A General Class of Nonlinear Normalized Adaptive Filtering Algorithms

Sudhakar Kalluri, *Member, IEEE*, and Gonzalo R. Arce, *Senior Member, IEEE*

***Abstract*— The normalized least mean square (NLMS) algorithm is an important variant of the classical LMS algorithm for adaptive linear filtering. It possesses many advantages over the LMS algorithm, including having a faster convergence and providing for an automatic time-varying choice of the LMS *step-size parameter* that affects the stability, steady-state mean square error (MSE), and convergence speed of the algorithm. An *auxiliary* fixed step-size that is often introduced in the NLMS algorithm has the advantage that its stability region (step-size range for algorithm stability) is *independent* of the signal statistics.**

In this paper, we generalize the NLMS algorithm by deriving a class of *nonlinear normalized LMS-type (NLMS-type) algorithms* that are applicable to a wide variety of nonlinear filter structures. We obtain a general nonlinear NLMS-type algorithm by choosing an optimal time-varying step-size that minimizes the next-step MSE at each iteration of the general nonlinear LMS-type algorithm. As in the linear case, we introduce a dimensionless auxiliary step-size whose stability range is independent of the signal statistics. The stability region could therefore be determined empirically for any given nonlinear filter type. We present computer simulations of these algorithms for two specific nonlinear filter structures: *Volterra filters* and the recently proposed class of *Myriad filters*. These simulations indicate that the NLMS-type algorithms, in general, converge faster than their LMS-type counterparts.

***Index Terms*—Adaptive filtering, least mean square, nonlinear adaptive algorithms, nonlinear filters, normalized LMS.**

I. Introduction

The LEAST MEAN square (LMS) algorithm [1] is widely used for adapting the weights of a linear FIR filter that minimizes the mean square error (MSE) between the filter output and a desired signal. Consider an input (observation) vector of N samples $\boldsymbol{x} \triangleq [x_1, x_2, \ldots, x_N]^T$ and a weight vector of N weights $\boldsymbol{w} \triangleq [w_1, w_2, \cdots, w_N]^T$. Denote the linear filter output by $y \equiv y(\boldsymbol{w}, \boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$. The filtering error, in estimating a desired signal d, is then $e \triangleq y - d$. Under the mean square error (MSE) criterion, the optimal filter weights minimize the cost function $J(\boldsymbol{w}) \triangleq E\{e^2\}$, where $E\{\cdot\}$ denotes statistical expectation. In an environment of unknown or time-varying signal statistics, the standard LMS algorithm [1] continually attempts to reduce the MSE by updating the weight vector, at each time instant n, as

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) - \mu e(n) \boldsymbol{x}(n) \tag{1}$$

where $\mu > 0$ is the so-called *step-size* of the update. The computational simplicity of the LMS algorithm has made it an attractive choice for several applications in linear signal processing, including noise cancellation, channel equalization, adaptive control, and system identification [1]. However, it suffers from a slow rate of convergence. Further, its implementation requires the choice of an appropriate value for the step-size μ that affects the stability, steady-state MSE, and convergence speed of the algorithm. The stability (convergence) of the LMS algorithm has been extensively studied in the literature [1]. The stability region for mean-square convergence of the LMS algorithm is given by $0 < \mu < (2/\text{trace}(\boldsymbol{R}))$ [1], where $\boldsymbol{R} \triangleq E\{\boldsymbol{x}(n)\boldsymbol{x}^T(n)\}$ is the correlation matrix of the input vector $\boldsymbol{x}(n)$. When the input signal statistics are unknown or the environment is nonstationary, it is difficult to choose a step-size that is guaranteed to lie within the stability region.

The so-called normalized LMS (NLMS) algorithm [1] addresses the problem of step-size design by choosing a *time-varying step-size* $\mu(n)$ in (1) such that the next-step MSE $J_{n+1} \triangleq E\{e^2(n+1)\}$ is minimized at each iteration. This algorithm can be developed from several different points of view; we shall focus on the criterion of minimization of the next-step MSE (see [1, Problem 14]) since this will be the most convenient interpretation when we later consider the case of a general nonlinear filter. The step-size that minimizes the next-step MSE is given by $\mu(n) \approx (1/\|\boldsymbol{x}(n)\|^2)$, where $\|\boldsymbol{x}(n)\|^2 \triangleq \Sigma_{i=1}^N x_i^2(n)$ is the squared Euclidean norm of the input vector $\boldsymbol{x}(n)$. After incorporating an *auxiliary* step-size $\tilde{\mu} > 0$, the NLMS algorithm is written as

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) - \frac{\tilde{\mu}}{\|\boldsymbol{x}(n)\|^2}\, e(n)\, \boldsymbol{x}(n). \tag{2}$$

The theoretical bounds on the stability of the NLMS algorithm are given by $0 < \tilde{\mu} < 2$ [1]. A significant advantage here is that unlike the LMS step-size μ of (1), the auxiliary step-size $\tilde{\mu}$ is *dimensionless*, and the stability region for $\tilde{\mu}$ is independent of the signal statistics. This allows for an easier step-size design with guaranteed stability (convergence) of the algorithm. Further, the NLMS algorithm has a potentially faster convergence than the LMS algorithm [2]. The NLMS algorithm can also be alternatively interpreted as a modification of the LMS algorithm of (1), where the update term is divided (normalized)

Manuscript received May 4, 1998; revised January 26, 1999. This work was supported in part by the National Science Foundation under Grant MIP-9530923. The associate editor coordinating the review of this paper and approving it for publication was Dr. Hitoshi Kiya.

The authors are with the Department of Electrical and Computer Engineering, University of Delaware, Newark, DE 19716 USA.

Publisher Item Identifier S 1053-587X(99)05419-7.

by the squared-norm $||\boldsymbol{x}(n)||^2$ so that the update stays bounded even when the input vector $\boldsymbol{x}(n)$ becomes large in magnitude.

