Electronic Transactions on Numerical Analysis.
Volume 4, pp. 14-36, March 1996.

ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

# LMS-NEWTON ADAPTIVE FILTERING USING FFT–BASED CONJUGATE GRADIENT ITERATIONS *

MICHAEL K. NG† AND ROBERT J. PLEMMONS‡

**Abstract.** In this paper, we propose a new fast Fourier transform (FFT) based LMS-Newton (LMSN) adaptive filter algorithm. At each adaptive time step $t$, the $n$th-order filter coefficients are updated by using the inverse of an $n$-by-$n$ Hermitian, positive definite, Toeplitz operator $T(t)$. By applying the cyclic displacement formula for the inverse of a Toeplitz operator, $T(t)^{-1}$ can be constructed using the solution vector of the Toeplitz system $T(t)\mathbf{u}(t) = \mathbf{e}_n$, where $\mathbf{e}_n$ is the last unit vector. We apply the FFT–based preconditioned conjugate gradient (PCG) method with the Toeplitz matrix $T(t-1)$ as preconditioner to solve such systems at the step $t$. As both matrix vector products $T(t)\mathbf{v}$ and $T(t-1)^{-1}\mathbf{v}$ can be computed by circular convolutions, FFTs are used throughout the computations. Under certain practical assumptions in signal processing applications, we prove that with probability 1 that the condition number of the preconditioned matrix $T(t-1)^{-1}T(t)$ is near to 1. The method converges very quickly, and the filter coefficients can be updated in $O(n \log n)$ operations per adaptive filter input. Preliminary numerical results are reported in order to illustrate the effectiveness of the method.

**Key words.** LMS-Newton adaptive filter algorithm, finite impulse response filter, Toeplitz matrix, circulant matrix, preconditioned conjugate gradient method, fast Fourier transform.

**AMS subject classification.** 65F10.

## 1. Introduction.

Adaptive *finite impulse response* (FIR) filters are used extensively in many signal processing and control applications: for instance, in system identification, equalization of telephone channels, spectrum analysis, noise cancellation, echo cancellation and in linear predictive coding [12] and [21]. The main concerns in the design of adaptive filter algorithms are their convergence performance and their computational requirements. These concerns are especially important when the filters are used in real-time signal processing applications or where the sizes of the filters are very large (as is the case in acoustic echo or active noise cancellation problems [12]).

The most popular adaptive filter algorithm is the well-known *Least Mean Square* (LMS) algorithm. It allows a simple implementation and requires only $O(n)$ operations for computing the filter coefficients per adaptive filter input, where $n$ is the size of the FIR filter [12]. However, a significant drawback of the LMS algorithm is that it is based on first order statistics, and therefore its convergence rate depends on the input signal spectrum. When the input signal process is white, good convergence performance is obtained. But when input signal process is highly colored, the LMS algorithm converges very slowly; see for instance Widrow [22, pp. 146-147].

In order to reduce the effect of the input signal spectrum on the convergence rate of the adaptive system, Gitlin and Magee [9] proposed an *LMS-Newton* (LMSN) adaptive filter algorithm. The approach is to use the second order statistics of the input signal to eliminate the dependence of the convergence of the LMS algorithm on the input signal process. To present the LMS-Newton (LMSN) algorithm properly,

* Received November 17, 1995. Accepted for publications March 16, 1996. Communicated by L. Reichel.

† Computer Sciences Laboratory, Research School of Information Sciences and Engineering, The Australian National University, Canberra ACT 0200, Australia. mng@cslab.anu.edu.au. This research was supported by the Cooperative Research Centre for Advanced Computational Systems.

‡ Department of Mathematics and Computer Science, Wake Forest University, Box 7388, Winston-Salem, NC 27109. This research was supported by the NSF under grant no. CCR–92–01105 and the U.S. Air Force Office of Scientific Research under grant no. F49620–94–1–0261.

ETNA
Kent State University
etna@mcs.kent.edu



we first introduce some notation from adaptive filter theory [12, p. 18]:

- discrete time index or step: $t$
- order of filter: $n$
- input sample scalar: $x(t)$
- input sample column vector: $\mathbf{x}(t) = [x(t), x(t-1), \cdots, x(t-n+1)]^T$
- filter coefficients column vector: $\mathbf{w}(t) = [w_1(t), w_2(t), \cdots, w_n(t)]^T$
- desired signal scalar: $d(t)$
- filter output scalar: $o(t) = \mathbf{w}(t)^*\mathbf{x}(t)$, where $*$ denotes the conjugate transpose.
- estimation error: $e(t) = d(t) - o(t) = d(t) - \mathbf{w}(t)^*\mathbf{x}(t)$

In the notation above, the LMS method, in its simplest form, for recursively updating the filter coefficients column vector $\mathbf{w}(t)$ can be expressed as

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)e(t)\mathbf{x}(t),$$

where $\mu(t)$ is a step size. The method is remarkable in its simplicity, but as noted earlier, slow convergence can often be a problem.

For the LMSN algorithm, the filter coefficients column vector $\mathbf{w}(t)$ is recursively updated by

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)e(t)T(t)^{-1}\mathbf{x}(t), \tag{1.1}$$

where $\mu(t)$ is again a step size, and now $T(t)$ is an estimate of the $n$-by-$n$ input signal autocorrelation matrix at time step $t$. In many signal processing applications, the input signal is generally assumed to come from a *wide sense stationary* (stationary up to the second order, see [12, p. 80]) stochastic process. It is well known that the corresponding autocorrelation matrix is Hermitian and Toeplitz, i.e. it is constant along diagonals; see for instance [12, p. 139]. Thus $T(t)$ is also Hermitian and Toeplitz in practical implementations of the LMSN algorithm. The basic and costly part of the LMSN algorithm is to compute the matrix-vector multiplication $T(t)^{-1}\mathbf{x}(t)$, or solve a Toeplitz system $T(t)\mathbf{z}(t) = \mathbf{x}(t)$, at each adaptive time step $t$. The Toeplitz structure of $T(t)$ allows one to find $\mathbf{z}(t)$ with direct methods that require many fewer operations than the $O(n^3)$ operations used in Gaussian elimination. Several direct methods (see for instance, Levinson, 1947 [14]) have been derived to solve such Toeplitz systems, and these methods require $O(n^2)$ operations. It follows that the computational requirement is $O(n^2)$ operations per adaptive filter input. The Toeplitz structure of the data matrix allows one to develop computationally efficient algorithms, which require only $O(n)$ operations per adaptive filter input. The numerical stability of these algorithms has always been in question [15, 20]. In particular, Luo and Qiao [15] have recently shown that the entire family of infinite memory fast recursive least squares algorithms is unstable when the forgetting factor, used to diminish the effects of the old data, is less than one. Numerical results given in [18] show that the fast transversal filter $O(n)$ operations algorithms do not converge when noise is added to the adaptive system.

The purpose of this paper is to propose a new fast Fourier transform (FFT) based LMSN adaptive filter algorithm with reasonable complexity and fast convergence. Based on the convergence performance of LMSN, our algorithm converges rapidly regardless of the input signal statistics. Moreover, the basic tool of our adaptive filter algorithm is the FFT. Since the FFT is highly parallelizable and has been implemented on multiprocessors efficiently [1, p.238], our algorithm can be expected to perform efficiently on a parallel machine for large-scale or real-time applications.

**ETNA**
Kent State University
etna@mcs.kent.edu



The sample autocorrelation matrices $T(t)$ are assumed to be positive definite at each adaptive time step. One can compute the inverse of $T(t)$ by solving a linear system

$$T(t)\mathbf{u}(t) = \mathbf{e}_n, \tag{1.2}$$

where $\mathbf{e}_n$ is the last unit vector. By using the solution vector $\mathbf{u}(t)$ in (1.2), Ammar and Gader [2] showed that there exists a circulant matrix $B_1(t)$ and a skew-circulant matrix $B_2(t)$ such that

$$T(t)^{-1} = \frac{1}{2[\mathbf{u}(t)]_n}[B_2(t)B_1(t)^* + B_2(t)^*B_1(t)], \tag{1.3}$$

where $[\cdot]_n$ denotes the last entry of $n$-vector, (see §2.2). The equation (1.3) is called by Ammar and Gader the cyclic displacement representation of $T(t)^{-1}$. The main problem left is how to compute $\mathbf{u}(t)$ efficiently at each adaptive time step. Our strategy is to apply the preconditioned conjugate gradient (PCG) method to solve the linear system (1.2). It is well known that the convergence performance of the conjugate gradient method depends on the spectrum and, in particular, on the condition number of the coefficient matrix. If the condition number of the coefficient matrix is near 1, convergence will be rapid. Thus, to make the conjugate gradient method a useful iterative method (converge rapidly), one *preconditions* the system. In our case, as the cyclic representation of $T(t-1)^{-1}$ has already been obtained at the time step $t-1$, we therefore use $T(t-1)$ as preconditioner. That means, instead of solving the original system (1.2), one solves the preconditioned system by the conjugate gradient method at the time step $t$.

The outline of the paper is as follows. In §2, we formulate the updating computations of $T(t)$ and introduce our FFT–based preconditioners. In §3, we present our FFT–based LMSN adaptive filtering algorithm and analyze the convergence rate of the PCG method probabilistically. In §4, numerical experiments on some adaptive filtering simulation models are reported to illustrate the effectiveness of our algorithm. Finally, some concluding remarks are given in §5.

**2. Updating in the LMSN Algorithm.** Here we consider the case where input signal comes from a discrete-time complex-valued process. Since $T(t)$ is an Hermitian Toeplitz matrix, it is completely determined by its first column. Let the first column of $T(t)$ be denoted by

$$[\gamma_0(t), \gamma_1(t), \ldots, \gamma_{n-1}(t)]^T.$$

The parameter $\gamma_k(t)$, $0 \le k \le n-1$, is the estimate of the $k$th-lag autocorrelation of the input signal process at the time step $t$ and also $\gamma_0(t)$ is real. In practical situations, there is no prior knowledge of the autocorrelations of the input signal process. In this case, the autocorrelations are estimated from the finite number of input signal samples received up to the time step $t$, i.e. $\{x(j)\}_{j=1}^{t}$. In the following discussion, we consider the estimates $\{\gamma_k(t)\}_{k=0}^{n-1}$ constructed from the convolution of input data samples given by

$$\gamma_k(t) = \sum_{j=1}^{t-k} \frac{1}{t}\overline{x(j)}x(j+k), \quad k = 0, 1, \ldots, n-1. \tag{2.1}$$

In signal processing terminology, the correlation windowing method is used, and the data samples prior to $j = 0$ and after $j = t$ are assumed to be zero at time step $t$ [12,

**ETNA**
Kent State University
etna@mcs.kent.edu



p. 373]. We remark that the correlation windowing method always leads to a positive semi-definite Toeplitz matrix, see for example, Ng and Chan [17]. If the input signal process is stationary, $\gamma_k(t)$ is the common estimator of the $k$th-lag autocorrelation of input stationary process in the time-series literature. Assuming stationarity, we remark that $\gamma_k(t)$ has a smaller mean square error than other estimators, see Priestley [19, p. 322].

