Training of Neural Networks Using a New Filter Based on the Kalman Filter

MOHAMED ELHASNAOUI
Service d'Automatique CP 165/55
Free University of Brussels
50, Av. F. Roosevelt, 1050 Brussels
BELGIUM
melhasna@ulb.ac.be

Abstract: In this paper we describe a recursive estimator for filtering nonlinear systems, [2]. It is based on the Kalman filter. Unlike the Extended Kalman filter, which is widely used for the prediction of nonlinear system, the linearisation is not required and avoids the calculation of the jacobian. This new filter is used for the training of the feedforward neural networks and is compared with the extended Kalman filter.

Key-Words: Neural networks, Kalman filter, Nonlinear system identification.

1 Introduction

Multilayer neural networks are general tools for modeling nonlinear functions since they can approximate any nonlinear function to any desired accuracy. It has been rigorously proved that any continuous function can be uniformly approximated by a feedforward neural network. The weights are fixed using a learning algorithm. Given a training set of input/output data, the original learning rule for multilayer perceptrons is the backpropagation algorithm [5]. It is a steepest descent algorithm, which is known to converge slowly. Many attempts have been made to improve the algorithm. Some of the more interesting methods involve the coupling of partial least squares with neural networks, or the use of conjugate gradient optimization [1]. These methods are significantly faster than backpropagation, and they can give more accurate results as well.

Other methods use the Kalman filter for training the feedforward and recurrent neural networks. The Kalman filter is one of the most widely used methods for estimating the state of processes described by linear stochastic dynamic model. However, in most applications of interest the system dynamics are nonlinear. The extended Kalman filter (EKF) was proposed in order to deal with the nonlinear systems. The EKF linearises the model around the current state estimate, and applies the traditional Kalman filter to the resulting time-varying linear model. The EKF is not an optimal state estimator and its convergence is not guaranteed..

The success of a Kalman filter depends on the accuracy of its predictions. Inaccurate predictions degrade filter performance and can lead to inconsistency. In linear systems an exact form solution exists. However when the system models are nonlinear there are significant problems with accurately predicting the state of the system.

Although the EKF is a widely used filtering strategy, it has in practice some well-known drawbacks: it is difficult to implement, difficult to tune and only reliable for systems which are almost linear on the time scale of update intervals. Linearisation can produce highly unstable filter performance if the time step intervals are not sufficiently small and the derivation of the jacobian matrices are non trivial in most applications and often lead to significant implementation difficulties. To over-

come those problems, a new recursive method was proposed in [2] which is based on a new parametrisation and avoid the linearisation steps required by the EKF.

In this paper, the new approach for filter prediction is used for the training of feedforward neural networks and it is compared with the standard EKF.

The structure of this paper is as follows. In section 2 we recall the Kalman filter. In section 3 we give the extended Kalman filter and the new filter. Section 4 deals with its application to the training of feedforward neural models. The test of the new algorithm is given in section 5.

2 The Kalman Filter

We consider the following discrete time process model, represented in state space form by:

$$\begin{aligned} x(k+1) &= f(x(k), u(k), \omega(k)) \\ y(k) &= g(x(k), u(k), \upsilon(k)) \end{aligned} \tag{1}$$

where $x(k) \in \mathbb{R}^n$ is the state of the system, $u(k) \in \mathbb{R}^q$ is a known input signal, $\omega(k) \in \mathbb{R}^m$ is an unknown disturbance signal, $\upsilon(k) \in \mathbb{R}^p$ is the measurement noise and $y(k) \in \mathbb{R}^p$ is the measured output. The function $f(.) : \mathbb{R}^n \to \mathbb{R}^n$ describes the dynamic relationship between the current state and the next state while $g(.) : \mathbb{R}^n \to \mathbb{R}^p$ describes the relationship of the measured outputs to the states and the inputs.

We assume that the process noise is a Gaussian, zero mean random variable and has covariance $Q(k)$. It is further assumed that the process noise at any time is independent of the process noises or the state of the system which have occurred at any previous time:

$$E\left[\upsilon(k)x(j)^T\right] = 0,\ E\left[\upsilon(k)\omega(j)^T\right] = 0$$

Once the system is in this form, the problem becomes the estimation of the state x from the measurement of y.

