FREQUENCY-DOMAIN REALIZATIONS OF ADAPTIVE PARALLEL-CASCADE QUADRATIC FILTERS

Linshan Li

College of Marine Engineering
Northwestern Polytechnical University
Xi'an 710072, P. R. China
e-mail: lili@eng.utah.edu

V. John Mathews

Department of Electrical Engineering
University of Utah
Salt Lake City, UT 84112, USA
e-mail: mathews@ee.utah.edu

ABSTRACT

Parallel-cascade realizations of truncated Volterra systems implement higher-order systems using a parallel connection of multiplicative combinations of lower-order systems. Such realizations are modular and permit efficient approximations of truncated Volterra systems. Frequency-domain realizations of the least-mean-square (LMS) adaptive filter and the normalized LMS adaptive filter that implements the system model using the parallel-cascade structure are presented in this paper. Computational complexity analysis and simulation results show that the normalized frequency-domain, parallel-cascade LMS adaptive quadratic filter has the advantages of computational simplicity and superior performance over direct form realizations.

1. INTRODUCTION

Truncated Volterra system models have found a variety of uses in adaptive filtering applications in the recent past. One difficulty with Volterra system models is the large number of coefficients they require to adequately characterize the behavior of many physical systems. Parallel-cascade realizations that implement higher-order Volterra systems as parallel combinations of multiplicative connections of lower-order systems were recently shown to be capable of approximating truncated Volterra systems efficiently [1, 2]. Stochastic gradient adaptive filters employing such structures were also presented in these papers. This paper introduces frequency-domain realizations of block-adaptive parallel-cascade truncated Volterra filters. A frequency-domain implementation of direct form adaptive Volterra filters is described in [3]. We demonstrate using a variety of experiments that frequency-domain realizations of parallel-cascade adaptive Volterra filters exhibit faster convergence speeds and lower computational complexity than their direct form counterparts.

For the sake of simplicity , all the derivations and experiments in this paper are done for homogeneous quadratic systems with input-output relationship

$$y(n) = \sum_{m_1=0}^{N-1} \sum_{m_2=0}^{N-1} h_2(m_1, m_2) x(n-m_1) x(n-m_2), \quad (1)$$

where $h_2(m_1, m_2)$ represents the coefficients of the quadratic kernel, N denotes the memory length in number of samples, and $x(n)$ and $y(n)$ are the input and output signals, respectively, of the system. We assume that the coefficients are symmetric so that $h_2(m_1, m_2) = h_2(m_2, m_1)$, and that the coefficients as well as the input signals are real-valued.

2. BLOCK ADAPTIVE REALIZATIONS

We consider a block-adaptive truncated quadratic filter whose coefficients for the lth input block is defined by the matrix $\mathbf{H}(l)$. We assume in our discussion that the block length $M = 2N$ and that the distance between the starting samples of adjacent segments is N samples. Let $h(i, j; l)$ denote the (i, j)th element of $\mathbf{H}(l)$. Because of the symmetry of coefficients $h(i, j; l)$, $\mathbf{H}(l)$ can be defined as a symmetric matrix. Then, the output of the adaptive filter during the lth block is given by

$$\hat{d}(lN+n) = \mathbf{x}^T(lN+n)\mathbf{H}(l)\mathbf{x}(lN+n) \; ; \; 0 \le n \le N-1 \; , \quad (2)$$

where the N-element input vector $\mathbf{x}(lN+n)$ is defined as

$$\mathbf{x}(lN+n) = [x(lN+n), \cdots, x(lN+n-N+1)]^T \; . \quad (3)$$

Let us express the coefficient matrix $\mathbf{H}(l)$ using singular value decomposition as

$$\mathbf{H}(l) = \sum_{i=1}^{r} \sigma_i(l) \mathbf{u}_i(l) \mathbf{u}_i^T(l) \; , \quad (4)$$