Adaptive filter algorithms are often used to process signals that result from time-varying environments. The estimates of the autocorrelations are often obtained by limiting the filter memory. A useful technique is an exponential data weighting infinite memory method controlled by a forgetting factor $\alpha$, with $0 < \alpha \leq 1$ [12, p. 478]. Roughly speaking, the inverse of $1-\alpha$ is a measure of the memory of the adaptive filter algorithm. For the construction of the estimates of the autocorrelations, we consider the exponentially weighted data samples $\{\alpha^{(t-1)/2}x(1), \alpha^{(t-2)/2}x(2), \ldots, x(t)\}$ instead of $\{x(1), x(2), \ldots, x(t)\}$.

**2.1. Updating Computations for $T(t)$.** In adaptive systems, data samples arrive continuously. It is necessary to update the autocorrelations from time $t-1$ to time $t$. Using (2.1) with forgetting factor $\alpha$, the relation between $\gamma_k(t)$ and $\gamma_k(t-1)$ is given by

$$\gamma_k(t) = \frac{(t-1)\alpha}{t}\gamma_k(t-1) + \frac{\alpha^{k/2}}{t}x(t)\overline{x(t-k)}, \quad k = 0, 1, 2 \ldots, n-1, \tag{2.2}$$

see for instance Widrow [22, p. 148]. In matrix form, the sample autocorrelation matrix $T(t)$ can be written as follows:

$$T(t) = \frac{(t-1)\alpha}{t}T(t-1) + \frac{x(t)}{t}V_T(t), \quad t \geq 2, \tag{2.3}$$

where $V_T(t)$ is a Toeplitz matrix with first column given by

$$\left[\overline{x(t)}, \alpha^{1/2}\overline{x(t-1)}, \cdots, \alpha^{(n-2)/2}\overline{x(t-n+2)}, \alpha^{(n-1)/2}\overline{x(t-n+1)}\right].$$

In each iteration of the PCG method, a matrix-vector multiplication $T(t)\mathbf{v}$ is required. In general, the matrix-vector product can be done in $O(n^2)$ operations. However, we can embed $T(t)$ into the $2n$-by-$2n$ circulant matrix:

$$C(t) = \begin{bmatrix} T(t) & S(t) \\ S(t) & T(t) \end{bmatrix}.$$

Here $S(t)$ is so constructed such that $C(t)$ is a circulant matrix. The first column of $C(t)$ is given by

$$[\gamma_0(t), \gamma_1(t), \ldots, \gamma_{n-1}(t), 0, \overline{\gamma_{n-1}(t)}, \ldots, \overline{\gamma_1(t)}]^T. \tag{2.4}$$

The matrix $C(t)$ can be diagonalized by using the discrete Fourier matrix $F_{2n}$ with entries given by $[F_{2n}]_{j,k} = \frac{1}{\sqrt{2n}}e^{-2\pi ijk/2n}$. The spectral decomposition of $C(t)$ is given by

$$C(t) = F_{2n}\Lambda(t)F_{2n}^*, \tag{2.5}$$

where $\Lambda(t)$ is a $2n$-by-$2n$ diagonal matrix holding the eigenvalues of $C(t)$. We see that if $\mathbf{e}_1$ and $\mathbf{1}_{2n}$ denote the first unit vector and the $2n$-vector of all ones respectively, then the eigenvalues of $C(t)$ are related to its first column. We have

$$\Lambda(t)\mathbf{1}_{2n} = \sqrt{2n}F_{2n}^*C(t)\mathbf{e}_1. \tag{2.6}$$

It follows that the matrix-vector product $T(t)\mathbf{v}$ can be computed by using the FFT in a total of $O(n \log n)$ operations, by first embedding $T(t)$ into $C(t)$.

Instead of computing the eigenvalues of $C(t)$ explicitly at each adaptive time step, we directly update $\Lambda(t)$ from $\Lambda(t-1)$. The updating scheme for the eigenvalues of $C(t)$ is stated in Algorithm 1 below. Before we begin, we define $n$-vectors $\mathbf{f}_1(t)$ and $\mathbf{f}_2(t)$ by

$$\mathbf{f}_1(t) \equiv [\overline{x(t)}, \alpha^{1/2}\overline{x(t-1)}, \ldots, \alpha^{(n-1)/2}\overline{x(t-n+1)}]^T, \tag{2.7}$$

and

$$\mathbf{f}_2(t) \equiv [0, \alpha^{(n-1)/2}x(t-n+1), \ldots, \alpha^{3/2}x(t-2), \alpha^{1/2}x(t-1)]^T, \tag{2.8}$$

at each time step $t$. We let $F_{2n}^*[k:j]$ denote the sub-matrix formed from the $k$th column to the $j$th column of the discrete Fourier matrix $F_{2n}^*$ where $j \geq k$.

**Algorithm 1: Updating the Eigenvalues $\Lambda(t)$.** *Given $\Lambda(t-1)$ and a new input signal sample $x(t)$, we let $\mathbf{g}_1(t)$ and $\mathbf{g}_2(t)$ be the following $n$-vectors*

$$\mathbf{g}_1(t-1) = F_{2n}^*[1:n]\mathbf{f}_1(t-1) \quad \text{and} \quad \mathbf{g}_2(t-1) = F_{2n}^*[n+1:2n]\mathbf{f}_2(t-1).$$

- **Compute**

$$\mathbf{g}_1(t) = \overline{x(t)}F_{2n}^*[1:1] + \alpha^{1/2}\left(\mathbf{g}_1(t-1) - F_{2n}^*[n:n]\alpha^{(n-1)/2}\overline{x(t-n+2)}\right) \odot F_{2n}^*[2:2],$$

and

$$\begin{aligned}\mathbf{g}_2(t) &= \alpha^{1/2}\left(\mathbf{g}_2(t-1) - F_{2n}^*[n+2:n+2]\alpha^{(n-1)/2}x(t-n+2)\right) \odot F_{2n}^*[2n:2n] + \\ &\quad \alpha^{1/2}x(t-1)F_{2n}^*[2n:2n],\end{aligned}$$

where $\odot$ denotes element–wise multiplication of two $2n$-vectors.

- **Update**

$$\Lambda(t)\mathbf{1}_{2n} = \frac{(t-1)\alpha}{t}\Lambda(t-1)\mathbf{1}_{2n} + \frac{\sqrt{2n}x(t)}{t}\left[\mathbf{g}_1(t) + \mathbf{g}_2(t)\right]. \tag{2.9}$$

As for the validity of Algorithm 1, it is suffices to establish the following Lemma.

LEMMA 2.1. *Let $\mathbf{g}_1(t)$ and $\mathbf{g}_2(t)$ be computed in Algorithm 1. Then we have*

$$\mathbf{g}_1(t) = F_{2n}^*[1:n]\mathbf{f}_1(t) \quad \text{and} \quad \mathbf{g}_2(t) = F_{2n}^*[n+1:2n]\mathbf{f}_2(t).$$

*Proof.* We let $Z_n$ denote the $n$-by-$n$ downshift matrix.

$$\begin{aligned}\mathbf{g}_1(t) &= \alpha^{1/2}\left(F_{2n}^*[1:n]\mathbf{f}_1(t-1) - F_{2n}^*[n:n]\alpha^{(n-1)/2}\overline{x(t-n+2)}\right) \odot F_{2n}^*[2:2] + \\ &\quad \overline{x(t)}F_{2n}^*[1:1] \\ &= \overline{x(t)}F_{2n}^*[1:1] + \alpha^{1/2}Z_nF_{2n}^*[1:n]\mathbf{f}_1(t-1) \odot F_{2n}^*[2:2] = F_{2n}^*[1:n]\mathbf{f}_1(t).\end{aligned}$$

By using a similar argument, we can prove $\mathbf{g}_2(t) = F_{2n}^*[n+1:2n]\mathbf{f}_2(t)$. □

By using (2.2), (2.4), (2.6), (2.7), (2.8) and Lemma 1, we obtain

$$
\begin{aligned}
\Lambda(t)\mathbf{1}_{2n} = \sqrt{2n}F_{2n}^{*}C(t)\mathbf{e}_1 &= \frac{\sqrt{2n}(t-1)\alpha}{t}F_{2n}^{*}C(t-1)\mathbf{e}_1 \\
&\quad + \frac{\sqrt{2n}x(t)}{t}F_{2n}^{*}\begin{pmatrix} \mathbf{f}_1(t) \\ \mathbf{f}_2(t) \end{pmatrix} \\
&= \frac{(t-1)\alpha}{t}\Lambda(t-1)\mathbf{1}_{2n} \\
&\quad + \frac{\sqrt{2n}x(t)}{t}[\mathbf{g}_1(t)+\mathbf{g}_2(t)]
\end{aligned} \tag{2.10}
$$

It follows that the eigenvalues of $C(t)$ can easily be updated from time step $t-1$ to $t$. As for the storage requirement, the vectors $\mathbf{g}_1(t)$, $\mathbf{g}_2(t)$ and $\Lambda(t)\mathbf{1}_{2n}$ are needed for the updating step. The total cost of Algorithm 1 is $O(n)$ operations. We remark that in a parallel processing environment with $O(n)$ processors, the complexity of Algorithm 1 is reduced to $O(1)$ time steps, *i.e.* the number of time steps is bounded independently of $n$.

**2.2. FFT–based Preconditioners.** In this subsection, we explain the choice of the Toeplitz matrix $T(t-1)$ over other preconditioners in the preconditioned conjugate gradient iterations for solving $T(t)\mathbf{u}(t) = \mathbf{e}_n$. We remark that, in general, a preconditioner $P$ should be chosen to satisfy the following criteria:

(P1) The inverse of $P$ should be easy to compute.

(P2) The matrix-vector product $P^{-1}\mathbf{v}$ should be easy to form for a given vector $\mathbf{v}$.

(P3) The condition number of the preconditioned matrix should be close to 1, and/or the spectrum should be clustered around 1.

In recent years, the use of circulant matrices or the inverses of Toeplitz matrices as preconditioners for solving Toeplitz systems $T\mathbf{z} = \mathbf{v}$ has been proposed and studied extensively; see for instance Chan and Strang [5], Ku and Kuo [13], T. Chan [6], and Chan and Ng [4]. They used circulant or Toeplitz matrices to approximate $T^{-1}$. In many practical situations, the spectrum of the preconditioned matrix is clustered around 1. In such cases, the PCG method can be shown to converge superlinearly, see Chan and Strang [5]. However, the inverse of a Toeplitz matrix is non-Toeplitz in general. Thus circulant matrices or the inverse of Toeplitz matrices may not always be good preconditioners for solving Toeplitz systems. We note from (2.3) that $V_T(t)$ is a matrix with small norm when $t$ is sufficiently large. Thus we expect that the condition number of our preconditioned matrices will be close to 1 (c.f. (P3)). A detailed convergence analysis will be given in §3.1.

