Derivative Free Training of Recurrent Neural Networks
A Comparison of Algorithms and Architectures

Branimir Todorović[1,4], Miomir Stanković[2,4] and Claudio Moraga[3,4]
[1]*Faculty of Natural Sciences and Mathematics, University of Niš, Niš, Serbia*
[2]*Faculty of Occupational Safety, University of Niš, Niš, Serbia*
[3]*European Centre for Soft Computing, 33600 Mieres, Spain*
[4]*Technical University of Dortmund, 44221 Dortmund, Germany*

Keywords: Recurrent Neural Networks, Bayesian Estimation, Nonlinear Derivative Free Estimation, Chaotic Time Series Prediction.

Abstract: The problem of recurrent neural network training is considered here as an approximate joint Bayesian estimation of the neuron outputs and unknown synaptic weights. We have implemented recursive estimators using nonlinear derivative free approximation of neural network dynamics. The computational efficiency and performances of proposed algorithms as training algorithms for different recurrent neural network architectures are compared on the problem of long term, chaotic time series prediction.

1 INTRODUCTION

In this paper we consider the training of Recurrent Neural Networks (RNNs) as derivative free approximate Bayesian estimation. RNNs form a wide class of neural networks with feedback connections among processing units (artificial neurons). Neural networks with feed forward connections implement static input-output mapping, while recurrent networks implement the mapping of both input and internal state (represented by outputs of recurrent neurons) into the future internal state.

In general, RNNs can be classified as locally recurrent, where feedback connections exist only from a processing unit to itself, and globally recurrent, where feedback connections exist among distinct processing units. The modeling capabilities of globally recurrent neural networks are much richer than that of the simple locally recurrent networks.

There exist a group of algorithms for training synaptic weights of recurrent neural networks that are based on the exact or approximate computation of the gradient of an error measure in the weight space. Well known approaches that use methods for exact gradient computation are back-propagation through time (BPTT) and real time recurrent learning (RTRL) (Williams and Zipser, 1989; Williams and Zipser, 1990). Since BPTT and RTRL are using only first-order derivative information, they exhibit slow convergence. In order to improve the speed of the RNN training, a technique known as ***teacher forcing*** has been introduced (Williams and Zipser, 1989). The idea is to use the desired outputs of the neurons instead of the obtained to compute the future outputs. In this way the training algorithm is focused on the current time step, given that the performance is correct on all earlier time steps.

However, in its basic form teacher forcing is not always applicable. It clearly cannot be applied in networks where feedback connections exist only from hidden units, for which the target outputs are not explicitly given. The second important case is the training on noisy data, where the target outputs are corrupted by noise. Therefore, to apply teacher forcing in such cases, a true target outputs of neurons have to be estimated somehow.

The well-known extended Kalman filter (Anderson and Moore, 1979), as a second order sequential training algorithm and state estimator offers the solution to the both stated problems. It improves the learning rate by exploiting second order information on criterion function and generalizes the teacher forcing technique by estimating the true outputs of the neurons.

The extended Kalman filter can be considered as the approximate solution of the recursive Bayesian state estimation problem. The problem of estimating the hidden state of a dynamic system using observations which arrive sequentially in time is

Todorović B., Stanković M. and Moraga C..
Derivative Free Training of Recurrent Neural Networks - A Comparison of Algorithms and Architectures.
DOI: 10.5220/0005081900760084
In *Proceedings of the International Conference on Neural Computation Theory and Applications* (NCTA-2014), pages 76-84
ISBN: 978-989-758-054-3

very important in many fields of science, engineering and finance. The hidden state of some dynamic system is represented as a random vector variable, and its evolution in time $\{x_k, k = 1,2,...\}$ is described by a so called dynamic or process equation:

$$x_k = f_k(x_{k-1}, u_k, d_k), \quad (1)$$

where $f_k : R^{n_x \times n_d} \to R^{n_x}$ is nonlinear function, and $\{d_k, k = 1,2,...\}$ is an i.i.d. process noise sequence, while n_x and n_d are dimensions of the state and process noise vectors respectively. The hidden state is known only through the measurement (observation) equation:

$$y_k = h_k(x_k, v_k), \quad (2)$$

where $h_k : R^{n_x \times n_v} \to R^{n_y}$ is nonlinear function, and $\{v_k, k = 1,2,...\}$ is an i.i.d. measurement noise sequence, and n_y and n_v are dimensions of the measurement and measurement noise vectors, respectively.

In a sequential or recursive Bayesian estimation framework, the state filtering probability density function (pdf) $p(x_k / y_{0:k})$, (where $y_{0:k}$ denotes the set of all observations $y_{0:k} = \{y_0, y_1, ..., y_k\}$ up to the time step *k*), represents the complete solution. The optimal state estimate with respect to any criterion can be calculated based on this pdf.

