TRANSFORM DOMAIN ADPTIVE VOLTERRA FILTER ALGORITHM BASED ON CONSTRAINED OPTIMIZATION

Yuexian Zou, Shing-Chow Chan and Tung-Sang Ng

Department of Electrical and Electronic Engineering
The University of Hong Kong, Pokfulam Road, Hong Kong
{yxzou,scchan,tsng}@eee.hku.hk

ABSTRACT

In this paper, a transform domain normalised partially decoupled *LMS* (*TP-LMS*) algorithm is proposed based on the partially decoupled transform domain Volterra filter. It is formulated by applying an orthogonal transform to the first order of the partially decoupled Volterra filter, in which the filter weights of a given order are optimised independently of those in the higher order. This approach results in solving the minimum mean square error (*MSE*) filtering problem as a series of constrained optimisation problem and the modular structure order by order. Simulation of a system identification application indicates *TP-LMS* algorithm provides a trade-off between convergence speed and computational complexity.

1. INTRODUCTION

Nonlinear adaptive filtering techniques employing truncated Volterra series model have become a powerful tools in some important applications. Adaptive Volterra filters have been developed for echo cancellation [2], system identification [4], interference and noise reduction [3]. With the Volterra expansion, the nonlinear filter weights can be treated in a fashion analogous to linear filtering problem since there is a linear relationship between its filter output and filter weights. The widely used linear adaptive algorithm such as stochastic gradient descent *(SGD)* algorithms and recursive least square (*RLS*) algorithms can be directly applied to Volterra filter. However, the convergence speed of Volterra filter is exacerbated not only by the input signal statistic but also by the nonlinearity itself [5,12]. Although *RLS* algorithm is insensitive to the condition number and has fast convergence, it has complexity of $O(N^6)$ in direct implementation and $O(N^3)$ in fast form implementation [6] for the quadratic Volterra filter realization with *N*-sample memory. The general lattice implementation of the Volterra filter suffered from the over-parameterised problem [4]. Another fast *RLS* algorithm for adaptive quadratic Volterra filter has been developed with $O(N^3)$ [6] by exploring the forward predictor error and backward predictor error. All of these fast algorithms suffer from the numerical unstable problem [10]. Recently, a time domain partially decoupled Volterra filter has been proposed in [7], in which its filter weights of a given order are optimised independently of those in the higher order. The Volterra filtering problem is formulated as a series of constrained optimization problem. The advantage of this approach is its possibility to increase the convergence speed by using different adaptive algorithm for different order. In [7], a new normal equation is derived and the partially decoupled *LMS* (*P-LMS*) algorithm is also developed which is with the similar form of the *LMS* algorithm but the stepsize bound can be computed straightforward.

In this paper, we propose a modification to the partially decoupled Volterra filter. We apply an orthogonal transform to the first order Volterra filter input signal as shown in Fig.1. A transform domain normalised partially decoupled least mean square (*TP-LMS*) algorithm is then derived. The proposed approach improves the convergence rate over time domain approach with little increase in computational complexity. Some performance analyses are presented. Simulation results on system identification and comparison with *RLS* and *P-LMS* algorithms are given.

2. BASIC THEORY

2.1 The Volterra Filter Theory

A nonlinear system can be modelled as a *N*-memory and the *Z*-order Volterra filter with the input-output relationship,

$$y(n) = \mathbf{H}_V^T[Z]\mathbf{X}_V(n,Z)\,. \tag{1}$$

where, $\mathbf{X}_V(n,Z) = [\mathbf{X}_1^T(n),...,\mathbf{X}_j^T(n),\cdots,\mathbf{X}_Z^T(n)]^T$ denotes the $Z-$order Volterra filter observation vector and $\mathbf{H}_V^T[Z] = [\mathbf{H}_1^T,...,\mathbf{H}_j^T,...,\mathbf{H}_Z^T]$ denotes the corresponding filter weight vector; n, j are time and order index respectively; $\mathbf{X}(n) = [x_1, x_2, \cdots, x_N]^T, x_i \equiv x(n-i+1)$ is the input vector of the system. $\mathbf{X}_j(n)$, $\mathbf{H}_j$ denote the *jth*-order input vector containing all the *jth*-order products of the elements in $\mathbf{X}(n)$ and the corresponding filter weight vector, respectively; $y(n)$ is the output of the system. For example, the second-order filter input vector can be described as,