Determination of the Kalman filter:

The filter estimates the state using the process model, the observation model and a sequence of observations. By $\widehat{x}(i \mid k)$, we will mean the estimate of $x(k)$ using all observations up to and including time step k, $Y^k = \{y(0), y(1), \ldots, y(k)\}$. Defining the estimation error as

$$\widetilde{x}(i \mid k) = x(i) - \widehat{x}(i \mid k)$$

the covariance of $\widehat{x}(i \mid k)$ is given by

$$P(i \mid k) = E\left[\widetilde{x}(i \mid k)\widetilde{x}^T(i \mid k) \mid Y^k\right]$$

The estimate is obtained by updating the prediction with the current observation. The Kalman filter uses a particular form of its update rule: the estimate is equal to the prediction plus the weighted sum of the innovation vector,

$$\widehat{x}(k \mid k) = \widehat{x}(k \mid k-1) + \mathcal{W}(k)\nu(k)$$

the innovation vector is defined to be the difference between the actual observation, $y(k)$ and the predicted observation $\widehat{y}(k \mid k-1)$,

$$\nu(k) = y(k) - \widehat{y}(k \mid k-1)$$

$\mathcal{W}$ is the Kalman filter gain. It determines the degree to which the innovation affects the new estimate. The Kalman filter, in its general form, can be grouped in the following set of equations [2]:

$$\begin{aligned} \widehat{x}(k|k) &= \widehat{x}(k|k-1) + \mathcal{W}(k)\nu(k) \\ P(k|k) &= P(k|k-1) - P_{\nu\nu}(k|k-1)\mathcal{W}^T(k) \\ \mathcal{W}(k) &= P_{xy}(k|k-1)P_{\nu\nu}^{-1}(k|k-1) \\ \nu(k) &= y(k) - \widehat{y}(k|k-1) \end{aligned} \tag{2}$$

$P_{xy}(k \mid k-1)$ is the predicted cross-correlation matrix between $\widehat{x}(k \mid k-1)$ and $\widehat{y}(k \mid k-1)$
$P_{\nu\nu}(k \mid k-1)$ is the covariance of the innovation.

Linear system:

Let x be the state vector associated with the linear system:

$$x(k+1) = A(k)x(k) + w(k)$$

$A(k)$ is a known $n \times n$ matrix and $w(k)$ is an n-dimensional white process noise. It is assumed that

$$Q(k) = E\left[w(k)w(k)^T\right]$$

is known for each k, and that $E\left[w(k)x(k)^T\right] = 0$ for $j \leq k$.
The measurement equation is given by

$$y(k) = C(k)x(k) + \upsilon(k)$$

where $C(k)$ is a known $m \times n$ matrix, $\upsilon(k)$ is the measurement noise,

$$R(k) = E\left[\upsilon(k)\upsilon(k)^T\right]$$

is known, and the conditions

$$E\left[\upsilon(k)w(j)^T\right] = 0 \text{ and } E\left[\upsilon(k)x(j)^T\right] = 0$$

hold for all j and k.

Theorem 1 *(Kalman,1960). The linear minimum variance estimator estimator $\widehat{x}(k \mid k)$ may be generated recursively by:*
(a) $\widehat{x}(k+1 \mid k+1) = A\widehat{x}(k \mid k) + \mathcal{W}(k+1)\left[y(k+1) - CA\widehat{x}(k|k)\right]$, where $\mathcal{W}(k+1)$, the Kalman gain matrix is given by:
(b) $\mathcal{W}(k+1) = P(k+1|k)C^T\left[CP(k+1|k)C^T + \right.$
$\left. + R(k+1)\right]$

and $P(k+1 \mid k)$ is generated recursively by the equations
(c) $P(k|k) = \left[I - \mathcal{W}(k)C\right]P(k|k-1)$ (covariance update) and
(d) $P(k+1|k) = AP(k|k)A^T + Q(k)$ (the covariance extrapolations).

In the case of nonlinear systems, the optimal estimation problem remains unsolved, but there are several approaches which yield suboptimal performance. The simplest and most widely used approach is the extended Kalman filter which is a recursive algorithm. The recursive structure allows simple computations, as only new data is operated upon to produce a new estimate; all old data can be thrown away. The EKF is not an optimal state estimator and its convergence is not guaranteed, but nevertheless it is used in many applications. The next section deals with this type of filter.