In this paper, we generalize the NLMS algorithm of (2) by deriving a class of nonlinear normalized LMS-type (NLMS-type) algorithms [3] that are applicable to a wide variety of nonlinear filter structures. Although linear filters are useful in a number of applications, several practical situations exist in which nonlinear processing of the signals involved is essential in order to maintain an acceptable level of performance. Applications of nonlinear models and filtering include polynomial (Volterra) filters used in nonlinear channel equalization and system identification [4]–[6] and the class of *order statistic* filters used in image processing [7], [8]. Several *adaptive* nonlinear filters have also been developed based on Volterra and order statistic filters [9]–[11]. Consider now the case of an arbitrary nonlinear filter whose output is given by $y \equiv y(\boldsymbol{w}, \boldsymbol{x})$. Under the MSE criterion, the LMS algorithm of (1) can be generalized to yield the class of nonlinear LMS-type adaptive algorithms (see Section II)

$$w_i(n+1) = w_i(n) - \mu e(n)\, \frac{\partial y}{\partial w_i}\,(n) \qquad i = 1, 2, \cdots, N. \tag{3}$$

Note that (3) can be applied to any nonlinear filter for which the derivatives $(\partial y/\partial w_i)(n)$ exist. The above algorithm inherits the main problem of the LMS algorithm of (1), namely, the difficulty in choosing the step-size $\mu > 0$. Unlike the linear case, where step-size bounds are available, no theoretical analysis of the LMS-type algorithm of (3) has been performed to derive the stability range for μ. This is due to the mathematical complexity inherent in most nonlinear filters. Consequently, there is a strong motivation (much more than in the linear case) to develop automatic step-size choices to guarantee stability of the LMS-type algorithm. The class of nonlinear normalized LMS-type algorithms developed in this paper addresses this problem of step-size design. Just as the linear NLMS algorithm of (2) is developed from the classical LMS algorithm of (1), we obtain a general nonlinear NLMS-type algorithm from the LMS-type algorithm of (3) by choosing a time-varying step-size that minimizes the next-step MSE at each iteration. As in the linear case, we introduce a dimensionless auxiliary step-size whose stability range has the advantage of being independent of the signal statistics. The stability region could therefore be determined empirically for any given nonlinear filter. We show through computer simulations that these NLMS-type algorithms have, in general, a faster convergence than their LMS-type counterparts. As one of our anonymous reviewers pointed out, normalization has been employed as a heuristic method for stability in the areas of numerical analysis and optimization. However, it has *never* been used in the field of *nonlinear adaptive signal processing*. We have introduced this valuable tool for the first time into the area of nonlinear signal processing and provided a theoretical basis for this technique as well.

The paper is organized as follows. Section II presents LMS-type nonlinear adaptive algorithms. The class of nonlinear normalized LMS-type (NLMS-type) algorithms is derived in Section III. In Section IV, we apply the NLMS-type algorithms to two specific nonlinear filter structures: the well-known polynomial (Volterra) filters and the recently proposed class of *Myriad* filters. The performance of these algorithms is demonstrated through computer simulations presented in Section V.

II. Nonlinear LMS-Type Adaptive Algorithms

In this section, we briefly review the derivation of nonlinear LMS-type adaptive algorithms that have been used in the literature for the optimization of several types of nonlinear filters. Consider a general nonlinear filter with the filter output given by $y \equiv y(\boldsymbol{w}, \boldsymbol{x})$, where $\boldsymbol{x}$ and $\boldsymbol{w}$ are the N-long input and weight vectors, respectively. The *optimal* filter weights minimize the mean square error (MSE) cost function

$$J(\boldsymbol{w}) = E\{e^2\} = E\{(y(\boldsymbol{w}, \boldsymbol{x}) - d)^2\} \tag{4}$$

where d is the desired signal, and $e = y - d$ is the filtering error. The *necessary* conditions for filter optimality are obtained by setting the gradient of the cost function equal to zero

$$\frac{\partial J(\boldsymbol{w})}{\partial w_i} = 2E\left\{ e \frac{\partial y}{\partial w_i} \right\} = 0, \qquad i = 1, 2, \cdots, N. \tag{5}$$

Due to the nonlinear nature of $y(\boldsymbol{w}, \boldsymbol{x})$, and, consequently, of the equations in (5), it is extremely difficult to solve for the optimal weights in closed form. The *method of steepest descent* is a popular technique that addresses this problem by updating the filter weights using the following equation in an attempt to continually reduce the MSE cost function:

$$w_i(n+1) = w_i(n) - \frac{1}{2}\, \mu \frac{\partial J}{\partial w_i}\,(n) \qquad i = 1, 2, \cdots, N \tag{6}$$

where $w_i(n)$ is the ith weight at iteration n, $\mu > 0$ is the *step-size* of the update, and the gradient at the nth iteration is given from (5) as

$$\frac{\partial J}{\partial w_i}\,(n) = 2E\left\{ e(n) \frac{\partial y}{\partial w_i}\,(n) \right\}, \qquad i = 1, 2, \cdots, N. \tag{7}$$

In order to utilize (6), we require a knowledge of the statistics of the signals involved. When the signal statistics are either unknown or rapidly changing (as in a nonstationary environment), we use *instantaneous estimates* for the gradient $(\partial J/\partial w_i)(n)$. To this end, removing the expectation operator in (7) and substituting into (6), we obtain the class of nonlinear LMS-type adaptive algorithms:

$$w_i(n+1) = w_i(n) - \mu\, e(n) \frac{\partial y}{\partial w_i}\,(n) \qquad i = 1, 2, \cdots, N. \tag{8}$$

Note that for a linear filter $(y = \boldsymbol{w}^T \boldsymbol{x})$, we have $(\partial y/\partial w_i) = x_i$, and (8) reduces, as expected, to the LMS algorithm of (1). As mentioned in Section I, the mathematical complexity inherent in nonlinear filtering has prevented a theoretical analysis of (8) in order to determine the bounds on the step-size μ for stable operation of the algorithm. There is therefore a strong motivation for the development of automatic step-size choices that guarantee the stability of the LMS-type algorithm.

The normalized LMS-type algorithms derived in the following section address this problem by choosing an optimal time-varying step-size $\mu(n)$ at each iteration.