To acheive (P1), we wish to construct the inverse of $T(t-1)$ easily. According to formula (1.3), the inverse of $T(t-1)$ can be constructed efficiently by using the solution vector $\mathbf{u}(t-1)$ of the Toeplitz system $T(t-1)\mathbf{u}(t-1) = \mathbf{e}_n$. We remark that, in the preconditioned conjugate gradient iterations, the solution vector $\mathbf{u}(t-1)$ is computed up to a given accuracy. Therefore, we employ a certain approximation $\tilde{T}(t-1)$ of $T(t-1)$ as a preconditioner to solve $T(t)\mathbf{u}(t) = \mathbf{e}_n$ at time step $t$. Given the computed solution vector

$$\tilde{\mathbf{u}}(t-1) = [\tilde{u}_0(t-1), \tilde{u}_1(t-1), \ldots, \tilde{u}_{n-1}(t-1)]^T,$$

we use a cyclic displacement formula that expresses the inverse of our preconditioner

$\tilde{T}(t-1)^{-1}$ in the following form:

$$\tilde{T}(t-1)^{-1} = \frac{1}{2\tilde{u}_{n-1}(t-1)}[\tilde{B}_2(t-1)\tilde{B}_1(t-1)^* + \tilde{B}_2(t-1)^*\tilde{B}_1(t-1)], \tag{2.11}$$

where $\tilde{B}_1(t-1)$ is a circulant matrix with its first row given by

$$[\tilde{u}_{n-1}(t-1), \overline{\tilde{u}_0(t-1)}, \ldots, \overline{\tilde{u}_{n-3}(t-1)}, \overline{\tilde{u}_{n-2}(t-1)}]$$

and $\tilde{B}_2(t-1)$ is a skew-circulant matrix with its first row given by

$$[\tilde{u}_{n-1}(t-1), -\overline{\tilde{u}_0(t-1)}, \ldots, -\overline{\tilde{u}_{n-3}(t-1)}, -\overline{\tilde{u}_{n-2}(t-1)}].$$

Since $T(t-1)$ is Hermitian positive definite matrix, we can assume that the component $\tilde{u}_{n-1}(t-1)$ is real and positive. By direct verification, we have the following lemma about the matrices that are constructed by using a cyclic displacement formula.

LEMMA 2.2. *Let $\mathbf{q}$ be any $n$-vector with $[\mathbf{q}]_1$ a real number. If the first column of both circulant matrix $Q_1$ and skew-circulant matrix $Q_2$ is given by $\mathbf{q}$, then $Q_2Q_1^* + Q_2^*Q_1$ is a Hermitian matrix.*

It follows from Lemma 2.2 that $\tilde{T}(t-1)^{-1}$ is a Hermitian matrix. As for the cost of the matrix-vector product $\tilde{T}(t-1)^{-1}\mathbf{v}$, both circulant and skew-circulant matrix-vector products can be done efficiently by using the n-dimensional FFT, in $O(n\log n)$ operations (c.f. (P2)).

In [10], Gohberg and Semencul presented the displacement formulas for the decomposition of the inverse of a Toeplitz matrix $T$,

$$T^{-1} = LL^* - U^*U, \tag{2.12}$$

where $L$ and $U$ are lower triangular and upper triangular Toeplitz matrices respectively. Using (2.12), one embeds the lower and upper triangular Toeplitz matrices into an $2n$-by-$2n$ circulant matrices and then uses the $2n$-dimensional FFT to compute $\tilde{T}(t-1)^{-1}\mathbf{v}$. This approach is more expensive than our method of using the cyclic displacement formula for computing $\tilde{T}(t-1)^{-1}\mathbf{v}$. The method only involves the $n$-dimensional FFT, see Ammar and Gader [2] for details of the approach.

In practice, we solve the scaled preconditioned system

$$\hat{\alpha}T(t-1)^{-1}T(t)\mathbf{u}(t) = \hat{\alpha}T(t-1)^{-1}\mathbf{e}_n, \tag{2.13}$$

where $\hat{\alpha} = \frac{t}{(t-1)\alpha}$ by the conjugate gradient iterations at each step $t$.

## 3. FFT–based LMSN Algorithm.

In this section, we summarize our FFT–based LMSN adaptive filter algorithm. Figure 1 displays the block diagram of the algorithm. We take the initial filter coefficient vector $\mathbf{w}(1)$ and the initial guess $\mathbf{u}(0)$ to be zero vectors, and we also assume that $x(1) \neq 0$. Thus, we have $\mathbf{x}(1) = [x(1), 0, \ldots, 0]^T$.

**Algorithm 2: FFT–based LMSN Adaptive Filter Algorithm**. *For* $t = 1, 2, 3, \ldots$

- **Compute** the estimation error $e(t) = d(t) - \mathbf{w}(t)^*\mathbf{x}(t)$.
- **Update** $\Lambda(t)$ in (2.9) using Algorithm 1.
- **Apply** the conjugate method to solving the preconditioned system represented as in (2.13), with starting initial guess $\mathbf{u}(t-1)$.
- **Generate** the cyclic displacement representation of $\tilde{T}(t)^{-1}$ using formula (1.3).
- **Update** $\mathbf{w}(t)$ by $\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)e(t)\tilde{T}(t)^{-1}\mathbf{x}(t)$.

**ETNA**
Kent State University
etna@mcs.kent.edu



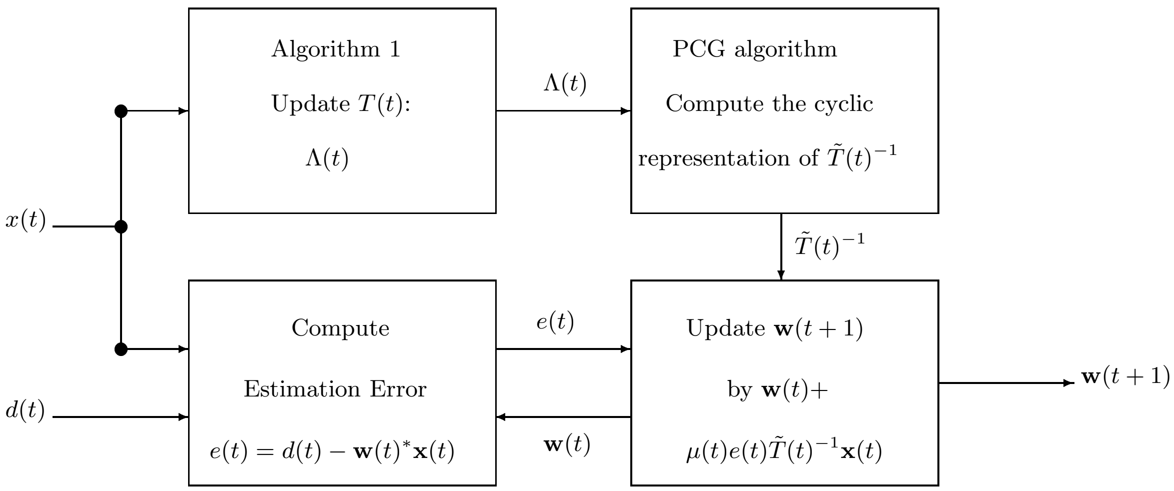


FIG. 1. *Block Diagram of the FFT–based LMSN Algorithm.*

**3.1. Convergence Analysis of the PCG Iterations.** In this section, we analyze the convergence rate of the preconditioned conjugate gradient method at each adaptive time step $t$. In the following, we assume that the computed solution vector $\tilde{\mathbf{u}}(t)$ is represented as

$$\tilde{\mathbf{u}}(t) = \mathbf{u}(t) + \mathbf{y}(t), \tag{3.1}$$

where

$$\mathbf{y}(t) = [y_0(t), y_1(t), \ldots, y_{n-1}(t)]^T.$$

Our first theorem gives the perturbation $\tilde{T}(t)^{-1}$ of $T(t)^{-1}$ when the computed solution vector $\tilde{\mathbf{u}}(t)$ is used to construct $\tilde{T}(t)^{-1}$.

THEOREM 3.1. *Let $\tilde{\mathbf{u}}(t)$ be given by (3.1) and define the error matrix $E(t)$ by*

$$E(t) = \tilde{T}(t)^{-1} - T(t)^{-1} \tag{3.2}$$

*at time step $t$. Then $E(t)$ is Hermitian and, moreover, if $\|\mathbf{y}(t)\|_2\|T(t)\|_2 < 1$, then*

$$\|E(t)\|_2 \leq \frac{\|\mathbf{y}(t)\|_2\|T(t)\|_2}{1 - \|\mathbf{y}(t)\|_2\|T(t)\|_2}\left[4n\|\mathbf{y}(t)\|_2 + 8n\|T(t)^{-1}\|_2 + \frac{\kappa_2(T(t))}{2}\right], \tag{3.3}$$

*where $\kappa_2(T(t))$ is the spectral condition number of $T(t)$.*

The proof of Theorem 3.1 can be found in the Appendix. As $T(t)^{-1}$ is positive definite, it follows from Theorem 3.1 that if $E(t)$ is sufficiently small then $\tilde{T}(t)^{-1}$ is also positive definite. We note from (3.3) that the error matrix $E(t)$ depends on $\|\mathbf{y}(t)\|_2$, and the maximum and minimum eigenvalues of $T(t)$. Thus the sensitivity of the $\ell_2$ norm of the error matrix $E(t)$ is initially determined by the conditioning of $T(t)$.

As we deal with signal data samples from random input signal processes, the convergence rate is considered in a probabilistic way, which is quite different from the deterministic case. We first make the following assumption on the input signal processes so that the results of the convergence rate can be derived.

**Assumption (A):** Assume that there exist constants $C_i$ independent of $t$ such that

(**A1**) $\left|\mathcal{E}\left(x(t)\overline{x(t-k)}\right)\right| < C_1, \quad k = 0, 1, \ldots, n-1.$

(**A2**) $\mathrm{Var}\left(x(t)\overline{x(t-k)}\right) < C_2, \quad k = 0, 1, \ldots, n-1.$

(**A3**) $\|T(t)\|_2 < nC_3 \quad \text{and} \quad \|T(t)^{-1}\|_2 < C_4.$

Here we consider these assumptions when the input signal processes under consideration are wide sense stationary (stationary up to the second order, see [12, p. 80]). *Autoregressive processes* (AR), *moving-average processes* (MA) and *autoregressive and moving-average processes* (ARMA) are commonly used wide sense stationary input processes in many signal processing applications [12, pp. 96–113].

1. The assumption (**A1**) is often true for a stationary input signal process. In the stationary environment, the $k$-th lag autocovariance function $r_k$ of the input process is given by

$$r_k = \mathcal{E}\left((x(t)-\beta)\overline{(x(t-k)-\beta)}\right), \tag{3.4}$$

where $\beta$ is the mean of the stationary input process, see Haykin [12, p. 79]. Equation (3.4) implies that

$$\mathcal{E}\left(x(t)\overline{x(t-k)}\right) = \beta\overline{\beta} + r_k. \tag{3.5}$$

As $|r_k|$ is always less than the variance $r_0$ of the input process, the constant $C_1$ in (**A1**) can be set to be $|\beta|^2 + r_0$ depending on the input stationary process.

2. The variance of $x(t)\overline{x(t-k)}$ is given by

$$\mathrm{Var}\left(x(t)\overline{x(t-k)}\right) = \mathcal{E}\left(x(t)\overline{x(t-k)x(t)}x(t-k)\right) - (\beta\overline{\beta} + r_k)(\beta\overline{\beta} + \overline{r_k}).$$

Assumption (**A2**) is satisfied when the input signal process is Gaussian process (see Haykin [12, p. 109] for definition). The variance of $x(t)\overline{x(t-k)}$ is bounded by

$$\mathrm{Var}\left(x(t)\overline{x(t-k)}\right) \le 40|\beta|^4 + 12|\beta|^2 r_0 + 3r_0^2. \tag{3.6}$$

The proof of (3.6) can be found in the Appendix.