$$\mathbf{X}_2(n) = [x_{2,1},\cdots x_{2,N_2}]^T = [x_1^2,\cdots x_N^2, x_1x_2,\cdots x_1x_N,\cdots x_{N-1}x_N]^T\,, \tag{2}$$

where, $N_2 = N(N+3)/2$ with corresponding weight vector $\mathbf{H}_2(n)$. Based on the Volterra expansion, $\mathbf{X}_j(n)$ consists of $N_j = \binom{N+j-1}{j}$ elements, so does $\mathbf{H}_j(n)$. Therefore, there are $M_Z = \sum_{j=1}^{Z} N_j$ elements in $\mathbf{X}_V(n,Z)$ and $\mathbf{H}_V[Z]$. The input correlation matrix and cross-correlation vector of the $Z-$order Volterra filter can be defined as,

$$\mathbf{R}_V[Z] = E[\mathbf{X}_V(n,Z)\mathbf{X}_V^T(n,Z)] = \begin{bmatrix} \mathbf{R}_{1,1} & \cdots & \mathbf{R}_{1,Z} \\ \vdots & \ddots & \vdots \\ \mathbf{R}_{Z,1} & \cdots & \mathbf{R}_{Z,Z} \end{bmatrix}, \tag{3}$$

$$\mathbf{P}_V[Z] = E[d(n)\mathbf{X}_V(n,Z)] = [\mathbf{P}_1^T,...,\mathbf{P}_Z^T]^T\,. \tag{4}$$

where, $\mathbf{R}_{i,j} = E[\mathbf{X}_i(n)\mathbf{X}_j^T(n)]$ is a correlation matrix of $\mathbf{X}_i(n)$ and $\mathbf{X}_j(n)$, $\mathbf{P}_j = E[d(n)\mathbf{X}_j(n)]$ is cross-correlation vector of $d(n)$ and $\mathbf{X}_j(n)$. Here, the Volterra filter problem can be solved as a linear filtering problem since there is a linear relationship between its output and the filter weights as shown in (1). The minimisation of the *MSE* results in the Wiener normal equation,

$$\mathbf{R}_V[Z]\mathbf{H}_{opt}[Z] = \mathbf{P}_V[Z]\,, \tag{5}$$

0-7803-5471-0/99/$10.00©1999 IEEE

and the minimum mean square error (*MMSE*) is given by [10]:

$$J_{\min} = \sigma_d^2 - \mathbf{P}_V^T[Z]\mathbf{H}_{V,opt}[Z] = \sigma_d^2 - (\mathbf{R}_V[Z]\mathbf{H}_{V,opt}[Z])^T\mathbf{H}_{V,opt}[Z]$$
$$= \sigma_d^2 - \mathbf{P}_V^T[Z](\mathbf{R}_V[Z])^{-1}\mathbf{P}_V[Z] \quad (6)$$

where, $\mathbf{H}_{V,opt}[Z]$ is the optimal solution of the Volterra filter in MSE sense; $\sigma_d^2 = E[d^2(n)]$ is the desired signal variance.

2.2 Partially Decoupled Volterra Filter [7]

The partially decoupled Volterra filter was proposed in [7], which optimises the given order filter weights independently of the higher order filter. For $Z=1$, the first order filter weights $\mathbf{H}_1$ is optimised independently of the higher order filter without any constrained condition. The *MSE* is defined as $J_1[\mathbf{H}_1] = E\{[d(n) - \mathbf{H}_1^T\mathbf{X}_1(n)]^2\}$ and the optimal filter weights can be obtained by $\mathbf{H}_{1,opt} = \mathbf{R}_{1,1}^{-1}\mathbf{P}_1$. The quadratic Volterra filter is constructed by adding the second order filter on top of the first one. So, the Volterra filter weights $\mathbf{H}_V[2]$ can be optimised by minimising the constrained *MSE* cost function,

$$J_2[\mathbf{H}_1, \mathbf{H}_2, \mathbf{C}_1] = E[e^2(n)] + \mathbf{C}_1^T(\mathbf{R}_{1,1}\mathbf{H}_1 - \mathbf{P}_1), \quad (7)$$