3 The extended Kalman filter

We consider the nonlinear system described by the equation (1). The extended Kalman filter (EKF) predicts the future state of the system under the assumption that its process and observation models are linear on the scale of the error. Expanding the equation (1) as a taylor series about $\widehat{x}(k \mid k)$, the true system propagates according to :

$$\begin{aligned} x(k+1) = f(\widehat{x}(k \mid k), u(k), 0, k) + \nabla f_x \widetilde{x}(k \mid k) \\ + \nabla f_\omega \omega(k) + h.o.t \end{aligned}$$

where ∇f_x is the Jacobian of $f(.)$ with respect to $x(k)$ and ∇f_ω is the jacobian with respect to $\omega(k)$. We assume that the second and higher order terms in this series are negligible, the predicted mean is

$$\begin{aligned} \widehat{x}(k+1 \mid k) &\approx E\left[f(\widehat{x}(k \mid k), u(k), 0, k)\right. \\ &\quad \left. + \nabla f_x \widetilde{x}(k \mid k) + \nabla f_\omega \omega(k) \mid Y^{k-1}\right] \\ &= f(\widehat{x}(k \mid k), u(k), 0, k) \end{aligned}$$

provided that both $\widetilde{x}(k-1 \mid k-1)$ and $\omega(k-1)$ are zero-mean random variables. The prediction error committed by this approximation is

$$\begin{aligned} \widetilde{x}(k+1 \mid k) &= x(k+1) - \widehat{x}(k+1 \mid k) \\ &\approx \nabla f_x \widetilde{x}(k \mid k) + \nabla f_\omega \omega(k) \end{aligned}$$

and the prediction covariance is

$$\begin{aligned} P(k+1 \mid k) = \nabla f_x P(k \mid k) \nabla^T f_x \\ + \nabla f_\omega Q(k) \nabla^T f_\omega \end{aligned}$$

in order to complete the set of equations in (2), we can easily compute the following expressions

$$\begin{aligned} \widehat{y}(k+1|k) &= g(\widehat{x}(k|k), u(k), 0, k) \\ P_{yy}(k+1|k) &= \nabla g_x P(k+1|k) \nabla^T g_x + \\ &\quad \nabla g_\omega R(k) \nabla^T g_\omega \\ P_{xy}(k+1|k) &= P(k+1|k) \nabla^T g_x \end{aligned}$$

The Extended Kalman Filter approach is, thus, apply the standard Kalman filter to nonlinear systems by continually updating a linearisation around the previous state estimate, starting with an initial guess. In other words, we only consider a linear Taylor approximation of the system function at a previous state estimate and that of the observation function at the corresponding predicted position.

The EKF suffers of some drawbacks. In order to implement the EKF, it is necessary to evaluate, analytically, the Jacobian matrices of the process and the observation models. In most applications, the Jacobian matrices are difficult to derive. The linearisation can produce highly unstable filters if the assumption of local linearity is violated. To overcome those problems a new filter has been proposed by Julier *et al* [2]. It is based on the approximation of the distribution rather that the approximation of the process or observation model. It uses a set of chosen points to parametrise the means and covariances of probability distributions. In the new filter, it is not necessary to calculate the Jacobian.

***Derivation of the new filter*, [2]:**

The problem is as follows: having the mean $\widehat{x}(k \mid k)$ and covariance $P(k \mid k)$ we would like to predict $\widehat{x}(k+1 \mid k)$ and $P(k+1 \mid k)$ through the nonlinear function $f(.)$. There are three steps to follows:

1. Compute the set $\sigma(k \mid k)$ of $2n$ points from the rows or columns of the matrices $\pm\sqrt{nP(k \mid k)}$. This set is zero mean and covariance $P(k \mid k)$. Compute a set of points with the same covariance but with mean $\widehat{x}(k \mid k)$, by translating each of the points as $\chi_i(k \mid k) = \sigma_i(k \mid k) + \widehat{x}(k \mid k)$.
2. Transform each point as $\chi_i(k+1 \mid k) = f\left[\chi_i(k \mid k), u(k+1), k+1\right]$.
3. Compute $\widehat{x}(k+1 \mid k)$ and $P(k+1 \mid k)$ by computing the mean and covariance of the $2n$ points in the set $\chi_i(k+1 \mid k)$.

In appendix(.) we give the detail of the new filter.

4 Application to neural networks

The extended Kalman Filter as a tool for the recursive parameter estimation of static and dynamic nonlinear models, has been applied to the estimation of the weights of feedforward and recurrent neural networks [3] [4].

