Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Dead-zone Kalman filter algorithm for recurrent neural networks

José de Jesús Rubio, Wen Yu

Abstract— **Compared to normal learning algorithms, for example backpropagation, Kalman filter-based algorithm has some better properties, such as faster convergence, although this algorithm is more complex and sensitive to the nature of noises. In this paper, extended Kalman filter is applied to train state-space recurrent neural networks for nonlinear system identification. In order to improve robustnees of Kalman filter algorithm dead-zone robust modification is applied to Kalman filter. Lyapunov method is used to prove that the Kalman filter training is stable.**

I. Introduction

Resent results show that neural network technique seems to be very effective to identify a broad category of complex nonlinear systems when complete model information cannot be obtained. Neural networks can be classified as feedforward and recurrent ones [7]. Feedforward networks, for example Multilayer Perceptrons (MLP), are implemented for the approximation of nonlinear functions in the right hand side of dynamic model equations. The main drawback of these neural networks is that the weights' updating do not utilize information on the local data structure and the function approximation is sensitive to the training data [13]. Since recurrent networks incorporate feedback, they have powerful representation capability and can successfully overcome disadvantages of feedforward networks [8]. Even though backpropagation (BP) has been widely used as a practical training method for neural networks, the limitations are that it may converge very slowly, there exists local minima problem and, the training process is sensitive to measurement noise. The stability of modified backpropagation algorithm is proved in [20].

Gradient-like learning laws are relatively slow. In order to solve this problem, many descendent methods in the identification and filter theory have been proposed to estimate the weights of neural networks. For example the extended Kalman filter is applied to train neural networks in [1], [9], [17], [18] and [19], they can give solutions of least-square problems. Most of them use static neural networks, sometimes the output layer must be linear and the hidden layer weights are chosen at randomly [3]. A faster convergence with the extended Kalman filter is reached, because it employes less interaction [9]. However, the computational complexity in each interaction is increased, it require of large amount of memory. Decoupling technique is used to decrease computational burden [14], the decoupled Kalman filter with diagonal matrix P is similar to gradient algorithm [7].

José de Jesús Rubio and Wen Yu are with the Departamento de Control Automatico, CINVESTAV-IPN, Av.IPN 2508, México D.F., 07360, México yuw@ctrl.cinvestav.mx

There are not so many stability analyses for Kalman filter training, in spite of reported successful Kalman filter applications. [6] analyzed convergence and stability properties of the Kalman filter for linear stochastic time-varying regression models. [16] proved that the estimation error of Kalman filter remains bounded if the system satisfies a detectability condition. [15] stated that the Kalman filter is exponentially stable when the covariance P is bounded and filter error is small enough, these conditions are very hard. [4] presented a new approach to finite-horizon guaranteed state prediction for discrete-time systems affected by bounded noise and unknown-but-bounded parameter uncertainty. There are only a few published results on stability analysis of neural networks training with Kalman filter. [12] used H_∞-learning to improved the robustness of Kalman filter training. By using results on stochastic stability of Kalman filter, [1] analyzed the convergence of the weights of neural networks with the assumption of the covariance P_k being bounded. The lack of robustness in Kalman filter with respect to noise was demonstrated in [12]. Several robust modification techniques were proposed for the least square algorithm [5] which is special cases of Kalman filter. To the best of our knowledge, Kalman filter training for recurrent neural networks and Lyapunov stability analysis have not yet been established in the literature.

In this paper the extended Kalman filter is modified with dead-zone technique, and is applied for state-space recurrent neural networks training. Both hidden layers and output layers can be updated. Stability analysis of identification error with the Kalman filter algorithm is given by the Lyapunov stability technique. A simple simulation gives the effectiveness of the suggested algorithm.

II. Recurrent neural networks training with extended Kalman filter

Consider following unknown discrete-time nonlinear system

$$x(k+1) = f\left[x(k), u(k)\right] \tag{1}$$

where $u\left(k\right) \in \Re^m$ is the input vector, $\left|u(k)\right|^2 \leq \overline{u}$, $x\left(k\right) \in \Re^n$ is a state vector, f is general nonlinear smooth function $f \in C^\infty$. We use the following state-space recurrent neural network to identify the nonlinear plant (1)

$$\widehat{x}(k+1) = A\widehat{x}(k) + V_{1,k}\sigma\left[W_{1,k}x(k)\right] + V_{2,k}\phi\left[W_{2,k}x(k)\right]u(k) \tag{2}$$

where $\widehat{x}\left(k\right) \in \Re^n$ represents the internal state of the neural network . The matrix $A \in \Re^{n\times n}$ is a stable diagonal matrix which will be specified after, $A = diag(a_i)$, $|a_i| < 1$. The weights in output layer are $V_{1,k}$, $V_{2,k} \in R^{n\times m}$, the

0-7803-9568-9/05/$20.00 ©2005 IEEE

weights in hidden layer are $W_{1,k},\ W_{2,k} \in R^{m\times n}$, σ is $m-$dimension vector function $\sigma = [\sigma_1 \cdots \sigma_n]^T$, $\phi(\cdot)$ is $\Re^{m\times m}$ diagonal matrix. According to the Stone-Weierstrass theorem and density properties of recurrent neural networks [8], the unknown nonlinear system (1) can be written in the following form

$$\begin{array}{l} x(k+1) = Ax(k) + V_{1,k}\sigma\left[W_{1,k}x(k)\right] \\ +V_{2,k}\phi\left[W_{2,k}x(k)\right]u(k) - \eta(k) \end{array} \tag{3}$$

