A Stable Total Least Square Adaptive Algorithm for the Nonlinear Volterra Filter

Xiangyu Kong
Electronic & Information Engr
Xi'an Jiaotong University
Xi'an, Shaanxi
710049, P.R. China
xiangyukong01@163.com

Chongzhao Han
Electronic & Information Engr
Xi'an Jiaotong University
Xi'an, Shaanxi
710049, P.R. China
czhan@mail.xjtu.edu.cn

Ruixuan Wei
School of Engr, Air Force
Engineering University
Xi'an, Shaanxi
710038, P.R. China
rxwei@mail.xjtu.edu.cn

Abstract - *Aiming at the nonlinear filtering problem that exists when the input and output observation data are both corrupted by noises, a stable total least square adaptive algorithm for the nonlinear Volterra filter is proposed in this paper. Taking the minimum Rayleigh Quotient of the augmented Volterra weight vectors and a constraint to the last element of the augmented Volterra weight vectors via a Lagrange multiplier as the overall cost function, the recursive formula of the Volterra weight vector is derived. The stability property of the algorithm is analyzed and the step-size parameter to guarantee the stability of the algorithm is educed. The proposed algorithm is implemented without requiring normalization. The simulation results have shown that, in addition to the fine convergence, the robust anti-noise performance and the stable convergence precision of the proposed algorithm are remarkably higher than other total least square algorithms for the nonlinear system.*

Keywords: total least square, Volterra filter, the Rayleigh Quotient, adaptive algorithm.

1 Introduction[1]

It is well known that nonlinear adaptive filter has played an important role in tracking fusion system, communication system, and control system etc. Because Volterra series can completely describe the input and output transfer characteristic of a large type of nonlinear system, the Volterra adaptive filter has been widely studied in recent years. The traditional algorithm for Volterra adaptive filter is developed by thinking Volterra series model as a pseudo-linear operator on the whole and directing analogy with the linear filter algorithm. Usually, the input and output data sampled from a practical system are all corrupted by noises, it is necessary to use the total least squares approach for the nonlinear filter. Though there has been much reseach regarding the TLS problem and its solution[1], iterative solution forms suitable for use in adaptive filters have only recently receievd much attention[2-6]. Much of this work appears under the guise of minor component analysis (MCA), with many of the algorithms that have been proposed by researchers appearing in the neural network litearature[7-10]. Despite the large number of algorithms proposed to date, there are few algorithms that posses the following three desirable properties: (1) proven stable adaptation behavior, (2) computational simplicity in the sense that normalization operations are not required in the implementation, and (3) fine robust anti-noise performance and convergence performance.

Of the TLS or MCA algorithms, the MCA EXIN algorithm is the best in terms of stability, speed, and accuracy [10], but the algorithm is applied to total least squares problem, the normalized operation is required in the each updating like other MCA algorithms, and the MCA EXIN algorithm like other MCA algorithms does not give the bound of the step-size parameter to guarantee the stability of the algorithm. The stable simplified gradient algorithm for total least squares filtering[6] is proven stable and has computational simplicity in the sense that normalization operations are not required in the implementation, but its simplicity can cause the robust anti-noise performance and the convergence performance to reduce and can cause the problem of no finite time divergence, and its learning rate must be selected very small to keep the algorithm convergent. And, the two algorithms are for linear filters, and are not proved to be efficient for the nonlinear filters.

By referring to the above algorithms and combining the principle of the nonlinear Volterra series, taking the minimum Rayleigh Quotient of the augmented Volterra weight vectors and a constraint to the last element of the augmented Volterra weight vectors via a LaGrange multiplier as the overall cost function, by the steepest descent principle, a stable total least square adaptive algorithm for nonlinear Volterra filter is proposed in this paper, the algorithm does not require normalization, and has fine robust anti-noise performance and convergence performance for the nonlinear filter.

The paper is organized as follows: the section 2 is devoted for Volterra total least squares adaptive filter based on MCA EXIN algorithm. The stable Volterra

[1] The paper is sponsored by the national key fundamental research & development programs of P.R.China, and natural science foundation of P.R.China.

total least squares algorithm is derived in section 3, meanwhile, its stable performance is analyzed and the step-size parameter to guarantee the stability of the algorithm is educed. The simulation results are shown in section 4. Finally, the section 5 is the conclusion.