where r is the rank of $\mathbf{H}(l)$, $\sigma_i(l)$'s are its singular values and $\mathbf{u}_i(l)$'s are the corresponding singular vectors. Substituting (4) in (2) results in

$$\hat{d}(lN+n) = \sum_{i=1}^{r} \sigma_i(l) y_i^2(lN+n) \; , \quad (5)$$

where $y_i(lN+n) = \mathbf{u}_i^T(l)\mathbf{x}(lN+n)$ is the output of a linear filter whose coefficients are defined by the singular vector $\mathbf{u}_i(l)$. Thus, the output of a homogeneous quadratic filter can be obtained as a weighted sum of the squared outputs of r linear filters as shown in Figure 1. This structure is known as a parallel-cascade realization of the quadratic filter. One advantage of parallel-cascade systems is that reduced complexity approximations can be easily obtained by discarding the branches corresponding to lower values of the singular values of the coefficient matrix.

The coefficients of a block LMS adaptive filter that attempts to reduce the sum of the squared estimation errors

$$J_l = \sum_{n=0}^{N-1} \left(d(lN+n) - \sum_{i=1}^{r} \sigma_i(l) y_i^2(lN+n) \right)^2 \quad (6)$$

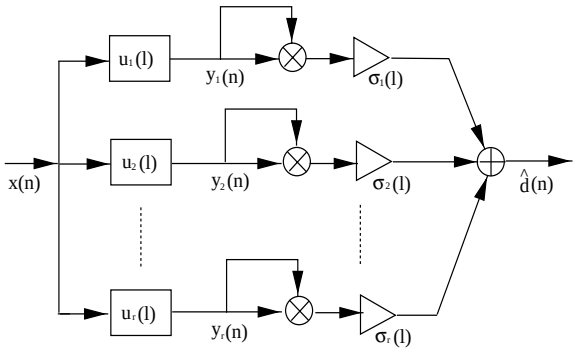

Figure 1: Block diagram of a parallel-cascade realization of a homogeneous quadratic Volterra filter.

during each iteration are updated as

$$\sigma_i(l+1) = \sigma_i(l) - \frac{\mu}{2}\frac{\partial J_l}{\partial \sigma_i(l)}$$

$$= \sigma_i(l) + \mu \sum_{n=1}^{N-1} e(lN+n) y_i^2(lN+n) \qquad (7)$$

and

$$\mathbf{u}_i(l+1) = \mathbf{u}_i(l) - \frac{\mu}{2}\frac{\partial J_l}{\partial \mathbf{u}_i(l)}$$

$$= \mathbf{u}_i(l) + 2\mu \sum_{n=0}^{N-1} \sigma_i(l) e(lN+n) \times y_i(lN+n)\mathbf{x}(lN+n)\ . \quad (8)$$

where μ is a positive step size.

The above update algorithm exhibits show slow and input-dependent convergence behavior similar to the characteristics of direct form LMS adaptive filters. In what follows, we derive a normalized filter that shows superior convergence behavior.

2.1. Normalized LMS Adaptive Parallel-Cascade Filters

Let $\sigma_i(l+1) = \sigma_i(l) + \triangle\sigma_i(l)$ and $\mathbf{u}_i(l+1) = \mathbf{u}_i(l) + \triangle\mathbf{u}_i(l)$ be the coefficients updated during the iterations on the ith branch for the lth block of data. We wish to choose $\triangle\sigma_i(l)$ and $\triangle\mathbf{u}_i(l)$ such that

$$d(lN+n) = \sum_{i=1}^{r} \left(\sigma_i(l) + \triangle\sigma_i(l)\right) \tilde{y}_i^2(lN+n);$$
$$n = 0, \ldots, N-1\ , \qquad (9)$$

where $\tilde{y}_i(lN+n) = \left(\mathbf{u}_i(l) + \triangle\mathbf{u}_i(l)\right)^T \mathbf{x}(lN+n)$. Our goal is to solve for $\triangle\sigma_i(l)$ and $\triangle\mathbf{u}_i(l)$ by minimizing the magnitude of the increments defined as $\sum_{i=1}^{r}\left(\|\triangle\sigma_i(l)\|^2 + \|\triangle\mathbf{u}_i(l)\|^2\right)$, subject to the equality of (9). For this purpose, we define a cost function