where $\eta(k) = f\left[x(k), u(k)\right] - Ax(k) - V_{1,k}\sigma\left[W_{1,k}x(k)\right] - V_{2,k}\phi\left[W_{2,k}x(k)\right]u(k)$ is modeling error with respect to the weights $V_{1,k},\ V_{2,k},\ W_{2,k}$ and $W_{2,k}$, they are time-varying weights which will be updated by identification error. By [8] we know that the term $\mu(k)$ can be made arbitrarily small by simply selecting appropriate the number of neurons in the hidden layer (in this paper, it is m). In the case of two independent variables, a smooth function f has Taylor formula as

$$f = \sum_{k=0}^{l-1}\frac{1}{k!}\left[\left(x_1 - x_1^0\right)\frac{\partial}{\partial x_1} + \left(x_2 - x_2^0\right)\frac{\partial}{\partial x_2}\right]_0^k f + \varepsilon \tag{4}$$

where ε is the remainder of the Taylor formula. If we let x_1 and x_2 correspond $W_{1,k}x(k)$ and $V_{1,k},\ x_1^0, x_2^0$ correspond $W_1^0x(k)$ and V_1^0, and define $\widetilde{W}_{1,k} = W_{1,k} - W_1^0$, $\widetilde{V}_{1,k} = V_{1,k} - V_1^0$, V_1^0, V_2^0, W_1^0 and W_2^0 are set of known initial constant weights.

$$V_{1,k}\sigma\left[W_{1,k}x(k)\right] = V_1^0\sigma\left[W_1^0x(k)\right] + \Theta_{1,k}B_{1,k} + \varepsilon_1 \tag{5}$$

where $B_{1,k} = \left[\sigma, \sigma' V_{1,k}^T x\right]^T \in R^{2m\times 1}$, $\Theta_{1,k} = \left[V_{1,k}, W_{1,k}^T\right]^T \in R^{n\times 2m}$, σ' is the derivative of nonlinear activation function $\sigma(\cdot)$ at the point of $W_{1,k}x(k)$. Similar

$$V_{2,k}\phi\left[W_{2,k}x(k)\right]u(k) = V_2^0\phi\left[W_2^0x(k)\right]u(k) + \Theta_{2,k}B_{2,k} + \varepsilon_2 \tag{6}$$

where $B_{2,k} = \left[\phi u, \phi' \mathrm{diag}(u) V_{2,k}^T x(k)\right]^T$, $\Theta_{2,k} = \left[\begin{array}{cc} V_{2,k} & W_{2,k}^T \end{array}\right]^T$. We define the modelling error $\zeta(k) = \varepsilon_1 + \varepsilon_2 - \eta(k)$, substituting (5) and (6) into (3) we have

$$y(k) = B_k\Theta_k + \zeta(k) \tag{7}$$

where $\Theta_k = \left[\begin{array}{cc} \Theta_{1,k} & \Theta_{2,k} \end{array}\right]^T$, $B_k = \left[\begin{array}{cc} B_{1,k} & B_{2,k} \end{array}\right]$, the output $y(k)$ is

$$\begin{array}{l} y(k) = x(k+1) - Ax(k) \\ -V_1^0\sigma\left[W_1^0x(k)\right] - V_2^0\phi\left[W_2^0x(k)\right]u(k) \end{array}$$

Now we use Kalman filter technique to train the recurrent neural networks (2) such that the identification error $\epsilon_i(k)$ between the plant (1) and the neural networks (2), *i.e.*, $\epsilon_i(k) = \widehat{x}_i(k) - x_i(k)$, is minimized. The parameter matrix Θ_k is assumed to be an unknown constant plus a small random walk component, an artificial process noise Ω_k is defined to serve this change, *i.e.*, $\Theta_{k+1} = \Theta_k + \Omega_k$, $\Omega_k = [\omega_1(k) \cdots \omega_n(k)]^T$. We rewrite (7) in state-space with single output

$$\left\{\begin{array}{l} \theta_i(k+1) = \theta_i(k) + \omega_i(k) \\ y_i(k) = B_k^T\theta_i(k) + \zeta_{i,k} \end{array}\right. \tag{8}$$

where $i = 1\cdots n$, $\theta_i(k) \in R^{4m\times 1}$, $\Theta_k = [\theta_1(k), \cdots \theta_n(k)]$, $y(k) = [y_1(k), \cdots y_n(k)]$, $\zeta = [\zeta_1(k) \cdots \zeta_n(k)]^T$.

$$\begin{array}{l} y_i(k) = x_i(k+1) - a_ix_i(k) \\ +V_1^0\sigma\left[W_1^0x_i(k)\right] + U(k)V_2^0\phi\left[W_2^0x_i(k)\right] \end{array}$$

Because $\omega_i(k)$ is a random noise, we assumed it is independent from $\theta_i(k)$, so $\zeta_{i,k}$ and $\omega_i(k)$ are un-correlated, $E\left\{\omega_i(k)\zeta_{i,k}^T\right\} = 0$.

Remark 1: The state vector $\theta_i(k)$ is assumed to be an unknown constant with a small random walk component to allow for adaptive estimation. This random walk serves the role of an artificial process noise $\omega_i(k)$. We assume it satisfies

$$E\left\{\omega_i(k)\right\} = 0, \quad E\left\{\omega_i(k)\omega_i(k)^T\right\} = R_1 \tag{9}$$

R_1 can be chosen as αI, where α is small and positive. In fact, if we do not have a change during the interval of time, R_1 tends to zero, it becomes the least square algorithm. The plant is not expected to change at all during the time of interest, the covariance of 'process noise' $\zeta_{i,k}$ can be assumed as

$$E\left\{\zeta_{i,k}\zeta_{i,k}^T\right\} = R_2$$

These ideas can be also found in [2] and [3], but [3] applied Kalman filter to feedforward neural networks. In this paper, we use Kalman filter for recurrent neural networks.