III. Nonlinear Normalized LMS-Type (NLMS-Type) Algorithms

We derive the class of nonlinear normalized LMS-type (NLMS-type) algorithms by choosing a *time-varying* step-size $\mu(n) > 0$ in the LMS-type algorithm of (8). In order to do this, we start by rewriting the steepest descent algorithm of (6), using (5) to obtain

$$w_i(n+1) = w_i(n) - \mu E\left\{e(n)\frac{\partial y}{\partial w_i}(n)\right\}. \tag{9}$$

Now, at the nth iteration, the *next-step* MSE is defined by

$$J_{n+1} \triangleq J(\boldsymbol{w}(n+1)) = E\{e^2(n+1)\} \tag{10}$$

where the next-step filtering error $e(n+1)$ is given by

$$\begin{aligned} e(n+1) &= y(n+1) - d(n+1) \\ &\equiv y(\boldsymbol{w}(n+1), \boldsymbol{x}(n+1)) - d(n+1). \end{aligned} \tag{11}$$

Note that J_{n+1} depends on the next-step weight vector $\boldsymbol{w}(n+1)$, which in turn is a function of $\mu > 0$. We obtain the NLMS-type algorithm from (9) by determining the *optimal* step-size, which is denoted by $\mu_o(n)$, that minimizes $J_{n+1} \equiv J_{n+1}(\mu)$:

$$\mu_o(n) \triangleq \arg\min_{\mu > 0} J_{n+1}(\mu). \tag{12}$$

As mentioned in Section I, the criterion of minimization of the next-step MSE is one of the several interpretations of the normalized LMS algorithm of (2) (see [1, Problem 14]). We use this criterion here since, out of all the interpretations, this extends most easily to the case of a general nonlinear filter. To determine $\mu_o(n)$, we need an expression for the derivative function $(\partial/\partial\mu)J_{n+1}(\mu)$. Referring to (10) and (11), we can use the chain rule to write

$$\frac{\partial}{\partial \mu} J_{n+1}(\mu) = \sum_{j=1}^{N} \frac{\partial J_{n+1}(\cdot)}{\partial w_j(n+1)} \cdot \frac{\partial w_j(n+1)}{\partial \mu}. \tag{13}$$

To evaluate the expressions in (13), we first define the following functions for notational convenience [see (7)]:

$$\begin{aligned} \Lambda_j(n) \triangleq\ -\frac{1}{2}\frac{\partial J}{\partial w_j}(n) &= -E\left\{e(n)\frac{\partial y}{\partial w_j}(n)\right\} \\ j &= 1, 2, \cdots, N. \end{aligned} \tag{14}$$

We can then rewrite the update in (9) as

$$w_i(n+1) = w_i(n) + \mu \cdot \Lambda_i(n), \qquad i = 1, 2, \cdots, N. \tag{15}$$

Using (15), we obtain one of the terms to be evaluated in (13) as

$$\frac{\partial w_j(n+1)}{\partial \mu} = \Lambda_j(n), \qquad j = 1, 2, \cdots, N. \tag{16}$$

The other term in (13) can be written, using (14), as

$$\begin{aligned} \frac{\partial J_{n+1}(\cdot)}{\partial w_j(n+1)} &\equiv \frac{\partial}{\partial w_j(n+1)} J(\boldsymbol{w}(n+1)) \\ &\equiv \frac{\partial J}{\partial w_j}(n+1) = -2\Lambda_j(n+1). \end{aligned} \tag{17}$$

Returning to the derivative function in (13), we substitute (16) and (17) into (13) to obtain

$$\frac{\partial}{\partial \mu} J_{n+1}(\mu) = -2\sum_{j=1}^{N} \Lambda_j(n+1)\Lambda_j(n). \tag{18}$$

Before simplifying (18) further, we note from (15) that

$$\mu = 0 \Rightarrow \boldsymbol{w}(n+1) = \boldsymbol{w}(n). \tag{19}$$

This leads to the significant observation that $\mu = 0$ *corresponds to quantities at time* n, *whereas* $\mu > 0$ *corresponds to quantities at time* $(n+1)$. Consequently, we notice in (18) that $\Lambda_j(n+1)$ depends on μ, whereas $\Lambda_j(n)$ does not. To emphasize this fact, we define the function

$$\begin{aligned} F_j(\mu) \triangleq\ \Lambda_j(n+1) &= -E\left\{e(n+1)\frac{\partial y}{\partial w_j}(n+1)\right\} \\ j &= 1, 2, \cdots, N. \end{aligned} \tag{20}$$

It follows that

$$\begin{aligned} F_j(0) = \Lambda_j(n) &= -E\left\{e(n)\frac{\partial y}{\partial w_j}(n)\right\} \\ j &= 1, 2, \cdots, N. \end{aligned} \tag{21}$$

Using (20) and (21), we have the following expression for the derivative of $J_{n+1}(\mu)$:

$$\frac{\partial}{\partial \mu} J_{n+1}(\mu) = -2\sum_{j=1}^{N} F_j(\mu)F_j(0). \tag{22}$$

Due to the nonlinearity of the quantities in the above equation [see (20) and (21)], it is generally very difficult to simplify (22) further in closed form. We therefore resort to an approximation by employing a first-order Taylor's series expansion (linearization) of the functions $F_j(\mu)$ about the point $\mu = 0$, assuming a small step-size $\mu > 0$:

$$\begin{aligned} F_j(\mu) &\approx F_j(0) + \mu F_j'(0) \\ &= F_j(0) + \mu\left[\frac{\partial}{\partial \mu} F_j(\mu)\bigg|_{\mu=0}\right]. \end{aligned} \tag{23}$$

Using this approximation in (22), we obtain

$$\frac{\partial}{\partial \mu} J_{n+1}(\mu) \approx -2\left[\sum_{j=1}^{N} F_j^2(0) + \mu \sum_{j=1}^{N} F_j'(0)\, F_j(0)\right]. \tag{24}$$

Notice that this also implies a linearization of the derivative function $(\partial/\partial\mu)J_{n+1}(\mu)$. This, in turn, is equivalent to approximating the next-step MSE $J_{n+1}(\mu)$ as a *quadratic* function of μ. Under these assumptions, the optimal step-size