2 Volterra TLS Adaptive Filter Based on MCA EXIN Algorithm

2.1 Volterra Series

For SISO time-invariant casual nonlinear system, its output $y(t)$ can be written as the Volterra series in time domain as following [14]:

$$y(t) = \sum_{n=1}^{\infty} y_n(t), \tag{1}$$

where $y_n(t)=\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} h_n(\tau_1,\cdots,\tau_n)\prod_{i=1}^{n} u(t-\tau_i)d\tau_i$, $n\in \boldsymbol{N}$, and $\boldsymbol{N}$ is the natural number set. $y_n(t)$ is the output of the n*th*-order Volterra series, and $h_n(\tau_1,\tau_2\cdots\tau_n)$ is the n*th*-order Volterra time-domain kernel or the n*th*-order impulse response function.

In practical filter problem, the discrete truncated form of Volterra series is used as following:

$$\begin{aligned} y(k) &= \sum_{n=1}^{N} y_n(k) \\ &= \sum_{n=1}^{N}\sum_{m_1=0}^{M-1}\cdots\sum_{m_n=0}^{M-1} h_n(m_1,m_2,\cdots,m_n)\prod_{i=1}^{n} u(k-m_i), \end{aligned} \tag{2}$$

where M is a memory length of the Volterra kernel, and N is the maximum order of the truncated Volterra series.

2.2 Volterra TLS Adaptive Filter Based on MCA EXIN Algorithm

Consider that the input and output observation data are both corrupted by noises, and write the input and output at k as $\tilde{u}(k)=u(k)+n_i(k)$ and $\tilde{d}(k)=d(k)+n_o(k)$, where $n_i(k)$ and $n_o(k)$ are the additive noise. Define the input observation vector of Volterra filter at k as:

$$\tilde{\boldsymbol{X}}_V(k)=\left[\boldsymbol{x}_1^T(k) \quad \boldsymbol{x}_2^T(k) \quad \cdots \quad \boldsymbol{x}_N^T(k)\right]^T \tag{3}$$

where $\mathbf{x}_i(k)(i=1,\cdots,N)$ is the *ith* order input observation vector of the Volterra filter, for example:

$$\boldsymbol{x}_2(k)=\left[u_0^2 \quad u_0u_1 \quad \cdots \quad u_0u_{m-1} \quad u_1^2 \quad \cdots \quad u_1u_{m-1} \quad u_2^2 \quad \cdots \quad u_{M-1}^2\right]^T \tag{4}$$

where $u_{m-1}=\tilde{u}(k-m+1)$. Note that, as usual, the Volterra kernel is assumed to be symmetric without loss of generality[14]. Corresponding define the i*th* order kernel vector, for example:

$$\boldsymbol{h}_2=[h_2(0,0) \ \cdots \ h_2(0,M-1) \ \ h_2(1,1) \ \cdots \ h_2(M-1,M-1)]^T \tag{5}$$

And then define the Volterra kernel vector as:

$$\boldsymbol{H}_V = [\boldsymbol{h}_1^T, \boldsymbol{h}_2^T, \cdots, \boldsymbol{h}_N^T]^T. \tag{6}$$

By describing the Volterra filter as a pseudo-linear one and taking the minimum Rayleigh Quotient of the augmented Volterra weight vectors as the cost function, the Volterra MCA EXIN adaptive filter algorithm can be built as follow:

$$\boldsymbol{W}(k+1)=\boldsymbol{W}(k)-\mu\varepsilon(k)\frac{(\|\boldsymbol{W}(k)\|_2^2\boldsymbol{Z}(k)-\varepsilon(k)\boldsymbol{W}(k))}{\|\boldsymbol{W}(k)\|_2^4} \tag{7}$$

where $\boldsymbol{W}=[\boldsymbol{H}_V^T, \ -1]^T$ is the Volterra augmented kernel vector, $\boldsymbol{Z}(k)=[\tilde{\boldsymbol{X}}_V^T(k), \ \tilde{d}(k)]^T$ is called as the Volterra augmented observation vector, and $\varepsilon(k)=\boldsymbol{Z}^T(k)\boldsymbol{W}(k)$. μ is the step-size factor that controls the stability and rate of convergence of the adaptive algorithm. If the Eq. (7) is convergent, it is proved that the solution of the total least squares problem is the eigenvector associated with the smallest eigenvalue of the input autocorrelation matrix $\boldsymbol{R}$. where $\boldsymbol{R}=E\{\boldsymbol{Z}(k)\boldsymbol{Z}^T(k)\}$.