$$C(l) = \sum_{i=1}^{r}\left(\|\triangle\sigma_i(l)\|^2 + \|\triangle\mathbf{u}_i(l)\|^2\right) + \lambda(l) \times$$

$$\sum_{m=0}^{N-1}\left(d(lN+m) - \sum_{i=1}^{r}\left(\sigma(l) + \triangle\sigma_i(l)\right)\tilde{y}_i^2(lN+m)\right), \quad (10)$$

where $\lambda(l)$ is a Lagrange multiplier. We note here that we employ a single constraint in (10). Even though this constraint can be met by choices of coefficients that do not force the error values at each instant to be zero, this formulation results in a relatively simple adaptive filter structure. Experimental results presented later show that the adaptive filter that results from this approach performs in a superior manner to competing structures.

In order to solve for $\triangle\sigma_i(l)$ and $\triangle\mathbf{u}_i(l)$, we take the partial derivatives of $C(l)$ with respect to $\triangle\sigma_i(l)$ and $\triangle\mathbf{u}_i(l)$, and equate the results to zero to get

$$\frac{\partial C(l)}{\partial \triangle\sigma_i(l)} = 2\triangle\sigma_i(l) - \lambda(l)\sum_{m=0}^{N-1}\tilde{y}_i^2(lN+m) = 0 \qquad (11)$$

and

$$\frac{\partial C(l)}{\partial \triangle\mathbf{u}_i(l)} = 2\triangle\mathbf{u}_i(l) - 2\lambda(l)\sum_{m=0}^{N-1}\left(\sigma_i(l) + \triangle\sigma_i(l)\right) \times$$
$$\tilde{y}_i(lN+m)\mathbf{x}(lN+m) = 0, \qquad (12)$$

respectively.

Solving for $\triangle\sigma_i(l)$ and $\triangle\mathbf{u}_i(l)$ from (11) and (12) requires the simultaneous solution of a set of $2r$ coupled, nonlinear equations. In what follows, we employ an approximation similar to that used for deriving the time-domain realization of the normalized LMS adaptive parallel-cascade Volterra filter in [2]. We assume that the coefficients experience only small changes during the updates so that we can employ the approximate relationships $\sigma_i(l) \approx \sigma_i(l) + \triangle\sigma_i(l)$ and $\mathbf{u}_i(l) \approx \mathbf{u}_i(l) + \triangle\mathbf{u}_i(l)$ in appropriate parts of the derivation. Even though this approximation can be rigorously justified only during the final stages of adaptation, experimental results have shown that the resulting algorithm provides fast convergence behavior at the initial stages of operation of the adaptive filter also.

Multiplying (11) with $\tilde{y}_i^2(lN+n)$, adding $2\sigma_i(l)\tilde{y}_i^2(lN+n)$ to the result, then using $\sigma_i(l)$ and $\mathbf{u}_i(l)$ to approximate $(\sigma_i(l) + \triangle\sigma_i(l))$ and $(\mathbf{u}_i(l) + \triangle\mathbf{u}_i(l))$, respectively, on the right-hand-side of the resulting equations and adding the results over all i gives

$$2e(lN+n) = \lambda(l)\sum_{i=1}^{r}\sum_{m=0}^{N-1} y_i^2(lN+m)y_i^2(lN+n). \qquad (13)$$