Here, $e(n) = d(n) - \mathbf{H}_V^T[2]\mathbf{X}_V(n,2)]$. $\mathbf{C}_1$ is $N\times 1$ the Lagrange constant vector. The constrained minimum is obtained by taking partial gradients with respect to $\mathbf{H}_1, \mathbf{H}_2$ and $\mathbf{C}_1$ in (7) and let them to zeros which results in a new normal equation

$$\begin{bmatrix} \mathbf{R}_{1,1} & \mathbf{0} \\ \mathbf{R}_{2,1} & \mathbf{R}_{2,2} \end{bmatrix}\begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \end{bmatrix}. \quad (8)$$

From (8), we can see that $\mathbf{H}_1$ is decoupled from $\mathbf{X}_2(n)$ and $\mathbf{H}_2$. Similarly for the Z-order Volterra filter, the constrained normal equation can be obtained [7]:

$$\mathbf{R}_V^{\Delta}[Z]\mathbf{H}_{V,opt}[Z] = \mathbf{P}_V[Z], \quad (9a)$$

where,

$$\mathbf{R}_V^{\Delta}[Z] = \begin{bmatrix} \mathbf{R}_{1,1} & 0 & \cdots & 0 \\ \mathbf{R}_{2,1} & \mathbf{R}_{2,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{R}_{Z,1} & \mathbf{R}_{Z,2} & \cdots & \mathbf{R}_{Z,Z} \end{bmatrix}, \quad (9b)$$

Note that, the upper triangular entries of $\mathbf{R}_V^{\Delta}[Z]$ vanish due to the $Z-1$ sets of constrained conditions and compared with (3),

$$\mathbf{R}_V[Z] = \mathbf{R}_V^{\Delta}[Z] + (\mathbf{R}_V^{\Delta}[Z])^T - \mathbf{Q}. \quad (10)$$

where $\mathbf{Q} = diag[diag(\mathbf{R}_V[Z])]$, is a block-diagonal matrix.

2.3. Partially Decoupled *SGD* and *LMS* algorithm

In general, the direct solution of (9a) for higher order Volterra filter in may applications is formidable [6,7,10]. Obviously, many adaptive algorithms can be applied to get the solution of (9a). Without loss of generality, we choose $Z=2$ as an example. The stochastic gradient descent (*SGD*) algorithm for the quadratic Volterra filter is [10],

$$\mathbf{H}_1(n+1) = \mathbf{H}_1(n) + \frac{1}{2}\mu_1(-\frac{\partial J_1}{\partial \mathbf{H}_1}) = \mathbf{H}_1(n) - \mu_1\nabla_1(n). \quad (11)$$

$$\mathbf{H}_2(n+1) = \mathbf{H}_2(n) + \frac{1}{2}\mu_2(-\frac{\partial J_2}{\partial \mathbf{H}_2}) = \mathbf{H}_2(n) - \mu_2\nabla_2(n). \quad (12)$$

where, $\nabla_1(n) = \mathbf{R}_{1,1}\mathbf{H}_1(n) - \mathbf{P}_1$, $\nabla_2(n) = \mathbf{R}_{2,2}\mathbf{H}_2(n) - \mathbf{P}_2 + \mathbf{R}_{2,1}\mathbf{H}_1(n)$ are the gradient vector. μ_1, μ_2 are two stepsizes. The range of the stepsize μ_j to guarantee convergence for the *jth* order Volterra filter has been proved to be [7]:

$$0 < \mu_j < 2/\lambda_{j\max}. \quad (13)$$

where, $\lambda_{j\max}$ is the maximum eigenvalues of the submatrix $\mathbf{R}_{j,j}$ in $\mathbf{R}_V^{\Delta}[Z]$. From (11), (12), we can see that, for quadratic Volterra filter, the *SGD* algorithm needs $O(N^4)$ operations in each iteration to update $\mathbf{H}_1$, $\mathbf{H}_2$. In order to reduce the computational complexity, the *LMS* algorithm has been suggested[7], and gradient vectors $\hat{\nabla}_1(n) = e(n)\mathbf{X}_1(n)$, $\hat{\nabla}_2(n) = e(n)\mathbf{X}_2(n)$ replace gradient vectors $\nabla_1(n)$ and $\nabla_2(n)$ in (11), (12) respectively. The filter weights can be updated by

$$\mathbf{H}_1(n+1) = \mathbf{H}_1(n) + \mu_1\hat{\nabla}_1(n) = \mathbf{H}_1(n) + \mu_1 e(n)\mathbf{X}_1(n), \quad (14)$$

$$\mathbf{H}_2(n+1) = \mathbf{H}_2(n) + \mu_2\hat{\nabla}_2(n) = \mathbf{H}_2(n) + \mu_2 e(n)\mathbf{X}_2(n). \quad (15)$$

Equation (14), (15) are called the partially decoupled *LMS* (*P-LMS*) algorithm to distinguish the conventional *LMS* algorithm. Analysis also shown that the stepsizes satisfy the same conditions as showed in (13) for convergence of the algorithm [7].

3. TRANSFORM DOMAIN PARTIALLY ECOUPLED *LMS* (*TP-LMS*) ALGORITHM

The main problem of *LMS* and *P-LMS* algorithm is their slow convergence rate. In the present work, we focus on improving the convergence rate of *P-LMS* algorithm by a transform approach with power normalisation. The block diagram of transform domain adaptive Z-order Volterra filter is shown in Fig.1.

$$\overline{\mathbf{X}}(n) = \mathbf{F}\mathbf{X}(n) = [\overline{x}_1, \cdots, \overline{x}_N]^T. \quad (16)$$

where, $\overline{x}_k = \mathbf{f}_k^T\mathbf{X}(n) = \sum_{i=1}^{N} f_{i,k}x_i$, $k = 1,2,\cdots N$ and $\mathbf{f}_k^T = [f_{1,k}, \cdots, f_{N,k}]$ is the *kth* row vector of the orthogonal transform matrix $\mathbf{F}$ of rank N. In particular, variables with the over bar represent the transform domain variables. Obviously, the transform domain Volterra filter weights can also be adapted by the *P-LMS* algorithm,

$$\overline{\mathbf{H}}_1(n+1) = \overline{\mathbf{H}}_1(n) + \overline{\mu}_1\overline{e}(n)\overline{\mathbf{X}}_1(n), \quad (17)$$

$$\overline{\mathbf{H}}_2(n+1) = \overline{\mathbf{H}}_2(n) + \overline{\mu}_2\overline{e}(n)\overline{\mathbf{X}}_2(n). \quad (18)$$

where, $\overline{\mathbf{H}}_1(n+1), \overline{\mathbf{H}}_2(n+1)$ are the first and the second order weight vector at time $(n+1)$. $\overline{e}(n) = d(n) - \overline{\mathbf{H}}_V^T[2]\overline{\mathbf{X}}_V(n,2)$ is the error signal. $\overline{\mathbf{X}}_1(n) = \overline{\mathbf{X}}(n)$, $\overline{\mathbf{X}}_2(n)$ contains all the second-order products of the elements in $\overline{\mathbf{X}}(n)$. $\overline{\mu}_1, \overline{\mu}_2$ are two different stepsizes which satisfy the condition

$$0 < \overline{\mu}_j < 2/\overline{\lambda}_{j\max}, j = 1,2, \quad (19)$$

where, $\overline{\lambda}_{j\max}$ is the maximum eigenvalues of the $\overline{\mathbf{R}}_{i,j} = E[\overline{\mathbf{X}}_i(n)\overline{\mathbf{X}}_j^T(n)]$. The transform approach in Fig.1 will result in an equivalent non-orthogonal transform for the Volterra filter

observation vector $\mathbf{X}_V(n,Z)$. Considering the case $Z=2$, the pth element in $\overline{\mathbf{X}}_2(n)$ can be computed as:

$$\bar{x}_{2,p}=\bar{x}_i\bar{x}_k=\sum_{m=1}^{N}\sum_{l=1}^{N}f_{im}f_{kl}x_mx_l \quad 1\le i,k\le N,\ i\le k,\ p=1,\cdots N_2 \quad (20)$$