$\mu_o(n)$ of (12) is found by setting the (approximate) derivative of $J_{n+1}(\mu)$ to zero as

$$\mu_o(n)\colon \frac{\partial}{\partial\mu}\, J_{n+1}(\mu)\bigg|_{\mu=\mu_o(n)} = 0. \tag{25}$$

In order to see if (25) leads to a *minimum* (rather than a maximum) of $J_{n+1}(\mu)$, we note from (22) that

$$\frac{\partial}{\partial\mu} J_{n+1}(\mu)\bigg|_{\mu=0} = -2\sum_{j=1}^{N} F_j^2(0) < 0. \tag{26}$$

Thus, $J_{n+1}(\mu)$ is (predictably) *decreasing* at $\mu = 0$. Therefore, it is reasonable to assume that the quadratic approximation of $J_{n+1}(\mu)$ attains its global *minimum* at some step-size $\mu > 0$. Using (24) in (25), we then obtain a closed-from, albeit approximate, expression for the optimal step-size:

$$\mu_o(n) \approx -\frac{\sum_{j=1}^{N} F_j^2(0)}{\sum_{j=1}^{N} F_j'(0)\, F_j(0)} \tag{27}$$

where, from (21)

$$F_j(0) = -E\left\{e(n)\frac{\partial y}{\partial w_j}\,(n)\right\}, \qquad j = 1, 2, \cdots, N \tag{28}$$

is independent of μ and depends only on the signal statistics at time n. We see from (27) that our remaining task is to evaluate $F_j'(0)$; this expression is derived in the Appendix and is given by

$$\begin{aligned} F_j'(0) = -\sum_{k=1}^{N} F_k(0)\, E\bigg\{&e(n)\, \frac{\partial^2 y}{\partial w_k \partial w_j}\,(n) \\ &+ \frac{\partial y}{\partial w_k}\,(n)\, \frac{\partial y}{\partial w_j}\,(n)\bigg\}. \end{aligned} \tag{29}$$

We can now substitute (29) and (28) into (27) and obtain an expression for the optimal step-size $\mu_o(n)$. Note that (29) and (28) involve statistical expectations $E\{\cdot\}$. These expectations are difficult to obtain in an environment of unknown or time-varying signal statistics. We therefore resort to using *instantaneous estimates* of these expectations, just as in the derivation of the conventional (linear) LMS algorithm of (1) or of the nonlinear LMS-type algorithm of (8). To this end, removing the expectation operator in (29) and (28), using the resulting expressions in (27), and performing some straightforward simplifications, we obtain an expression for the optimal step-size:

$$\mu_o(n) \approx \left(\frac{1}{1+\mathcal{E}(n)}\right) \cdot \frac{1}{\sum_{j=1}^{N}\left(\frac{\partial y}{\partial w_j}\,(n)\right)^2} \tag{30}$$

where

$$\mathcal{E} \triangleq e \cdot \frac{\sum_{j=1}^{N} \frac{\partial y}{\partial w_j}\left[\sum_{k=1}^{N} \frac{\partial y}{\partial w_k}\, \frac{\partial^2 y}{\partial w_k \partial w_j}\right]}{\left[\sum_{j=1}^{N}\left(\frac{\partial y}{\partial w_j}\right)^2\right]^2}. \tag{31}$$

In order to obtain a more manageable and practically useful expression for $\mu_o(n)$, we make the simplifying assumption that

$$|\mathcal{E}(n)| \ll 1. \tag{32}$$

This is a justifiable assumption for two reasons. First, we see from (31) that $\mathcal{E}(n)$ is proportional to the filtering error $e(n)$. Since the error $e(n)$ is continually reduced in magnitude by the adaptive algorithm of (9), $|\mathcal{E}(n)|$ in turn becomes smaller during succeeding iterations, making (32) a progressively better approximation. Second, referring to the numerator of $\mathcal{E}$ in (31), we see that for mild nonlinearities in the filter, the approximation in (32) amounts to neglecting the effects of the second-order *cross-derivatives* $\partial^2 y/\partial w_k \partial w_j$ in relation to the first-order derivatives $\partial y/\partial w_j$. Using the assumption of (32) in (30), we finally obtain a simplified expression for the optimal step-size:

$$\mu_o(n) \approx \frac{1}{\sum_{j=1}^{N}\left(\frac{\partial y}{\partial w_j}\,(n)\right)^2}. \tag{33}$$

After incorporating an *auxiliary* step-size $\tilde{\mu} > 0$, just as in the conventional (linear) NLMS algorithm of (2), we can then write the time-varying step-size to be used in the steepest-descent algorithm of (9) as

$$\mu(n) = \tilde{\mu} \cdot \mu_o(n) \approx \frac{\tilde{\mu}}{\sum_{j=1}^{N}\left(\frac{\partial y}{\partial w_j}\,(n)\right)^2}. \tag{34}$$

Finally, on using instantaneous estimates by removing the expectation operator in the steepest-descent algorithm of (9), we obtain the following class of *nonlinear normalized LMS-type adaptive filtering algorithms*:

$$\begin{aligned} w_i(n+1) &= w_i(n) - \frac{\tilde{\mu}}{\sum_{j=1}^{N}\left(\frac{\partial y}{\partial w_j}\,(n)\right)^2} \cdot e(n)\, \frac{\partial y}{\partial w_i}\,(n) \\ & \qquad i = 1, 2, \cdots, N. \end{aligned} \tag{35}$$

This algorithm has several advantages.

- It is applicable to a wide variety of nonlinear filters; in fact, it applies to any nonlinear filter for which the filter output y is an analytic function of each of the filter weights w_i (so that derivatives of all orders exist).
- The auxiliary step-size $\tilde{\mu}$ is dimensionless, and the stability region for $\tilde{\mu}$ is independent of the signal statistics. As a result, the stability region could be determined *empirically* for any particular nonlinear filter of interest.

- This algorithm has a potentially faster convergence than its LMS-type counterpart of (8), as demonstrated by our simulation results in Section V.
- It can also be interpreted as a modification of the LMS-type algorithm of (8) in which the update term is divided (*normalized*) by the Euclidean squared norm of the set of values $(\partial y/\partial w_i)(n), i = 1, 2, \cdots, N$ in order to ensure algorithm stability when these values become large in magnitude.