The derivation of this step assumes that the constraint in (9) is satisfied by the solution. Adding (13) over all n in the block gives

$$2\sum_{n=0}^{N-1} e(lN+n) = \lambda(l)\sum_{i=1}^{r}\|\mathbf{y}_i(l)\|^4, \qquad (14)$$

where $\mathbf{y}_i(l) = [y_i(lN), y_i(lN+1), \ldots, y_i(lN+N-1)]^T$. In a similar manner, we can show that

$$\sum_{n=0}^{N-1} e(lN+n) = \lambda(l)\sum_{i=1}^{r}\sigma_i^2(l)\|\mathbf{X}(l)\mathbf{y}_i(l)\|^2, \qquad (15)$$

where $\mathbf{X}(l) = [\mathbf{x}(lN), \mathbf{x}(lN+1), \ldots, \mathbf{x}(lN+N-1)]$. Combining (14) with (15) and solving for $\lambda(l)$ gives

$$\lambda(l) = \frac{3P_e(l)}{P_\sigma(l) + P_u(l)}, \qquad (16)$$

where $P_e(l) = \sum_{n=0}^{N-1} e(lN+n)$, $P_\sigma(l) = \sum_{i=1}^{r}\|\mathbf{y}_i(l)\|^4$, and $P_u(l) = \sum_{i=1}^{r}\sigma_i^2(l)\|\mathbf{X}(l)\mathbf{y}_i(l)\|^2$.

Substituting (16) in (11) and (12) after invoking the slow coefficient variation approximation gives the desired solutions for $\triangle\sigma_i(l)$ and $\triangle\mathbf{u}_i(l)$. Employing the resulting expressions in the coefficient update equations may create large fluctuations in the coefficients. Consequently, the normalized block LMS adaptive

parallel-cascade filter changes the coefficients a fraction of the distance suggested by the solution to the constrained optimization problem. Thus, the coefficient updates are given by

$$\sigma_i(l+1) = \sigma_i(l) + \frac{1}{2}\mu\lambda(l)\sum_{m=0}^{N-1} y_i^2(lN+m) \qquad (17)$$

and

$$\mathbf{u}_i(l+1) = \mathbf{u}_i(l) + \mu\lambda(l)\sum_{m=0}^{N-1}\sigma_i(l)y_i(lN+m)\mathbf{x}(lN+m), \quad (18)$$

where μ is a positive step size value and $\lambda(l)$ is given by (16). Deriving a rigorous bound on μ for the stable operation of the adaptive filter is difficult. However, we can show using heuristic arguments that a choice of μ in the rang (0,2) should result in a stable algorithm. In general, we use much smaller values of the step size than the upper bound of 2.

2.2. Frequency-Domain Realizations

2.2.1. LMS Adaptive Parallel-Cascade Filter

We start our discussion by considering the update equations for the coefficients given by (7) and (8). We note that $\sigma_i(l)$ is a scalar element. There is no significant computational simplification possible by implementing (7) in the frequency-domain. Consequently, our algorithm implements the updates for the scaling factors in the time-domain, using the expression in (7).

Let $g_{i,l}(m)$ denote the mth element of the gradient vector of the lth segment

$$\frac{\partial J_l}{\partial \mathbf{u}_i(l)} = -4\sum_{n=0}^{N-1}\sigma_i(l)e(lN+n)y_i(lN+n)\mathbf{x}(lN+n)\ . \quad (19)$$

Then

$$g_{i,l}(m) = -4\sum_{n=0}^{N-1}\sigma_i(l)e(lN+n)y_i(lN+n)x(lN+n-m)\ . \qquad (20)$$

Using arguments similar to the derivation of the frequency-domain direct form adaptive quadratic filter [3], we can show that (8) can be implemented in the frequency-domain using the overlap-save technique in the following manner.

