\varepsilon(k)\} = 0$ and $E\{\varepsilon^2(k)\} = \sigma_\varepsilon^2 < \infty$.*

Let $\mathcal{F}$ be a set of nonlinear functionals parameterized by an m-dimensional weight vector $\boldsymbol{w} = [w_1, w_2, \ldots, w_m]^\top$ ($^\top$ denotes the transpose operator). In general it is assumed that the functionals are nonlinear in $\boldsymbol{w}$. Multi–layer feed–forward neural networks with hidden units are examples of $\mathcal{F}$. Let $f(\cdot) \in \mathcal{F}$ then the non-recursive model with additive error is defined by:

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

where $e(k; \boldsymbol{w})$ is the additive error [2]. The prediction of $y(k)$, say $\hat{y}(k)$, is given by

$$\hat{y}(k) = f(\boldsymbol{x}(k); \boldsymbol{w}). \tag{3}$$

When referring to a nonlinear model, $f(\cdot)$ is considered to be nonlinear in the weights; otherwise, the model is linear, i.e., $\hat{y}(k) = \boldsymbol{w}^\top \boldsymbol{x}(k)$.

Normally neural networks models are used in situations where only scanty structural knowledge of the "true" system $g(\cdot)$ is available. This may be due to the fact that the task possesses a very complex nature which — more or less — precludes conventional approaches. In such cases neural network models are flexible tools since they carry the universal approximation ability, see e.g., [7]. However, the framework involves some hurdles:

- In general, the neural network model does not enable interpretations of the structure of $g(\cdot)$; in fact the purpose of model design is here considered to be low generalization error, i.e., high prediction accuracy.
- Both the weights and the model architecture have to be estimated from data. Consequently, it is expected that a relatively large data set is required.
- Models with a finite number of weights are infrequently capable of modeling the system perfectly.

[2] Note the dependence on the weights sometimes is accentuated.

In connection with the last item we make the following definition:

Definition 1. *If* $\exists\, \boldsymbol{w}^\circ,\ \forall \boldsymbol{x}:\ g(\boldsymbol{x}) \equiv f(\boldsymbol{x};\ \boldsymbol{w}^\circ)$ *the model is signified as complete otherwise as incomplete.* $\boldsymbol{w}^\circ$ *is denoted the true weight vector.*

Consequently, we claim that most neural network models are incomplete.

The paper is organized as follows: In Section 2 the fundamentals of training and generalization are presented. Various generalization error estimators are reviewed in Section 3, and the *GEN*–estimator [3] [9, 10] is suggested for the estimating the generalization error of nonlinear, incomplete models. Section 4 provides some comparative numerical experiments which illustrate the properties of the *GEN*–estimator, and finally Section 5 states the conclusions.

2. TRAINING AND GENERALIZATION

Given a training set: $\mathcal{T} = \{\boldsymbol{x}(k),\ y(k)\},\ k = 1, 2, \ldots, N$, where N is the training set size, the model is estimated by minimizing some cost function, say $S_N(\boldsymbol{w})$. In this work the classical least squares (LS) cost is employed [4]:

$$S_N(\boldsymbol{w}) = \frac{1}{N}\sum_{k=1}^{N} e^2(k;\ \boldsymbol{w}) = \frac{1}{N}\sum_{k=1}^{N}\Big[y(k) - f(\boldsymbol{x}(k);\ \boldsymbol{w})\Big]^2. \tag{4}$$

Assumption 2. *Define* Ω *as the m-dimensional weight space. Assume that* $\widehat{\boldsymbol{w}}$ *uniquely minimizes* $S_N(\boldsymbol{w})$ *within a compact subset* $\mathcal{W} \subseteq \Omega$, *and furthermore*

$$\frac{\partial S_N(\widehat{\boldsymbol{w}})}{\partial \boldsymbol{w}} = 0, \quad \boldsymbol{a}^\top \frac{\partial^2 S_N(\widehat{\boldsymbol{w}})}{\partial \boldsymbol{w}\partial \boldsymbol{w}^\top}\boldsymbol{a} > 0, \quad \forall \boldsymbol{a} \neq 0. \tag{5}$$

Note that $\widehat{\boldsymbol{w}}$ is merely a particular minimizer and not necessarily the global minimum — even though it is preferred. The occurrence of multiple minima — including the global — is well–known for feed–forward neural networks, e.g., due to symmetries as mentioned in e.g., [5].

The training performance $S_N(\widehat{\boldsymbol{w}})$ is usually not a reliable measure of the quality of a model because it depends on the actual training set. A reliable quality measure is the *generalization error*, G, (e.g., [11, 12]) which is defined as the expected, squared prediction error on a test sample, $\{\boldsymbol{x}_t,\ y_t\}$ (denoting t for test), which is *independent* of the training set but with identical distribution, i.e.,

$$G(\boldsymbol{w}) = E_{\boldsymbol{x}_t, e_t}\Big\{[y_t - f(\boldsymbol{x}_t;\ \boldsymbol{w})]^2\Big\}. \tag{6}$$

[3] Generalization error estimate for incomplete nonlinear models.

[4] A more sophisticated cost function which incorporates a regularization term — like weight decay regularization (see e.g., [6] — is considered in [10].

$E_{\boldsymbol{x}_t,\boldsymbol{\varepsilon}_t}\{\cdot\}$ denotes expectation with respect to the joint p.d.f. of $[\boldsymbol{x}_t,\ \boldsymbol{\varepsilon}_t]$. We mainly focus on the generalization error evaluated at the estimated weights, i.e., $G(\widehat{\boldsymbol{w}})$; however, consider also $G(\boldsymbol{w})$ as a function of the weights. In that context we state the following assumption:

Assumption 3. *Assume the existence of a compact subset $\mathcal{W}^* \subseteq \Omega$ such that the optimal weight vector $\boldsymbol{w}^*$ uniquely minimizes $G(\boldsymbol{w})$ within $\mathcal{W}^*$, and furthermore*

$$\frac{\partial G(\boldsymbol{w}^*)}{\partial \boldsymbol{w}} = 0, \quad \boldsymbol{a}^\top \frac{\partial^2 G(\boldsymbol{w}^*)}{\partial \boldsymbol{w} \partial \boldsymbol{w}^\top} \boldsymbol{a} > 0, \quad \forall \boldsymbol{a} \neq 0. \tag{7}$$

Note that the optimal weight vector reflects the "best" model within the actual set $\mathcal{F}$ and within the compact subset $\mathcal{W}^*$; that is, the model obtained by training on an infinite training set [5].