We consider the following process

$$y(k) = f(x(k)) + \upsilon(k)$$

with u and y are, respectively, the input and the output vectors, υ is a random variable with zero mean and variance σ_υ^2. The modeling problem consist in finding a neural network which gives a good estimation of the regression in the input domain of interest. Since feedforward neural networks has been shown to be universal approximators, there exists at least a feedforward neural network $\mathcal{N}(u, \mathcal{W})$, where $\mathcal{W} \in \mathbb{R}^M$ represents the weights of the network, such that

$$|f(u) - \mathcal{N}(x, \mathcal{W})| < \varepsilon$$

in the input domain of interest, with $\varepsilon > 0$ an arbitrary small scalar. The new filter developed in the previous section is used as a recursive algorithm for the estimation of the weights of the neural model by considering the following dynamic system

$$\begin{cases} \mathcal{W}(k+1) = \mathcal{W}(k) \\ \widehat{y}(k) \quad\quad = \mathcal{N}(x(k), \mathcal{W}) + \upsilon(k) \end{cases}$$

where $x(k)$ is the input vector at time k and $\widehat{y}(k)$ is the output vector at time k, $\upsilon(k)$ is assumed to be a white noise vector with the covariance matrix $R(k)$ due to the modelling error. The purpose of the weight learning of the multilayer neural network is to estimate the weight vector $\mathcal{W}$ such that the output $\widehat{y}(k)$ tracks the desired output $y(k)$ of the process using a training set $\left\{x^k, y^k\right\}_{k=1:N}$. In absence of any prior knowledge on the process, the weights are initialised with small random values.

5 Simulation example

In this section we will give a numerical exam-

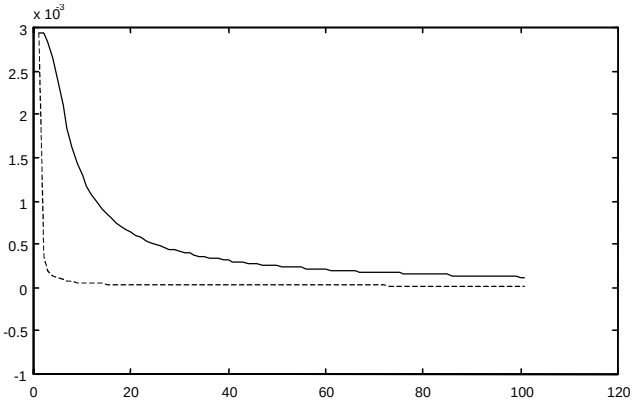

Fig. 1. Evolution of the errors: EKF (full line) and the new filter (dashed line)

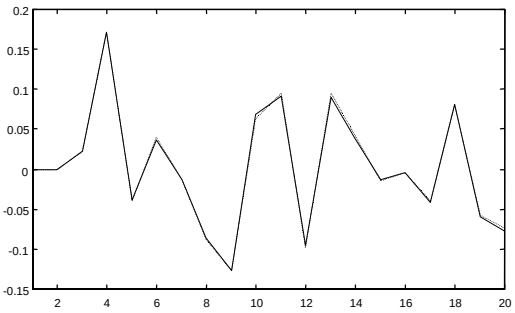

Fig. 2. Nonlinear system identification using the new filter: output of the process (full line) and the output of the neural network (dashed line)

ple. It deals with the modeling of a nonlinear simulated process, which is described by the following equation:

$$y_k = \frac{1}{5}\left(\frac{y_{k-2}y_{k-1}\left(y_{k-1}+2.5\right)}{1+y_{k-2}^2+y_{k-1}^2}+u_{k-2}\right)$$

A feedforward neural network is trained to model the process, having one hidden layer of 5 nodes with $tanh$ as activation function. the training control input sequence $\{u_k\}$ consists of 200 random variables with a uniform distribution in $[-1,1]$. The performance of the new filter is compared with the standard extended Kalman filter. The results of the simulation are shown in fig. 1 and fig. 2. It is remarkable that, in this example, the proposed filter converges more fast than the EKF.

6 Conclusion

We have presented a new algorithm for the training of the feedforward neural networks. It is based on the Kalman filter. Unlike the extended kalman filter, it avoids the linearisation and the calculation of the jacobian of the nonlinear function.

References

[1] C. Chalambous. Conjugate gradient algorithm for efficient trainnig of artificial neural networks. *IEE Proceeding-G*, Vol.139,No.3, 1992,pp.301-310.