Let us define a windowed version of the product signal

$$\tilde{e}_{y_{i,l}}(n) = \begin{cases} 0; & 0 \le n \le N-1 \\ e((l-1)N+n)y_i((l-1)N+n); & N \le n \le 2N-1, \end{cases} \qquad (21)$$

and let $\tilde{E}_{y_{i,l}}(k)$ denote the $2N$-point DFT of $\tilde{e}_{y_{i,l}}(n)$. Then

$$g_{i,l}(m) = -4\sigma_i(l)\sum_{k=0}^{2N-1}\tilde{E}_{y_{i,l}}(k)X_l^*(k)e^{j(\frac{2\pi}{2N})km}\ ; \quad 0 \le m \le 2N-1\ , \qquad (22)$$

where $X_l^*(k)$ denotes the complex conjugate of $X_l(k)$. In a similar manner, let us define two windowed signals

$$\tilde{g}_{i,l}(m) = \begin{cases} g_{i,l}(m); & 0 \le m \le N-1 \\ 0; & N \le m \le 2N-1\ , \end{cases} \qquad (23)$$

and

$$\tilde{u}_{i,l}(m) = \begin{cases} u_{i,l}(m); & 0 \le m \le N-1 \\ 0; & N \le m \le 2N-1\ . \end{cases} \qquad (24)$$

Also, let $\hat{\nabla}_{g_{i,l}}(k)$ and $U_{i,l}(k)$ represent the $2N$-point DFTs of $\tilde{g}_{i,l}(m)$ and $\tilde{u}_{i,l}(m)$, respectively. We can obtain the frequency-domain implementation of the update equation as

Table 1: Frequency-domain LMS parallel-cascade realization of the adaptive quadratic filter.

Given: $\mathbf{x}_0 = [0,\cdots,0,x(0),\cdots,x(N-1)]^T$ $\mathbf{x}_l = [x(lN-N),\cdots,x(lN+N-1)]^T;\ l \ge 1$ $\mathbf{d}_l = [0,\cdots,0,d(lN),\cdots,d(lN+N-1)]^T$
Compute at block time l, $1 \le i \le r$: $X_l(k) = \text{DFT}\{\mathbf{x}_l\}$ $\tilde{\mathbf{y}}_{i,l} = [\tilde{y}_{i,l}(0),\cdots,\tilde{y}_{i,l}(2N-1)]^T$ $= \text{IDFT}\{X_l(k)U_{i,l}(k)\}$ $\mathbf{y}_{i,l}^{(2)} = [0,\cdots,0,\tilde{y}_{i,l}^2(N),\cdots,\tilde{y}_{i,l}^2(2N-1)]^T$ $\hat{\mathbf{d}}_l = [0,\cdots,0,\hat{d}(lN),\cdots,\hat{d}(lN+N-1)]^T$ $= \sum_{i=1}^{r}\sigma_i(l)\mathbf{y}_{i,l}^{(2)}$ $\mathbf{e}_l = \mathbf{d}_l - \hat{\mathbf{d}}_l = [0,\cdots,0,e(lN),\cdots,e(lN+N-1)]^T$ $\tilde{\mathbf{e}}_{y_{i,l}} = [0,\cdots,0,\tilde{e}_{y_{i,l}}(N),\cdots,\tilde{e}_{y_{i,l}}(2N-1)]^T$ $\tilde{E}_{y_{i,l}}(k) = \text{DFT}\{\tilde{\mathbf{e}}_{y_{i,l}}\}$ $\tilde{\mathbf{g}}_{i,l} = [\tilde{g}_{i,l}(0),\cdots,\tilde{g}_{i,l}(2N-1)]^T$ $= \text{IDFT}\left\{\sigma_i(l)\tilde{E}_{y_{i,l}}(k)X_l^*(k)\right\}$ $\mathbf{g}_{i,l} = [\tilde{g}_{i,l}(0),\cdots,\tilde{g}_{i,l}(N-1),0,\cdots,0]^T$ $\hat{\nabla}_{g_{i,l}}(k) = \text{DFT}\{\mathbf{g}_{i,l}\}$ $U_{i,l+1}(k) = U_{i,l}(k) + 2\mu\hat{\nabla}_{g_{i,l}}(k)$ $\hat{\nabla}_{\sigma_{i,l}} = \mathbf{e}_l^T\mathbf{y}_{i,l}^{(2)}$ $\sigma_i(l+1) = \sigma_i(l) + \mu\hat{\nabla}_{\sigma_{i,l}}$

$$U_{i,l+1}(k) = U_{i,l}(k) - \frac{\mu_u}{2}\hat{\nabla}_{g_{i,l}}(k)\ . \qquad (25)$$