3. Assumption (**A3**) is satisfied when the underlying spectral density function $f(\theta)$ of the input stationary process is positive and in the Wiener class, i.e. the autocovariances $\{r_k\}_{k=-\infty}^{\infty}$ of the process are absolutely summable, $\sum_{k=0}^{\infty}|r_k| \le \infty$. We remark that the spectral density function $f(\theta)$ is always non-negative. We have the following lemma about the smallest and largest eigenvalues of $T(t)$, proved in Ng [16] when the forgetting factor $\alpha$ is equal to 1.

Lemma 3.2. *Let the spectral density function $f(\theta)$ of the input stationary process be in the Wiener class, and let the mean of the process be $\beta$. Then for any given $\epsilon > 0$ and $0 < \delta < 1$, there exists a positive integer $N$ such that for $n > N$ (where $n$ is size of $T(t)$),*

$$\Pr\{\lambda_{\min}(T(t)) \ge f_{\min} - \epsilon\} \ge 1 - \delta$$

*and*

$$\Pr\left\{\lambda_{\max}(T(t)) \le f_{\max} + n\beta^2 + \epsilon\right\} \ge 1 - \delta,$$

*for sufficiently large $t$ where $f_{\min}$ and $f_{\max}$ are the minimum and maximum values of $f(\theta)$.*

Before beginning the convergence analysis, we first denote $\mathcal{E}(Y)$ as the expected value of a random matrix $Y$, where the entries of $\mathcal{E}(Y)$ are the expected values of the elements of $Y$, i.e. the $(j,k)$th entry of $\mathcal{E}(Y)$ is given by $[\mathcal{E}(Y)]_{j,k} = \mathcal{E}([Y]_{j,k})$. The following Lemma will be useful later in the analysis of the convergence rate of the method.

LEMMA 3.3. *Let the input signal process satisfy assumption* (**A2**). *Then for any given $\epsilon > 0$, we have*

$$\Pr\left\{\left|\sum_{k=0}^{n-1}\left[\frac{x(t)\overline{x(t-k)}}{t} - \mathcal{E}\left(\frac{x(t)\overline{x(t-k)}}{t}\right)\right]\right| \le \epsilon\right\} \ge 1 - \frac{n^3 C_2}{\epsilon^2 t^2}.$$

*Proof.* By using a Lemma in Fuller [8, p. 182] and Chebyshev's inequality [8, p. 185], we obtain

$$\begin{aligned} &\Pr\left\{\left|\sum_{k=0}^{n-1}\left[\frac{x(t)\overline{x(t-k)}}{t} - \mathcal{E}\left(\frac{x(t)\overline{x(t-k)}}{t}\right)\right]\right| \ge \epsilon\right\} \\ \le\ &\sum_{k=0}^{n-1}\Pr\left\{\left|\frac{x(t)\overline{x(t-k)}}{t} - \mathcal{E}\left(\frac{x(t)\overline{x(t-k)}}{t}\right)\right| \ge \frac{\epsilon}{n}\right\} \le \sum_{k=0}^{n-1}\frac{\operatorname{Var}(x(t)\overline{x(t-k)})n^2}{\epsilon^2 t^2} \le \frac{n^3 C_2}{\epsilon^2 t^2}. \end{aligned}$$

□

As both matrices $\tilde{T}(t-1)$ and $T(t)$ are Hermitian positive definite, the preconditioned matrices $\hat{\alpha}\tilde{T}(t-1)^{-1/2}T(t)\tilde{T}(t-1)^{-1/2}$ are similar to the matrices $\hat{\alpha}T(t)^{1/2}\tilde{T}(t-1)^{-1}T(t)^{1/2}$. Next, we prove that the condition number of the matrix $\hat{\alpha}T(t)^{1/2}\tilde{T}(t-1)^{-1}T(t)^{1/2}$ is close to 1, with probability 1.

THEOREM 3.4. *Let the input signal process satisfy assumption* (**A**). *If*

$$||E(t-1)||_2 \le \frac{\epsilon_1}{nC_3} < 1, \tag{3.7}$$

*then for any given $0 < \epsilon < 1 - \epsilon_1$ and $0 < \delta < 1$, there exists an integer $t_0$, which depends upon $n$, $\epsilon$, $\delta$, $\alpha$, $C_1$, $C_2$, $C_3$ and $C_4$, such that for $t \ge t_0$,*

$$\Pr\left\{\kappa_2(\hat{\alpha}T(t)^{1/2}\tilde{T}(t-1)^{-1}T(t)^{1/2}) \le \frac{1+\epsilon_1+\epsilon}{1-\epsilon_1-\epsilon}\right\} > 1 - \delta.$$

*Proof.* By (3.2), the matrix $\hat{T}(t)$ can be written as follows:

$$\begin{aligned} \hat{T}(t) &= \hat{\alpha}\Big[I_n + T(t)^{1/2}E(t-1)T(t)^{1/2} + \\ &\qquad T(t)^{1/2}\left(T(t-1)^{-1} - T(t)^{-1}\right)T(t)^{1/2}\Big]. \end{aligned} \tag{3.8}$$

Next we estimate the $\ell_2$ norm of the matrices $T(t)^{1/2}E(t-1)T(t)^{1/2}$ and $T(t)^{1/2}(T(t-1)^{-1}-T(t)^{-1})T(t)^{1/2}$ in the right hand side of (3.8). From (**A3**) and the hypothesis on

**ETNA**
Kent State University
etna@mcs.kent.edu



$\|E(t-1)\|_2$, the $\ell_2$ norm of the matrix $T(t)^{1/2}E(t-1)T(t)^{1/2}$ is bounded by $\epsilon_1$. As both $T(t)$ and $T(t-1)$ are Hermitian positive definite matrices, the matrix $T(t)^{1/2}(T(t-1)^{-1}-T(t)^{-1})T(t)^{1/2}$ is similar to the matrix $T(t-1)^{-1/2}(T(t)-T(t-1))T(t-1)^{-1/2}$. Therefore it suffices to estimate the $\ell_2$ norm of $T(t-1)^{-1/2}(T(t)-T(t-1))T(t-1)^{-1/2}$. By (2.3), we have

$$\begin{aligned} & T(t-1)^{-1/2}\left(T(t)-T(t-1)\right)T(t-1)^{-1/2} \\ = \ & T(t-1)^{-1/2}\left[\frac{x(t)}{t}V_T(t)-\mathcal{E}\left(\frac{x(t)}{t}V_T(t)\right)\right]T(t-1)^{-1/2}+ \\ & T(t-1)^{-1/2}\mathcal{E}\left(\frac{x(t)}{t}V_T(t)\right)T(t-1)^{-1/2}. \end{aligned}$$

With

$$\begin{aligned} \left\|\frac{x(t)}{t}V_T(t)-\mathcal{E}\left(\frac{x(t)}{t}V_T(t)\right)\right\|_2 &\le \left\|\frac{x(t)}{t}V_T(t)-\mathcal{E}\left(\frac{x(t)}{t}V_T(t)\right)\right\|_1 \\ &= 2\left|\sum_{k=0}^{n-1}\alpha^{k/2}\left[\frac{x(t)\overline{x(t-k)}}{t}-\mathcal{E}\left(\frac{x(t)\overline{x(t-k)}}{t}\right)\right]\right|, \end{aligned}$$

using Lemma (3.3) and considering probability argument, we obtain

$$\begin{aligned} & \Pr\left\{\left\|T(t-1)^{-1/2}\left[\frac{x(t)}{t}V_T(t)-\mathcal{E}\left(\frac{x(t)}{t}V_T(t)\right)\right]T(t-1)^{-1/2}\right\|_2\frac{\epsilon}{2}\right\} \\ \ge \ & 1-\frac{8n^3C_2C_4^2\alpha^{n-1}(1-\alpha^n)}{(1-\alpha)\epsilon^2t^2}. \end{aligned}$$

By (**A1**), we also have

$$\left\|\mathcal{E}\left(\frac{x(t)}{t}V_T(t)\right)\right\|_2 \le \frac{1}{t}\left\|\mathcal{E}\left(x(t)V_T(t)\right)\right\|_1 \le \frac{nC_1}{t}.$$

Thus there exists $t_0$ given by $t_0=\max\left\{\frac{2nC_1}{\epsilon},\sqrt{\frac{8n^3C_2C_4^2}{\delta\epsilon^2}}\right\}$, such that for $t\ge t_0$, then we have

$$\|T(t)^{1/2}(T(t-1)^{-1}-T(t)^{-1})T(t)^{1/2}\|_2\le\epsilon,$$

with probability $1-\delta$. It follows from (3.8) that the minimum and maximum eigenvalues of $\hat\alpha T(t)^{1/2}\tilde T(t-1)^{-1}T(t)^{1/2}$ are bounded as follows with probability $1-\delta$:

$$\begin{aligned} \hat\alpha(1-\epsilon_1-\epsilon) &\le \lambda_{\min}\left(\hat\alpha T(t)^{1/2}\tilde T(t-1)^{-1}T(t)^{1/2}\right) \\ &\le \lambda_{\max}\left(\hat\alpha T(t)^{1/2}\tilde T(t-1)^{-1}T(t)^{1/2}\right)\le\hat\alpha(1+\epsilon_1+\epsilon) \end{aligned} \tag{3.9}$$

Hence the theorem follows. □

Using Theorem 3.4, we can estimate the number of iterations required for convergence. We let $\mathbf{s}^{(k)}(t)$ be the error vector given by

$$\mathbf{s}^{(k)}(t)=\mathbf{u}(t)-\mathbf{u}^{(k)}(t),$$

after the $k$th iteration of preconditioned conjugate gradient method is applied to solving the preconditioned system. By convergence of the PCG method, we mean that $\|\mathbf{s}^{(k)}(t)\|_2$ is sufficiently small at the $k$th iteration so that the $\ell_2$ norm of the error matrix satisfies $\|E(t)\|_2 \leq \epsilon_1/nC_3$. We remark that if $\|E(t)\|_2$ is sufficiently small then the filter coefficients can be computed accurately at each time step and the number of PCG iterations can be reduced.

LEMMA 3.5. *Let the input signal process satisfy assumption* (**A3**). *For any given* $\epsilon_1 > 0$, *there exists* $\eta^* > 0$ *such that if* $\|\mathbf{y}(t)\|_2 \leq \eta^*$ *then* $\|E(t)\|_2 \leq \dfrac{\epsilon_1}{nC_3}$.

*Proof.* Choose

$$\eta^* = \min\left\{\frac{1}{2C_3}, \sqrt{\frac{\epsilon_1}{24nC_1C_3}}, \frac{\epsilon_1}{48nC_1C_3C_4}, \frac{\epsilon_1}{3C_1C_3^2C_4}\right\} \cdot \frac{1}{nC_3},$$

and use the error bound $E(t)$ given in Theorem 3.1. The result follows. □

In the following, we let $N(t,\eta^*)$ denote the smallest positive integer $k$ such that $\|\mathbf{s}^{(k)}(t)\|_2 \leq \eta^*$. Therefore, $N(t,\eta^*)$ is the smallest number of iterations required for the convergence of the PCG method at time $t$. The following theorem gives an upper bound for $N(t,\eta^*)$.