The generalization error of the estimated model, $G(\widehat{\boldsymbol{w}})$, depends on the present training set through $\widehat{\boldsymbol{w}}$; consequently, if another training set of equal size where employed another generalization error emerges. The reason we believe that the model under consideration is pretty bad or good could be due to the nature of the actual training set rather than the chosen architecture. In order to eliminate such effects we focus on the *average generalization error* defined by:

$$\Gamma = E_{\mathcal{T}}\{G(\widehat{\boldsymbol{w}})\} \tag{8}$$

where $E_{\mathcal{T}}\{\cdot\}$ denotes expectation w.r.t. to the training set, $\mathcal{T}$. Note that determination of Γ requires knowledge of the system, the model and the joint p.d.f. of $[\boldsymbol{x},\ \boldsymbol{\varepsilon}]$; q.e., generally this quantity is not accessible. HOwowever, it is still possible to suggest various estimators, which is the topic of the next section.

3. GENERALIZATION ERROR ESTIMATION

In e literature several attempts have been made in order to estimate the generalization error of both linear and nonlinear models. Direct methods are based on cross–validation techniques, see e.g., [2, Sections 6.8 & 8.3], which are advantageous since they only require mild conditions on the error signal. The usual cross–validation estimator (called the C–estimator) consist in using $N - N_c$ of the samples in the training set for estimation of the weights (i.e., $\widehat{\boldsymbol{w}}$). The remaining — preferably independent — N_c samples for calculating the generalization error estimate as the average of the squared error signal, i.e.,

$$C_\nu(\mathcal{T}) = \frac{1}{N_c} \sum_{k=N-N_c+1}^{N} e^2(k;\widehat{\boldsymbol{w}}) \tag{9}$$

[5] This is due to the fact that $\lim_{N\to\infty} S_N(\boldsymbol{w}) = G(\boldsymbol{w})$ provided that $e^2(k;\ \boldsymbol{w})$ is a mean–ergodic sequence.

where the index ν is defined as $\nu = N - N_c/N \cdot 100\%$. The setting of N_c — or equivalently, ν — involves a bias/variance trade off. On the one hand N_c should be small so that all data can be used for training (small bias); on the other hand N_c should be large to obtain small variance since (under mild assumptions) $\lim_{N_c \to \infty} C_\nu = G(\hat{w})$.

A different cross–validation estimate is the leave–one–out cross–validation estimate (L–estimate). The idea is to successively leave out one sample in the training set for cross–validation and then use the rest for training [6], i.e.,

$$L(\mathcal{T}) = \frac{1}{N} \sum_{j=1}^{N} e^2(j; \hat{w}^{(j)}), \tag{10}$$

where $\hat{w}^{(j)}$ is the weight estimate obtained by training on the training set: $\{x(k); y(k)\}$,

$$\begin{array}{lll} k = 2, 3, \ldots, N, & & j = 1 \\ k = 1, 2, \ldots, j-1, j+1, \ldots, N, & & j \in [2; N-1] \\ k = 1, 2, \ldots, N-1, & & j = N. \end{array} \tag{11}$$

Notice that in the L–estimator $N - 1$ samples are used for training, i.e., we expect a small bias [7]. On the other hand, the variance may still be significant.

Indirect methods are based on deriving algebraic expression for the average generalization error from various assumptions on the system and the model. The immediate benefits are:

- All data can be used for training.
- It is possible to understand how characteristic parameters such as the number of weights and training samples influence the generalization error.

The classical Final Prediction Error estimator (*FPE*) [1] and the *FIS*–estimator [3] focus on complete models, while [8] and the Generalized Prediction Error estimator (*GPE*) [11, 12] focus on incomplete models, which are claimed to be the most common in a neural network modeling context.

In [8] a generalization error estimator for linear incomplete models is developed. The estimate requires knowledge of the estimated weights $\hat{w}_i$, $i = m+1, m+2, \ldots, m^\circ$ where m° denotes the dimension for which the model becomes complete. Unfortunately, these estimated weights are not accessible when fitting with only m weights. Therefore, the final result of [8] is essentially the *FPE*–estimator.

[6] The leave–one–out cross–validation technique is a special case of the general leave–v–out technique.

[7] It should be emphasized that dependence among the samples causes extra bias.

The *GPE*–estimator [11, 12] is claimed to estimate the generalization error for both nonlinear and incomplete (in [11] denoted biased) models when using the sum of $S_N(\boldsymbol{w})$ (the LS cost function) and a regularization term as the cost function. However, in Section 3.1 which presents the *GEN*–estimator [9, 10] with validity for both incomplete and nonlinear models, it is established that the error, $e(k;\ \boldsymbol{w})$, and the input, $\boldsymbol{x}(k)$, are *not* independent unless the model is complete. This dependence is not taken into account in the *GPE*–estimator.

3.1. The generalization error estimator for incomplete nonlinear models

In this subsection the generalization error estimate for incomplete nonlinear models, called *GEN* is presented. The estimator can be viewed as an extension of the *FPE*– and *GPE*–estimators. The advantages and drawbacks encumbered with the *GEN*–estimator are:

Advantages:

- All data in the training set are used for estimating the weights. This is especially important in situations where training data are sparse. This is not the case when considering the cross–validation estimators.
- The model may be nonlinear as well as linear in the weights.
- Both incomplete and complete models are treated.
- The input and the noise may in general be correlated and dependent.
- Noiseless systems are also considered.
- The weight estimate, $\widehat{\boldsymbol{w}}$, is not required to be the global minimum of the cost function.
- It is ensured that the estimator becomes an estimator of Γ, i.e., the estimate provides for fluctuations in both the input and the inherent noise and in that way extending the *GPE*–estimator.

Drawbacks [8]:

- A fundamental condition is that the training set is large; however, all considered estimators in fact require this assumption.
- The model is assumed to be properly approximated in the vicinity of $\widehat{\boldsymbol{w}}$ by a linear expansion in the weights.
- The weight estimate, $\widehat{\boldsymbol{w}}$, is assumed to be locally unique, and the cost function is required to have a non–zero curvature around $\widehat{\boldsymbol{w}}$.