We use n Kalman filters to estimate the weights. For ith subsystem the observer is

$$\left\{\begin{array}{l} \widehat{\theta}_i(k+1) = \widehat{\theta}_i(k) + K\left[y_i(k) - \widehat{y}_i(k)\right] \\ \widehat{y}_i(k) = B_k^T\widehat{\theta}_i(k) \end{array}\right. \tag{10}$$

where $\widehat{y}_i(k)$ is estimated state of $y_i(k)$, K is observer gain. The state estimation error is defined as

$$\widetilde{\theta}_i(k) = \theta_i(k) - \widehat{\theta}_i(k) \tag{11}$$

Substitute (8) and (10) into (11)

$$\widetilde{\theta}_i(k+1) = \left[I - KB_k^T\right]\widetilde{\theta}_i(k) + \omega_i(k) - K\zeta_{i,k} \tag{12}$$

We define a performance index $P_k \in R^{4m\times 4m}$ which uses the covariance

$$P_k = cov(\widetilde{\theta}_i) = E\left\{\left[\widetilde{\theta}_i - E\left(\widetilde{\theta}_i\right)\right]\left[\widetilde{\theta}_i - E\left(\widetilde{\theta}_i\right)\right]^T\right\}$$

By (8) and $E\left[\omega_i(k)\right] = 0$, we have $E\left[\theta_i(k+1)\right] = E\left[\theta_i(k)\right] + E\left[\omega_i(k)\right] = 0$, $E\left[\theta_i(k+1)\right] = E\left[\theta_i(k)\right] = E\left[\theta_i(1)\right] = 0$. Because $E(\widehat{\theta}_i(k)) = 0$, $E(\widetilde{\theta}_i(k)) = E(\theta_i(k)) - E(\widehat{\theta}_i(k)) = 0$. So

$$P_k = E\left\{\widetilde{\theta}_i(k)\widetilde{\theta}_i^T(k)\right\} \tag{13}$$

Substitute (12) into (13)

$$
\begin{aligned}
&P_{k+1} = E\left\{\widetilde{\theta}_i(k+1)\,\widetilde{\theta}_i^T(k+1)\right\} \\
&= E\left\{ \begin{array}{c} \left[\left(I - KB_k^T\right)\widetilde{\theta}_i(k) + \omega_i(k) - K\zeta_{i,k}\right] \\ \times \left[\left(I - KB_k^T\right)\widetilde{\theta}_i(k) + \omega_i(k) - K\zeta_{i,k}\right]^T \end{array} \right\} \\
&= \left(I - KB_k^T\right) E\left\{\widetilde{\theta}_i(k)\,\widetilde{\theta}_i^T(k)\right\}\left(I - KB_k^T\right)^T \\
&+ E\left\{\omega_i(k)\omega_i(k)^T\right\} + KE\left\{\zeta_{i,k}\zeta_{i,k}^T\right\}K^T \\
&+ \text{cross-terms}(\widetilde{\theta}_i, \omega_i(k), \zeta_{i,k})
\end{aligned}
$$

Because $\widetilde{\theta}_i(k)$, $\omega_i(k)$ and $\zeta_{i,k}$ are independent, the cross terms of $(\widetilde{\theta}_i, \omega_i(k), \zeta_{i,k})$ are zero. So P_{k+1} is

$$
\begin{aligned}
&P_{k+1} = P_k + R_1 - P_kB_k\left(R_2 + B_k^TP_kB_k\right)^{-1}B_k^TP_k \\
&+ \left[K - P_kB_k\left(R_2 + B_k^TP_kB_k\right)^{-1}\right]\left(R_2 + B_k^TP_kB_k\right) \\
&\times \left[K - P_kB_k\left(R_2 + B_k^TP_kB_k\right)^{-1}\right]^T
\end{aligned}
$$

The last term $[\cdot]\left(R_2 + B_k^TP_kB_k\right)[\cdot]^T \geq 0$, and $P_k \geq 0$, $R_1 > 0$. If we want to minimize P_{k+1}, we have to make the last term zero, *i.e.*, $K - P_kB_k\left(R_2 + B_k^TP_kB_k\right)^{-1} = 0$. So the observer gain is selected as

$$
K = P_kB_k\left(R_2 + B_k^TP_kB_k\right)^{-1} \tag{14}
$$

By (14), P_{k+1} becomes

$$
\begin{aligned}
&P_{k+1} = P_k + R_1 - P_kB_k\left(R_2 + B_k^TP_kB_k\right)^{-1}B_k^TP_k \\
&= R_1 + \left[I - KB_k^T\right]P_k
\end{aligned} \tag{15}
$$

(10), (14) and (15) are the extended Kalman filter for the training of the neural networks' weights $\widehat{\theta}_i(k)$

$$
\begin{aligned}
&\widehat{\theta}_i(k+1) = \widehat{\theta}_i(k) - K_ke_i(k) \\
&\widehat{y}_i(k) = B_k^T\widehat{\theta}_i(k) \\
&P_{k+1} = R_1 + \left[I - K_kB_k^T\right]P_k \\
&K_k = P_kB_k\left(R_2 + B_k^TP_kB_k\right)^{-1}
\end{aligned} \tag{16}
$$

where $e_i(k) = \widehat{y}_i(k) - y_i(k)$.

Remark 2: The Kalman error $e_i(k)$ is not the same as the identification error $\epsilon_i(k) = \widehat{x}_i(k) - x_i(k)$, but they are minimized at the same time. From (2) and (3), we have

$$
\epsilon_i(k+1) = a_i\epsilon_i(k) + e_i(k) \tag{17}
$$

By the relation $\epsilon_i(2) = a_i\epsilon_i(1) + e_i(1) = a_i^2\epsilon_i(0) + a_ie_i(0) + e_i(1)$, $\epsilon_i(k) = a_i^k\epsilon_i(0) + \sum_{j=0}^{k-1} a_i^{k-j-1}e_i(j)$, and $|a_i| < 1$

$$
|\epsilon_i(k)| \leq |\epsilon_i(0)| + \sum_{j=0}^{k-1}|e_i(j)|
$$

Since $\epsilon_i(0)$ is a constant, the minimization of the Kalman error $e_i(j)$ means the upper bound of the identification error $\epsilon_i(k)$ is minimized.