The recursive Bayesian estimation algorithm consists of two steps: **prediction** and **update**. In the prediction step the previous posterior $p(x_{k-1} / y_{0:k-1})$ is projected forward in time, using the probabilistic process model:

$$p(x_k / y_{0:k-1}) = \int p(x_k / x_{k-1}) p(x_{k-1} / y_{0:k-1}) dx_k, \quad (3)$$

where the state transition density function $p(x_k / x_{k-1})$ is completely specified by $f(\cdot)$ and the process noise distribution $p(d_k)$.

In the second step, the predictive density is updated by incorporating the latest noisy measurement y_k using the observation likelihood $p(y_k / x_k)$ to generate the posterior:

$$p(x_k / y_{0:k}) = \frac{p(y_k / x_k) p(x_k / y_{0:k-1})}{\int p(y_k / x_k) p(x_k / y_{0:k-1}) dx_k} \quad (4)$$

This recursive estimation algorithm can be applied to RNN training after representing the time evolution of neurons outputs and connection weights, as well as their observations, in the form of the state space model. The hidden state of the recurrent neural network x_k is a stacked vector of recurrent neurons outputs s_k and connection weights w_k. Its evolution in time can be represented by the following dynamic equation.

$$\begin{bmatrix} s_k \\ w_k \end{bmatrix} = \begin{bmatrix} f(s_{k-1}, u_k, w_{k-1}, d_{s_k}) \\ w_{k-1} + d_{w_k} \end{bmatrix}, \quad (5)$$

where d_{s_k} and d_{w_k} represent dynamic noise vectors.

The outputs of the neurons are obtained through the following observation equation:

$$y_k = h(s_k, w_k, v_k). \quad (6)$$

The recurrence relations (3) and (4) are only conceptual solutions and the posterior density $p(x_k / y_{0:k})$ cannot be determined analytically in general. The restrictive set of cases includes the well known Kalman filter, which represents the optimal solution of (3) and (4) if the prior state density $p(x_0 / y_0) = p(x_0)$, the process noise as well as the observation noise densities are Gaussians, and $f(\cdot)$ and $h(\cdot)$ are linear functions.

In case of RNN training, $f(\cdot)$ and $h(\cdot)$ are nonlinear in general and an analytic solution is not tractable, therefore some approximations and suboptimal solutions have to be considered. The well known suboptimal solution is the Extended Kalman Filter (EKF), which assumes the Gaussian property of noise and uses the Taylor expansion of $f(\cdot)$ and $h(\cdot)$ (usually up to the linear term) to obtain the recursive estimation for $p(x_k / y_{0:k})$. The EKF has been successfully applied in RNN training (Todorović et al., 2003; Todorović et al., 2004) due to important advantages compared to RTRL and BPTT: faster convergence and generalization of teacher forcing. Recently, families of new derivative free filters have been proposed as an alternative to EKF for estimation in nonlinear systems. Divided Difference Filters (DDF), derived in (Nørgaard et al., 2000), are based on polynomial approximation of nonlinear transformations using a multidimensional extension of Stirling's interpolation formula. The Unscented Kalman Filter (UKF) (Julier and Uhlmann, 1997) uses the true nonlinear models and approximates the state distribution using deterministically chosen sample points. Surprisingly, both the DDF and the UKF result in similar equations and are usually called

derivative free filters (Van der Merwe and Wan, 2001).

The rest of the paper is organized as follows. In the second section recursive Bayesian estimator is approximated by linear minimum mean square error estimator (MMSE), which recursively updates only the first two moments of the relevant probability densities. The problem that remains to be solved is propagating these moments through the nonlinear mapping of the process equation and the observation equation. In the third section we describe three approaches to this problem: a linearization of the nonlinear mapping using a Taylor series expansion, a derivative free unscented transform and a derivative free polynomial approximation using a multidimensional extension of the Stirling's interpolation formula. In the fourth section we give the state space models of three globally recurrent neural networks: fully connected, Elman and Non-linear AutoRegresssive with eXogenous inputs (NARX) recurrent neural networks. We trained them by applying approximate recursive Bayesian joint estimation of the recurrent neurons outputs and synaptic weights. The results of applying three different estimation algorithms in training three different architectures of recurrent neural networks are given in the last section.

2 LINEAR MMSE ESTIMATION OF THE NONLINEAR STATE SPACE MODEL

An analytically tractable solution of the problem of recursive Bayesian estimation framework can be obtained based on the assumption that the state estimator $\hat{x}_k$ can be represented as a linear function of the current observation y_k :

$$\hat{x}_k = A_k y_k + b_k , \tag{7}$$

where matrix A_k and vector b_k are derived by minimizing mean square estimation error criterion:

$$R_k = \iint (x_k - \hat{x}_k)^T (x_k - \hat{x}_k) \cdot p(x_k, y_k / y_{0:k-1}) dx_k dy_k \cdot \tag{8}$$

Note that the condition $\partial R_k / \partial b_k = 0$ is equivalent to the requirement that the estimator is unbiased:

$$\iint (x_k - A_k y_k - b_k) \cdot p(x_k, y_k / y_{0:k-1}) dx_k dy_k = 0 , \tag{9}$$

from which we obtain,

$$b_k = \hat{x}_k^- - A_k \hat{y}_k^- , \quad \hat{x}_k = \hat{x}_k^- - A_k (y_k - \hat{y}_k^-) , \tag{10}$$

where $\hat{x}_k^- = E[x_k / y_{0:k-1}] = \int x_k p(x_k / y_{0:k-1}) dx_k$ and $\hat{y}_k^- = E[y_k / y_{0:k-1}] = \int y_k p(y_k / y_{0:k-1}) dy_k$.