[8] The assumptions mentioned also enter the derivation of the *FPE*– and *GPE*–estimators.

- Only local effects due to fluctuations in the weight estimate are considered [9].
- Training algorithm effects are not considered [10]. That means, normally the training procedure result in an estimate different from $\hat{\boldsymbol{w}}$ which locally minimizes $S_N(\boldsymbol{w})$.

In order to ensure the validity of the *GEN*–estimator we state some additional assumptions:

Assumption 4. *$\boldsymbol{x}(k)$ is an M–dependent stationary sequence, i.e., $\boldsymbol{x}(k)$, $\boldsymbol{x}(k+\tau)$ are independent $\forall\,|\tau| > M$ (A weaker assumption is that $\boldsymbol{x}(k)$ is a strongly mixing sequence [13, p.62]).*

Assumption 5. *Let the minimization of S_N on the training set result in the estimate: $\hat{\boldsymbol{w}}$ [11]. Assume the existence of an optimal weight vector $\boldsymbol{w}^*$, define the weight fluctuation, $\Delta\boldsymbol{w} = \hat{\boldsymbol{w}} - \boldsymbol{w}^*$, and let Ξ denote the region of validity around $\boldsymbol{w}^*$. While $\Delta\boldsymbol{w} \in \Xi$ the remainders of the following second order Taylor series expansion are assumed negligible:*

$$G(\hat{\boldsymbol{w}}) \approx G(\boldsymbol{w}^*) + \Delta\boldsymbol{w}^{\top}\mathbf{H}(\boldsymbol{w}^*)\Delta\boldsymbol{w} \tag{12}$$

where $\mathbf{H}(\boldsymbol{w}^*)$ is the nonsingular (by Eq.(7)) Hessian matrix

$$\mathbf{H}(\boldsymbol{w}^*) = \frac{1}{2}\frac{\partial^2 G(\boldsymbol{w}^*)}{\partial\boldsymbol{w}\partial\boldsymbol{w}^{\top}} = E_{\boldsymbol{x}_t,e_t}\Big\{\boldsymbol{\psi}_t(\boldsymbol{w}^*)\boldsymbol{\psi}_t^{\top}(\boldsymbol{w}^*) - \boldsymbol{\Psi}_t(\boldsymbol{w}^*)e_t(\boldsymbol{w}^*)\Big\}, \tag{13}$$

$\boldsymbol{\psi}_t(\boldsymbol{w}^*) = \partial f(\boldsymbol{x}_t;\,\boldsymbol{w}^*)/\partial\boldsymbol{w}$ and $\boldsymbol{\Psi}_t(\boldsymbol{w}^*) = \partial\boldsymbol{\psi}(\boldsymbol{x}_t;\,\boldsymbol{w}^*)/\partial\boldsymbol{w}^{\top}$.

Further, assume that the remainders of expanding S_N around $\hat{\boldsymbol{w}}$ to the second order are negligible, i.e.,

$$S_N(\boldsymbol{w}^*) \approx S_N(\hat{\boldsymbol{w}}) + \Delta\boldsymbol{w}^{\top}\mathbf{H}_N(\hat{\boldsymbol{w}})\Delta\boldsymbol{w} \tag{14}$$

where $\mathbf{H}_N(\hat{\boldsymbol{w}})$ is the nonsingular Hessian given by

$$\mathbf{H}_N(\hat{\boldsymbol{w}}) = \frac{1}{2}\frac{\partial^2 S_N(\hat{\boldsymbol{w}})}{\partial\boldsymbol{w}\partial\boldsymbol{w}^{\top}} = \frac{1}{N}\sum_{k=1}^{N}\boldsymbol{\psi}(k;\,\hat{\boldsymbol{w}})\boldsymbol{\psi}^{\top}(k;\,\hat{\boldsymbol{w}}) - \boldsymbol{\Psi}(k;\,\hat{\boldsymbol{w}})e(k;\,\hat{\boldsymbol{w}}), \tag{15}$$

$\boldsymbol{\psi}(k;\,\hat{\boldsymbol{w}}) = \partial f(\boldsymbol{x}(k);\,\hat{\boldsymbol{w}})/\partial\boldsymbol{w}$ and $\boldsymbol{\Psi}(k;\,\hat{\boldsymbol{w}}) = \partial\boldsymbol{\psi}(\boldsymbol{x}(k);\,\hat{\boldsymbol{w}})/\partial\boldsymbol{w}^{\top}$.

[9] By that, we explicitly mean fluctuations which take place within the hypersphere, Ξ, cf.As.5.

[10] In [4] the effects of training with the back–propagation algorithm (see e.g., [6]) are taken into account. The considered models are linear and complete.

[11] Note that the weight estimate is highly dependent on the chosen weight estimation algorithm due to local optimization, initial conditions, etc. An alternative algorithm used on the same training set may therefore result in a different weight estimate.

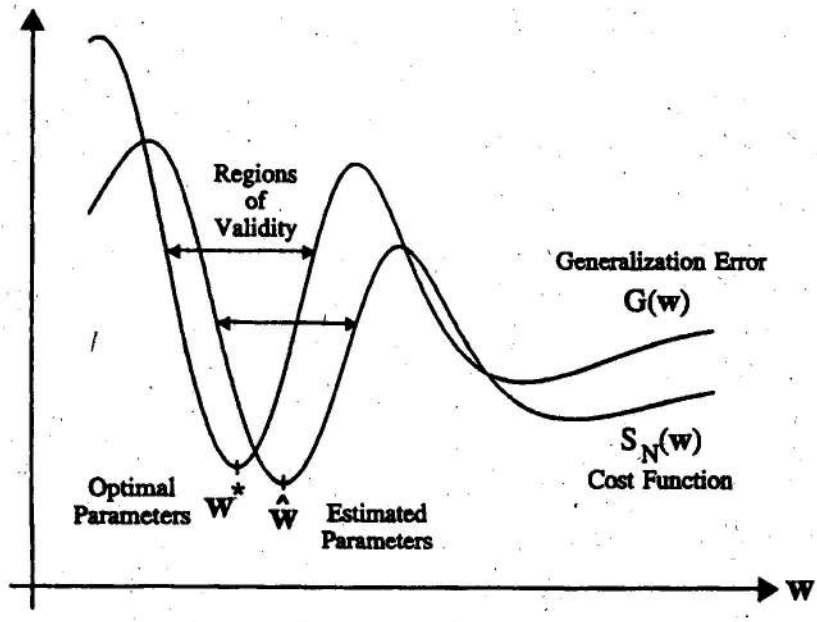

Fig. 1. Example of cost function $S_N(w)$ and generalization error $G(w)$ which fulfill As.5.