The lth segment of the output sequence $y_i(lN+n)$ for the ith linear filter can be calculated by taking the $2N$-point inverse DFT of the product $X_l(k)U_{i,l}(k)$ and retaining the first N samples of the resulting sequence, *i.e.*,

$$y_i(lN+n) = \sum_{k=0}^{2N-1}X_l(k)U_{i,l}(k)e^{j(\frac{2\pi}{2N})kn}\ ; 0 \le n \le N-1\ . \qquad (26)$$

The complete set of equations describing the frequency-domain realization of the LMS adaptive parallel-cascade quadratic filter is given in Table 1. Assuming that computation the of L-point DFTs require $\frac{L}{2}log_2(L)$ complex multiplications, we can show that the complexity of implementing one block of the adaptive parallel-cascade quadratic filter in the frequency-domain corresponds to $(4rN+N)log_2(2N)+7rN+r/4$ complex multiplications. This number compares with $(8N^2+4N)log_2(2N)+10N^2+3N$ complex multiplications required by the frequency-domain direct form filter [3] and $\frac{1}{8}(3N^3+3N^2+2N)$ complex multiplications per block for the time-domain realization of the block-adaptive direct form filter. Since the maximum number of branches we need for systems with N-sample memory is $r = N$, the maximum complexity of the system is about half of that of the frequency-domain realization of the direct form filter. For implementations involving lower values of r, the complexity can be significantly lower.

2.2.2. NLMS Adaptive Parallel-Cascade Filter

The only significant difference between the normalized and unnormalized algorithms is that the normalized algorithm requires calculation of the normalization factor $\lambda(l)$. We can update the scalar elements $\sigma_i(l)$ and $\lambda(l)$ efficiently in the time-domain. Let $f_{i,l}(m)$

denote the mth element of the increment vector in (18) for the lth segment. Then

$$f_{i,l}(m) = \sigma_i(l) \sum_{k=0}^{2N-1} Y_{i,l}(k) X_l^*(k) e^{j(\frac{2\pi}{2N})km}; 0 \leq m \leq 2N-1, \tag{27}$$

where $Y_{i,l}(k) = X_l(k)U_{i,l}(k)$. Let us define a windowed signal

$$\tilde{f}_{i,l}(m) = \begin{cases} f_{i,l}(m); & 0 \leq m \leq N-1 \\ 0; & N \leq m \leq 2N-1. \end{cases} \tag{28}$$

Now, the coefficient can be updated as

$$U_{i,l+1}(k) = U_{i,l}(k) + \mu\lambda(l)\hat{\nabla}_{f_{i,l}}(k), \tag{29}$$

where $\hat{\nabla}_{f_{i,l}}(k)$ represents the $2N$-point DFT of $\tilde{f}_{i,l}(m)$.

3. EXPERIMENTAL RESULTS

The adaptive filters were used in a system identification problem in which the unknown system was a homogeneous quadratic filter whose coefficients were given by

$$h_2(i,j) = \frac{1.5}{2\pi[1.5^2 + (i-\frac{63}{2})^2 + (j-\frac{63}{2})^2]^{1.5}}; \ 0 \leq i,j \leq 63. \tag{30}$$