3 A Stable Volterra Total Least Square Adaptive Algorithm

3.1 The Derivation of the Stable Total Least Square Algorithm

Define the Rayleigh Quotient of the Volterra augmented kernel vector and a constraint to the last element of the Volterra augmented kernel vector via a LaGrange multiplier as the overall cost function:

$$\eta(\boldsymbol{W},\lambda)=\frac{\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W}}{\boldsymbol{W}^T\boldsymbol{W}}+\lambda(-1-\boldsymbol{W}^T\boldsymbol{e}_{N+1}) \tag{8}$$

where $\boldsymbol{R}=E\{\boldsymbol{Z}(k)\boldsymbol{Z}^T(k)\}$, $\boldsymbol{e}_{N+1}=[0,0,\cdots,1]^T$. The adaptive filter is just to adjust the weight coefficients to minimize the loss function.

With

$$\nabla\eta_W(\boldsymbol{W},\lambda)=\frac{2}{[\boldsymbol{W}^T\boldsymbol{W}]^2}\{\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{R}\boldsymbol{W}-\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W}\boldsymbol{W}\}-\lambda\boldsymbol{e}_{N+1} \tag{9}$$

Setting

$$\lambda = \boldsymbol{e}^T{}_{N+1} \frac{2(\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{R}\boldsymbol{W} - \boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W}\boldsymbol{W})}{(\boldsymbol{W}^T\boldsymbol{W})^2} \tag{10}$$

keeps the gradient to the constrained surface. Therefore,

$$\nabla \eta_W(\boldsymbol{W}, \lambda) = 2\hat{\boldsymbol{I}}(\frac{\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{R}\boldsymbol{W} - \boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W}\boldsymbol{W}}{(\boldsymbol{W}^T\boldsymbol{W})^2}) \tag{11}$$

where $\hat{\boldsymbol{I}} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & 0 \end{bmatrix}$.

To develop the adaptive algorithm, we will use the standard steepest descent approach of

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) - \frac{1}{2}\mu\nabla\eta_W(\boldsymbol{W}, \lambda) \tag{12}$$

Then

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) - \mu\hat{\boldsymbol{I}}\frac{\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{R}\boldsymbol{W} - \boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W}\boldsymbol{W}}{(\boldsymbol{W}^T\boldsymbol{W})^2} \tag{13}$$

We replace the stochastic quantities with their immediate values, then

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) - \mu\hat{\boldsymbol{I}}\varepsilon(k)\frac{(\|\boldsymbol{W}(k)\|_2{}^2\boldsymbol{Z}(k) - \varepsilon(k)\boldsymbol{W}(k))}{\|\boldsymbol{W}(k)\|_2{}^4} \tag{14}$$

If the Eq. (14) is convergent, it can be proved that the only stable fixed point of the algorithm is the eigenvector associated with the minimum eignvalue of the extended autocorrelation matrix $\boldsymbol{R}$.

3.2 The Performance Analysis of the Proposed Algorithm

3.2.1 The Equilibrium Point of the Algorithm

The equilibrium points of the algorithm satisfy

$$\hat{\boldsymbol{I}}(\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{R}\boldsymbol{W} - \boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W}\boldsymbol{W}) = 0 \tag{15}$$

Clearly, if $\boldsymbol{W}$ is an eigenvector of $\boldsymbol{R}$, the condition expressed in Eq. (15) is met. There set of fixed points for the constraint TLS algorithm can be written as

$$\boldsymbol{W}_i = \alpha_i\boldsymbol{V}_i = [\boldsymbol{H}_i{}^T \; -1]^T \tag{16}$$

where the $\boldsymbol{V}_i$ are the eigenvectors of $\boldsymbol{R}$ with associated eigenvector λ_i and α_i scales the final component of the ith eigenvector to –1. If we let $\|\boldsymbol{V}_i\| = 1$, then $\alpha_i \geq 1$ and

$$\boldsymbol{R}\boldsymbol{W}_i = \lambda_i\boldsymbol{W}_i \quad \text{With } \boldsymbol{W}_i{}^T\boldsymbol{W}_i = \alpha_i{}^2 \tag{17}$$

Therefore, we have established that the eigenvectors of, automatically scaled to be consistent with the inherent input/output relationship, are the fixed points of the algorithm.