In Fig.1 an example of a cost function which fulfill As.5 is shown.

Assumption 6. *Assume large training sets, i.e., $N \to \infty$ and $N \gg M$ where M is the dependence length defined in As.4. Further, assume that m is finite.*

Definition 2. *Provided that the system* [12] *and the model is defined by Eq.(1), (2) and the As.2 through As.6 hold then GEN is defined as a consistent $(N \to \infty)$ estimator of Γ, Eq.(8).*

Theorem 1. *Suppose that the system and the model are given by Eq.(1), (2) and the model is* **incomplete** *or alternatively* **complete** *with the restriction that w^* defined by As.3, As.5 is not the global optimum of $G(w)$. Further, suppose that As.2 through As.6 hold. The GEN–estimate is then given by:*

$$GEN = S_N(\hat{w}) + \frac{2}{N} \cdot \mathrm{tr}\left[\left(\mathbf{R}(0) + \sum_{\tau=1}^{M} \frac{N-\tau}{N}\left(\mathbf{R}(\tau) + \right.\right.\right.$$
$$\left.\left.\left. + \mathbf{R}^{\top}(\tau)\right)\right)\mathbf{H}_N^{-1}(\hat{w})\right] \qquad (16)$$

[12]The formulation in this paper is based on a data generating system given by Eq.(1). This system definition is appealing as regards intèrpretation; however, it is possible give a formulation which does not require an explicit system definition. The final result of the estimator is unchanged.

where $\mathrm{tr}[\cdot]$ *is the trace operator, and the correlation matrices* $\mathbf{R}(\tau)$, $0 \le \tau \le M$ *are calculated as:*

$$\mathbf{R}(\tau) = \frac{1}{N}\sum_{k=1}^{N-\tau} \boldsymbol{\psi}(k;\,\widehat{\boldsymbol{w}})e(k;\,\widehat{\boldsymbol{w}})\boldsymbol{\psi}^{\top}(k+\tau;\,\widehat{\boldsymbol{w}})e(k+\tau;\,\widehat{\boldsymbol{w}}). \tag{17}$$

Sketch of Proof and Discussion. The basis of the proof is the Taylor series expansions given in Eq.(12), (14) and an order, $o(1/N)$, expansion. It turns out that the *GEN*–estimator is an unbiased estimator of Γ to $o(1/N)$ under the stated assumptions.

Taking the expectation, $E_{\mathcal{T}}\{\cdot\}$ (i.e., w.r.t. the training set) of Eq.(12), (14) it is possible to substitute Eq.(14) into Eq.(12) by using the identity

$$E_{\mathcal{T}}\{S_N(\boldsymbol{w}^*)\} = E_{\mathcal{T}}\{G(\boldsymbol{w}^*)\}. \tag{18}$$

That is, this enables an expression for $\Gamma = E_{\mathcal{T}}\{G(\widehat{\boldsymbol{w}})\}$ in terms of training data. When evaluating the expectations it is important to notice that the error (cf. Eq.(1) and (2))

$$e(k;\,\boldsymbol{w}) = \varepsilon(k) + g(\boldsymbol{x}(k)) - f(\boldsymbol{x}(k);\,\boldsymbol{w}) \tag{19}$$

depends *both* on $\boldsymbol{x}(k)$ and $\varepsilon(k)$ unless the model is complete and $\boldsymbol{w}^*$ is the global optimum since $g(\boldsymbol{x}) \equiv f(\boldsymbol{x};\,\boldsymbol{w}^*)$ in that case [13]. In [10] the details of the proof are given and the estimate is further extended to treat other cost functions, for instance the LS–cost with inclusion of a regularization term — e.g., weight decay regularization. It should be noted that the derivation is valid also when dealing with noise free systems, i.e., $\sigma_\varepsilon^2 = 0$.

In order to emphasize the difference between the *GEN*– and *GPE*–estimators [11, 12] consider a incomplete *linear* model and suppose there is no regularization term (e.g., no weight decay) in the cost function. In that case the *GPE*–estimator equals the *FPE*–estimator [1] $FPE = S_N(\widehat{\boldsymbol{w}})(N+m)/(N-m)$ which obviously is different from the one given in Eq.(16).

Theorem 2. *Suppose that the system and the model are given by Eq.(1), (2), that the model is* **complete**, *and that* $\boldsymbol{w}^*$ *defined by As.3, As.5 is the global optimum of* $G(\boldsymbol{w})$. *Further, suppose that As.2 trough As.6 hold and the noise variance* $E\{\varepsilon^2\} = \sigma_\varepsilon^2 \neq 0$. *The GEN–estimate then coincides with the FPE–estimate [1]:*

$$GEN = FPE = \frac{N+m}{N-m}S_N(\widehat{\boldsymbol{w}}), \quad N > m. \tag{20}$$

Proof See the sketch above and [10].

[13] Note that $g - f$ may be equal to a constant which is independent of $\boldsymbol{x}$. However, this case never occurs if the model contains a bias term, as e.g., in a multi–layer feed–forward neural network with a linear output neuron.

4. NUMERICAL EXPERIMENTS

In this section we present a comparative numerical study in order to demonstrate the usefulness of the *GEN*–estimator. The *GEN*–estimator is compared to the *FPE*–estimator Eq.(20), the cross–validation estimator, C_{50} Eq.(9), and the leave–one–out cross–validation estimator, L, Eq.(10).

We form Q *independent* training sets with sizes:

$$N = N_{\min},\ N_{\min}+1,\ \ldots,\ N_{\max}. \tag{21}$$

The s'th training set with size N, $\mathcal{T}_N^{(s)}$, is given by:

$$\mathcal{T}_N^{(s)} = \left\{ \boldsymbol{x}^{(s)}(k);\ y^{(s)}(k) \right\} \tag{22}$$

where $s \in [1; Q]$, $N \in [N_{\min};\ N_{\max}]$, and $k \in [1; N]$. The weight estimate obtained by using the training set, $\mathcal{T}_N^{(s)}$, is denoted by $\widehat{\boldsymbol{w}}^{(s)}$.