Both the condition $\partial R_k / \partial A_k = 0$ and the unbiasedness of the estimator result in the well known orthogonality principle, which states that the estimation error is orthogonal to the current observation, and, consequently:

$$\iint (x_k - \hat{x}_k^- - A_k (y_k - \hat{y}_k^-))(y_k - \hat{y}_k^-)^T \cdot p(x_k, y_k / y_{0:k-1}) dx_k dy_k = 0 \cdot \tag{11}$$

From (11) we obtain the matrix $A_k = P_{x_k y_k} P_{y_k}^{-1}$, where

$$P_{y_k} = E[(y_k - \hat{y}_k^-)(y_k - \hat{y}_k^-)^T / y_{0:k-1}] \tag{12b}$$

$$P_{x_k y_k} = E[(x_k - \hat{x}_k^-)(y_k - \hat{y}_k^-)^T / y_{0:k-1}] \tag{12c}$$

Note that P_{y_k} must be invertible, that is measurements y_k have to be linearly independent.

Finally, after replacing $A_k = P_{x_k y_k} P_{y_k}^{-1}$ we obtain the linear MMSE estimator:

$$\hat{x}_k = \hat{x}_k^- + P_{x_k y_k} P_{y_k}^{-1} (y_k - \hat{y}_k^-) . \tag{13a}$$

The matrix Mean Square Error (MSE) corresponding to (13):

$$E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T] = P_{x_k}^- - P_{x_k y_k} P_{y_k}^{-1} P_{x_k y_k}^T . \tag{13b}$$

is used as the approximation of the estimator covariance $P_{x_k} \approx E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T]$.

If the dynamic and the observation models are linear and process and observation noises are Gaussian, the linear MMSE estimator is the best MMSE estimator and is equal to the conditional mean $E[x_k / y_{0:k}]$, otherwise it is the best within the class of linear estimators.

The problem that remains to be solved is the estimation of the statics of a random variable propagated trough the nonlinear transformation.

$$\hat{x}_k^- = \iint f(x_{k-1}, u_k, d_k) p(x_{k-1} / y_{0:k-1}) p(d_k) dx_{k-1} dd_k \tag{14a}$$

$$P_{x_k}^- = \iint (f(x_{k-1}, u_k, d_k) - \hat{x}_k^-)(f(x_{k-1}, u_k, d_k) - \hat{x}_k^-)^T p(x_{k-1} / y_{0:k-1}) p(d_k) dx_{k-1} dv_k \tag{14b}$$

$$\hat{y}_k^- = \iint h(x_k, u_k, v_k) p(x_k / y_{0:k-1}) p(v_k) dx_k dv_k \tag{14c}$$

$$P_{y_k} = \iint (h(x_k, u_k, v_k) - \hat{y}_k^-)(h(x_k, u_k, v_k) - \hat{y}_k^-)^T \cdot p(x_k / y_{0:k-1}) p(v_k) dx_k dv_k \tag{14d}$$

$$P_{x_k y_k} = \iint (x_k - \hat{x}_k^-)(h(x_k, u_k, v_k) - \hat{y}_k^-)^T \cdot p(x_k / y_{0:k-1}) p(v_k) dx_k \, dv_k \quad (14e)$$

The problem can be considered in a general. Suppose that x is a random variable with mean $\hat{x}$ and covariance P_x. A random variable y is related to x through the nonlinear function $y = f(x)$. We wish to calculate the mean $\hat{y}$ and the covariance P_y of y.

2.1 Extended Kalman Filter

The extended Kalman filter is based on the multidimensional Taylor series expansion of $f(x)$. We shall consider only the first order EKF, obtained by excluding the nonlinear terms of Taylor series expansion:

$$f(x) = f(\hat{x} + \Delta x) \approx f(\hat{x}) + f'_x(\hat{x})\Delta x \quad (15)$$

where $f'_x(\hat{x}) = \partial f / \partial x \big|_{x=\hat{x}}$ is the Jacobian of the nonlinear function and Δx is a zero mean random variable with covariance P_x.

In this way the prediction of the state is given by:

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_k) + \bar{d}_k \quad (16a)$$

$$P_{x_k}^- = F_k P_{x_{k-1}} F_k^T + G_k Q_k G_k^T \quad (16b)$$

where $F_k = \nabla_{\hat{x}_{k-1}} f$, $G_k = \nabla_{\bar{d}_k} f$, $\bar{d}_k = E[d_k]$, $Q_k = E[d_k d_k^T]$.

