Dual Extended Kalman Filter Algorithm for Training RBF Networks

Iulian Ciocoiu*

Abstract

We propose a new supervised learning algorithm for training RBF networks. It uses a pair of parallel running Kalman filters to sequentially update both the weights and the centres of the network. Simulation results for solving the classical 2-spirals classification problem are reported, and a comparison with other approaches is discussed.

1 Introduction

Radial Basis Function (RBF) networks have been traditionally used as a multidimensional interpolation technique of general mappings $f: R^m \to R$ according to [1, 2]:

$$f(\boldsymbol{X}) = w_0 + \sum_{i=1}^{M} w_i \Phi(\left\| \boldsymbol{X} - \boldsymbol{C}^i \right\|) \qquad (1)$$

where ϕ is a nonlinear function selected from a set of typical ones, $\| . \|$ denotes the Euclidean norm, w_i are the tap weights and $C^i \in R^m$ are called RBF centers. It is easy to see that the formula above is equivalent to a special form of a 2-layer perceptron, which is *linear in the parameters* by fixing all the centers and nonlinearities in the hidden layer. The output layer simply performs a linear combination of the (nonlinearly) transformed inputs and thus the tap weights w_i can be obtained by using the standard LMS algorithm or its momentum version. This leads to a dramatic reduction of the computing time with the supplementary benefit of avoiding the problem of local minima, usually encountered when simulating standard multilayer perceptrons.

The approximation capabilities of RBF networks critically depend on the choice of the centres. Most existing approaches use a hybrid training strategy, using an unsupervised algorithm (*e.g.,* k-means clustering or Kohonen's self-organizing maps [2]) to pick the centres, followed by a supervised one to obtain the weights. Anyway, in order to acquire optimal performance the centres training procedure should also include the target data [3], leading to a form of supervised learning which proved superior in several applications [4].

In this paper we propose the use of the Kalman filter as a framework for the supervised training of both weights and centres of the network, following the line of previous work related to considering the training procedure of a neural network as an estimation problem [5, 6]. In the context of RBF networks the Kalman filter was used for weights estimation only [7], and combined weights and centres estimation, by concatenating them into a joint state vector [8]. In the following we analyse the efficiency of a novel approach, based on using a *pair* of parallel running Kalman filters to sequentially update both the weights and the centres of the network. We call it a Dual Extended Kalman Filter (DEKF) algorithm, by analogy with a similar concept introduced in [9], where a pair of Kalman filters was used for combined estimation of the states and the weights of a standard multilayer perceptron.

2 The proposed algorithm

Kalman filtering theory requires the formulation of the problem within a state-space framework [10]. It was originally introduced for linear models, but linearization can be used to extend the method for the nonlinear case too. Given a controllable and observable system we may write:

$$\begin{aligned} \boldsymbol{X}[k] &= \Psi[k, k-1]\boldsymbol{X}[k-1] + \boldsymbol{v}[k-1] \\ \boldsymbol{y}[k] &= \boldsymbol{C}[k])\boldsymbol{X}[k] + \boldsymbol{q}[k] \end{aligned} \qquad (2)$$

where Ψ[k,k-1] is the state transition matrix, **v**[k] is the input driving noise, and **q**[k] is the measurement noise, which are specified by:

*Technical University of Iasi, Romania
Faculty of Electronics and Telecommunications

$$E\{\boldsymbol{v}[k]\boldsymbol{v}^T[n]\} = \delta_{kn}\boldsymbol{Q}[n]$$
$$E\{\boldsymbol{q}[k]\boldsymbol{q}^T[n]\} = \delta_{kn}\sigma^2{}_q[n] = \delta_{kn}\boldsymbol{R}[n] \quad (3)$$
$$E\{\boldsymbol{v}[k]\boldsymbol{q}^T[n]\} = \boldsymbol{0}$$

When nonlinear models are used the linearization of the equations around the current operating point is needed, thus approximating a nonlinear function by a time-varying linear one. The matrix Ψ[k,k-1] must be replaced with the Jacobian of the (nonlinear) function appearing in the state transition equation, leading to the formulation of the Extended Kalman filter algorithm. In the following, we will recast equation (2) in order to cope with the above requirements.