The "true" average generalization error, Γ, is estimated by:

$$\widehat{\Gamma}_G = \left\langle G(\widehat{\boldsymbol{w}}^{(s)}) \right\rangle = \frac{1}{Q} \sum_{s=1}^{Q} G(\widehat{\boldsymbol{w}}^{(s)}) \tag{23}$$

where $\langle \cdot \rangle$ denotes the average w.r.t. the Q training sets. Since we are able to design the experiment by hand, it is assumed that it is possible to determine the generalization error, $G(\widehat{\boldsymbol{w}}) = E_{\boldsymbol{x}_t, e_t}\{e_t^2(\widehat{\boldsymbol{w}})\}$. Obviously, $\lim_{Q\to\infty} \widehat{\Gamma}_G = \Gamma$, so when Q is large [14] Γ_G becomes an accurate estimate of Γ.

The quality of the various estimators is quantified by three different measures: relative bias, *RB*, averaged squared error, *ASE*, and probability of proximity, Π which are defined by:

$$RB = \frac{\widehat{\Gamma}_G - \langle \widehat{\Gamma}(\mathcal{T}_N^{(s)}) \rangle}{\widehat{\Gamma}_G} \tag{24}$$

$$ASE = \left\langle \left[\widehat{\Gamma}(\mathcal{T}_N^{(s)}) - \widehat{\Gamma}_G \right]^2 \right\rangle \tag{25}$$

$$\begin{aligned} \Pi &= \Pr\left\{ |GEN - \Gamma| < |\widehat{\Gamma} - \Gamma| \right\} \approx \\ &\approx \left\langle \mu\left(\left| \widehat{\Gamma}(\mathcal{T}_N^{(s)}) - \widehat{\Gamma}_G \right| - \left| GEN(\mathcal{T}_N^{(s)}) - \widehat{\Gamma}_G \right| \right) \right\rangle. \end{aligned} \tag{26}$$

$\widehat{\Gamma} \in \{GEN,\ FPE,\ C_{50},\ L\}$ denotes a particular estimator and $\mu(\cdot)$ is the step function.

A linear system and a simple neural network are under consideration.

[14] Note that Q can be chosen arbitrarily large independent of the number of training samples, N.

4.1. Linear system

The linear system is given by:

$$y(k) = y^{\circ}(k) + \varepsilon(k) = [x(k),\ x^2(k)]w^{\circ} + \varepsilon(k) \tag{27}$$

where $w^{\circ} = [1, 1]^{\mathsf{T}}$. The input $x(k) = \sum_{n=0}^{15} b_n u(k-n)$ where $u(k)$ is an i.i.d. Gaussian sequence with zero mean and unit variance. b_n is designed to implement a low–pass filter [15] with normalized cutoff frequency 0.01. $x(k)$ is consequently colored and M–dependent (see As.4 above) with $M = 15$. $\varepsilon(k)$ is an i.i.d. Gaussian noise sequence with zero mean, $\sigma_{\varepsilon}^2 = 0.2 \cdot E_{x(k)}\{(y^{\circ})^2(k)\}$, and independent of $u(k)$. The model used is incomplete [16] and given by:

$$y(k) = wx(k) + e(k;\ w). \tag{28}$$

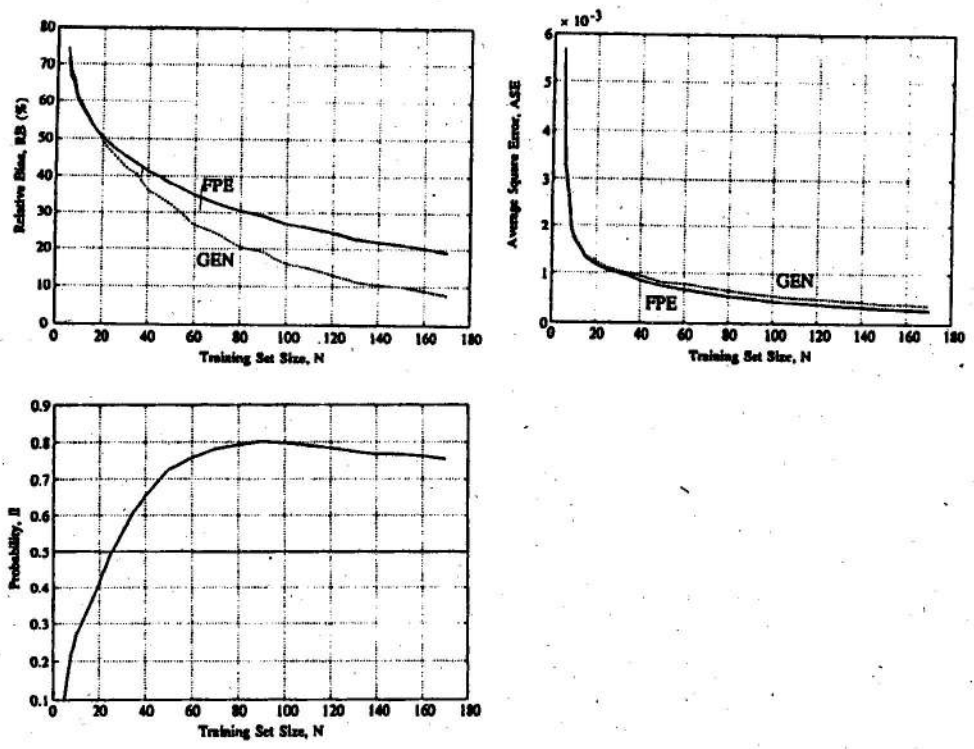

Fig. 2. Comparison of *GEN* and *FPE* within the linear model Eq.(28). We used $Q = 3 \cdot 10^4$ as $5 \le N \le 9$ and $Q = 2 \cdot 10^4$ as $10 \le N \le 170$.

[15] The design is performed by the MATLAB (The Math Works, Inc.) M–file "fir1" which uses a Hamming windowed ideal impulse response (i.e., $\mathrm{sinc}(x)$).

[16] The $x^2(k)$ term is left out.