The prediction of the observation is given by:

$$\hat{x}_k^- = h(\hat{x}_k^-) + \bar{v}_k \quad (17a)$$

$$P_{y_k} = H_k P_{x_k}^- H_k^T + L_k R_k L_k^T \quad (17b)$$

where $H_k = \nabla_{\hat{x}_k^-} h$, $L_k = \nabla_{\bar{v}_k} h$, $\bar{v}_k = E[v_k]$, and $R_k = E[v_k v_k^T]$.

2.2 Divided Difference Filter

In (Nørgaard et al., 2000) Nørgaard et al. proposed a new set of estimators based on a derivative free polynomial approximation of nonlinear transformations using a multidimensional extension of Stirling's interpolation formula. This formula is particularly simple if only the first and second order polynomial approximation is considered:

$$f(x) \approx f(\hat{x}) + \tilde{D}_{\Delta x} f + \tilde{D}_{\Delta x}^2 f \quad (18)$$

where divided difference operators are defined by:

$$\tilde{D}_{\Delta x} f = \frac{1}{h}\left(\sum_{p=1}^{n} \Delta x_p \mu_p \delta_p \right) f(\bar{x}) \quad (19)$$

δ_p is a "partial" difference operator:

$$\delta_p f(\hat{x}) = f(\hat{x} + 0.5 \cdot h \cdot e_p) - f(\bar{x} - 0.5 \cdot h \cdot e_p) \quad (20)$$

and μ_p is an average operator:

$$\mu_p f(\hat{x}) = 0.5 \cdot (f(\hat{x} + 0.5 \cdot h \cdot e_p) + f(\hat{x} - 0.5 \cdot h \cdot e_p)) \quad (21)$$

where e_p is the p-th unit vector.

Applying a stochastic decoupling of the variables in x by the following transformation $z = S_x^{-1} x$, (S_x is the Cholesky factor of the covariance matrix $P_x = S_x S_x^T$), an approximation of the mean and the covariance of $y = f(x)$ is obtained:

$$\hat{y} = \frac{h^2 - n}{h^2} f(\bar{x}) + \frac{1}{2h^2} \sum_{p=1}^{n} (f(\bar{x} + h s_{x,p}) + f(\bar{x} - h s_{x,p})) \quad (22a)$$

$$\begin{aligned} P_y = {} & \frac{1}{4h^2} \sum_{p=1}^{n} (f(\hat{x} + h s_{x,p}) - f(\hat{x} - h s_{x,p})) \\ & \cdot (f(\hat{x} + h s_{x,p}) - f(\hat{x} - h s_{x,p}))^{\mathrm{T}} \\ & + \frac{h^2 - 1}{4h^4} \sum_{p=1}^{n} (f(\hat{x} + h s_{x,p}) + f(\hat{x} - h s_{x,p}) - 2 f(\hat{x})) \\ & \cdot (f(\hat{x} + h s_{x,p}) + f(\hat{x} - h s_{x,p}) - 2 f(\hat{x}))^{\mathrm{T}} \end{aligned} \quad (22b)$$

Nørgaard et al. have derived the alternative covariance estimate as well (Nørgaard et al., 2000):

$$\begin{aligned} P_y = {} & \frac{1}{2h^2} \sum_{p=1}^{n} (f(\hat{x} + h s_{x,p}) - \hat{y}) f(\hat{x} + h s_{x,p}) - \hat{y})^{\mathrm{T}} \\ & + \frac{1}{2h^2} \sum_{p=1}^{n} (f(\hat{x} - h s_{x,p}) - \hat{y}) f(\hat{x} - h s_{x,p}) - \hat{y})^{\mathrm{T}} \\ & + \frac{h^2 - n}{h^2} (f(\hat{x}) - \hat{y}) f(\hat{x}) - \hat{y})^{\mathrm{T}} \end{aligned} \quad (23)$$

This estimate is less accurate than (22b). Moreover, for $h^2 < n$ the last term becomes negative semi-definite with a possible implication that the covariance estimate (23) becomes non-positive definite too. The reason why this estimate is considered here is to provide a comparison with the covariance estimate obtained by the Unscented Transformation described in the next subsection.

2.3 Unscented Kalman Filter

Julier and Uhlman proposed the Unscented Transformation (UT) (Julier and Uhlmann, 1997) in order to calculate the statistics of a random variable x propagated through the nonlinear function $y = f(x)$. The n_x-dimensional continuous random

variable x with mean $\hat{x}$ and covariance P_x is approximated by $2n_x+1$ sigma points $\mathcal{X}_p$ with corresponding weights ω_p, $p=0,1,...,2n_x$:

$$\mathcal{X}_0=\hat{x},\quad \omega_0=\lambda/(n+\lambda),\quad \lambda=\alpha^2(n_x+\kappa)-n_x \text{ fo}$$
$$\text{r } p=1,2...,n$$

$$\mathcal{X}_p=\hat{x}+\sqrt{n+\lambda}\cdot s_{x,p}\qquad \omega_p=0.5/(n+\lambda)$$

$$\mathcal{X}_{p+n_x}=\hat{x}-\sqrt{n+\kappa}\cdot s_{x,p}\quad \omega_{p+n_x}=0.5/(n+\lambda)$$

where α determines the spread of the sigma points around $\hat{x}$ (usually $1.e-4\le\alpha\le 1$) and $\kappa\in\Re$ is the scaling parameter, usually set to 0 or $3-n_x$ (Julier and Uhlmann, 1997). $s_{x,p}$ is the p-th row or column of the matrix square root of P_x.