The adaptive Volterra filter was implemented using a homogeneous quadratic system model with 64-sample memory, and the data block size M was chosen to be 128, twice the system memory length. For the parallel-cascade realizations, the coefficients σ_i and the ith element of the linear filter coefficient vector $\mathbf{u}_i$ were initialized to 1 and all other coefficients were initialized to zero. The coefficients of the direct form LMS adaptive filter were initialized to the values corresponding to the initial coefficient values of the adaptive parallel-cascade filter. The desired response signal $d(n)$ was obtained by adding to the output of the unknown system a zero-mean, white, Gaussian noise that was uncorrelated with the input signal and with variance equal to 0.001. The results presented in this section are averages obtained over fifty independent runs. The convergence behavior of the normalized direct form LMS quadratic filter was much slower than that of its unnormalized counterpart, and therefore the results are not included in this paper. The input sequence was obtained as the output of a one-pole filter with transfer function $A(z) = 0.8/(1-0.6z^{-1})$ when its input was an i.i.d. Gaussian signal with zero mean value and variance equals to 0.1. Figure 2 shows the mean-square-error (MSE) between the adaptive filter output and the desired response signal for the LMS direct form, LMS parallel-cascade and the normalized LMS parallel-cascade adaptive filters. The step sizes for the three adaptive filters were selected in such a way that the steady-state excess mean-square estimation error was approximately the same for all three filters. The step sizes selected and the corresponding excess mean-square error values are given in Table 2. We see from the figure that the normalized LMS parallel-cascade realization converges much faster than the other two filters. Figure 3 shows the evolution of the mean-square error for the NLMS parallel-cascade filter employing three cases of 4, 16 and 64 branches in the system when a step size value of $\mu = 0.015$ was employed. All three curves exhibit similar convergence behaviors, but reach slightly different steady-state values. We observe from this result that much reduced complexity without a significant loss of performance can be achieved with parallel-cascade approximations for the system model in this experiment.

Table 2: Parameters and excess MSE in the Experiment.

Type	μ	excess MSE
LMS direct form	1.408E-05	7.881E-07
LMS Parallel-cascade	5.791E-04	8.055E-07
NLMS Parallel-cascade	1.500E-02	8.001E-07

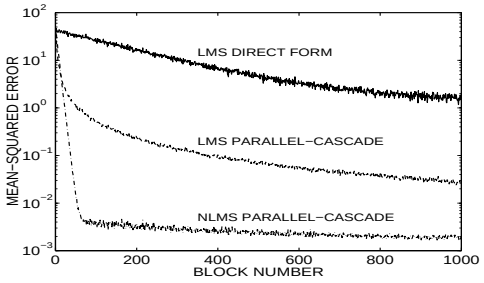

Figure 2: MSE of the system output.

4. CONCLUDING REMARKS

This paper presented frequency-domain realizations of adaptive parallel-cascade quadratic filters. The greatest advantage of these algorithms is their reduced computational complexity over the corresponding direct form realizations. The computational complexity of the parallel-cascade adaptive filter can be further reduced when the underlying system kernels have low rank or can be adequately represented using low-rank coefficient matrices. The efficiency of the frequency-domain parallel-cascade realizations as well as their performance should make adaptive Volterra filters attractive and feasible in many practical applications.

5. REFERENCES

[1] Y. Lou, C.L. Nikias and A.N. Venetsanopoulos, "Efficient VLSI Array Processing Structures for Adaptive Quadratic Digital Filters," *Circuits, Systems and Signal Processing*, Vol. 7, No. 2, pp. 253-273, Feb. 1988.

[2] T.M. Panicker, V. J. Mathews and G.L. Sicuranza, "Parallel-Cascade Adaptive Volterra Filters," *Proc. EUSIPCO'96*, pp. 1937-1940, Trieste, Italy, 1996.

[3] S. Im and E. J. Powers, "A Third-Order Frequency-Domain Adaptive Volterra Filter", *IEEE Signal Processing Letters*, Vol. 4, No. 3, pp.75-78, March 1997.

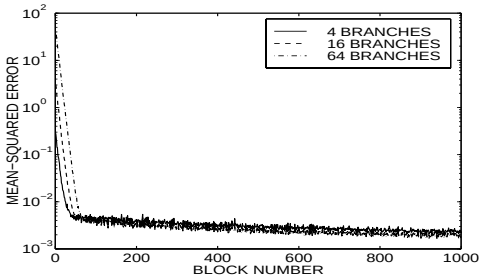

Figure 3: MSE for 4, 16, and 64 branches.