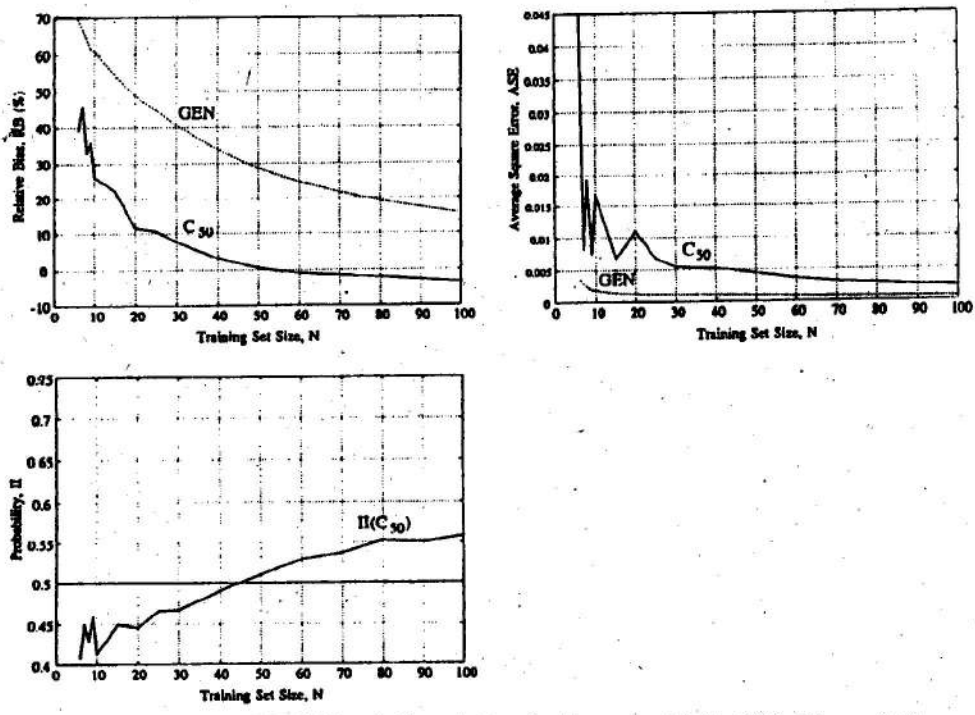

Fig. 3. Comparison of *GEN* and C_{50} within the linear model Eq.(28). We used $Q = 4 \cdot 10^4$, $\forall N$.

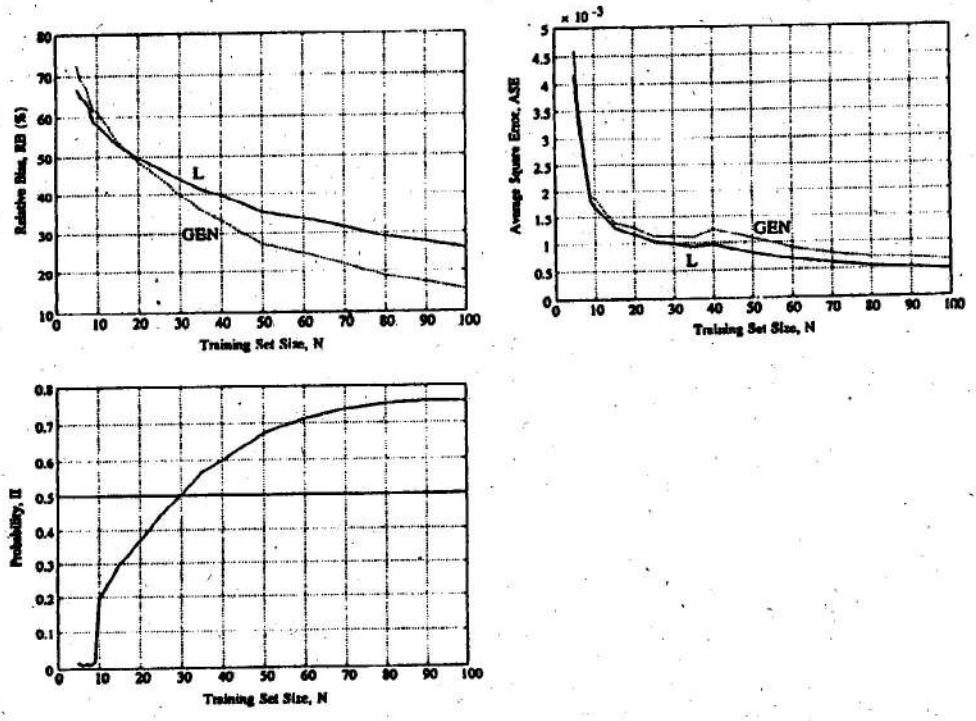

Fig. 4. Comparison of *GEN* and *L* within the linear model Eq.(28). We used $Q = 1 \cdot 10^4$ as $5 \leq N \leq 9$ and $Q = 5 \cdot 10^3$ as $10 \leq N \leq 100$.

The weight, w, is estimated by:

$$\widehat{w} = \left[\sum_{k=1}^{N} x^2(k)\right]^{-1} \cdot \sum_{k=1}^{N} x(k)y(k). \tag{29}$$

Knowing the details of the system Eq.(27) it is possible to compute analytically the true generalization error $G(\widehat{w})$ according to Eq.(6). Let $E\{\cdot\}$ denote expectation w.r.t. x_t and ε_t. Now, noting that x_t is Gaussian:

$$\begin{aligned} G(\widehat{w}) &= E\left\{\left[w_1^\circ x_t + w_2^\circ x_t^2 + \varepsilon_t - \widehat{w}x_t\right]^2\right\} = \\ &= (w_1^\circ - \widehat{w})^2 E\{x_t^2\} + 3(w_2^\circ E\{x_t^2\})^2 + \sigma_\varepsilon^2. \end{aligned} \tag{30}$$

The result of comparing *GEN* with *FPE*, C_{50}, and L is shown in Fig.2g.4. As shown in Fig.2 $|RB(GEN)| < |RB(FPE)|$ when $N > 15$ and vice versa when $N \leq 15$ [17]. When $N > 100$ the $|RB(GEN)|$ is less than or equal to one half of the $|RB(FPE)|$. The *ASE* of *GEN* is slightly higher than that of *FPE* for most values of N. One could then argue that nothing speaks in favor of using the *GEN*–estimator since what is gained in lower bias is lost in increased variance. However, notice that *ASE* is merely one particular measure which equally balance the squared first and second order moments (i.e., squared bias and variance) of the estimator distribution over different training sets. Inspecting the probability of proximity, Π, it is seen that $\Pi \approx 0.7$ in the interval $60 \leq N \leq 170$ indicating that *GEN* is closer to Γ than *FPE*. Hence, it is concluded that one should prefer *GEN* when $N \geq 25$.