Remark 3: The observer (10) is for each subsystem. This method can decrease computational burden when we estimate the weights of the recurrent neural network, see [7] and [14]. By (5) and (6) we know the data matrix B_k depends on the parameters $V_{1,k}^T$ and $V_{2,k}^T$, this will not effect parameter updating algorithm (16), because the unknown parameter $\widehat{\theta}_i(k+1)$ is calculated by the known parameters $\widehat{\theta}_i(k)$ and data B_k . For each element of Θ_k^T and B_k in (16), we have

$$
\begin{aligned}
&V_{1,k+1} = V_{1,k} - \tfrac{P_k}{R_2 + B_k^TP_kB_k}\sigma\left[W_{1,k}x(k)\right]e^T(k) \\
&W_{1,k+1} = W_{1,k} - \tfrac{P_k}{R_2 + B_k^TP_kB_k}\sigma'\left[W_{1,k}x(k)\right]V_{1,k}^Tx(k)e^T(k)
\end{aligned} \tag{18}
$$

It has the same form as the backpropagation [7], but the learning rate is not positive constant, it is a matrix $\frac{P_k}{R_2 + B_k^TP_kB_k}$ which changes though the time. That is main reason why Kalman filter training has a faster convergence speed. As [14] point out, $R_2 > 0$ can increase convergence speed.

III. Stability analysis

In this section we will use Lyapunov stability theory to prove that the Kalman filter algorithm (16) with dead zone is stable for system identification.

Lemma 1: If the neural network $\Theta^*(V_1^*, V_2^*, W_1^*, W_2^*))$ is the optimal model of the system as in (3) for some data $(B_k, y_i(k))$, there exist at least one $\Theta_k(V_{1,k}, V_{2,k}, W_{1,k}, W_{2,k})$ such the modelling error $\zeta(k)$ is also minimized, and Θ^* and Θ_k satisfy

$$
\begin{aligned}
B_k &= TB^* \\
\Theta_k &= \Theta^*T^{-1}
\end{aligned} \tag{19}
$$

where $T \in R^{4m\times 4m}$ is a linear transformation, it is an invertible matrix.

Proof: From (7) $y(k) = \Theta_kB_k + \xi(k)$, we know

$$
\Theta^*B^* = y(k) - \xi^*
$$

Choose an invertible matrix T, compute Θ_kB_k according to (19)

$$
\Theta_kB_k = \left(\Theta^*T^{-1}\right)\left(TB_k^*\right) = \Theta^*B^*
$$

So

$$
y(k) - \xi(k) = y(k) - \xi^* \tag{20}
$$

that is, the function reconstruction error remains unchanged. ■

Because of there exists $\Theta_k(V_{1,k}, V_{2,k}, W_{1,k}, W_{2,k})$ such that the model error is minimized, we can design an algorithm to find an acceptable solution. This Lemma gives conditions for an exact match of the optimal error vector. However, for all practical purposes, it is sufficient that neural networks Θ_kB_k generate error

$$
\xi^{*T}\xi^* = \zeta^T(k)\,\zeta(k)
$$

that is, the two errors are similar in the sense of the norm but they are not required to match.

By (8), (10) becomes

$$
e_i(k) = B_k^T\widetilde{\theta}_i(k) + \zeta_{i,k} \tag{21}
$$

where $\widetilde{\theta}_i(k) = \theta_i(k) - \widehat{\theta}_i(k)$, $i = 1 \cdots n$. We modify the extended Kalman filter (16) as dead-zone Kalman filter

$$\begin{aligned} \widehat{\theta}_i(k+1) &= \widehat{\theta}_i(k) - \tfrac{1}{R_k} P_k B_k s_k \\ \widehat{y}_i(k) &= B_k^T \widehat{\theta}_i(k) \end{aligned} \tag{22}$$

where $R_k = R_2 + B_k^T P_k B_k$, $s_k = \begin{cases} e_i(k) & e_i^2(k) \geq 3\overline{\zeta}_i^2 \text{ AND } \frac{1}{R_k} P_k B_k B_k^T P_k \geq R_1 \\ 0 & e_i^2(k) < 3\overline{\zeta}_i^2 \text{ OR } \frac{1}{R_k} P_k B_k B_k^T P_k < R_1 \end{cases}$, $\overline{\zeta}_i$ is the upper bound of the uncertainty $\zeta_{i,k}$, $\left|\zeta_{i,k}\right| < \overline{\zeta}_i$.

Theorem 1: The dead-zone Kalman filter algorithm (22) can assure the identification error $\epsilon_i(k)$ and the weights of the neural networks bounded. For any $T \in (0, \infty)$ the output error $e_i(k)$ converges to the residual set

$$D_{e_i} = \left\{ e_i(k) \mid e_i^2(k) \leq 3\overline{\zeta}_i^2 \right\} \tag{23}$$

Proof: We define the following Lyapunov function

$$V(k) = \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) \tag{24}$$

Let us define $\overline{P}_{k+1} = \left[I - K_k B_k^T\right] P_k$, because $P_{k+1} = \overline{P}_{k+1} + R_1$, $R_1 > 0$, so $x^T P_{k+1} x > x^T \overline{P}_{k+1} x > 0$ and $x^T \overline{P}_{k+1}^{-1} x \geq x^T P_{k+1}^{-1} x > 0$. Apply to (24)

$$\begin{aligned} \Delta V(k) &= \widetilde{\theta}_i^T(k+1) P_{k+1}^{-1} \widetilde{\theta}_i(k+1) - \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) \\ &\leq \widetilde{\theta}_i^T(k+1) \overline{P}_{k+1}^{-1} \widetilde{\theta}_i(k+1) - \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) \end{aligned} \tag{25}$$