THEOREM 3.6. *Let the input signal process satisfy assumption* (**A**). *For any given* $\epsilon$ *and* $\delta$ *satisfying* $0 < \epsilon < 1-\epsilon_1$ *and* $0 < \delta < 1$, *there exists an integer* $t_0$, *which depends on* $n$, $\epsilon$, $\delta$, $\alpha$, $C_1$, $C_2$, $C_3$ *and* $C_4$, *such that for* $t \geq t_0$, *if*

$$N(t,\eta^*) \leq \frac{1}{2}\left(\frac{1+\epsilon_1+\epsilon}{1-\epsilon_1-\epsilon}\right)\log\left[\frac{2(1+\epsilon_1+\epsilon)\|\mathbf{s}^{(0)}(t)\|_2}{(1-\epsilon_1-\epsilon)\eta^*}\right]+1, \tag{3.10}$$

*then*

$$\Pr\left\{\|\mathbf{s}^{(k)}(t)\|_2 \leq \eta^*\right\} > 1-\delta.$$

*Proof.* By using Theorem 3.4 and the convergence rate of the conjugate gradient iterations in [3, Theorem 1.12, p. 26], one can prove that if $N(t,\eta^*)$ is bounded above as stated as in (3.10), we have

$$\Pr\left\{|||\mathbf{s}^{(k)}(t)||| \leq \left[\frac{(1-\epsilon_1-\epsilon)\eta^*}{(1+\epsilon_1+\epsilon)\|\mathbf{s}^{(0)}(t)\|_2}\right]|||\mathbf{s}^{(0)}(t)|||\right\} > 1-\delta. \tag{3.11}$$

Here $|||\cdot|||$ is the *energy norm* corresponding to the preconditioned matrix

$$\hat{\alpha}\tilde{T}(t-1)^{-1/2}T(t)\tilde{T}(t-1)^{-1/2},$$

defined by

$$|||\mathbf{v}||| = \mathbf{v}^*\hat{\alpha}\tilde{T}(t-1)^{-1/2}T(t)\tilde{T}(t-1)^{-1/2}\mathbf{v}.$$

As the minimum and maximum eigenvalues of the preconditioned matrix are bounded as stated in (3.9), for any vector $\mathbf{v}$, we obtain

$$\hat{\alpha}(1-\epsilon_1-\epsilon)\|\mathbf{v}\|_2 \leq |||\mathbf{v}||| \leq \hat{\alpha}(1+\epsilon_1+\epsilon)\|\mathbf{v}\|_2. \tag{3.12}$$

Putting (3.12) into (3.11), the result follows. □

Using Theorem 3.6, we easily note that the conjugate gradient method, when applied to the preconditioned system

$$\hat{\alpha}\tilde{T}(t-1)^{-1}T(t)\mathbf{u}(t) = \hat{\alpha}\tilde{T}(t-1)^{-1}\mathbf{e}_n,$$

**ETNA**
Kent State University
etna@mcs.kent.edu



converges rapidly with probability 1, provided that $t$ is sufficiently large and $\|E(t-1)\|_2$ is sufficiently small.

We recall that in each iteration of the preconditioned conjugate gradient method the work is of order $O(n \log n)$ operations. Therefore, the work for obtaining the solution vector $\mathbf{u}(t)$ to a given accuracy is also of order $O(n \log n)$ operations. Hence the total work of obtaining $\mathbf{w}(t)$ at each adaptive time step is of order only $O(n \log n)$ operations, if the input process satisfies assumption (**A**). Finally, we remark that our LMSN algorithm is highly parallelizable. In a parallel environment with $O(n)$ processors, the total work of updating the filter coefficient vector $\mathbf{w}(t)$ at each adaptive time step is $O(\log n)$ operations.

**4. Numerical Experiments.** In this section, some numerical experiments are performed to test the convergence rate of the preconditioned conjugate gradient algorithm. All the computations are done using Matlab. In the numerical tests, the stopping criterion used for the preconditioned conjugate gradient (PCG) method is $\tau_k/\tau_0 < 10^{-7}$ as stated in PCG algorithm. Moreover, in all tests, the forgetting factor is set to 0.99, for simplicity.

In the first set of numerical tests, we illustrate our method by using an adaptive filtering model problem from [7]. Figure 2 shows the block diagram of the adaptive filtering simulation model. The common input to the unknown (reference) system and adaptive filter is a MA(16) input process that is obtained by passing a Gaussian white noise with unity variance through a FIR filter whose impulse responses is given in Table 1. The unknown (reference) system is an 14-th order FIR system with transfer function $\mathcal{P}(z)$ given by $\mathcal{P}(z) = 1 + 4z^{-6} - z^{-8} + 0.2z^{-10} + 0.1z^{-13}$. Different levels of variances of the Gaussian white noise $\{n(t)\}$ are used to test the performance of the FFT–based LMSN algorithm. The length $n$ of the adaptive filter is selected to be 16.

For the comparison, T. Chan circulant preconditioners [6] are used in the tests. Figure 3 shows the condition numbers of different preconditioned matrices for one realization of input signal. We note that the condition numbers of our preconditioned systems are near to 1 when $t$ is large enough. Figure 4 shows the average number of iterations of the preconditioned system at each step, averaged over 100 independent runs of the algorithm for different levels of Gaussian noise (variances of noise = 0.01 and 0.1) added into the adaptive systems. We remark that for different levels of noise, the Toeplitz systems $T(t)\mathbf{u}(t) = \mathbf{e}_n$ are the same. We see from the numerical results that the conjugate gradient method for our preconditioned systems converges faster than that of the circulant-preconditioned systems and non-preconditioned systems. Figures 5–6 show the corresponding ensemble-averaged learning curves of these adaptive filtering models. For this comparison, the learning curves of the LMS algorithms are presented together in Figures 5–6. We used the same step size 0.025 for the FFT–based LMSN and LMS algorithms. From the numerical results of the FFT–based LMSN algorithm, the mean square error (estimation error $e(t)$) decreases rapidly to a steady-state value of the average mean square error. Also the decrease of mean square error using FFT–based LMSN algorithm is faster than that by using the LMS algorithm. We also compare our FFT–based LMSN algorithm with fast transversal filter $O(n)$ operations algorithms [12, pp. 586–599]. Figure 7 shows that the fast transversal filter algorithm does not converge when noise is added to the adaptive FIR system. In order to get accurate result, it is necessary to reinitialize the filter coefficients before the error becomes large; see [12]. However, the performance of the FFT–based LMSN algorithm is quite stable.

**ETNA**
Kent State University
etna@mcs.kent.edu



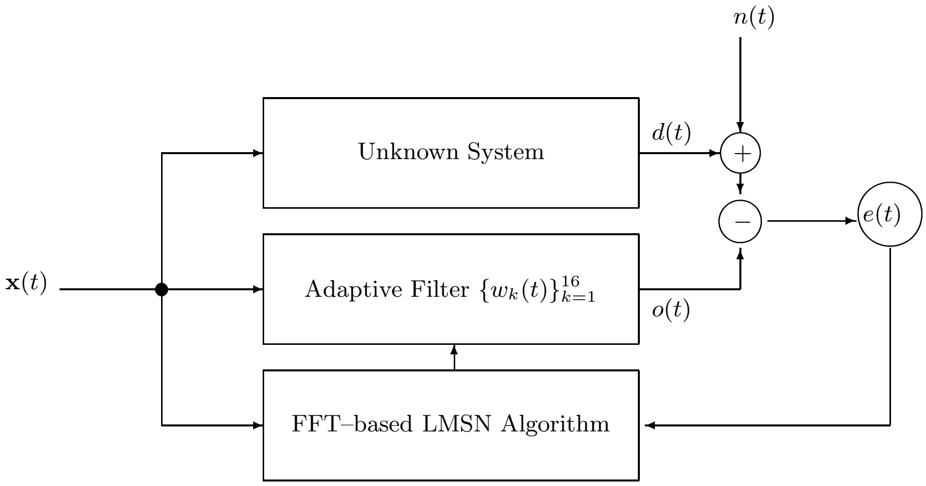


FIG. 2. *Adaptive Filtering Simulation Model*

| | | | |
|---|---|---|---|
| $h(1) = -0.024476535$ | $h(5) = 0.024523262$ | $h(9) = -0.005495879$ | $h(13) = 0.224112905$ |
| $h(2) = -0.122125943$ | $h(6) = -0.051290013$ | $h(10) = -0.055246398$ | $h(14) = 0.015079975$ |
| $h(3) = 0.133007741$ | $h(7) = -0.024950762$ | $h(11) = -0.017690268$ | $h(15) = -0.475178548$ |
| $h(4) = -0.050216528$ | $h(8) = 0.096703820$ | $h(12) = -0.058418098$ | $h(16) = 0.409890873$ |

TABLE 1
*The impulse responses of the FIR filter with $h(k) = h(32 - k + 1)$ used for generation of the input signal process.*

For the second set of simulations, we consider a non-Gaussian input process and desired responses. We consider the channel equalization problem as shown as in Figure 8; see [7] and [12, p. 342]. The random sequence $\{a(t)\}$ applied to the channel input is in polar form, with $a(t) = \pm 1$ with equal probability. The impulse responses of the channel are given in Table 2. A Gaussian white noise $\{n(t)\}$ is added to the output of the channel. The channel input $\{a(t)\}$ is delayed by 15 samples to provide the desired responses for the equalizer. Figure 9 shows the condition number of different preconditioned matrices for one realization of input signal. We note that the condition numbers of our preconditioned systems are near to 1 when $t$ is sufficiently large. Figure 10 shows the average number of iterations of our preconditioned system, circulant-preconditioned system and non-preconditioned system at each step, averaged over 100 independent runs of the algorithm. The preconditioned conjugate gradient method with our FFT–based preconditioners converges faster than the others at each step. Figure 11 also shows the ensemble-averaged learning curves of the FFT–based LMSN and LMS algorithms where the variance of noise is 0.01. The step size used in two algorithms is 0.01. We note from the numerical results that the decrease of the of mean square error using the FFT–based LMSN algorithm is faster than that by using the LMS algorithm. We observe from Figure 12 that the fast transversal filter algorithm converges faster than the FFT–based LMSN algorithm, however, its mean square error is larger than the FFT–based LMSN algorithm. Therefore the performance of the FFT–based LMSN algorithm is better than the others.