From (16), (20), we can get the relationship between $\overline{\mathbf{X}}_V(n,2)$ and $\mathbf{X}_V(n,2)$,

$$\overline{\mathbf{X}}_V(n,2)=\mathbf{B}\mathbf{X}_V(n,2)=\begin{bmatrix}\mathbf{F} & 0\\ 0 & \overline{\mathbf{F}}_2\end{bmatrix}\begin{bmatrix}\mathbf{X}_1(n)\\ \mathbf{X}_2(n)\end{bmatrix}. \quad (21)$$

where, $\mathbf{B}$ is a new equivalent linear transform matrix. Each element of $\overline{\mathbf{F}}_2$ can be computed from (20) according to the corresponding sequence order of $\mathbf{X}_2(n)$. It is easy to derive the following equation,

$$\overline{\mathbf{R}}_V^{\Delta}(n,2)=\mathbf{B}\mathbf{R}_V^{\Delta}(n,2)\mathbf{B}^T=\begin{bmatrix}\mathbf{F}\mathbf{R}_{1,1}\mathbf{F}^T & 0\\ \overline{\mathbf{F}}_2\mathbf{R}_{2,1}\mathbf{F}^T & \overline{\mathbf{F}}_2\mathbf{R}_{2,2}\overline{\mathbf{F}}_2^T\end{bmatrix}=\begin{bmatrix}\overline{\mathbf{R}}_{1,1} & 0\\ \overline{\mathbf{R}}_{2,1} & \overline{\mathbf{R}}_{2,2}\end{bmatrix}. \quad (22)$$

In (22), $\overline{\mathbf{R}}_{1,1}$ is approximately diagonalized by orthogonal transform $\mathbf{F}$, $\overline{\mathbf{R}}_{2,2}$ is a band matrix resulting from non-orthogonal transform $\overline{\mathbf{F}}_2$. With the power normalized approach, the transform domain partially decoupled normalised *LMS* (*TP-LMS*) algorithm can be directly derived from (17), (18)

$$\overline{\mathbf{H}}_1(n+1)=\overline{\mathbf{H}}_1(n)+\overline{\mu}_1\overline{\Lambda}_1^{-2}\overline{e}(n)\overline{\mathbf{X}}_1(n), \quad (23a)$$

$$\overline{\mathbf{H}}_2(n+1)=\overline{\mathbf{H}}_2(n)+\overline{\mu}_2\overline{\Lambda}_2^{-2}\overline{e}(n)\overline{\mathbf{X}}_2(n). \quad (23b)$$

where, $\overline{\Lambda}_j^{-2}=diag(\overline{\sigma}_{j,1}^{-2}(n),\ldots,\overline{\sigma}_{j,N_j}^{-2}(n)),\ j=1,2$ are diagonal matrixes. We can estimate the signal power by a single-pole low pass filter as following in a recursive way [11],

$$\overline{\sigma}_{1,i}^2(n)=\alpha\overline{\sigma}_{1,i}^2(n-1)+(1-\alpha)\bar{x}_i^2,\ i=1,\ldots,N. \quad (24)$$

$$\overline{\sigma}_{2,i}^2(n)=\alpha\overline{\sigma}_{2,i}^2(n-1)+(1-\alpha)\bar{x}_{2,i}^2,\ i=1,\ldots,N_2. \quad (25)$$

where, $0<\alpha\le 1$ is a forgetting factor.

4. PEFORMANCE ANALYSIS

4.1 The Convergence Performance

The convergence conditions for *P-LMS* algorithm in time domain have been explored in [7]. For *P-LMS* algorithm, the stepsize need to satisfy the sufficient condition in (13). For *TP-LMSA*, it is well-known that the convergence speed can be improved over time domain approach by using the data dependent stepsize $\hat{\mu}_{j,i}=\overline{\mu}_j/\overline{\sigma}_{j,i}^2$ in (23a), (23b) and $\overline{\mu}_j$ satisfies (19).