First, we consider the case $e_i^2(k) \geq 3\overline{\zeta}_i^2$ and $\frac{1}{R_k} P_k B_k B_k^T P_k \geq R_1$. So $s_k = e_i(k)$. Substituting (21) into (22) gives

$$\widetilde{\theta}_i(k+1) = \widetilde{\theta}_i(k) - \frac{1}{R_k} P_k B_k B_k^T \widetilde{\theta}_i(k) - \frac{1}{R_k} P_k B_k \zeta_i(k) \tag{26}$$

From (22), (21) and (26), we have

$$\begin{aligned} \widetilde{\theta}_i(k+1) &= \widetilde{\theta}_i(k) - \tfrac{1}{R_k} P_k B_k e_i(k) \\ \widetilde{\theta}_i(k+1) &= \left[I - \tfrac{1}{R_k} P_k B_k B_k^T\right] \widetilde{\theta}_i(k) - \tfrac{1}{R_k} P_k B_k \zeta_{i,k} \end{aligned} \tag{27}$$

From (15) we know $P_{k+1} = \overline{P}_{k+1} + R_1$. So $\overline{P}_{k+1} = P_k - \frac{1}{R_k} P_k B_k B_k^T P_k$, $\overline{P}_{k+1} P_k^{-1} = I - \frac{1}{R_k} P_k B_k B_k^T$, (27) becomes

$$\overline{P}_{k+1}^{-1} \widetilde{\theta}_i(k+1) = P_k^{-1} \widetilde{\theta}_i(k) - \frac{1}{R_k} \overline{P}_{k+1}^{-1} P_k B_k \zeta_{i,k} \tag{28}$$

substituting (28) into (25) gives

$$\begin{aligned} \Delta V(k) \leq & \left[\widetilde{\theta}_i(k+1) - \widetilde{\theta}_i(k)\right]^T P_k^{-1} \widetilde{\theta}_i(k) \\ & - \tfrac{1}{R_k} \widetilde{\theta}_i^T(k+1) \overline{P}_{k+1}^{-1} P_k B_k \zeta_{i,k} \end{aligned} \tag{29}$$

here $\left[\widetilde{\theta}_i(k+1) - \widetilde{\theta}_i(k)\right]^T$ is substituted by (26), (29) is

$$\begin{aligned} \Delta V(k) \leq & -\tfrac{1}{R_k} \widetilde{\theta}_i^T(k) B_k B_k^T \widetilde{\theta}_i(k) - \tfrac{1}{R_k} \zeta_{i,k} B_k^T \widetilde{\theta}_i(k) \\ & - \tfrac{1}{R_k} \widetilde{\theta}_i^T(k+1) \overline{P}_{k+1}^{-1} P_k B_k \zeta_{i,k} \end{aligned} \tag{30}$$

The last term of (30) is

$$\begin{aligned} & \tfrac{\zeta_{i,k}}{R_k} \widetilde{\theta}_i^T(k+1) \overline{P}_{k+1}^{-1} P_k B_k = \tfrac{\zeta_{i,k}}{R_k} B_k^T P_k \overline{P}_{k+1}^{-1} \widetilde{\theta}_i(k+1) \\ &= \tfrac{\zeta_{i,k}}{R_k} B_k^T P_k \left[P_k^{-1} \widetilde{\theta}_i(k) - \tfrac{1}{R_k} \overline{P}_{k+1}^{-1} P_k B_k \zeta_{i,k}\right] \\ &= \tfrac{\zeta_{i,k}}{R_k} B_k^T \widetilde{\theta}_i(k) - \tfrac{\zeta_{i,k}^2}{R_k^2} B_k^T P_k \overline{P}_{k+1}^{-1} P_k B_k \end{aligned}$$

Now we apply the matrix inversion lemma to $\overline{P}_{k+1} = P_k - \frac{1}{R_k} P_k B_k B_k^T P_k$

$$(A + BCD)^{-1} = A^{-1} - A^{-1} B \left(DA^{-1} B + C^{-1}\right)^{-1} DA^{-1} \tag{31}$$

with $A^{-1} = P_k$, $B = B_k$, $C^{-1} = R_2$, $D = B_k^T$, $R_k = R_2 + B_k^T P_k B_k$, so $\overline{P}_{k+1}^{-1} = P_k^{-1} + \frac{1}{R_2} B_k B_k^T$. (30) becomes

$$\begin{aligned} & \Delta V(k) \leq -\tfrac{1}{R_k} \widetilde{\theta}_i^T(k) B_k B_k^T \widetilde{\theta}_i(k) - \tfrac{1}{R_k} \zeta_{i,k} B_k^T \widetilde{\theta}_i(k) \\ & - \tfrac{\zeta_{i,k}}{R_k} B_k^T \widetilde{\theta}_i(k) + \tfrac{\zeta_{i,k}^2}{R_k^2} B_k^T P_k \overline{P}_{k+1}^{-1} P_k B_k \\ &= -\tfrac{1}{R_k} \left[\widetilde{\theta}_i^T(k) B_k B_k^T \widetilde{\theta}_i(k) + 2\zeta_i(k) B_k^T \widetilde{\theta}_i(k) + \zeta_i^2(k)\right] \\ & + \left(\tfrac{1}{R_k} + \tfrac{1}{R_k^2} B_k^T P_k \left(P_k^{-1} + \tfrac{1}{R_2} B_k B_k^T\right) P_k B_k\right) \zeta_{i,k}^2 \\ &= -\tfrac{1}{R_k} \left[B_k^T \widetilde{\theta}_i(k) + \zeta_{i,k}\right]^2 \\ & + \tfrac{1}{R_k} \left(1 + \tfrac{1}{R_k} B_k^T P_k B_k + \tfrac{1}{R_k^2} \left(B_k^T P_k B_k\right)^2\right) \zeta_{i,k}^2 \end{aligned} \tag{32}$$