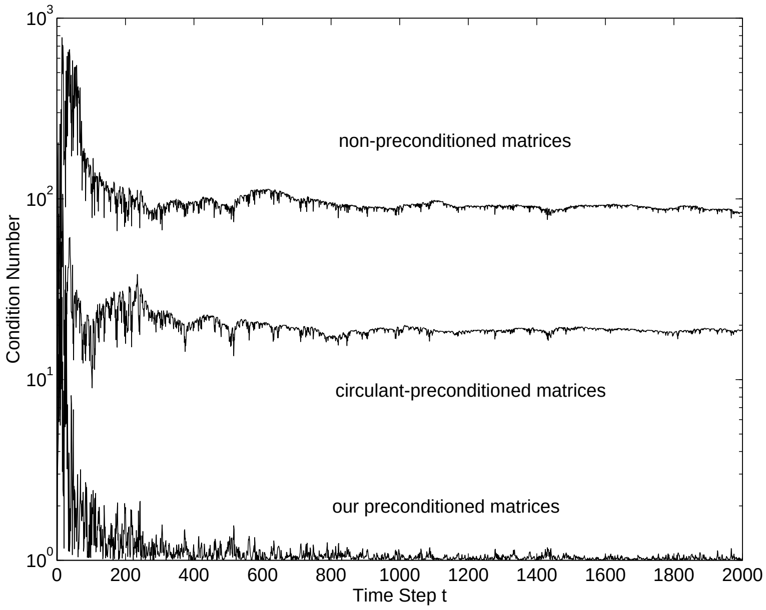


FIG. 3. *Condition Numbers of Different Preconditioned Matrices.*

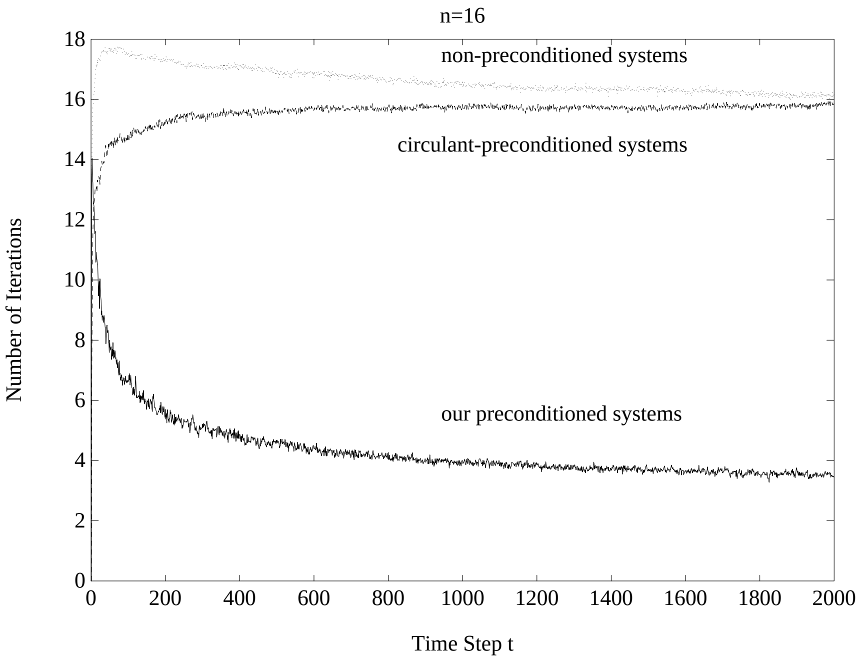


FIG. 4. *Numbers of Iterations for Convergence at Each Time Step with $n = 16$.*

**ETNA**
**Kent State University**
**etna@mcs.kent.edu**



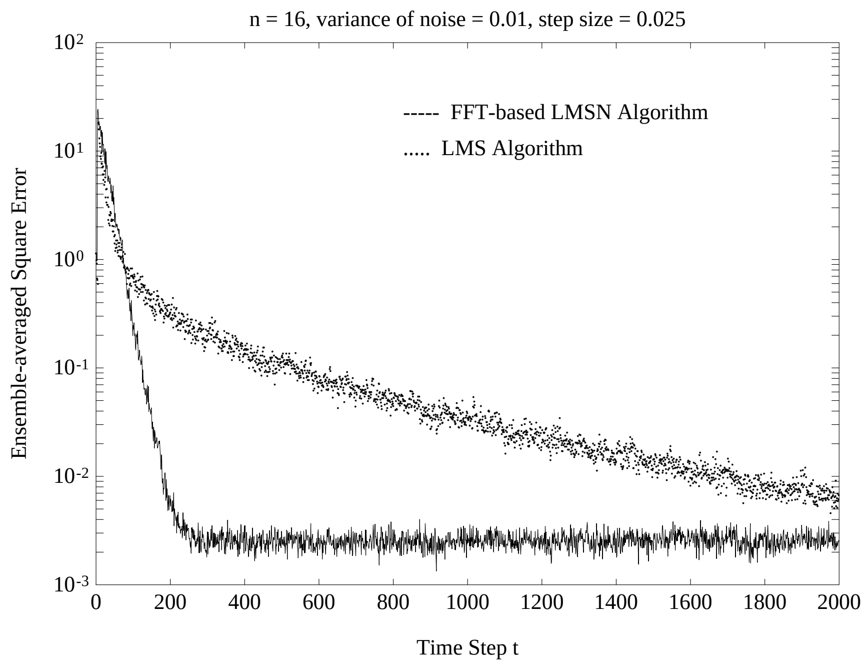


Fig. 5. *Learning Curves of the FFT–based LMSN and LMS Algorithms.*

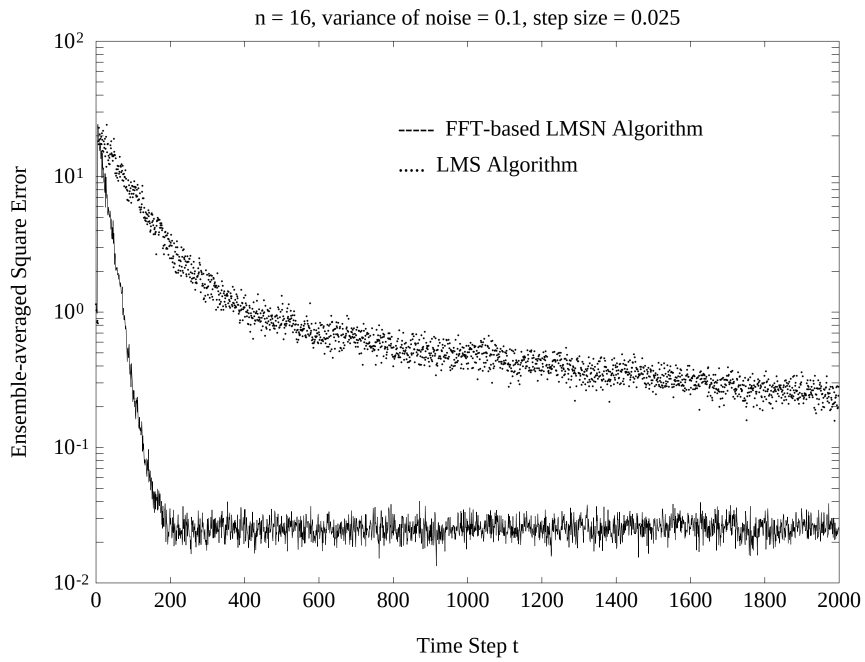


Fig. 6. *Learning Curves of the FFT–based LMSN and LMS Algorithms.*

ETNA
Kent State University
etna@mcs.kent.edu



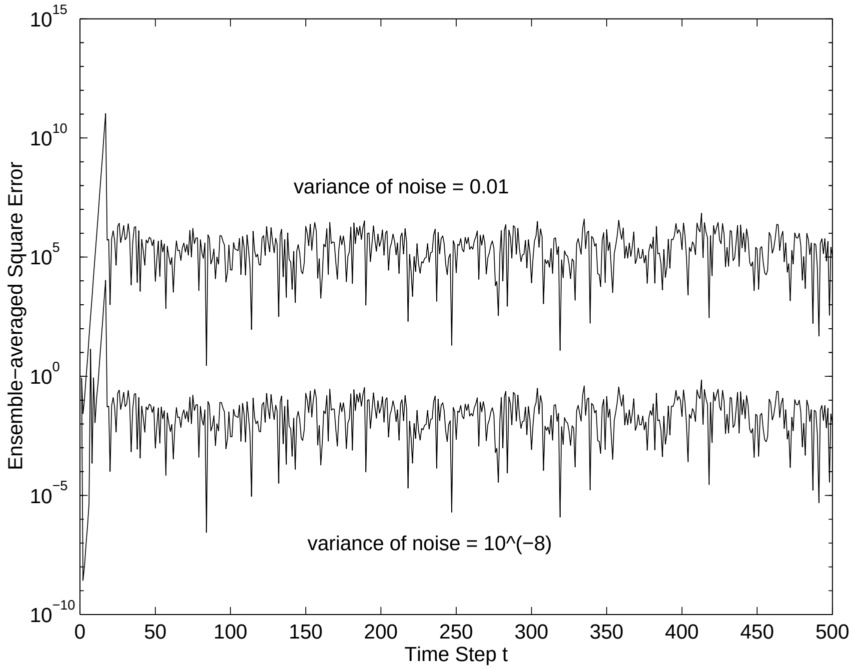


Fig. 7. *Learning Curves of the Fast Transversal Filter Algorithms.*

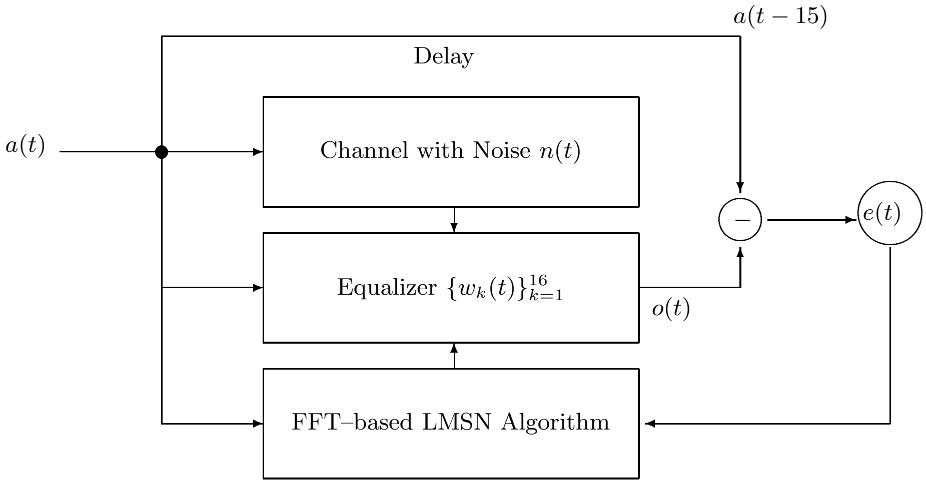


Fig. 8. *The Equalization Problem used for Simulation.*

| | | | |
|---|---|---|---|
| $h(1) = 0.0066$ | $h(5) = 0.3605$ | $h(9) = -0.2270$ | $h(13) = -0.0159$ |
| $h(2) = 0.0262$ | $h(6) = 0.6762$ | $h(10) = 0.1048$ | $h(14) = 0.0083$ |
| $h(3) = 0.1919$ | $h(7) = -0.2620$ | $h(11) = -0.0509$ | $h(15) = -0.0331$ |
| $h(4) = 0.2867$ | $h(8) = 0.3977$ | $h(12) = 0.0278$ | $h(16) = 0.0005$ |