When comparing *GEN* to C_{50}, as shown in Fig.3 it turns out that C_{50} has a smaller *RB*; however, the $ASE(C_{50})$ is significantly higher. In fact:

$$3 \lesssim \frac{ASE(C_{50})}{ASE(GEN)} \lesssim 13. \tag{31}$$

In addition, $\Pi(C_{50}) > 0.5$ as $N \geq 50$ and reach a level of approx 0.55. The probability of proximity seems not particularly high; however, it is judged that the huge *ASE* makes the C_{50}–estimator a dubious alternative.

Concerning the L–estimator cf.Fig.4 we found that $|RB(GEN)| < |RB(L)|$ as $N > 15$ and otherwise, vice versa. $|RB(GEN)|$ constitutes approximately the half of $|RB(L)|$ when $N = 100$. The *ASE* of the two estimators are fairly comparable for all N–values. Furthermore, $\Pi > 0.5$ as $N \geq 30$ and reaches as level at approx. 0.75 at $N = 100$. Hence, one may prefer *GEN* to L when $N \geq 30$.

[17] The inequalities are significant on a 0.5% significance level, see [10, Ch.7].

4.2. Simple neural network

Consider a simple nonlinear system which consists of a single nonlinear neuron:

$$y(k) = y^{\circ}(k) + \varepsilon(k) = h(\boldsymbol{x}^{\mathsf{T}}(k)\boldsymbol{w}^{\circ}) + \varepsilon(k), \tag{32}$$

$$h(z) = \exp\left(-\left(\frac{z-\nu}{\eta}\right)^2\right) - \exp\left(-\left(\frac{z+\nu}{\eta}\right)^2\right) \tag{33}$$

where $\boldsymbol{w}^{\circ} = [3, 3]^{\mathsf{T}}$. Let $\boldsymbol{u}(k)$ be a two–dimensional i.i.d. Gaussian sequence with zero mean and $E\{u_i^2(k)\} = 1$, $E\{u_1(k)u_2(k)\} = 0.5$. b_n is given as in the preceding subsection and $x_i(k) = \sum_{n=0}^{15} b_n u_i(k-n)$, $i = \{1, 2\}$. $\varepsilon(k)$ is an i.i.d. Gaussian noise sequence with zero mean, $\sigma_\varepsilon^2 = 0.1 \cdot E_{\boldsymbol{x}(k)}\{(y^{\circ})^2(k)\}$, and independent of $u_i(k)$. The activation function $h(z)$ is chosen to be a sum of two Gaussian functions in order to enable the evaluation of the true generalization error Eq.(6). In this simulation: $\nu = 2$ and $\eta = 1$. The employed incomplete nonlinear model of Eq.(32) is:

$$y(k) = h(\boldsymbol{w}'\boldsymbol{x}_1(k)) + e(k;\, \boldsymbol{w}). \tag{34}$$

According to Eq.(6), (32) and (34) ($E\{\cdot\}$ w.r.t. $[\boldsymbol{x}_t,\, \varepsilon_t]$):

$$\begin{aligned} G(\widehat{\boldsymbol{w}}) &= E\left\{\left[\varepsilon_t + h(\boldsymbol{x}^{\mathsf{T}}(k)\boldsymbol{w}^{\circ}) - h(\widehat{\boldsymbol{w}}\boldsymbol{x}_1(k))\right]^2\right\} = \\ &= E\left\{\left[h(\boldsymbol{x}^{\mathsf{T}}(k)\boldsymbol{w}^{\circ}) - h(\widehat{\boldsymbol{w}}\boldsymbol{x}_1(k))\right]^2\right\} + \sigma_\varepsilon^2. \end{aligned} \tag{35}$$

Evaluation of the first term in Eq.(35) is possible; however, due to the extent of the derivation it is omitted, see [10, Appendix D] for further details.

The weight, $\boldsymbol{w}$, in Eq.(34) is estimated using a modified Gauss–Newton algorithm [14, Ch.14]. That is, for each training set $\{\boldsymbol{x}_1^{(s)}(k), y^{(s)}(k)\}$, $s = 1, 2, \ldots, Q$ (below the s index is omitted for simplicity):

$$\boldsymbol{w}_{(i+1)} = \boldsymbol{w}_{(i)} + \mu\tilde{\mathbf{H}}_N^{-1}(\boldsymbol{w}_{(i)})\nabla(\boldsymbol{w}_{(i)}), \tag{36}$$

$$\tilde{\mathbf{H}}_N(\boldsymbol{w}_{(i)}) = \sum_{k=1}^{N}\left[h'(\boldsymbol{w}_{(i)}\boldsymbol{x}_1(k)) \cdot \boldsymbol{x}_1(k)\right]^2, \tag{37}$$

$$\nabla(\boldsymbol{w}_{(i)}) = \sum_{k=1}^{N} h'(\boldsymbol{w}_{(i)}\boldsymbol{x}_1(k)) \cdot \boldsymbol{x}_1(k) \cdot e(k;\, \boldsymbol{w}_{(i)}) \tag{38}$$

where $0 \leq \mu \leq 1$ is the step–size and $'$ denotes the derivative. For ea iteration μ is adjusted in order to ensure: $S_N(w_{(i+1)}) < S_N(w_{(i)})$. The employed stopping criterion [14, Sec.14.4] was: $(S_N(w_{(i+1)}) - S_N(w_{(i)}))/S_N(w_{(i)}) < 10^{-12}$.

The result of comparing *GEN* to *FPE* is shown in Fig.5.