It is important to note the following two approximations used in deriving the nonlinear NLMS-type algorithm of (35):

- Linearization of the function $F_j(\mu)$ defined in (20) about the point $\mu = 0$ [see (23)]: This approximation holds good for any nonlinear filter, as long as the time-varying step-size $\mu(n)$ is sufficiently small. Note that this is not, at least directly, a restriction on the *auxiliary* step-size $\tilde{\mu}$.
- The assumption that $|\mathcal{E}(n)| \ll 1$ [see (31) and (32)]: Note that the validity of this approximation has to be investigated on a case-by-case basis for each nonlinear filter of interest.

Consider now the special case of the linear filter, for which we have $y = \boldsymbol{w}^T\boldsymbol{x}$, leading to $(\partial y/\partial w_i) = x_i, i = 1, 2, \cdots, N$. It is then easily seen that (35) reduces predictably to the (linear) NLMS algorithm of (2). A significant point to note here is that we do not require any of the approximations [see (23) and (32)] that were used to obtain (35); the derivation in this case is exact. Indeed, when the filter is linear, the function $F_j(\mu)$ of (20) can be shown to be *linear* in μ, thus eliminating the need for the linearization approximation. Further, the expression $\mathcal{E}$ of (31) is identically equal to zero for the linear filter, making the approximation (32) unnecessary.

IV. Normalized Adaptive Volterra and Myriad Filters

As mentioned in Section III, the NLMS-type adaptive algorithm of (35) is applicable to any nonlinear filter for which the filter output y is an analytic function of the filter parameters (weights) $w_i, i = 1, 2, \cdots, N$ so that its derivatives of all orders exist. In this section, we apply this normalized adaptive algorithm to two specific types of nonlinear filter structures, namely, *Volterra* and *Myriad* filters. The performance of the resulting algorithms will be investigated through computer simulation examples in Section V.

A. Volterra Filters

The Volterra filter belongs to a particularly simple class of nonlinear filters having the property that *the filter output is linear in the filter parameters (weights)*. Given an $N \times 1$ input (observation) vector $\boldsymbol{x}$, the filter output in this class is given by

$$y = \boldsymbol{h}^T\boldsymbol{z} = \boldsymbol{h}^T f(\boldsymbol{x}) \tag{36}$$

where $\boldsymbol{h}$ is an $M \times 1$ vector of filter parameters, and $f: \mathcal{R}^N \mapsto \mathcal{R}^M$ is a (generally nonlinear) mapping that transforms the $N \times 1$ input vector $\boldsymbol{x}$ into an $M \times 1$ *modified observation vector* $\boldsymbol{z}$. Included in this class are linear filters, *Ll* filters based on order statistics [12], [13], permutation filters (which are generalizations of both linear and order statistic filters) [14], and Volterra filters [4], among others. In order to obtain the NLMS-type adaptive filtering algorithm for this class, we first see from (36) that

$$\frac{\partial y}{\partial h_i} = z_i, \qquad i = 1, 2, \cdots, M. \tag{37}$$

Using this in (8), the LMS-type algorithm to update the $M \times 1$ parameter vector $\boldsymbol{h}$ can be written as

$$\boldsymbol{h}(n+1) = \boldsymbol{h}(n) - \mu e(n)\boldsymbol{z}(n) \tag{38}$$

and the corresponding NLMS-type algorithm can be written from (35) as

$$\boldsymbol{h}(n+1) = \boldsymbol{h}(n) - \frac{\tilde{\mu}}{\|\boldsymbol{z}(n)\|^2}\, e(n)\boldsymbol{z}(n) \tag{39}$$

where $\|\boldsymbol{z}(n)\|^2 \triangleq \Sigma_{i=1}^{M}\ z_i^2(n)$. Notice the simplicity of the above two equations, which are similar to the (linear) LMS and NLMS algorithms of (1) and (2), respectively. However, unlike the linear case where there are N filter parameters (weights), the number of parameters M for a nonlinear filter in this class is typically significantly greater than the number of input samples N. Consider now the special case of the Volterra filter, which has found widespread use in nonlinear signal processing [6], [9]. The output of this filter is given by

$$\begin{aligned} y &\triangleq \sum_{i=1}^{N} w_1(i)\, x_i + \sum_{i=1}^{N}\sum_{j=1}^{N} w_2(i,j)\, x_i\, x_j + \cdots \\ &= \boldsymbol{h}_1^T\boldsymbol{z}_1 + \boldsymbol{h}_2^T\boldsymbol{z}_2 + \cdots \\ &= \boldsymbol{h}^T\boldsymbol{z} \end{aligned} \tag{40}$$

where

$$\begin{aligned} \boldsymbol{h} &\triangleq [\boldsymbol{h}_1^T|\boldsymbol{h}_2^T|\cdots]^T \\ &= [w_1(1), w_1(2), \cdots, w_1(N) \\ &\quad \cdot |w_2(1,1), \cdots, w_2(N,N)|w_3(1,1,1), \cdots]^T \end{aligned} \tag{41}$$

and

$$\begin{aligned} \boldsymbol{z} &\triangleq [\boldsymbol{z}_1^T|\boldsymbol{z}_2^T|\cdots]^T \\ &= [x_1, x_2, \cdots, x_N|x_1^2, x_1x_2, \cdots, x_N^2|\cdots]^T \end{aligned} \tag{42}$$

are the filter parameter vector and the modified observation vector, respectively. Thus, the modified observation vector $\boldsymbol{z}$ contains all possible cross-products of the input samples $\{x_i\}$. In practice, we use a *truncated Volterra series*, obtaining a pth-order Volterra filter by using p terms in the series of (40), with a parameter vector $\boldsymbol{h} = [\boldsymbol{h}_1^T|\boldsymbol{h}_2^T|\cdots|\boldsymbol{h}_p^T]^T$ and a modified observation vector $\boldsymbol{z} = [\boldsymbol{z}_1^T|\boldsymbol{z}_2^T|\cdots|\boldsymbol{z}_p^T]^T$. The normalized adaptive Volterra filtering algorithm is easily written down by substituting (40)–(42) into (39), where the quantity $\|\boldsymbol{z}(n)\|^2$ in (39) is given by

$$\begin{aligned} \|\boldsymbol{z}(n)\|^2 &= \|\boldsymbol{z}_1(n)\|^2 + \|\boldsymbol{z}_2(n)\|^2 + \cdots \\ &= \sum_{i=1}^{N} x_i^2(n) + \sum_{i=1}^{N}\sum_{j=1}^{N} x_i^2(n)\, x_j^2(n) + \cdots \end{aligned} \tag{43}$$