Using (21), (32) tells us

$$\begin{aligned} & \Delta V(k) \leq \tfrac{1}{R_k} [-e_i^2(k) \\ & + \left(1 + \tfrac{B_k^T P_k B_k}{R_2 + B_k^T P_k B_k} + \tfrac{\left(B_k^T P_k B_k\right)^2}{\left(R_2 + B_k^T P_k B_k\right)^2}\right) \zeta_{i,k}^2] \\ & \Delta V(k) \leq \tfrac{1}{R_k} \left[-e_i^2(k) + (1 + 1 + 1) \zeta_{i,k}^2\right] \end{aligned}$$

So

$$\Delta V(k) \leq \frac{1}{R_k} \left[-e_i^2(k) + 3\overline{\zeta}_i^2\right] \tag{33}$$

where $\left|\zeta_{i,k}\right| < \overline{\zeta}_i$. Because $e_i^2(k) \geq 3\overline{\zeta}_i^2$ and $R_k > 0$, so $\Delta V(k) \leq 0$. Also

$$\widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k) \leq \widetilde{\theta}_i^T(1) P_1^{-1} \widetilde{\theta}_i(1) \leq \lambda_{\max}\left(P_1^{-1}\right) \left\|\widetilde{\theta}_i(1)\right\|^2$$

From (16) we know $P_{k+1} = P_k - \frac{1}{R_k} P_k B_k B_k^T P_k + R_1$,because $\frac{1}{R_k} P_k B_k B_k^T P_k \geq R_1$, $P_{k+1}^{-1} \geq P_k^{-1}$

$$\lambda_{\min}\left(P_1^{-1}\right) \left\|\widetilde{\theta}_i(k)\right\|^2 \leq \widetilde{\theta}_i^T(k) P_1^{-1} \widetilde{\theta}_i(k) \leq \widetilde{\theta}_i^T(k) P_k^{-1} \widetilde{\theta}_i(k)$$

we choose $\lambda_{\min}\left(P_1^{-1}\right) \neq 0$, so

$$\left\|\widetilde{\theta}_i(k)\right\|^2 \leq \frac{\lambda_{\max}\left(P_1^{-1}\right)}{\lambda_{\min}\left(P_1^{-1}\right)} \left\|\widetilde{\theta}_i(1)\right\|^2$$

$\widehat{\theta}_i(k)$ is bounded. On the other side, if $e_i^2(k) < 3\overline{\zeta}_i^2$ or $\frac{1}{R_k} P_k B_k B_k^T P_k < R_1$ $s_k = 0$, $\widehat{\theta}_i(k+1) = \widehat{\theta}_i(k)$, $\widetilde{\theta}_i(k)$ is bounded. For the all case, the weights $\widehat{\theta}_i(k)$ are bounded. Form (21) we know the boundness of B_k^T, $\widetilde{\theta}_i(k)$ and $\zeta_{i,k}$ mean $e_i(k)$ is bounded, P_k is also bounded by (13). From (1) and (2) the identification error $\epsilon_i(k)$ is bounded.

Because $e_i^2(k)$ is bounded, the total time during which $e_i^2(k) \geq 3\overline{\zeta}_i^2$ is finite. Let T_j denotes the time interval during which $e_i^2(k) \geq 3\overline{\zeta}_i^2$. (a) If only finite times that $e_i^2(k)$ stay outside the circle of radius $3\overline{\zeta}_i^2$ (and then reenter), $e_i^2(k)$ will eventually stay inside of this circle. (b) If $e_i^2(k)$ leave the circle infinite times, since the total time $e_i^2(k)$ leave the circle is finite,

$$\sum_{j=1}^{\infty} T_j < \infty, \quad \lim_{j \to \infty} T_j = 0 \tag{34}$$

So $e_i^2(k)$ is bounded via an invariant set argument. Let $e_{i,j}^2(k)$ denote the largest tracking error during the T_j interval. Then (34) and bounded $e_{i,j}^2(k)$ imply that

$$\lim_{k\to\infty}\left[e_{i,j}^2(k)-3\overline{\zeta}_i^2\right]=0$$

So $e_{i,j}^2(k)$ will convergence to $3\overline{\zeta}_i^2$, (23) is achieved. ■

Remark 4: The dead-zone s_k depends on the two system noise in (8), $\omega_i(k)$ (corresponds to R_1) and $\zeta_{i,k}$ (corresponds to $\overline{\zeta}_i^2$). If the weights converge to some constants (not optimal weights), (9) can be expressed as

$$E\left\{\omega_i(k)\right\}=0,\quad E\left\{\omega_i(k)\omega_i(k)^T\right\}=R_1=\frac{1}{k}\overline{R}_1$$

We can change the dead-zone as $s_k=\begin{cases} e_i(k) & e_i^2(k)\geq 3\overline{\zeta}_i^2 \\ 0 & e_i^2(k)<3\overline{\zeta}_i^2 \end{cases}$ which only depends on the $\zeta_{i,k}$. In the proof of Theorem 1, after (33) we have

$$\begin{aligned}\widetilde{\theta}_i^T(k+1)P_{k+1}^{-1}\widetilde{\theta}_i(k+1)&\leq\widetilde{\theta}_i^T(1)P_1^{-1}\widetilde{\theta}_i(1)\\&\leq\lambda_{\max}\left(P_1^{-1}\right)\left\|\widetilde{\theta}_i(1)\right\|^2\end{aligned}$$