Table 2
*The Impulse Responses of the Channel.*

**ETNA**
Kent State University
etna@mcs.kent.edu



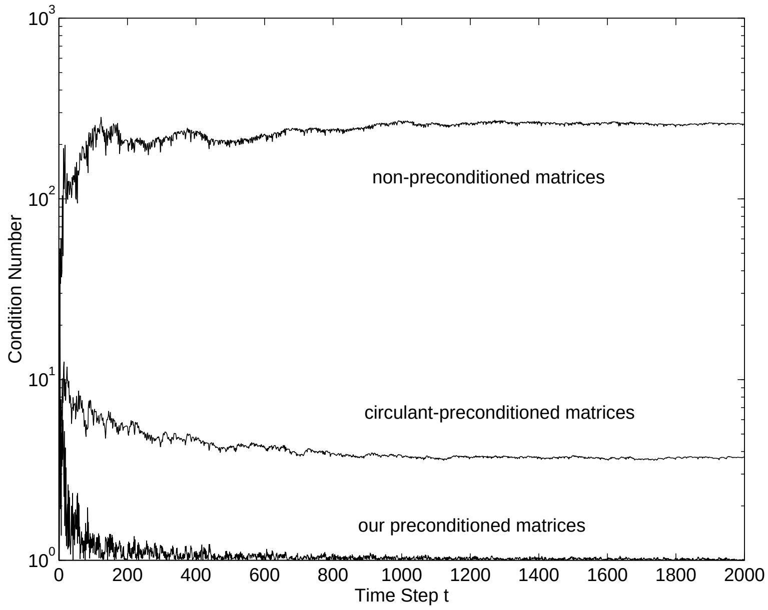


Fig. 9. *Condition Numbers of Different Preconditioned Matrices.*

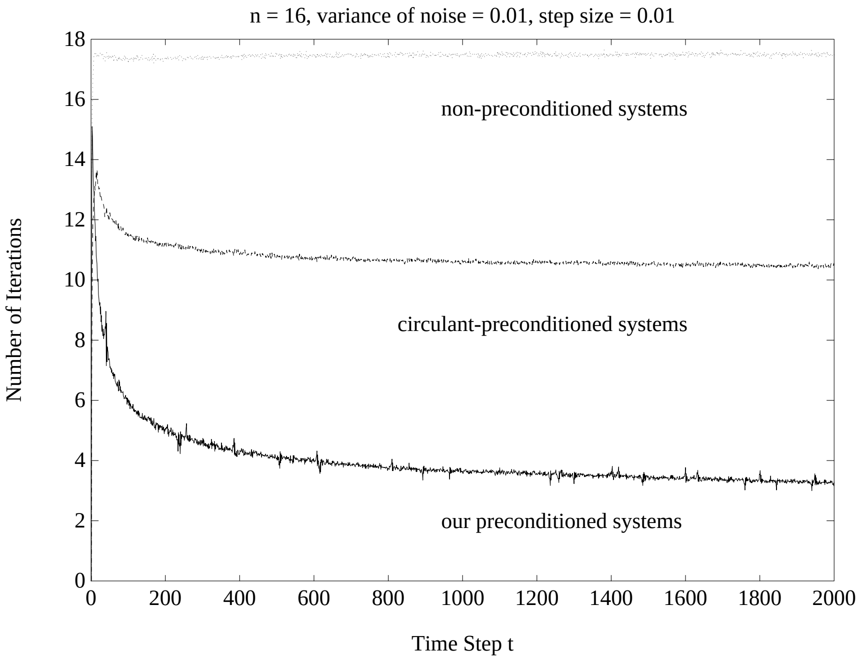


Fig. 10. *Numbers of Iterations for Convergence at Each Time Step with* $n = 16$*.*

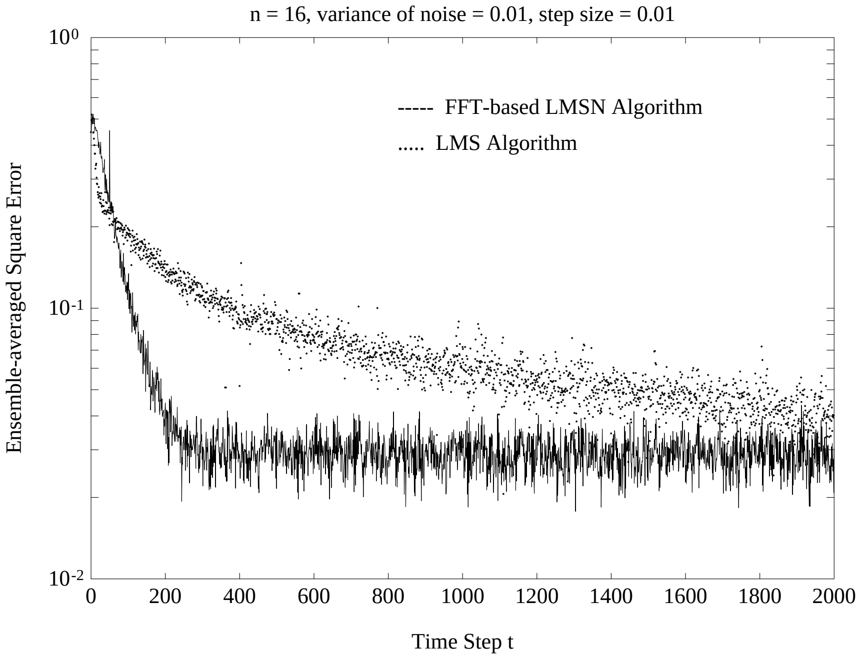


FIG. 11. *Learning Curves of the FFT–based LMSN and LMS Algorithms.*

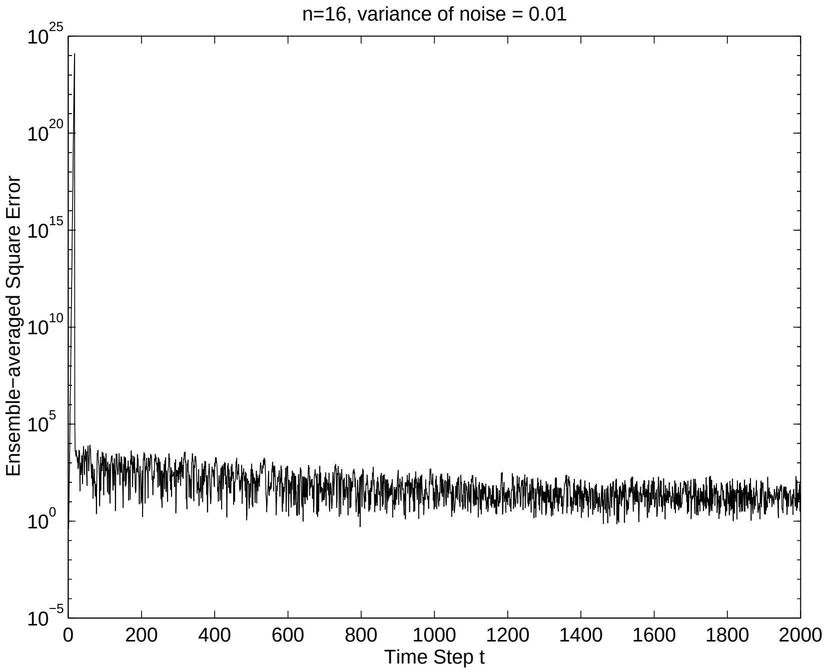


FIG. 12. *Learning Curves of the Fast Transversal Filter Algorithms.*

**5. Concluding Remarks.** In this paper, we have proposed a new FFT–based LMSN algorithm. Preliminary numerical results show that the performance of the algorithm is good in the sense of reasonable complexity and convergence. Also, the complexity of the algorithm is only $O(n \log n)$ operations per each adaptive time step, and the algorithm itself is highly parallelizable. These attractive features could lead to the use of the algorithm in diverse adaptive filtering applications. We remark that the FFT–based LMSN algorithm can be adapted to handle 2–D signal processing applications, for example 2–D linear prediction, multichannel filtering and spectrum analysis, as in [21]. The results for 1–D signal processing applications thus far look promising, and we intend to test a 2–D FFT–based LMSN algorithm in the next phase of our work.

**6. Appendix. Proof of Theorem 3.1:** We first let $B_1(t)$ and $B_2(t)$ be the circulant and skew-circulant matrices respectively. Their first rows are given by

$$[u_{n-1}(t), \overline{u_{n-2}(t)}, \ldots, \overline{u_0(t)}] \quad \text{and} \quad [u_{n-1}(t), -\overline{u_{n-2}(t)}, \ldots, -\overline{u_0(t)}]$$

respectively where $\mathbf{u}(t)$ is the actual solution vector of the linear system $T(t)\mathbf{u}(t) = \mathbf{e}_n$. It follows from Lemma 2.2 that $B_2(t)B_1(t)^* + B_2(t)^*B_1(t)$ is a Hermitian matrix. By (3.1), we obtain

$$\tilde{B}_1(t) = B_1(t) + E_1(t) \quad \text{and} \quad \tilde{B}_2(t) = B_2(t) + E_2(t).$$

Therefore, the matrix $E_1(t)$ is the circulant matrix with first row given by

$$[y_{n-1}(t), \overline{y_{n-2}(t)}, \ldots, \overline{y_0(t)}]^T, \tag{6.1}$$

and $E_2(t)$ is the skew-circulant matrix with first row given by

$$[y_{n-1}(t), -\overline{y_{n-2}(t)}, \ldots, -\overline{y_0(t)}]^T. \tag{6.2}$$

It follows that the error matrix $E(t)$ is equal to

$$\begin{aligned}
E(t) &= \frac{1}{2\tilde{u}_{n-1}(t)}\left[\tilde{B}_2(t)\tilde{B}_1(t)^* + \tilde{B}_2(t)^*\tilde{B}_1(t)\right] \\
&\quad -\frac{1}{2u_{n-1}(t)}\left[B_2(t)B_1(t)^* + B_2(t)^*B_1(t)\right] \\
&= \frac{1}{2}\left[\frac{1}{u_{n-1}(t)+y_{n-1}(t)}\right]\Big[\tilde{B}_2(t)\tilde{B}_1(t)^* + \tilde{B}_2(t)^*\tilde{B}_1(t) \\
&\quad -B_2(t)B_1(t)^* + B_2(t)^*B_1(t)\Big] \\
&\quad +\frac{1}{2}\left[\frac{1}{u_{n-1}(t)+y_{n-1}(t)} - \frac{1}{u_{n-1}(t)}\right]\left[B_2(t)B_1(t)^* + B_2(t)^*B_1(t)\right] \\
&= \frac{1}{2}\left[\frac{1}{u_{n-1}(t)+y_{n-1}(t)}\right]\Big[\tilde{B}_2(t)\tilde{B}_1(t)^* + \tilde{B}_2(t)^*\tilde{B}_1(t) \\
&\quad -B_2(t)B_1(t)^* - B_2(t)^*B_1(t)\Big] \\
&\quad +\frac{1}{2}\left[\frac{1}{\mathbf{u}_{n-1}(t)+y_{n-1}(t)} - \frac{1}{\mathbf{u}_{n-1}(t)}\right]T(t)^{-1}.
\end{aligned} \tag{6.3}$$