3.2.2 Local Stability Analysis

Assume that the adaptation process has nearly converged and the Volterra augmented kernel vector estimate is in the neighbourhood of one of the fixed points. In this case, we can define the convergence error in the neighborhood of the ith fixed point as

$$\tilde{\boldsymbol{W}}(k) = \boldsymbol{W}(k) - \boldsymbol{W}_i \tag{18}$$

We can observe the behavior of the algorithm about the fixed points by substituting Eq. (18) into the update Eq. (13). Since we are assumed to be in the local area of the fixed point, we can ignore second order and higher terms in the numerator of the gradient increment involving $\tilde{\mathbf{W}}(k)$ to give (for each of the ith fixed points)

$$\tilde{\boldsymbol{W}}(k+1) = \tilde{\boldsymbol{W}}(k) - \mu\frac{\alpha_i{}^2\tilde{\boldsymbol{I}}[(\boldsymbol{R} - \lambda_i\boldsymbol{I})\tilde{\boldsymbol{W}}]}{\|\boldsymbol{W}_i + \tilde{\boldsymbol{W}}(k)\|^4} \tag{19}$$

We can make the following approximation:

$$\begin{aligned}
\frac{\mu\alpha_i{}^2}{\|\boldsymbol{W}_i + \tilde{\boldsymbol{W}}(k)\|^4} &= \frac{\mu\alpha_i{}^2}{(\|\boldsymbol{W}_i\|^2 + 2\boldsymbol{W}_i{}^T\tilde{\boldsymbol{W}}(k) + \|\tilde{\boldsymbol{W}}(k)\|^2)^2} \\
&\approx \frac{\mu\alpha_i{}^2}{(\|\boldsymbol{W}_i\|^2 + 2\boldsymbol{W}_i{}^T\tilde{\boldsymbol{W}}(k))^2} \\
&\approx \frac{\mu\alpha_i{}^2}{(\|\boldsymbol{W}_i\|^4 + 4\|\boldsymbol{W}_i\|^2\boldsymbol{W}_i{}^T\tilde{\boldsymbol{W}}(k))} \approx \frac{\mu\alpha_i{}^2}{\alpha_i{}^4} = \frac{\mu}{\alpha_i{}^2}
\end{aligned} \tag{20}$$

Then, Eq. (19) can be written as:

$$\tilde{\boldsymbol{W}}(k+1) = \tilde{\boldsymbol{W}}(k) - \mu\frac{\tilde{\boldsymbol{I}}[(\boldsymbol{R} - \lambda_i\boldsymbol{I})\tilde{\boldsymbol{W}}(k)]}{\alpha_i{}^2} \tag{21}$$

We can rearrange Eq. (21) in order to factor out the dependence of the fixed final component. In doing so, we translate into the non_extended parameter domain to get

$$\tilde{\boldsymbol{H}}(k+1) = [\boldsymbol{I} - \frac{\mu}{\alpha_i{}^2}(\boldsymbol{R}_0 - \lambda_i\boldsymbol{I})]\tilde{\boldsymbol{H}}(k) \tag{22}$$

where $\boldsymbol{R}_0$ is $\boldsymbol{R}$ with the last row and column removed. If the convergence error exponentially decays to zero in the neighbourhood of the fixed point, we conclude that the fixed point is locally stable. Local stability is guaranteed if the eignvalues of the matrix multiplying in Eq. (22) all have magnitude less than unity.

Let the eignvalues of the submatrix $\boldsymbol{R}_0$ be denoted as $\xi_i \quad i=1,2,\cdots n$. Because $\boldsymbol{R}_0$ is a submatrix of $\boldsymbol{R}$ obtained by removing the last row and column, the eignvalues of $\boldsymbol{R}_0$ are related to those of $\boldsymbol{R}$ as [12]:

$$\lambda_{\min} \le \xi_{\min} \le \lambda_2 \le \xi_2 \le \cdots \le \xi_{\max} \le \lambda_{\max} \tag{23}$$

By inspection, the eignvalues of the matrix multiplying $\tilde{\boldsymbol{H}}$ in Eq. (22) are of the form:

$$1-\frac{\mu}{\alpha_i^2}(\xi_j-\lambda_i) \tag{24}$$

Now, since both $\boldsymbol{R}_0$ and $\boldsymbol{R}$ are correlation matrices, all ξ_i and λ_i are non-negative. Given a small enough step size $\mu > 0$, the only fixed point for which the magnitude of Eq. (24) can be less than unity for all j is the eigenvector associated with $\lambda_{\min}$. Therefore, the only stable fixed point of the algorithm is the eigenvector associated with the minimum eignvalue of the extended autocorrelation matrix $\boldsymbol{R}$. Thus, the desired TLS solution is the only locally stable equilibrium point of the algorithm.