From (16) we know

$$\begin{aligned}P_{k+1}&=P_k+R_1-\tfrac{1}{R_k}P_kB_kB_k^TP_k\\&\leq P_k+R_1\leq P_1+kR_1=P_1+\overline{R}_1\end{aligned}$$

So $P_{k+1}^{-1}\geq\left(P_1+\overline{R}_1\right)^{-1}$,

$$\begin{aligned}&\widetilde{\theta}_i^T(k+1)P_{k+1}^{-1}\widetilde{\theta}_i(k+1)\geq\widetilde{\theta}_i^T(k+1)\left(P_1+\overline{R}_1\right)^{-1}\widetilde{\theta}_i(k+1)\\&\geq\lambda_{\min}\left(\left(P_1+\overline{R}_1\right)^{-1}\right)\left\|\widetilde{\theta}_i(k+1)\right\|^2\end{aligned}$$

we choose P_1 such that $\lambda_{\min}\left(\left(P_1+\overline{R}_1\right)^{-1}\right)\neq 0$, so

$$\left\|\widetilde{\theta}_i(k+1)\right\|^2\leq\frac{\lambda_{\max}\left(P_1^{-1}\right)}{\lambda_{\min}\left(\left(P_1+\overline{R}_1\right)^{-1}\right)}\left\|\widetilde{\theta}_i(1)\right\|^2$$

The same results as Theorem 1 can be reached.

IV. Simulations

We will use the nonlinear system which proposed [11] to illustrate the behavior of the dead zone Kalman filter algorithm proposed in this paper

$$x_1(k+1)=\frac{x_1(k)x_2(k)x_3(k)}{1+x_1(k)^2+x_2(k)^2+x_3(k)^2}+2u(k)$$

where $x_2(k+1)=x_1(k)$, $x_3(k+1)=x_2(k)$, and $x_4(k)=u(k)$. The unknown nonlinear system has the standard form (1), we use the recurrent neural network (series-parallel model) given in (2) to identify it, where $\widehat{x}(k)\in R^4$, $A\in R^{4\times 4}$ is a stable diagonal matrix which is specified as $A=diag(0.1)$. In this paper, in order to exam the effectiveness of the Kalman filter training, we use 1 node in the hidden layer. The weights in output layer are $V_{1,k}\in R^{4\times 1}$, the weights in hidden layer are $W_{1,k}\in R^{1\times 4}$, $\sigma=[\sigma_1]^T$, $\phi(\cdot)$ is an element. The elements of the initial weights $W_{1,0}$ and $V_{1,0}$ are chosen in random number between $(0,1)$. The input is $u(t)=0.3\sin(3\pi kTs)+0.1\sin(4\pi kTs)+0.6\sin(\pi kTs)$.

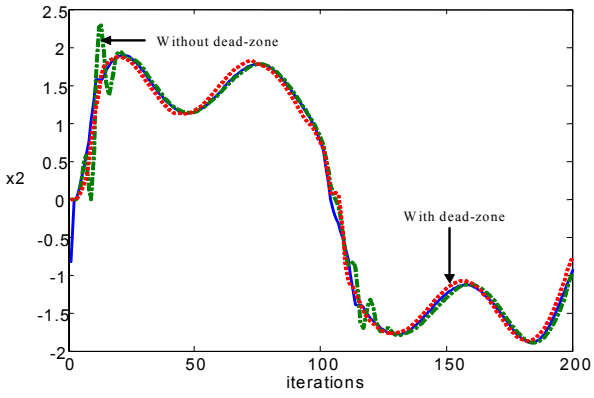

Fig. 1. Comparison of Kalman filter with dead zone

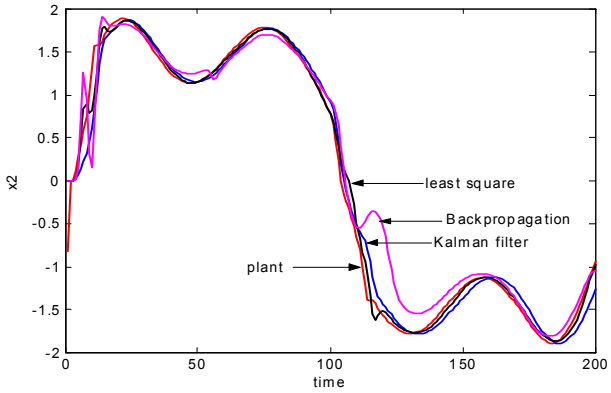

Fig. 2. Identification for x_2

We select $P=diag(100)\in R^{4\times 4}$, $R_1=diag(0.1)\in R^{4\times 4}$, $R_2=1.2$. The dead zone is selected as $\overline{\zeta}_i=0.1$. We compare the normal Kalman filter (16) with dead-zone Kalman filter (22). The identification results for $x_2(k)$ are shown in Fig.1. We can see that dead-zone Kalman filter has more robustness than normal one, but has bigger steady-state error.

Now we use the dead zone Kalman filter with back-propagation algorithm [7] with learning rate 0.02, and also recursive least square method [5] to train the neural network given in (2). we define the mean squared error for finite time as $J(N)=\frac{1}{2N}\sum_{k=1}^{N}e^2(k)$. The identification results for x_2 and $J(N)$ are shown in Fig.2and Fig. 3. We find that Kalman filter has best convergence property.