4.2 The Minimum Mean Square Error (*MMSE*)

Firstly, in this section, we will investigate the *MMSE* of the constrained approach comparing to the unconstrained one. We assume that the tape-input vectors of the different order of the Volterra filter and the desire signal are jointly stationary. Based on the definition, the constrained *MMSE* of $Z-$order Volterra filter can be obtained when $\mathbf{H}_{V,opt}[Z]=(\mathbf{R}_V^{\Delta}[Z])^{-1}\mathbf{P}_V[Z]$ [10],

$$\begin{aligned}J_{\min}^{\Delta}&=\sigma_d^2-\mathbf{P}_V^T[Z]\mathbf{H}_{V,opt}[Z]+\mathbf{C}_{P-1}^T(\mathbf{R}_V^{\Delta}[Z-1]\mathbf{H}_{V,opt}[Z-1]-\mathbf{P}_V[Z-1])\\&=\sigma_d^2-\mathbf{P}_V^T[Z](\mathbf{R}_V^{\Delta}[Z])^{-1}\mathbf{P}_V[Z]\end{aligned} \quad (26)$$

Comparing (6) to (26), we can see that $J_{\min}$ is determined by the $\mathbf{R}_V[Z]$ which contains all information of $\mathbf{X}_V(n,Z)$, but $J_{\min}^{\Delta}$ is by $\mathbf{R}_V^{\Delta}[Z]$ which loses some information between the cross-correlation between the different order. Only when the input vector is fully decoupled that means the $\mathbf{R}_V[Z]$ is a diagonal matrix, we have $J_{\min}=J_{\min}^{\Delta}$, otherwise, there exist some performance degrade for partially decoupled approach.

Secondly, we consider the transform domain *TP-LMS* algorithm. It is easy to get the relationship between time domain and transform domain without the power normalisation,

$$\begin{aligned}&\overline{\mathbf{R}}_V[Z]=E[\overline{\mathbf{X}}_V(n,Z)\overline{\mathbf{X}}_V^T(n,Z)]=\mathbf{B}\mathbf{R}_V[Z]\mathbf{B}^T,\\&\overline{\mathbf{P}}_V[Z]=E[d(n)\overline{\mathbf{X}}_V(n,Z)]=\mathbf{B}\mathbf{P}_V[Z],\\&\overline{\mathbf{H}}_V[Z]=(\mathbf{B}^T)^{-1}\mathbf{H}_V[Z],\\&\overline{\mathbf{R}}_V^{\Delta}[Z]=\mathbf{B}\mathbf{R}_V^{\Delta}[Z]\mathbf{B}^T.\end{aligned} \quad (27)$$

Based on the definition, we can compute the *MMSE* as,

$$\begin{aligned}\bar{J}_{\min}^{\Delta}&=\sigma_d^2-\overline{\mathbf{P}}_V^T[Z]\overline{\mathbf{H}}_{V,opt}[Z]+\overline{\mathbf{C}}_{P-1}^T(\overline{\mathbf{R}}_V^{\Delta}[Z-1]\overline{\mathbf{H}}_{V,opt}[Z-1]-\overline{\mathbf{P}}_V[Z-1])\\&=\sigma_d^2-\overline{\mathbf{P}}_V^T[Z](\overline{\mathbf{R}}_V^{\Delta}[Z])^{-1}\overline{\mathbf{P}}_V[Z]\\&=\sigma_d^2-\mathbf{P}_V^T[Z](\mathbf{R}_V^{\Delta}[Z])^{-1}\mathbf{P}_V[Z]\end{aligned} \quad (28)$$

Comparing (28) to (26), $\bar{J}_{\min}^{\Delta}=J_{\min}^{\Delta}$, it means the transform $\mathbf{B}$ does not change the *MMSE* just like the orthogonal transform does. The power normalisation process in orthogonal transform always increase the steady error [9,12]. So, it can be predictable that the normalised non-orthogonal transform will also increase the steady error.