A. Estimation of the weights

Similar to previous work [x,y], estimation of the weights is performed assuming the following *model*:

$$\begin{aligned} \boldsymbol{W}[k] &= \boldsymbol{W}[k-1] \\ y[k] &= \Phi(\boldsymbol{C}[k-1])\boldsymbol{W}[k] + q[k] \end{aligned} \quad (4)$$

where: $\boldsymbol{W}[k] = \left[w_0 \;\; w_1 \cdots w_M\right]^T$,

$\Phi(\boldsymbol{C}[k]) = \left[1 \;\; \Phi(\left\|\boldsymbol{X}[k]-\boldsymbol{C}^1\right\|) \;\; \Phi(\left\|\boldsymbol{X}[k]-\boldsymbol{C}^2\right\|) \ldots \Phi(\left\|\boldsymbol{X}[k]-\boldsymbol{C}^M\right\|)\right]$,

and q[k] is the measurement noise, assumed white with variance σ^2_q (in the following we will denote $\Phi(\mathbf{C}[k])$ simply by $\Phi[k]$). It is important to observe that in the case of RBF networks with *fixed* centres the estimation (learning) problem is a *linear* one, as opposed to the case of standard MLP networks. Moreover, the state transition matrix is simply an identity matrix, the process noise **v**[n] is null, and **Q**[n]=0.

According to the specific operating mode of the Kalman filter, we may compute the (least square) estimate of the weights vector $\hat{\boldsymbol{W}}[k]$ and its prediction $\hat{\boldsymbol{W}}^-[k]$, along with their respective error covariance matrices $P_{\boldsymbol{W}}[k]$, and $P_{\boldsymbol{W}}{}^-[k]$ according to [10]:

$$\boldsymbol{g}_w[k] = \boldsymbol{P}_{\boldsymbol{W}}{}^-[k]\Phi^T[k] * \{\Phi[k]\boldsymbol{P}_{\boldsymbol{W}}{}^-[k]\Phi^T[k] + \sigma^2{}_q\}^{-1} \quad (5)$$
$$\hat{\boldsymbol{W}}[k] = \hat{\boldsymbol{W}}^-[k] + \boldsymbol{g}_w[k]\{y[k] - \Phi[k]\hat{\boldsymbol{W}}^-[k]\} \quad (6)$$
$$\boldsymbol{P}_{\boldsymbol{W}}[k] = \boldsymbol{P}_{\boldsymbol{W}}{}^-[k] - \boldsymbol{g}_w[k]\Phi[k]\boldsymbol{P}_{\boldsymbol{W}}{}^-[k] \quad (7)$$
$$\hat{\boldsymbol{W}}^-[k+1] = \hat{\boldsymbol{W}}[k] \quad (8)$$
$$\boldsymbol{P}_{\boldsymbol{W}}{}^-[k+1] = \boldsymbol{P}_{\boldsymbol{W}}[k] \quad (9)$$

where $\mathbf{g}_\mathbf{W}$[k] designates the current value of the so-called *Kalman gain*.

B. Estimation of the centres

A second EKF is used to estimate the centres, which are described by the state equations:

$$\begin{aligned} \boldsymbol{C}[k] &= \boldsymbol{C}[k-1] \\ y[k] &= f\{\boldsymbol{X}[k], \Phi(\boldsymbol{C}[k]), \boldsymbol{W}[k-1]\} + q[k] \end{aligned} \quad (10)$$

The adaptation algorithm requires the linearization of the equation above and is given by:

$$\boldsymbol{g}_{\boldsymbol{C}}[k] = \boldsymbol{P}_{\boldsymbol{C}}{}^-[k]J^T[k] * \{\boldsymbol{J}[k]\boldsymbol{P}_{\boldsymbol{C}}{}^-[k]\boldsymbol{J}^T[k] + \sigma^2{}_q\}^{-1} \quad (11)$$
$$\hat{\boldsymbol{C}}[k] = \hat{\boldsymbol{C}}^-[k] + \boldsymbol{g}_{\boldsymbol{C}}[k]\{y[k] - \Phi(\hat{\boldsymbol{C}}^-[k])\hat{\boldsymbol{W}}^-[k]\} \quad (12)$$
$$\boldsymbol{P}_{\boldsymbol{C}}[k] = \boldsymbol{P}_{\boldsymbol{C}}{}^-[k] - \boldsymbol{g}_{\boldsymbol{C}}[k]\boldsymbol{J}[k]\boldsymbol{P}_{\boldsymbol{C}}{}^-[k] \quad (13)$$

where:

$$\boldsymbol{J}[k] = \frac{\partial f[\hat{\boldsymbol{C}}[k], \hat{\boldsymbol{W}}]}{\partial \hat{\boldsymbol{C}}[k]} \quad (14)$$

$$\hat{\boldsymbol{C}}^-[k+1] = \hat{\boldsymbol{C}}[k] \quad (15)$$

$$\boldsymbol{P}_{\boldsymbol{C}}{}^-[k+1] = \boldsymbol{P}_{\boldsymbol{C}}[k] \quad (16)$$

and:

$$J_{ij}[k] = \{y[k] - d[k]\}w_{kj} e^{-\frac{\left\|\boldsymbol{X}[k]-\boldsymbol{C}^j\right\|^2}{2\sigma_j^2}} \frac{X_i[k] - \boldsymbol{C}_i{}^j}{\sigma^2{}_j} \quad (17)$$