Each sigma point is instantiated through the function $f(\cdot)$ to yield the set of transformed sigma points $\mathcal{Y}_i=f(\mathcal{X}_0)$, and the mean $\hat{y}$ of the transformed distribution is estimated by:

$$\hat{y}=\sum_{p=0}^{2n_x}\omega_p\mathcal{Y}_p=\frac{\lambda}{n+\lambda}f(\hat{x})+\frac{1}{2(n+\lambda)}\sum_{i=1}^{n}(f(\hat{x}+\sqrt{n+\lambda}\cdot s_{x,i})+f(\hat{x}-\sqrt{n+\lambda}\cdot s_{x,i})) \tag{24}$$

The covariance estimate obtained by the unscented transformation is:

$$P_y=\sum_{p=0}^{2n_x}\omega_p(\mathcal{Y}_p-\hat{y})(\mathcal{Y}_p-\hat{y})^{\mathrm{T}}=\frac{\lambda}{n+\lambda}(f(\hat{x})-\hat{y})f(\hat{x})-\hat{y})^{\mathrm{T}}+\frac{1}{2(n+\lambda)}\sum_{p=1}^{n}(f(\hat{x}+\sqrt{n+\lambda}\cdot s_{x,p})-\hat{y})f(\hat{x}+\sqrt{n+\lambda}s_{x,p})-\hat{y})^{\mathrm{T}}+\frac{1}{2(n+\lambda)}\sum_{p=1}^{n}(f(\hat{x}-\sqrt{n+\lambda}s_{x,p})-\hat{y})f(\hat{x}-\sqrt{n+\lambda}s_{x,p})-\hat{y})^{\mathrm{T}} \tag{25}$$

It can be easily verified that for $h=\sqrt{n+\lambda}$, the estimates of the mean (22a) and the covariance (23) obtained by DDF are equivalent to the estimates (24) and (25) obtained by UKF. The interval length h is set equal to the kurtosis of the prior random variable x. For Gaussians it holds $h^2=3$.

3 STATE SPACE MODELS OF GLOBALLY RECURRENT NEURAL NETWORKS

In order to apply approximate Bayesian estimators as training algorithm of recurrent neural networks we need to represent dynamics of RNN in a form of state space model. In this section we define the state space models of three representative architectures of globally recurrent neural networks: Elman, fully connected, and NARX recurrent neural network.

3.1 Elman Network State Space Model

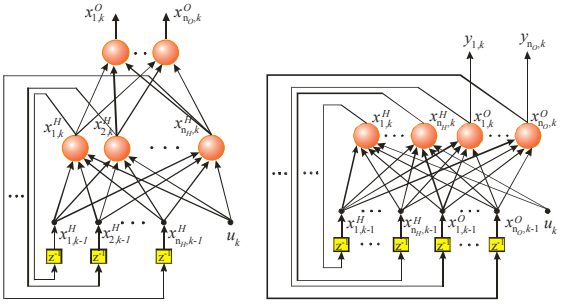
In Elman RNNs adaptive feedbacks are provided between every pair of hidden units. The network is illustrated in Fig.1.a), and the state space model of the Elman network is given by equations

$$\begin{bmatrix}x_k^H\\ w_k^O\\ w_k^H\end{bmatrix}=\begin{bmatrix}f(x_{k-1}^H,w_{k-1}^H,u_k)\\ w_{k-1}^O\\ w_{k-1}^H\end{bmatrix}+\begin{bmatrix}d_{x_k^H}\\ d_{w_k^O}\\ d_{w_k^H}\end{bmatrix} \tag{26a}$$

$$y_k=x_k^O+v_k,\quad x_k^O=h(x_k^H,w_k^O) \tag{26b}$$

where x_k^H represents the output of the hidden neurons in the k-th time step, x_k^O is the output of the neurons in the last layer, w_{k-1}^O is the vector of synaptic weights between the hidden and the output layer and w_{k-1}^H is the vector of recurrent adaptive connection weights. Note that in the original formulation of Elman, these weights were fixed. Random variables $d_{x_k^H}$, $d_{w_k^O}$, $d_{w_k^H}$ represent the process noises.

It is assumed that the output of the network $x_k^O=h(x_k^H,w_k^O)$ is corrupted by the observation noise v_k.



a) Elman b) Fully connected

Figure 1: Elman and fully connected RNN.

3.2 Fully Connected Recurrent Network State Space Model

In fully connected RNNs adaptive feedbacks are provided between each pair of processing units (hidden and output). The state vector of a fully connected RNN consists of outputs (activities) of

hidden x_k^H and output neurons x_k^O, and their synaptic weights w_k^H and w_k^O. The activation functions of the hidden and thee output neurons are $f^H(x_k^O, x_k^H, w_{k-1}^H, u_k)$ and $f^O(x_k^O, x_k^H, w_{k-1}^O, u_k)$, respectively. The network structure is illustrated in Fig. 1.b).