[2] S. J. Julier and J. K. Uhlmann. A new approach for filtering nonlinear systems. *The Proceedings of the American Control Conference, Seattle, Washington*, 1995, pp.1628–1632.

[3] G. V. Puskorius and L. A. Feldkamp. Neurocontrol of nonlinear dynamical systems with kalman filter trained recurrent networks. *IEEE Trans. on Neural Networks*,Vol.5, pp.279-297, 1994.

[4] I. Rivals and L. Personnaz. A recursive algorithm based on the extended kalman filter for training of feedforward neural models. *To appear in Neurocomputing.*

[5] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representation by error propagation. Parallel distributed processing: Explorations in the microstructure of cognition, *MIT Press, Cambridge*, 1986, pp.318-360.

Appendix:

In this appendix we summarise the steps of the new filter, [2]:

1. The set of σ points is computed from the $n \times n$ matrix $P(k \mid k)$ as

$$\begin{aligned}
&\sigma(k \mid k) \leftarrow 2n \text{ columns from } \sqrt{\pm(n+\kappa)P(k \mid k)},\ \kappa \in \mathbb{R} \\
&\chi_0(k \mid k) = \widehat{x}(k \mid k) \\
&\chi_i(k \mid k) = \sigma_i(k \mid k) + \widehat{x}(k \mid k)
\end{aligned}$$

2. The predicted mean is computed as

$$\widehat{x}(k+1 \mid k) = \frac{1}{n+\kappa}\left\{\kappa\chi_0(k+1 \mid k) + \frac{1}{2}\sum_{i=1}^{2n}\chi_i(k+1 \mid k)\right\}$$

   where $\chi_i(k+1 \mid k) = f\left[\chi_i(k \mid k), u(k+1), k+1\right]$

3. The predicted covariance is computed as

$$\begin{aligned}
P(k+1 \mid k) = \frac{1}{n+\kappa}\{&\kappa\left[\chi_0(k+1 \mid k) - \widehat{x}(k+1 \mid k)\right]\left[\chi_0(k+1 \mid k) - \widehat{x}(k+1 \mid k)\right]^T \\
&+\frac{1}{2}\sum_{i=1}^{2n}\left[\chi_i(k+1 \mid k) - \widehat{x}(k+1 \mid k)\right]\left[\chi_i(k+1 \mid k) - \widehat{x}(k+1 \mid k)\right]^T
\end{aligned}$$

4. The predicted measure is calculated by

$$\widehat{y}(k+1 \mid k) = \frac{1}{n+\kappa}\left\{\kappa\mathcal{Y}_0(k+1 \mid k) + \frac{1}{2}\sum_{i=1}^{2n}\mathcal{Y}_i(k+1 \mid k)\right\}$$

   where $\mathcal{Y}_i(k+1 \mid k) = f\left[\chi_i(k+1 \mid k), u(k+1), k+1\right]$

5. The corresponding covariance is determined by

$$\begin{aligned}
P_{yy}(k+1 \mid k) = \frac{1}{n+\kappa}\Big\{&\kappa\left[\mathcal{Y}_0(k+1 \mid k) - \widehat{y}(k+1 \mid k)\right]\left[\mathcal{Y}_0(k+1 \mid k) - \widehat{y}(k+1 \mid k)\right]^T \\
&+\frac{1}{2}\sum_{i=1}^{2n}\left[\mathcal{Y}_i(k+1 \mid k) - \widehat{y}(k+1 \mid k)\right]\left[\mathcal{Y}_i(k+1 \mid k) - \widehat{y}(k+1 \mid k)\right]^T\Big\}
\end{aligned}$$

   where $P_{\nu\nu}(k+1 \mid k) = P_{yy}(k+1 \mid k) + R(k+1)$.

6. Finally the cross correlation matrix is determined by

$$\begin{aligned}
P_{xy}(k+1 \mid k) = \frac{1}{n+\kappa}\Big\{&\kappa\left[\chi_0(k+1 \mid k) - \widehat{x}(k+1 \mid k)\right]\left[\mathcal{Y}_0(k+1 \mid k) - \widehat{y}(k+1 \mid k)\right]^T \\
&+\frac{1}{2}\sum_{i=1}^{2n}\left[\chi_i(k+1 \mid k) - \widehat{x}(k+1 \mid k)\right]\left[\mathcal{Y}_i(k+1 \mid k) - \widehat{y}(k+1 \mid k)\right]^T\Big\}
\end{aligned}$$