Furthermore, conditions can be derived to determine a bound on μ in order to guarantee local stability. Setting $\left|1-\frac{\mu}{\alpha_i^2}(\xi_j-\lambda_{\min})\right|<1$, the step-size parameter μ to guarantee the stability of the algorithm is derived:

$$0<\mu<\frac{2\alpha_i^2}{(\xi_{\max}-\lambda_{\min})} \tag{25}$$

4 Simulation

In the simulations, two algorithms are used for nonlinear adaptive filter, i.e. the Volterra MCA EXIN adaptive filter algorithm [10] and the presented stable Volterra total least squares adaptive filter algorithm. Their convergence performances are compared under different extent noise environment and by using different learning factor. The nonlinear system is given as following:

$$\begin{aligned} y(k) &= 0.62u(k) - 0.25u(k-1) + 0.64u(k-2) \\ &+ 0.82u^2(k) - 1.48u(k-1)u(k-2) \end{aligned} \tag{26}$$

Define the learning error of the Volterra kernel vector as:

$$Error(k) = 20\log\left[\left\|\boldsymbol{H}_V - \hat{\boldsymbol{H}}_V(k)\right\|^2\right], \tag{27}$$

where

$\boldsymbol{H}_V = [0.62\ \ -0.25\ \ 0.64\ \ 0.82\ \ 0\ \ 0\ \ 0\ \ 0\ \ -0.74\ \ 0\ \ -0.74\ \ 0]^T$

is the true Volterra kernel vector of the analyzed nonlinear system, $\hat{\boldsymbol{H}}_V(k)$ is estimated Volterra kernel vector.

Assume that the SNR (signal-noise rate) of the input signal is equal to the SNR of the output signal. The additive noise is independent zero-mean white noise. And

$$SNR = 10\log\left(\frac{E\{\|\boldsymbol{y}(k)\|^2\}}{E\{\|\boldsymbol{n}_o(k)\|^2\}}\right), \tag{28}$$

where $\mathbf{y}(k)$ is the output signal vector, $\boldsymbol{n}_o(k)$ is the interference of the output. Fig 1、Fig2 and Fig 2 show respectively the learning curves for SNR=30dB 、SNR=20dB and SNR=10dB with different learning factors.

The simulation results have shown that the presented stable Volterra total least squares algorithm for nonlinear adaptive filter has excellent convergence performance, and the robust anti-noise performance and the stable convergence precision of the proposed algorithm are remarkably higher Volterra MCA EXIN algorithms, and it permits to use a larger learning factor, that is very fine for the practical applications.

5 Conclusion

We propose a stable total least square algorithm for the nonlinear Volterra filter with the corrupted input and output signal. The proposed does not require normalization. The simulation results have shown that, in addition to the fine convergence, the robust anti-noise performance and the stable convergence precision of the proposed algorithm are remarkably higher than other total least square algorithms, which is more efficient to a practical application.