The state space model of the network is given by:

$$\begin{bmatrix} x_k^O \\ x_k^H \\ w_k^O \\ w_k^H \end{bmatrix} = \begin{bmatrix} f^O(x_k^O, x_k^H, w_{k-1}^O, u_k) \\ f^H(x_k^O, x_k^H, w_{k-1}^H, u_k) \\ w_{k-1}^O \\ w_{k-1}^H \end{bmatrix} + \begin{bmatrix} d_{x_k^O} \\ d_{x_k^H} \\ d_{w_k^O} \\ d_{w_k^H} \end{bmatrix} \quad (27a)$$

$$y_k = H \cdot \begin{bmatrix} x_k^O \\ x_k^H \\ w_k^O \\ w_k^H \end{bmatrix} + v_k, \quad H = [I_{n_O \times n_O} \; 0_{n_O \times (n_S - n_O)}] \quad (27b)$$

The dynamic equation describes the evolution of neuron outputs and synaptic weights. In the observation equation, the matrix H selects the activities of output neurons as the only visible part of the state vector, where n_S is the number of hidden states which are estimated: $n_S = n_O + n_H + n_{W^O} + n_{W^H}$, n_O and n_H are the numbers of output and hidden neurons respectively, n_{W^O} is the number of adaptive weights of the output neurons, n_{W^H} is the number of adaptive weights of the hidden neurons.

3.3 NARX Recurrent Neural Network State Space Model

The non-linear AutoRegressive with eXogenous inputs (NARX) recurrent neural network usually outperforms the classical recurrent neural networks, like Elman or fully connected RNN, in tasks that involve long term dependencies for which the desired output depends on inputs presented at times far in the past.

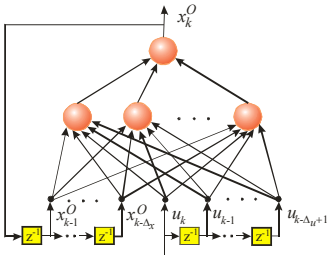


Figure 2: NARX recurrent neural network.

Here we define the state space model of a NARX RNN. Adaptive feedbacks are provided between the output and the hidden units. These feedback connection and possible input connections are implemented as FIR filters. The state vector consists of outputs of the network in Δ_x time steps $x_k^O, x_{k-1}^O, ..., x_{k-\Delta_x+1}^O$, the output w_k^O, and hidden synaptic w_k^H weights.

$$\begin{bmatrix} x_k^O \\ x_{k-1}^O \\ \vdots \\ x_{k-\Delta_x+1}^O \\ w_k^O \\ w_k^H \end{bmatrix} = \begin{bmatrix} f(x_{k-1}^O, .., x_{k-\Delta_x}^O, u_{k-1}, .., u_{k-\Delta_u}, w_{k-1}) \\ x_{k-1}^O \\ \vdots \\ x_{k-\Delta_x+1}^O \\ w_{k-1}^O \\ w_{k-1}^H \end{bmatrix} + \begin{bmatrix} d_{x_k^O} \\ 0 \\ \vdots \\ 0 \\ d_{w_k^O} \\ d_{w_k^H} \end{bmatrix} \quad (28)$$

The dynamic equation describes the evolution of network outputs and synaptic weights.

$$y_k = H \cdot \begin{bmatrix} x_k^O \\ x_{k-1}^O \\ \vdots \\ x_{k-\Delta_x+1}^O \\ w_k^O \\ w_k^H \end{bmatrix} + v_k, \quad H = [I_{n_O \times n_O} \; 0_{n_O \times (n_S - n_O)}] \quad (29)$$

As in previous examples, n_O represents the number of output neurons. n_S is the number of hidden states of the NARX RNN: $n_S = n_O + n_{W^O} + n_{W^H}$, n_{W^O} is the number of adaptive weights of output neurons, n_{W^H} is number of adaptive weights of hidden neurons.

All considered models have nonlinear hidden neurons and linear output neurons. Two types of nonlinear activation functions have been used in the following tests: the sigmoidal and the Gaussian radial basis function.

4 EXAMPLES

In this section we compare derived algorithms for sequential training of RNN.

We have evaluated the performance of algorithms in training three different architectures of globally recurrent neural networks: fully connected RNN, Elman RNN with adaptive recurrent connections and NARX recurrent neural network. The problem at hand was the long term prediction of chaotic time series. Implementation of Divided Difference Filter and Unscented Kalman filter did

not required linearization of the RNN state space models. However, in order to apply Extended Kalman Filter we had to linearize the RNNs state space models that are to calculate Jacobian of the RNN outputs with respect to the inputs and synaptic weights. Note that we did not apply back propagation through time but standard back propagation algorithm to calculate the Jacobian. This was possible because of the joint estimation of RNN outputs and synaptic weights.