As noted before, the algorithm of (39) is similar to the linear NLMS algorithm of (2) since the filter output is linear in the filter parameters. Therefore, it is reasonable to expect the stability region for $\tilde{\mu}$ to be the same as in the linear case, where it is $0 < \tilde{\mu} < 2$ [1]. Our simulations confirm this fact for the case of the second-order Volterra filter (see Section V).

B. Myriad Filters

As a second example of nonlinear NLMS-type adaptive filtering, we consider the class of *weighted myriad filters*, which have recently been developed for robust signal processing in impulsive environments [15]–[18]. These filters have been derived based on the properties of the heavy-tailed class of α-stable distributions [19], which accurately model impulsive processes. Nonlinear LMS-type adaptive algorithms have also been derived for the optimization of these filters [17]. Given an input vector $\boldsymbol{x}$ and a vector of real-valued weights $\boldsymbol{w}$, both of length N, the weighted myriad filter output is given by

$$y \equiv y_K(\boldsymbol{w}, \boldsymbol{x}) \stackrel{\Delta}{=} \arg\min_{\theta} \sum_{i=1}^{N} \cdot \log[K^2 + |w_i| \cdot (\theta - \text{sgn}(w_i)\, x_i)^2] \tag{44}$$

where sgn$(\cdot)$ is the *sign function*, and K is called the *linearity parameter* since y_K reduces to the (linear) *weighted mean* of the input samples as $K \to \infty$: $y_\infty = \Sigma_{j=1}^N w_j x_j / \Sigma_{j=1}^N w_j$. For finite K, however, the filter output depends only on the N-long filter parameter vector $\boldsymbol{h} \stackrel{\Delta}{=} \boldsymbol{w}/K^2$:

$$y \equiv y(\boldsymbol{h}, \boldsymbol{x}) \stackrel{\Delta}{=} \arg\min_{\theta} \sum_{i=1}^{N} \cdot \log[1 + |h_i| \cdot (\theta - \text{sgn}(h_i) \cdot x_i)^2]. \tag{45}$$

The filter can therefore be adaptively optimized by updating the parameter vector $\boldsymbol{h}$. Now, it can be shown [18] that

$$\frac{\partial y}{\partial h_i} = -\frac{\delta_i}{\Delta}, \qquad i = 1, 2, \cdots, N \tag{46}$$

where

$$\delta_i = \frac{u_i}{(1 + |h_i| u_i^2)^2}, \qquad i = 1, 2, \cdots, N$$

and

$$\Delta = \sum_{j=1}^{N} |h_j| \, \frac{1 - |h_j| u_j^2}{(1 + |h_j| u_j^2)^2}$$

with $u_i = \text{sgn}(h_i) \cdot y - x_i, i = 1, 2, \cdots, N$. Substituting (46) into (8) and (35), we obtain the LMS-type weighted myriad filtering algorithm:

$$h_i(n+1) = h_i(n) + \mu\, e(n)\, \frac{\delta_i(n)}{\Delta(n)} \qquad i = 1, 2, \cdots, N. \tag{47}$$

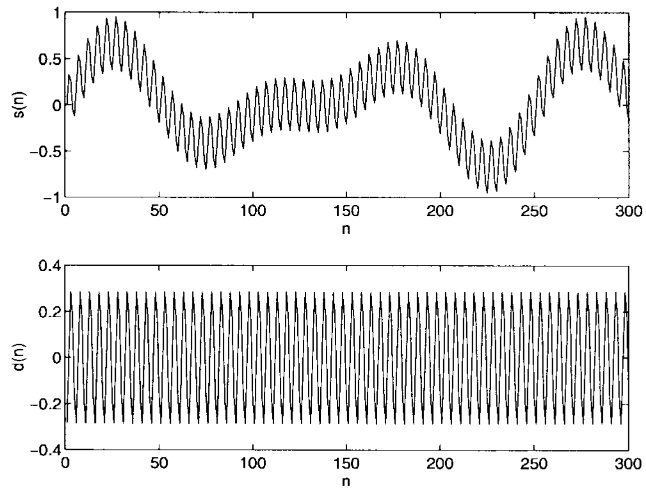


Fig. 1. Clean sum of sinusoids $s(n)$ (top) and desired highpass component $d(n)$ (bottom).

The corresponding normalized adaptive (NLMS-type) algorithm is given by

$$h_i(n+1) = h_i(n) + \tilde{\mu}\, e(n)\, \frac{\Delta(n)}{\sum_{j=1}^{N} \delta_j^2(n)}\, \delta_i(n) \qquad i = 1, 2, \cdots, N. \tag{48}$$

V. Simulation Results

The normalized adaptive algorithms of Section IV are investigated in this section through two computer simulation examples. In the first example, the weighted myriad filter is applied to the problem of adaptive highpass filtering of a sum of sinusoids in an impulsive noise environment. The second example involves identification of a nonlinear system using measurements of its input and output in a noiseless environment with the input–output relationship modeled by a truncated Volterra series.