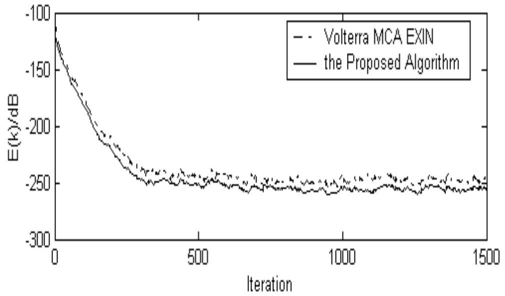

Fig. 1(a). learning curve at $\mu=0.05$, 30dB

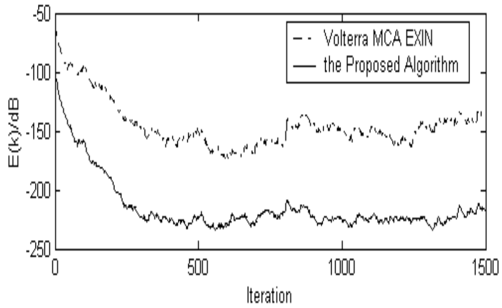

Fig. 1(b). learning curve at $\mu=0.1$, 30dB

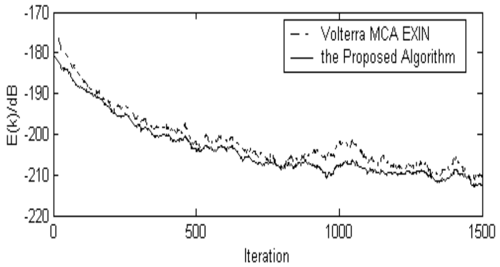

Fig. 2(a). learning curve at $\mu=0.005$, 20dB

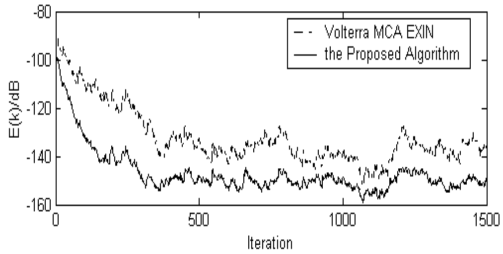

Fig. 2(b). learning curve at $\mu=0.075$, 20dB

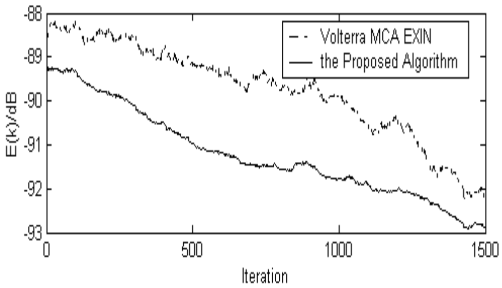

Fig. 3(a). learning curve at $\mu=0.001$, 10dB

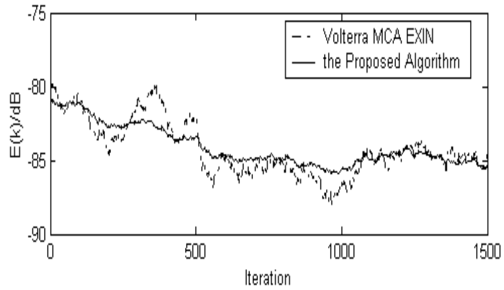

Fig. 3(b). learning curve at $\mu=0.005$, 10dB

References

[1] G H Golub and C F V Loan. An analysis of total least squares problem. SIAM Numer. Anal., 17(6): 883-893, 1980.

[2] L Xu, E Oja and C Y Suen. Modified Hebbian learning for curve and surface fitting. Neural Networks, 5(3): 441-457, 1992.

[3] K Q Gao, M O Ahmad and M N S Swamy. Constrained anti-Hebbian learning algorithm for total least-squares estimation with applications to adaptive FIR and IIR filtering. IEEE Trans. on Circuits and Systems II: Analog and Digital Signal Processing, 41(11): 718-729, 1994.

[4] C E Davila. An efficient recursive total least squares algorithm for FIR adaptive filtering. IEEE Trans. on Signal Processing, 42(2): 268-280, 1994.

[5] D Z Feng, Z Bao and L C Jiao. Total least mean squares algorithm. IEEE Trans. on Signal Processing, 46(8): 2122-2130, 1998.

[6] B E Dunne and G A Williamson. Stable Simplified Gradient Algorithms for Total Least Squares Filtering. in Proceedings of 32nd Ann. Asilomar Conference On Signals, Systems, and Computers. Pp. 1762-1766,November 2000, Pacific Grove, CA.

[7] F L Luo, R Unbehauen and A Cichocki. A minor component analysis algorithm. Neural Networks, 10(2): 291-297, 1997.

[8] S Ouyang, Z Bao and G liao. Adaptive step-size minor component extraction algorithm. ELECTRONICS LETTERS, 35(18): 443-444, 1999.

[9] S Ouyang, Z Bao, and P C Ching. Adaptive Minor Component Extraction With Modular Structure. IEEE Trans On Signal Processing, 49(9): 2127-2137, 2001.

[10] Giansalvo Cirrincione, Maurizio Cirrincione. The MCA EXIN Neuron for the Minor Component Analysis. IEEE Trans On Neural Networks, 13(1): 160~187, 2002.

[11] S Haykin. Adaptive filter theory. Englewood Cliffs, NJ: Prentice Hall, 1991.

[12] G Strang, Linear Algebra and Its Applications. New York: Academic Press, Section 6.4, 1980.

[13] M V John, Adaptive polynomial filters, IEEE Signal Processing Magazine, 8 (3): 10-26, 1991.

[14] M Schetzen, The Volterra and Wiener Theories of Nonlinear Systems. New York: Wiley, 1980.

[15] Chongzhao Han, Liqi Wang, Xiaoquan Tang and Yingnong Dang, Identification of Nonparametric GFRF Model for a Class of Nonlinear Dynamic Systems, Control Theory and Applications (Chinese Journal), 16 (6): 816-825, 1999.