In the process of the evaluation, recurrent neural networks were trained sequentially on the certain number of samples. After that they were iterated for a number of samples, by feeding back just the predicted outputs as the new inputs of the recurrent neurons. Time series of iterated predictions were compared with the test parts of the original time series by calculating the Normalized Root Mean Squared Error (NRMSE):

$$NRMSE = \sqrt{\frac{1}{\sigma^2 N}\sum_{k=1}^{N}(y_k - \hat{y}_k^-)^2} \qquad (30)$$

where σ is the standard deviation of chaotic time series, y_k is the true value of sample at time step k, and $\hat{y}_k^-$ is the RNN prediction.

Mean and variance of the NRMSE obtained on 30 independent runs, average time needed for training and number of hidden neurons and adaptive synaptic weights are given in tables for comparison.

The variance of the process noise $d_{s,k}$ and $d_{w,k}$ were exponentially decayed form 1.e-1 and 1.e-3 to 1.e-10, and the variance of the observation noise v_k was also exponentially decayed from 1.e-1 to 1.e-10 during the sequential training.

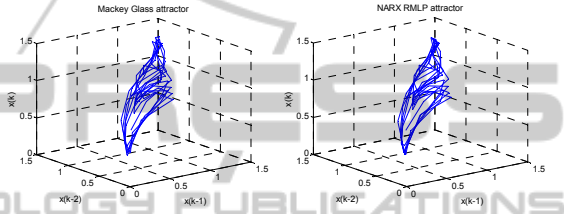
4.1 Mackey Glass Chaotic Time Series Prediction

In our first example we have considered the long term iterated prediction of the Mackey Glass time series. We have applied Divided Difference Filter, Unscented Kalman Filter and Extended Kalman Filter for joint estimation of synaptic weights and neuron outputs of three different RNN architectures: Elman, fully connected and non-linear AutoRegressive with eXogenous inputs (NARX) recurrent neural network.

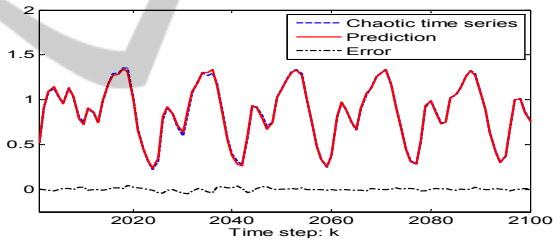
After sequential adaptation on 2000 samples, a long term iterated prediction of the next $N = 100$ samples is used to calculate the NRMSE.

Table 1. contains mean and variance of NRMSE obtained after 30 independent trials of each estimator applied on each architecture. We also give the number of hidden units, the number of adaptable parameters and time needed for training on 2000 samples. Given these results we can conclude that the NARX network is superior in both NRMSE of long term prediction and time needed for training, compared to other two architectures. As for the approximate Bayesian estimators, although slightly slower in our implementation, derivative free filters (DDF and UKF) are consistently better than EKF, that is they produced RNN's with significantly lower NRMSE.

Sample results of long term prediction using NARX network with sigmoidal neurons, trained using DDF are shown in Fig. 3.



a) Phase plot of x_k versus x_{k-1} and x_{k-2} for the Mackey Glass time series and the NARX RMLP iterated prediction



b) Comparison of the original chaotic time series and the NARX RMLP iterated prediction

Figure 3: Mackey Glass chaotic time series prediction.

Table 1: NRMSE of the long term iterated prediction for various RNN architectures and training algorithms.

	Mean	Var	n_H	n_W	T[s]
DDF_ELMAN_SIG	0.316	8.77e-3	10	121	20.88
UKF_ELMAN_SIG	0.419	3.43e-2	10	121	20.91
EKF_ELMAN_SIG	0.429	5.89e-2	10	121	14.98
DDF_FC_SIG	0.269	7.15e-3	10	131	23.43
UKF_FC_SIG	0.465	8.51e-2	10	131	23.78
EKF_FC_SIG	0.359	8.64e-2	10	131	17.38
DDF_NARX_SIG	0.0874	2.91e-4	5	41	5.64
UKF_NARX_SIG	0.119	1.89e-3	5	41	5.68
EKF_NARX_SIG	0.153	3.37e-3	5	41	4.76

4.2 Hénon Chaotic Time Series Prediction

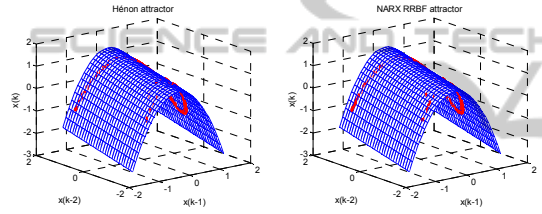
In our first example, we consider the prediction of

the long-term behavior of the chaotic Hénon dynamics:

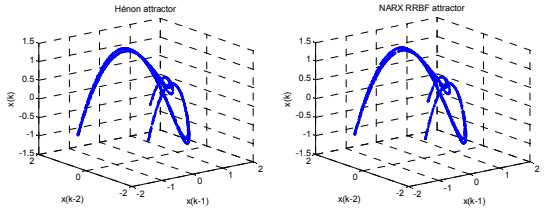
$$x_k = 1 - 1.4x_{k-1}^2 + 0.3x_{k-2} \quad (31)$$

RNNs with sigmoidal and Gaussian hidden neurons (we call this network Recurrent Radial Basis Function network – RBF network) were trained sequentially on 3000 samples. After training networks were iterated for 20 samples by feeding back the current outputs of the neurons as the new inputs. Figures 4 and 5 show results of prediction using NARX_RBF network trained by DDF on a Hénon chaotic time series.