Here we recall that $T(t)^{-1} = B_2(t)B_1(t)^* + B_2(t)^*B_1(t)$. From $\mathbf{u}(t) = T(t)^{-1}\mathbf{e}_n$, we get

$$\mathbf{u}_{n-1}(t) = [T(t)^{-1}]_{n,n}.$$

ETNA
Kent State University
etna@mcs.kent.edu



As the diagonal elements of $T(t)^{-1}$ are always greater than or equal to $\frac{1}{\|T(t)\|_2}$, we have

$$u_{n-1}(t) \geq \frac{1}{\|T(t)\|_2}.$$

Therefore, the $\ell_2$ norm of the second term of the right hand side in (6.3) is given by

$$\left\| \frac{1}{2} \left[ \frac{1}{u_{n-1}(t) + y_{n-1}(t)} - \frac{1}{u_{n-1}(t)} \right] T(t)^{-1} \right\|_2 \leq \frac{\|\mathbf{y}(t)\|_2 \kappa_2(T(t))}{2[1 - \|\mathbf{y}(t)\|_2 \|T(t)\|_2]}. \tag{6.4}$$

As for the first term of right hand side in (6.3), we first rewrite it as follows:

$$\begin{aligned}
& \tilde{B}_2(t)\tilde{B}_1(t)^* + \tilde{B}_2(t)^*\tilde{B}_1(t) - B_2(t)B_1(t)^* - B_2(t)^*B_1(t) \\
= \; & \tilde{B}_2(t)\tilde{B}_1(t)^* - B_2(t)\tilde{B}_1(t)^* + B_2(t)\tilde{B}_1(t)^* - B_2(t)B_1(t)^* + \tilde{B}_2(t)^*\tilde{B}_1(t)^* - B_2(t)^*\tilde{B}_1(t) + \\
& B_2(t)^*\tilde{B}_1(t) - B_2(t)^*B_1(t) \\
= \; & E_2(t)\tilde{B}_1(t)^* + B_2(t)E_1(t) + E_2(t)^*\tilde{B}_1(t) + B_2(t)^*E_1(t) \\
= \; & E_2(t)\tilde{B}_1(t)^* - E_2(t)B_1(t)^* + E_2(t)B_1(t)^* + B_2(t)E_1(t) + E_2(t)^*\tilde{B}_1(t) - E_2(t)B_1(t) + \\
& E_2(t)B_1(t) + B_2(t)^*E_1(t) \\
= \; & E_2(t)E_1(t)^* + E_2(t)B_1(t)^* + B_2(t)E_1(t) + E_2(t)^*E_1(t) + E_2(t)B_1(t) + B_2(t)^*E_1(t).
\end{aligned}$$

We see from (6.1) and (6.2) that

$$\|E_2(t)E_1(t)^*\|_2 \leq \|E_2(t)\|_2 \|E_1(t)^*\|_2 \leq 4\|\mathbf{y}(t)\|_1 \|\mathbf{y}(t)\|_1 \leq 4\|\mathbf{y}(t)\|_1^2$$

and that

$$\begin{aligned}
\|B_2(t)E_1(t)^*\|_2 \leq \|B_2(t)\|_2 \|E_1(t)^*\|_2 & \leq 4\|\mathbf{u}(t)\|_1 \|\mathbf{y}(t)\|_1 \\
& = 4\|T(t)^{-1}\mathbf{e}_n\|_1 \|\mathbf{y}(t)\|_1 \leq 4\|T(t)^{-1}\|_1 \|\mathbf{y}(t)\|_1.
\end{aligned}$$

We can establish similar results for the matrices $E_2(t)^*E_1(t)$, $E_2(t)B_1(t)^*$, $B_2(t)^*E_1(t)$ and $E_2(t)B_1(t)$. Thus we obtain

$$\begin{aligned}
& \|\tilde{B}_2(t)\tilde{B}_1(t)^* + \tilde{B}_2(t)^*\tilde{B}_1(t) - B_2(t)B_1(t)^* - B_2(t)^*B_1(t)\|_2 \\
\leq \; & 8\|\mathbf{y}(t)\|_1^2 + 16\|T(t)^{-1}\|_1 \|\mathbf{y}(t)\|_1 \leq 8n\|\mathbf{y}(t)\|_2^2 + 16n\|T(t)^{-1}\|_2 \|\mathbf{y}(t)\|_2.
\end{aligned} \tag{6.5}$$

Combining (6.5) and (6.4) into (6.3), the result follows.

**Proof of (3.6):** By expanding the expression $\mathcal{E}\left((x(t)-\beta)(\overline{x(t-k)}-\overline{\beta})(\overline{x(t)}-\overline{\beta})(x(t-k)-\beta)\right)$, we can obtain

$$\begin{aligned}
\mathcal{E}\left(x(t)\overline{x(t-k)x(t)}x(t-k)\right) = \; & \mathcal{E}\left((x(t)-\beta)(\overline{x(t-k)}-\overline{\beta})\right. \\
& \left.\times(\overline{x(t)}-\overline{\beta})(x(t-k)-\beta)\right) - \beta^2\overline{\beta}^2 \\
& +\beta^2\overline{\beta}\mathcal{E}\left(\overline{x(t)}+\overline{x(t-k)}\right) \\
& +\beta\overline{\beta}^2\mathcal{E}(x(t)+x(t-k)) \\
& +2\beta\mathcal{E}\left(\overline{x(t-k)x(t)}x(t-k)\right) \\
& +2\overline{\beta}\mathcal{E}\left(x(t)x(t-k)\overline{x(t)}\right)
\end{aligned} \tag{6.6}$$

**ETNA**
Kent State University
etna@mcs.kent.edu



$$
\begin{aligned}
&-\beta\overline{\beta}\mathcal{E}\left(x(t)\overline{x(t-k)}+\overline{x(t)}x(t-k)\right.\\
&\left.+x(t-k)\overline{x(t-k)}+x(t)\overline{x(t)}\right)\\
&+\beta^2\mathcal{E}\left(\overline{x(t)x(t-k)}\right)+\overline{\beta}^2\mathcal{E}(x(t)x(t-k)).
\end{aligned}
$$

Similarly we can derive

$$
\begin{aligned}
\mathcal{E}\left(\overline{x(t-k)x(t)}x(t)\right) &= \mathcal{E}\left((\overline{x(t)}-\overline{\beta})(x(t)-\beta)(\overline{x(t-k)}-\overline{\beta})\right)\\
&+\beta\overline{\beta}^2-\beta\overline{\beta}\mathcal{E}\left(\overline{x(t-k)}+\overline{x(t)}\right)-\overline{\beta}^2\mathcal{E}\left(x(t)\right)\\
&+\overline{\beta}\mathcal{E}\left(\overline{x(t-k)}x(t-k)+\overline{x(t)}x(t)\right)+\beta\overline{x(t-k)x(t)}.
\end{aligned}
\tag{6.7}
$$

As $\{x(t)-\beta\}$ is a zero-mean complex Gaussian stationary process up to the second order with the variances $r_0$, we have

$$
\begin{aligned}
&\mathcal{E}\left((x(t)-\beta)(\overline{x(t-k)}-\overline{\beta})(\overline{x(t)}-\overline{\beta})(x(t-k)-\beta)\right)\\
&\qquad=2\mathcal{E}\left((x(t)-\beta)(\overline{x(t-k)}-\overline{\beta})\right)^2\le 2r_0^2
\end{aligned}
\tag{6.8}
$$

and

$$
\mathcal{E}\left((\overline{x(t-k)}-\overline{\beta})(\overline{x(t)}-\overline{\beta})(x(t-k)-\beta)\right)=0,
\tag{6.9}
$$

see Haykin [12, pp. 110–111], Also we have

$$
\left|\mathcal{E}\left(x(t)\overline{x(t-k)}\right)\right|\le|\beta|^2+r_0.
\tag{6.10}
$$

Combining (6.6), (6.7), (6.8), (6.9), (6.10) and (3.5) together, it follows that

$$
\left|\mathcal{E}\left(x(t)\overline{x(t-k)x(t)}x(t-k)\right)\right|\le 40|\beta|^4+12|\beta|^2r_0+3r_0^2,
$$

establishing (3.6).

## REFERENCES

[1] S. Aki, *The Design and Analysis of Parallel Algorithms*, Prentice-Hall International Inc., London, 1989.

[2] G. Ammar and P. Gader, *New Decompositions of the Inverse of a Toeplitz Matrix*, in Signal Processing, Scattering and Operator Theory, and Numerical Methods (Proceedings of the International Symposium MTNS-89, Volume III), M.A. Kaashoek, J.H. van Schuppen and A.C.N. Ran, eds., Birkhäuser, Boston, 1990, pp. 421–428.

[3] O. Axelsson and V. Barker, *Finite Element Solution of Boundary Value Problems, Theory and Computation*, Academic, New York, 1984.

[4] R. Chan and M. Ng, *Toeplitz preconditioners for Hermitian Toeplitz systems*, Linear Algebra and Appl., 190 (1993), pp. 181-208.

[5] R. Chan and G. Strang, *Toeplitz equations by conjugate gradients with circulant preconditioner*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 104-119.

[6] T. Chan, *An optimal circulant preconditioner for Toeplitz systems*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 766-771.

[7] B. Farhang-Boroujeny, *Application of orthonormal transforms to implementation of quasi-LMS/Newton algorithm*, IEEE Trans. on Signal Process., 41 (1993), pp.1400-1405.

[8] W. Fuller, *Introduction to Statistical Time Series*, John Wiley & Sons, Inc., New York, 1976.

[9] R. Gitlin and F. Magee, *Self-orthogonalizing adaptive equalization algorithms*, IEEE Trans. Commun., 25 (1977), pp. 666-672.
[10] I. Gohberg and A. Semencul, *On the inversion of finite Toeplitz matrices and their continuous analogs* (in Russian), Mat. Issled, 2 (1972), pp. 201-223.
[11] G. Golub and C. Van Loan, *Matrix Computations*, Second ed., Johns Hopkins Press, Baltimore, 1989.
[12] S. Haykin, *Adaptive Filter Theory*, Second ed., Prentice-Hall, Englewood Cliffs, NJ, 1991.
[13] T. Ku and C. Kuo, *Design and analysis of Toeplitz preconditioners*, IEEE Trans. on Signal Process., 40 (1991), pp. 129-141.
[14] N. Levinson, *The Wiener rms (root-mean-square) error criterion in filter design and prediction*, J. Math. Phys., 25 (1947), pp. 261-278.
[15] X. Luo and S. Qiao, *An error analysis of the fast RLS algorithms*, Rept. no. 231, Comm. Res. Lab., McMaster Univ., Hamilton, Ontario, Canada, 1991.
[16] M. Ng, *Fast iterative methods for solving Toeplitz-plus-Hankel least squares problem*, Electron. Trans. Numer. Anal., 2 (1994), pp. 154-170.
[17] M. Ng and R. Chan, *Fast iterative methods for least squares estimations*, Numer. Algorithms, 6 (1994), pp. 353-378.
[18] M. Ng and R. Plemmons, *Fast RLS adaptive filtering by FFT–based conjugate gradient iterations*, to appear in SIAM J. Sci. Comp. (1996).
[19] M. Priestley, *Spectral Analysis and Time Series*, Academic Press, New York, 1981.
[20] S. Qiao, *Fast recursive least aquares algorithms for Toeplitz matrices*, Advanced Signal Processing Algorithms, Architectures, and Implementations II, SPIE, 1566 (1991), pp. 47-58.
[21] L. Sibul, *Adaptive Signal Processing*, IEEE Press, New York, 1987.
[22] B. Widrow and S. Stearn, *Adaptive Signal Processing*, Prentice-Hall, New Jersey, 1985.