4.3 Computational Complexity

Finally, for *TP-LMS* algorithm, we discuss the computational complexity of quadratic Volterra filter. Fast recursive transform algorithm in general would require complexity $O(N)$ [11]. The estimation of $\Lambda_j^{-2}, j=1,2$ in (23a), (23b) requires $N(N+5)$ multiplications, $N(N+3)/2$ additives and $N(N+3)/2$ divisions. The update of the filter weights in (23a), (23b) needs $N(N+3)+2$ multiplications and $N(N+3)/2$ additions. The total computation requirement is therefore of order $O(N^2)$ which is the same as the number of Volterra filter weights.

5. SIMULATION RESULTS

In this section, we present simulation studies using a system identification application to demonstrate the speed of convergence and *MSE* of *TP-LMS* algorithm. The adaptive filter has the same nonlinear structure as that of the unknown nonlinear system which can be described as

$$\begin{aligned}y(n)=&[-0.78x(n)-1.48x(n-1)+1.39x(n-2)+0.04x(n-3)\\&+0.54x^2(n)-1.63x^2(x-1)+1.41x^2(n-2)-0.13x^2(n-3)+\\&+3.72x(n)x(n-1)+1.86x(n)x(n-2)-0.76x(n)x(n-3)+\\&0.76x(n-1)x(n-2)-0.12x(n-1)x(n-3)-1.52x(n-2)x(n-3)]^T\end{aligned} \quad (29)$$

The coloured input signal $x(n)$ satisfies the input-output relationship: $x(n)=bx(n-1)+\sqrt{1-b^2}v(n)$ [5], where $v(n)$ is Guassian process with zero-mean, variance unit, $b=.9$ is a constant parameter which decides the level of correlation between adjacent samples of $x(n)$. The output *SNR* is defined as

$SNR = 20\log(\sigma_d^2/\sigma_n^2)$ and is chosen equal to 20 db; μ_1, μ_2, $\bar{\mu}_1, \bar{\mu}_2$ are chosen equal to .04,.02,.06,.02 respectively and $\mathbf{F}$ is chosen as *DCT*. The performance is evaluated by *MSE* and the normalised squared norm of the weight error (*NSWE*) [6],

$$NSWE = 10\log \sum_{i=1}^{M_2} \left| h_i(n) - h_i^* \right|^2 \Big/ \sum_{i=1}^{M_2} \left| h_i^* \right|^2 . \qquad (30)$$

where, $M_2 = 14$, h_i^* is the *ith* unknown system coefficients in (29) and $h_i(n)$ is the corresponding adaptive filter weight at time n. The ensemble average *MSE* and the *NSWE* obtained by averaging over 20 independent runs are plotted in Fig.2 and Fig.3 respectively. Simulation results in Fig.2 and Fig.3 demonstrate that the proposed *TP-LMS* algorithm has a better convergence rate than that of *P-LMS* algorithm, but slower than that of *RLS* algorithm. In particular, from curves (2), (3) in Fig.2 and Fig.3, we can see that the steady error of *TP-LMS* algorithm is a slightly greater than that of *P-LMS* algorithm.

6. CONCLUSION

A new transform domain partially decoupled normalised *LMS* (*TP-LMS*) algorithm is derived in this paper. By applying an orthogonal transform on the first order filter, we constructed the so-called partially decoupled transform domain Volterra filter, where the optimal weights satisfy the constrained normal equation. Based on this structure, the equivalent transform matrix is formulated from the elements of the orthogonal transform following the Volterra expansion. Power normalisation is applied to improve the convergence speed. The advantage of this constrained approach lies on the modular filter structure and the possibility of applying different adaptation algorithm to different Volterra filter order. Simulation results indicated that *TP-LMS* algorithm has better convergence speed than that of *P-LMS* algorithm but with small increase in complexity. It can be viewed as a trade-off between the convergence speed and complexity.

7. REFERENCE

[1] I. Pitas, A. N. Venetsanopoulos, *Nonlinear digital filters*, Kluwer Academic Publisher, 1990.