C_i^j denotes component i of centre vector $\mathbf{C}^j$, d[k] is the current desired output, and y[k] is the output of the RBF network.

The learning algorithm works on a pattern-by-pattern basis: the adaptation procedure consists in consecutively performing the modification of the centres (eq.(11)-(16)) and the weights (eq. (5)-(9)) on the arrival of each training input-output pair. Typically, the initial values of the centres should be obtained after an unsupervised training phase, while the weights are initialised to small random values. The initial values of the symmetric error covariance matrices reflect the uncertainty in locating the weights and the centres, and are typically chosen proportional to an identity matrix of proper order [8, 10]. The measurement noise variance σ^2_q may be estimated from noisy data as in [9].

3 Simulation results

We have tested the efficiency of our algorithm on a difficult classification task, namely the 2-spirals problem. A set of 194 training points lying on two distinct spirals in the x-y plane should be correctly classified. The spirals twist three times around the origin and around each other, as in Fig. 1.

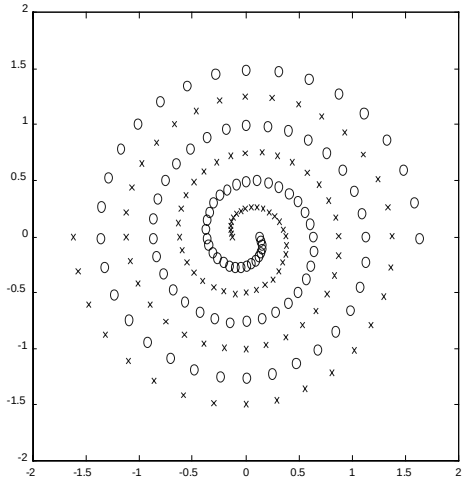
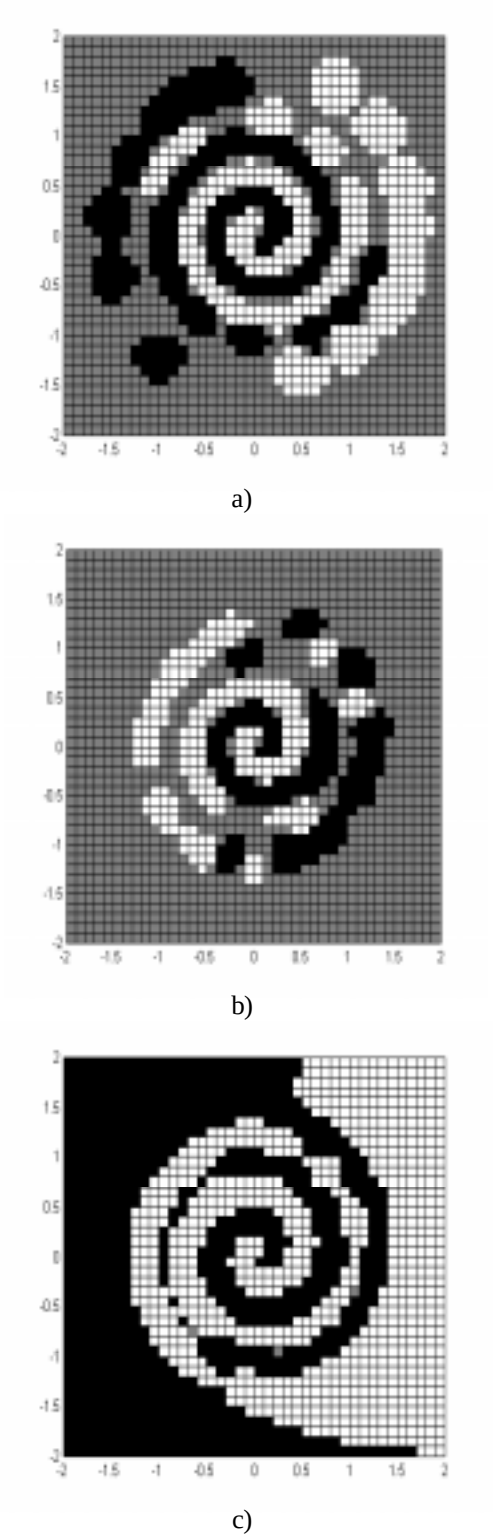


Fig. 1: The training dataset

This is a benchmark problem considered to be extremely difficult for standard multilayer networks trained with classical back-propagation class algorithms, although successful results were reported using other architectures or learning strategies [11,12,13].