It can be seen from Figures 4.a and 4.b, that, although the network was trained only using sample data chaotic attractor, which occupies small part of the surface defined by equation (31), the recurrent neural network was able to reconstruct that surface closely to the original mapping (Fig 4.a), as well as to reconstruct the original attractor (Fig 4.b).



a) Surface plot of the Hénon and NARX_RRBF map; Dots – chaotic attractor and NARX RRBF attractor



b) Phase plot of x_k versus x_{k-1} and x_{k-2} for the Hénon series and NARX RRBF iterated prediction

Figure 4: Hénon chaotic time series prediction: surface and phase plot.

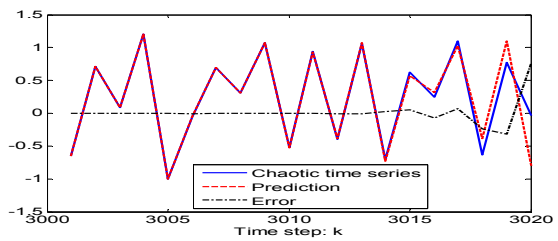


Figure 5: Comparison of the original chaotic time series and the NARX RRBF iterated prediction.

Results presented in Table 2. show that both DDF and UKF produce more accurate RNNs than EKF with comparable training time.

Table 2: Results of long term predictions of the Hénon chaotic time series.

	Mean	Var	n_H	n_W	T[s]
DDF_ELMAN_SIG	1.73e-2	6.19e-5	4	25	8.34
DDF_ELMAN_RBF	6.02e-2	3.89e-4	3	22	7.76
UKF_ELMAN_SIG	7.29e-2	9.46e-3	4	25	8.53
UKF_ELMAN_RBF	7.24e-2	1.79e-3	3	22	7.91
EKF_ELMAN_SIG	1.69e-1	5.17e-2	4	25	7.69
EKF_ELMAN_RBF	1.01e-1	7.50e-3	3	22	7.96
DDF_NARX_SIG	7.46e-3	3.39e-6	4	17	6.21
DDF_NARX_RBF	4.36e-3	4.15e-6	3	16	5.85
UKF_NARX_SIG	1.28e-2	2.68e-5	4	17	6.37
UKF_NARX_RBF	5.72e-3	7.14e-6	3	16	6.00
EKF_NARX_SIG	1.57e-2	1.65e-5	4	17	5.76
EKF_NARX_RBF	7.07e-3	1.35e-6	3	16	6.17

5 CONCLUSIONS

We considered the problem of recurrent neural network training as an approximate recursive Bayesian state estimation. Results in chaotic time series long term prediction show that derivative free estimators Divided Difference Filter and Unscented Kalman Filter considerably outperform Extended Kalman Filter as RNN learning algorithms with respect to the accuracy of the obtained network, while retaining comparable training times.

Experiments also show that of tree considered architectures: Elman, fully connected and non-linear AutoRegressive with eXogenous inputs (NARX) recurrent neural network, NARX is by far superior in both training time and accuracy of trained networks in long term prediction.

REFERENCES

Anderson, B. and J. Moore, 1979. *Optimal Filtering. Englewood Cliffs*, NJ, Prentice-Hall.

Julier, S. J., and Uhlmann, J. K., A new extension of the Kalman filter to nonlinear systems. 1997, *Proceedings of AeroSense: The 11th international symposium on aerospace/defence sensing, simulation and controls*, Orlando, FL.

Nørgaard, M., Poulsen, N. K., and Ravn, O., 2000, Advances in derivative free state estimation for nonlinear systems, *Technical Report, IMM-REP-1998-15*, Department of Mathematical Modelling, DTU.

Todorović, B., Stanković, M., and Moraga, C. 2003, On-line Learning in Recurrent Neural Networks using

Nonlinear Kalman Filters. In *Proc. of ISSPIT 2003*, Darmstadt, Germany.

Todorović, B., Stanković, M., and Moraga C., 2004, Nonlinear Bayesian Estimation of Recurrent Neural Networks. In *Proc. of IEEE 4th International Conference on Intelligent Systems Design and Applications ISDA 2004*, Budapest, Hungary, August 26-28, pp. 855-860.

Van der Merwe, R. and Wan, E. AS. 2001, Efficient Derivative-Free Kalman Filters for Online Learning. In *Proc. of ESSAN*, Bruges, Belgium.

Williams, R. J., & Zipser, D. 1989, A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1, 270-280.

Williams, R. J. and Zipser, D. 1990, Gradient-based learning algorithms for recurrent connectionist networks. *TR NU_CCS_90-9*. Boston, Northeastern University.