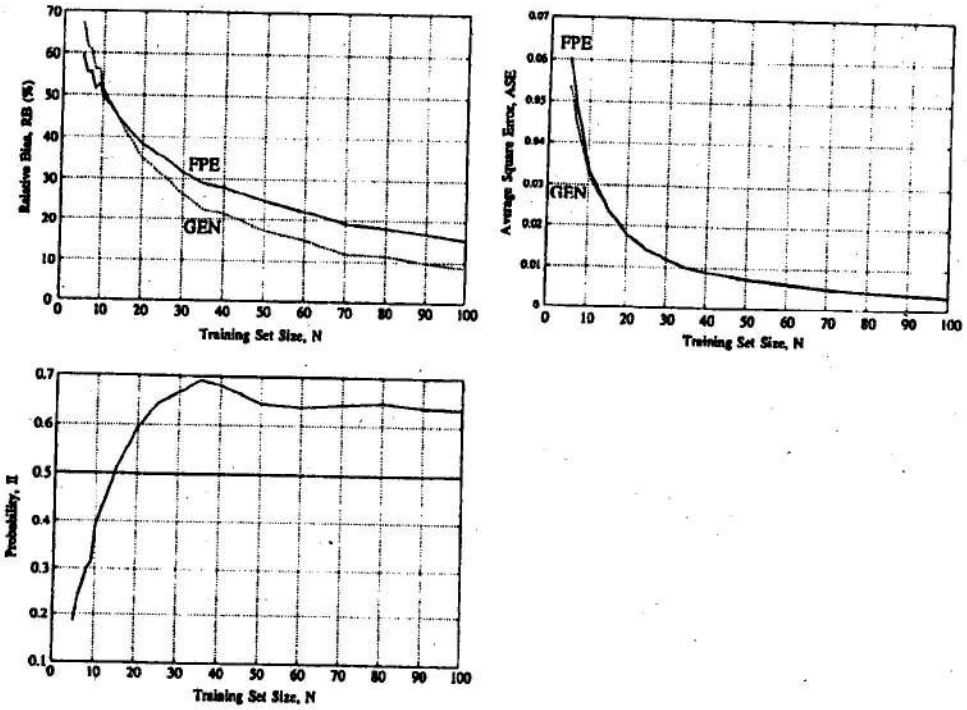

Fig. 5. Comparison of *GEN* and *FPE* within the neural model Eq.(34). We used $Q = 5 \cdot 10^3$, $\forall\, N$.

It turned out that $|RB(GEN)| < |RB(FPE)|$ as $N \gtrsim 15$; otherwise, vice versa. The improvement in relative bias is approximately a factor of 1.75 as $N = 100$. The average squared errors are approximately identical; however, $ASE(GEN)$ tend to be smaller when N is small. Finally, $\Pi > 0.5$ as $N \geq 15$, and $\Pi \approx 0.65$ as $N \geq 20$. In conclusion, *GEN* may be preferred as $N \gtrsim 15$.

5. CONCLUSION

In this paper we have suggested to optimize the architecture of a neural network by using generalization error estimators. The network architecture with minimal estimated average generalization error is selected as being optimal.

Different estimators of the average generalization error were discussed and a presentation of the *GEN*–estimator [9, 10] was given. The *GEN*–estimator is designed for the case of *incomplete, nonlinear* models. A model is signified

as incomplete if it is not capable of modeling the underlying data generating system perfectly. It was claimed that incomplete models are the typical case when employing neural networks. The *GEN*–estimator may be viewed as an extension of the Final Prediction Error estimator (*FPE*) [1] and the Generalized Prediction Error estimator (*GPE*) [11, 12]. A concise list of advantages and drawbacks of the *GEN*–estimator was also provided.

A numerical study for the comparison of *GEN*, *FPE*, the half–half split cross-validation estimator, C_{50}, and the leave–one–out cross–validation estimator, L, was setup. It turned out that in most cases the *GEN*–estimator is a preferable alternative. The relative bias (*RB*) of *GEN* compared to *FPE* is typically smaller, they possess similar averaged squared error (*ASE*), and the probability of proximity (Π) is significantly larger than 0.5. The *RB* of the C_{50} is normally much smaller than that of *GEN*; however, the *ASE* of C_{50} is often extremely high. Π is often only a little above 0.5; however, the high variance may rule out the C_{50}–estimator. The *RB* of *GEN* compared to that of L is often smaller while they possess similar variance; however, still Π is well above 0.5.

6. ACKNOWLEDGMENTS

Lars Kai Hansen is gratefully acknowledged for helpful discussions. The work was supported by the Danish Natural Science and Technical Research Councils through the Computational Neural Network Center.

REFERENCES

1. Akaike H. // Annals of the Institute of Statistical Mathematics. 1969. V.21. P.243.
2. Draper N.R. & Smith H. // Applied Regression Analysis, New York. — New York: John Wiley & Sons, 1981.
3. Fogel D.B. // IEEE Transactions on Neural Networks. 1991. V.2. N 5. P.490.
4. Hansen L.K. // Neural Networks. 1993. V.6. P.393.
5. Hansen L.K. & Salamon P. // IEEE Transactions on Pattern Analysis and Machine Intelligence. 1990. V.12. N 10. P.993.
6. Hertz J., Krogh A. & Palmer R.G. // Introduction to the Theory of Neural Computation. — Redwood City, California: Addison–Wesley Publishing Company, 1991.
7. Hornik K., Stinchcombe M. & White H. // Neural Networks. 1990. V.3. N 5. P.551.
8. Kannurpatti R. & Hart G.W. // IEEE Transaction on Information Theory. 1991. V.37. N 5. P.1441.

9. Larsen J. // Neural Networks for Signal. / S.Y.Kung, F.Fallside, J.Aa.Sørensen & C.A.Kamm. — Piscataway, New Jersey: IEEE, 1992. P.29.
10. Larsen J. // Ph.D.Thesis. Design of Neural Network Filters. — The Technical University of Denmark: Electronics Institute, March 1993.
11. Moody J. Proceedings of the first IEEE Workshop on Neural Networks for Signal Processing. / B.H.Juang, S.Y.Kung & C.A.Kamm. — Piscataway, New Jersey: IEEE, 1991. P.1.
12. Moody J. Advances in Neural Information Processing Systems 4, Proceedings of the 1991 Conference. / J.E.Moody, S.J.Hanson, R.P.Lippmann. — San Mateo, California: Morgan Kaufmann Publishers, 1992. P.847.
13. Rosenblatt M. // Stationary Sequences and Random Fields. — Boston, Massachusetts: Birkhäuser, 1985.
14. Seber G.A.F. & Wild C.J. // Nonlinear Regression. — New York: John Wiley & Sons, 1989.

The Computational Neural
Network Center Electronics
Institute, Building 349. The
Technical University of Denmark

Поступила в редакцию
16 сентября 1993 г.