V. Conclusions

A novel method of neural identification with Kalman filter training. The normal Kalman filter is modified with dead-zone such that it is not sensitive to system noise. Both hidden layers and output layers of the state-space recurrent neural networks can be updated. Lyapunov technique is used

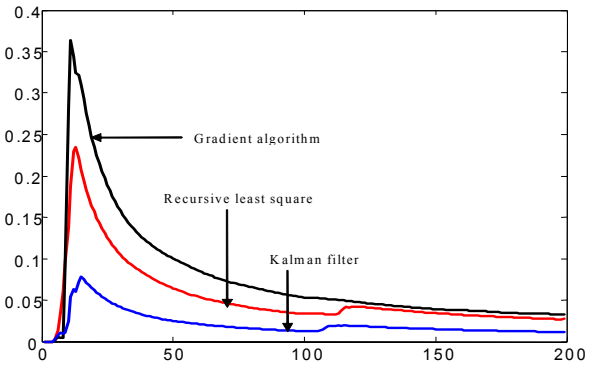

Fig. 3. Errors comparison

to prove that the Kalman filter training is stable. From a dynamic systems point of view, such training can be useful for all neural network applications requiring real-time updating of the weights.

REFERENCES

[1] A.Alessandri, MCuneo, S.Pagnan, M.Sanguineti, On the convergence of EKF-based parameters optimization for neural networks, *42nd IEEE Conference on Decision and Control,* 6181 - 6186, 2003

[2] R.K. Boel, M.R.James and I.R.Petersen, Robustness and Risk-Sensitive Filtering, *IEEE Trans. Automatic Control*, Vol.47, No.3, 451-462, 2002.

[3] F.N.Chowdhury, A new approach to real-time training of dynamic neural networks, *International Journal of Adaptive Control and Signal Processing*, 509-521, 2003.

[4] Robust filtering for discrete-time systems with bounded noise and parametric uncertainty, *IEEE Trans. Automatic Control,* Vol.46 , No.7, 1084 - 1089, 2001

[5] G. C. Goodwin and K. Sang Sin, *Adaptive filtering prediction and control*, Prentice Hall, Englewood Cliffis, NJ07632, 1984.

[6] L.Guo, Estimating time-varying parameters by the Kalman filter based algorithm: stability and convergence, *IEEE Trans. Automatic Control*, 141-147, 1990.

[7] S.Haykin, *Neural Networks- A Comprehensive Foundation*, Macmillan College Publ. Co., New York, 1994.

[8] E.B. Kosmatopoulos, M.M. Poly carpou, M.A. Christodoulou and P.A. Ioannou, High-Order Neural Network Structures for Identification of Dynamical Systems, *IEEE Transactions on Neural Networks*, Vol.6, No.2, 422-431, 1995.

[9] Y. Iiguni, H. Sakai, Member, IEEE and Higekatsu Tokumaru, A Real-Time Learning Algorithm for a Multilayered Neural Network Based on the Extended Kalman Filter, *IEEE Transactions on Signal Processing*, Vol.40, No.4, 959-966, 1992.

[10] S. Kollias and D. Anastassiou, An adaptive least square algorithm for efficient training of artificial neural networks, *IEEE Transactions on Circuits Systems*, Vol.36, No.5, 1092-1101, 1989.

[11] L.Jin, P. N. Nikiforunk, M.M. Gupta, Adaptive Model Reference Control of Discrete-Time Nonlinear Systems Using Neural Networks, *Control-Theory and Advanced Technolnogy*, Vol.10, No.4, Part.3, 1373-1399, September 1999.

[12] K.Nishiyama, K.Suzuki, H_∞-learning of layered neural networks, Neural Networks, *IEEE Trans.Neural Networks*, Vol.12, No. 6, 1265 - 1277, 2001

[13] A. G. Parlos, S. K. Menon and A. F. Atiya, An Algorithm Approach to Adaptive State Filtering Using Recurrent Neural Network, *IEEE Transactions on Neural Networks*, Vol.12, No.6, 1411-1432, 2001.

[14] G. V. Puskorius and L. A. Feldkamp, Neurocontrol of Nonlinear Dynamical Systems with Kalman Filter Trained Recurrent Networks, *IEEE Transactions on Neural Networks*, Vol.5, No.2, 279-297, March 1994.

[15] K. Reif and R. Unbehauen, The Extended Kalman Filter as an Exponential Observer for Nonlinear Systems, *IEEE Transactions on Signal Processing*, Vol.47, No.8, 2324-2328, August 1999.

[16] K.Reif, S.Gunther, E.Yaz, R.Unbehauen, Stochastic stability of the continuous-time extended Kalman filter, *IEE Proceedings-Control Theory and Applications*, 1350-2379, 2000

[17] D. W. Ruck, S. K. Rogers, M. Kabrisky, P. S. Maybeck and M. E. Oxley, Comparative Analysis of Backpropagation and the Extended Kalman Filter for Training Multilayer Perceptrons, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.14, No.6, 686-691, June 1992.

[18] S. Singhal and L. Wu, Training multilayer perceptrons with the extended Kalman algorithm, *Advances in Neural inform. Processing Syst. I*, 133-140, 1989.

[19] J.Sum, C.L., G. H. Young and W. Kan, On the Kalman Filtering Method in Neural-Network Training and Pruning, *IEEE Transactions on Neural Networks*, Vol.10, No.1, 161-166, 1999.

[20] W.Yu, Nonlinear system identification using discrete-time recurrent neural networks with stable learning algorithms, *Information Sciences*, Vol.158, No.1, 131-147, 2004.

[21] Wen Yu, Passivity analysis for dynamic multilayer neuro identifier, *IEEE Transactions on Circuits and Systems*, Part I, Vol.50, No.1, 173-178, 2003.

[22] Wen Yu, Xiaoou Li, Discrete-time neuro identification without robust modification, *IEE Proceedings - Control Theory and Applications*, Vol.150, No.3, 311-316, 2003.

[23] Wen Yu, Xiaoou Li, Fuzzy identification using fuzzy neural networks with stable learning algorithms, I*EEE Transactions on Fuzzy Systems*, Vol.12, No.3, 411-420, 2004.

[24] Wen Yu, América Morales, Gasoline blending system modeling via static and dynamic neural networks, *International Journal of Modelling and Simulation*, Vol.24, No.3, 151-160, 2004