Example 1: The normalized adaptive weighted myriad filtering algorithm of Section IV-B was used to extract the high-frequency tone from a sum of three sinusoidal signals corrupted by impulsive noise. The observed signal was given by $x(n) = s(n) + v(n)$, where $s(n) = \Sigma_{k=0}^{2} a_k \sin(2\pi f_k n)$ is the *clean* sum of sinusoids. Fig. 1 shows a segment of the signal $s(n)$ consisting of sinusoids at digital frequencies $f_0 = 0.008, f_1 = 0.012$, and $f_2 = 0.2$, with amplitudes $(a_0, a_1, a_2) = (0.4, 0.3, 0.3)$. The *desired signal* $d(n)$, which is also shown in the figure, is the sinusoid at the highest frequency f_2. The additive noise process $v(n)$ was chosen to have a zero-mean symmetric α-stable distribution [19] with a *characteristic exponent* $\alpha = 1.6$ and a *dispersion* $\gamma = 0.02$. Impulsive noise is well-modeled by the heavy-tailed class of α-stable distributions, which includes the Gaussian distribution as the special case when $\alpha = 2$. The characteristic exponent $(0 < \alpha \le 2)$ measures the heaviness of the tails (a smaller α indicates heavier tails), whereas the dispersion γ decides the spread of the distribution around the origin. These distributions

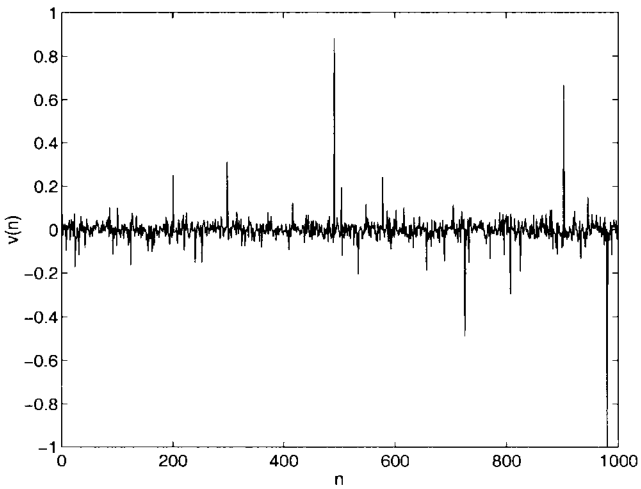


Fig. 2. Additive α-stable noise signal $v(n)$ (characteristic exponent $\alpha = 1.6$, dispersion $\gamma = 0.02$).

have infinite variance for $0 < \alpha < 2$. The dispersion γ is analogous to the variance of a distribution; when $\alpha = 2$, γ is equal to half the variance of the Gaussian distribution. Fig. 2 shows the additive α-stable noise signal $v(n)$ used in our simulations. As the figure shows, the chosen noise process simulates low-level Gaussian-type noise as well as impulsive interference consisting of occasional spikes in the signal.

The LMS-type algorithm of (47) and the normalized LMS-type algorithm of (48) in Section IV-B were used to train a weighted myriad filter to extract the desired highpass signal $d(n)$ from the noisy observed signal $x(n)$. The filter window length was chosen to be $N = 9$. In all cases, the initial weights in the adaptive algorithms were all identical and normalized to sum to unity: $w_i(0) = (1/N) = 0.11, i = 1, 2, \cdots, N$. The linearity parameter was arbitrarily chosen to be $K = 1.0$; recall from Section IV-B that the filter output depends only on $(\boldsymbol{w}/K^2)$. A step-size of $\mu = 0.05$ was used in the LMS-type algorithm; this pushed the algorithm fairly close to its stability limits while maintaining an acceptable final MSE. The normalized LMS-type algorithm was used with an auxiliary step-size $\tilde{\mu} = 1.0$, which is its default value, corresponding to the optimal step-size at each iteration step. Note that this implies an *automatic* step-size choice in the NLMS-type algorithm, without a need for step-size design. In our simulations, we also investigated other choices for $\tilde{\mu}$ and their effect on the final MSE. The final MSE increased with $\tilde{\mu}$, as expected. Further, it was found that the MSE increased much more rapidly with $\tilde{\mu}$ when $\tilde{\mu} > 1.8$, compared with its behavior for $\tilde{\mu} < 1.8$. It is recommended that $\tilde{\mu}$ be chosen nearer to 1.0 in practice to ensure a reliable performance of the algorithm.

Table I shows the final weights obtained by the two algorithms (with $\mu = 0.05$ and $\tilde{\mu} = 1.0$) using 1000 samples of the noisy observed signal $x(n)$. The final trained filters, using both the adaptive algorithms, were successful in accurately extracting the high-frequency sinusoidal component. We do not show the filter outputs (using the trained filter weights) here since they are very close to the desired signal. The

TABLE I
FINAL WEIGHTS OBTAINED BY THE ADAPTIVE WEIGHTED MYRIAD FILTERING ALGORITHMS

Weights	Weighted Myriad ($K = 1.0$)	
w_i	LMS-type	NLMS-type
w_1	0.0249	0.0084
w_2	-0.0945	-0.2118
w_3	-0.1119	-0.2429
w_4	0.0306	0.0151
w_5	0.3427	0.9579
w_6	0.0179	0.0049
w_7	-0.1225	-0.1773
w_8	-0.1011	-0.2284
w_9	0.0220	0.0099

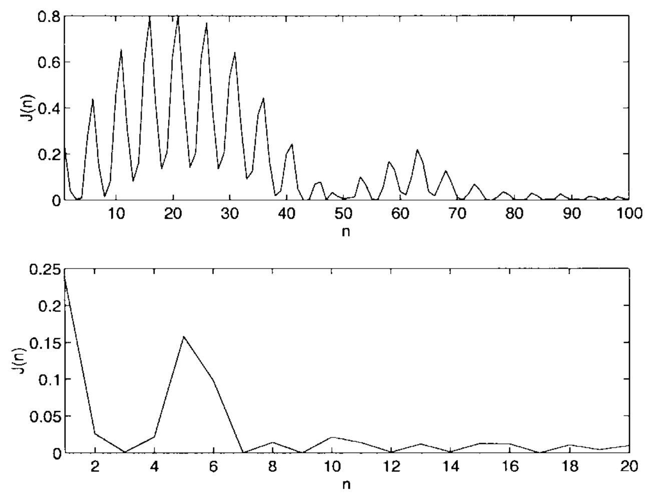


Fig. 3. MSE learning curves for adaptive weighted myriad filtering, LMS-type (top), and NLMS-type (bottom).