We performed intensive computer simulations using gaussian activation functions, variable number of centres and initialisation procedures. We tested three different algorithms, namely the Dual Extended Kalman Filtering approach described above (DEKF), standard Kalman Filter estimation of weights only (KF), and the gradient-descent (GD) procedure described in [3]. The classification performances were tested on a separate set of 41x41 points, uniformly distributed on the surface covered by the training data. In Fig. 2 we present the results for M=96 centres, evenly selected initially from the training database. Convergence was typically reached in about 150 training epochs with DEKF, while the gradient-descent required several thousands of epochs and a careful tuning of the learning parameters. Rigorously speaking, the σ_j parameters of the gaussian functions should also be estimated during the training phase, but considering an extra Kalman filter would render the computational cost excessive. Since the approximation capabilities of RBF networks are still preserved using a common value for those parameters [2], they were all taken equal to 0.3. We have also tested the possibility of heuristically adapting their values according to the distance between the centres, but no improvement was obtained.



a)

b)

c)

Fig. 2: M=96 centres: a) DEKF; b) KF; c) GD

4 Conclusions

We analysed the efficiency of a new training algorithm for RBF networks relying on the Kalman filter. It offers advantages when compared to standard gradient-descent procedures since it considers explicitly the correlation between the weights of the network. As a consequence, the convergence speed is much higher than for the LMS-type algorithms, and the final error values are smaller. In the case of large networks, the memory requirements for storing the (symmetric) error covariance matrices could become prohibitive, and pruning techniques need to be used.

References:

[1] Broomhead, D.S., and Lowe, D., "Multivariable functional interpolation and adaptive networks", *Complex Syst.*, vol. 2, pp. 321-355, 1988

[2] Haykin, S., *Neural Networks - A Comprehensive Foundation*, IEEE Press, 1994

[3] Bishop, C.M., *Neural Networks for Pattern Recognition*, NY: Oxford University Press, 1995

[4] Wettschereck, D., and T. Dietterich, " Improving the performance of radial basis function networks by learning centre locations", in *Advances in Neural Information Processing Systems 4* (J.E. Moody, S.J. Hanson, and R.P. Lippmann, eds.), pp. 1133-1140, San Mateo, CA: Morgan Kaufmann, 1992

[5] Connor, J., Martin, R., and Atlas, L., "Recurrent neural networks and robust time series prediction", *IEEE Trans. on Neural Networks*, vol. 5, no. 2, pp. 240-254, 1994

[6] Puskorious, G., and Feldkamp, L., "Neural Control of Nonlinear Dynamic Systems with Kalman Filter Trained Recurrent Networks", *IEEE Trans. on Neural Networks*, vol. 5, no. 2, pp. 279-297, 1994

[7] Kadirkamanathan, V., M. Niranjan, and F. Fallside, "Models of dynamic complexity for time-series prediction", in *Proc. ICASSP*, San Francisco, 1992

[8] Nabney, I.T., "Practical methods of tracking of non-stationary time series applied to real world problems", *AeroSense '96: Applications and Science of Artificial Neural Networks II* (S.K. Rogers, and D.W. Ruck, eds.), pp. 152-163, SPIE Proc. No. 2760, 1996

[9] Nelson, A.T., and E.A. Wan, "Neural Speech Enhancement Using Dual Extended Kalman Filtering", in *Proc. ICNN'97*, Houston, TX, pp. 2171-2175, 1997

[10] Haykin, S., *Adaptive Filter Theory*, 2nd ed., NY: Wiley, 1991

[11] Fahlman, S.E., C. Lebiere, "The cascade-correlation learning architecture", in *Advances in Neural Information Processing Systems 2*, pp. 524-532, San Mateo: Morgan Kaufmann, 1990

[12] Lengelle, R., and T. Den∩ux, " Training MLPs Layer by Layer Using an Objective Function for Internal Representations", *Neural Networks*, vol. 9, no. 1, pp. 83-97, 1996

[13] Platt, J., "A resource allocating network for function interpolation", *Neural Computation*, vol. 3, no. 2, 1991