[2] G. L. Sicuranza, A. Bucconi, P. Mitri, "Adaptive echo cancellation with nonlinear digital filters", *IEEE International Conference on Acoustics, Speech, and Signal Processing*, page 3.10.1-3.10.4, 1984.

[3] E. Biglieri, A. Gersho, R.D. Gitlin, T. L. Lim, "Adaptive cancellation of nonlinear intersymbol interference for voice band data transmission" *IEEE J. Selected Areas in Communications*. Vol. 2, page765-777, 1984.

[4] V. J. Mathews, "Orthogonalization of coloured Gaussian signals for Volterra system identification", *IEEE Signal Processing Letters*, Vol.2, No.10, 188-190, October 1995.

[5] V. J. Mathews, "Adaptive Volterra filters using orthogonal Structures", *IEEE Signal Processing Letters*. Vol.3, No.12, 307-309, Dec., 1996.

[6] Junghsi Lee, V. J. Mathews." A fast recursive Least Squares adaptive second-order Volterra filter and its performance analysis", *IEEE Trans. on Signal Processing*, Vol..41, No. 3, 1087-1102, March, 1993.

[7] David W. Griffith, "Partially decoupled Volterra filters: formulation and LMS adaptation", *IEEE Trans. On Signal Processing*, Vol. 45, No.6, 1485-1494, June, 1997.

[8] Xiaohui Li and W.K. Jemkins, C.W. Therrien, "A computationally efficient algorithm for adaptive quadratic Volterra filters", *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Hong Kong, page2184-2187, 1997.

[9] J. C. Lee and Chong Kwan Un, 'Performance of transform domain LMS adaptive digital filters', *IEEE Trans. On Acoustics, Speech, and Signal Processing*, Vol. ASSP-34, No. 3, 499-510, 1986.

[10] Simon Haykin, *Adaptive filter theory*, Englewood Cliffs, NJ: Prentice - Hall, 1991.

[11] S. S. Narayan, A. M. Peterson and M. J. Narasimha, "Transform domain LMS algorithm", *IEEE Trans. On Acoustics, Speech, and Signal Processing*, Vol. ASSP-31, page 609-615, 1983.

[12] D. F. Marshall and W.K. Jenkins, "The use of orthogonal transforms for improving performance of adaptive filters", *IEEE. Trans. On Circuits and Systems*, Vol. 36, No.4, page 474-484, 1989.

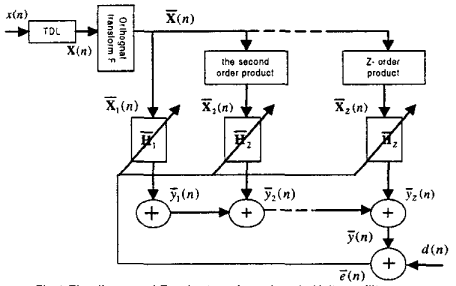

Fig.1 The diagram of Z-order transform domain Volterra filter

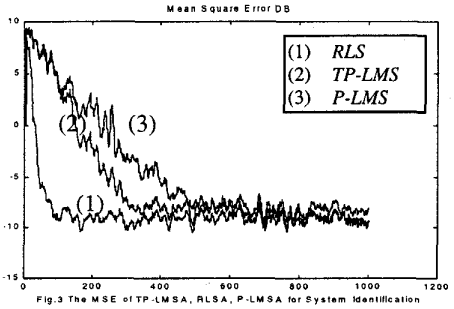

Fig.3 The MSE of TP-LMSA, RLSA, P-LMSA for System Identification

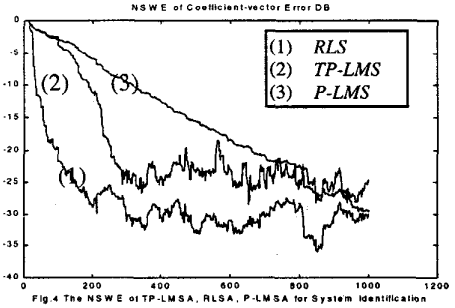

Fig.4 The NSWE of TP-LMSA, RLSA, P-LMSA for System Identification