MSE's achieved by the trained filters, using both the LMS-type and the NLMS-type algorithms, were almost the same (around 0.0022). This allows for a meaningful comparison of the convergence speeds of the two algorithms. Fig. 3 shows the learning curves (MSE as a function of algorithm iterations) for the LMS-type as well as the NLMS-type algorithms. The LMS-type algorithm converges in about 100 iterations to an MSE below 0.02. On the other hand, the NLMS-type algorithm is about *ten* times faster, converging to the same MSE in just ten iterations. The figure clearly indicates the dramatic improvement in convergence speed when employing the NLMS-type algorithm. Notice the values of the MSE's in these curves; the NLMS-type algorithm has a lower MSE at each iteration step. This is expected since the NLMS-type algorithm was derived to minimize the next-step MSE at each iteration of the LMS-type algorithm.

Example 2: We apply the normalized adaptive Volterra filtering algorithm of (39) to the problem of adaptive identification of an unknown nonlinear system in a noiseless environment. Fig. 4 shows the block diagram representing our simulation experiment. It is assumed that the unknown system can be adequately modeled using a Volterra filter. A known common *input signal* $x(n)$ is applied both to the unknown nonlinear system and to the adaptive Volterra filter. The *desired signal* $d(n)$ is the output of the unknown system.

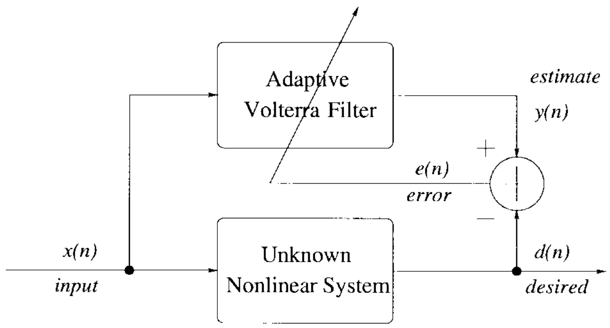


Fig. 4. Nonlinear system identification using a Volterra filter model.

TABLE II
FILTER TAP INPUTS AND CORRESPONDING TAP WEIGHTS OF THE SECOND-ORDER VOLTERRA NONLINEAR SYSTEM TO BE IDENTIFIED, WITH THE INPUT VECTOR $x = [x_1, x_2, x_3, x_4, x_5]^T$

	Tap Input Vector $\mathbf{z}$	Tap Weights H_i	Unknown System
Linear	x_1	H_1	0.30
	x_2	H_2	-0.50
	x_3	H_3	0.70
	x_4	H_4	-0.60
	x_5	H_5	0.20
Quadratic	x_1^2	H_6	0.10
	x_1x_2	H_7	0.25
	x_1x_3	H_8	-0.20
	x_1x_4	H_9	0.08
	x_1x_5	H_{10}	-0.03
	x_2^2	H_{11}	0.40
	x_2x_3	H_{12}	-0.30
	x_2x_4	H_{13}	0.50
	x_2x_5	H_{14}	0.06
	x_3^2	H_{15}	0.65
	x_3x_4	H_{16}	-0.35
	x_3x_5	H_{17}	-0.30
	x_4^2	H_{18}	0.45
	x_4x_5	H_{19}	0.20
	x_5^2	H_{20}	0.15

The objective of the adaptive algorithms is to minimize the mean squared value of the *error* $e(n)$ between the *desired signal* $d(n)$ and the output (*estimate*) $y(n)$ of the adaptive Volterra filter.

In our simulation example, both the unknown system and the adaptive filter were chosen to be second-order (quadratic) Volterra filters with the observation window size $N = 5$. For an N-long observation vector $\boldsymbol{x}$, the outputs of these systems are given by $d = \boldsymbol{H}^T\boldsymbol{z}$ and $y = \boldsymbol{h}^T\boldsymbol{z}$, where the modified observation vector $\boldsymbol{z} = [\boldsymbol{z}_1^T|\boldsymbol{z}_2^T]^T$ is given by (42). Table II shows the parameters (weights) H_i of the system to be identified, along with the corresponding tap inputs for an observation vector $\boldsymbol{x} = [x_1, x_2, x_3, x_4, x_5]^T$. The linear and quadratic parts of the system together constitute a parameter vector $\boldsymbol{H}$ of length $M = 20$. For the *input signal* $x(n)$, we chose a zero-mean white Gaussian process (i.i.d. samples) with unit variance. The objective of the adaptive algorithms is to train the adaptive Volterra filter of Fig. 4 such that its parameter vector $\boldsymbol{h}$ converges to the vector $\boldsymbol{H}$ of Table II.

The LMS-type algorithm of (38) and the NLMS-type algorithm of (39) were applied to train the adaptive Volterra filter. Each algorithm was run for 5000 iterations, with the initial filter weights chosen to be all zero: $\boldsymbol{h}(0) = \mathbf{0}$. This experiment was repeated for 100 independent trials, using a different realization of the input signal $x(n)$ for each trial, in order to obtain the average convergence behavior of the algorithms. For the NLMS-type algorithm, the auxiliary step-size used was its default value $\tilde{\mu} = 1.0$, corresponding to the optimal (automatic) step-size choice at each iteration step. The step-size for the LMS-type algorithm $\mu = 1.0 \times 10^{-2}$ was chosen such that the final ensemble-averaged mean square error (MSE) was comparable with that of the NLMS-type algorithm. As a secondary experiment, we also determined the stability region of the NLMS-type algorithm for second-order Volterra filtering. This was done by increasing $\tilde{\mu}$ until the onset of algorithm instability. The step-size bounds turned out to be around $0 < \tilde{\mu} < 2.0$, confirming our prediction in Section IV-A.

Fig. 5 shows the trajectories of some of the filter weights, averaged over all the 100 independent trials, for both the algorithms. In all cases, the algorithms converged to values that were very close to the weights of the unknown system given in Table II. As the figure shows, the filter weights converge in about 400–500 iterations using the LMS-type algorithm, whereas the convergence is much faster (in about 50–60 iterations) with the normalized LMS-type algorithm. This amounts to about a ten-fold increase in speed for most of the weights. Note that the step-size parameters were chosen so that the final MSE's of the two algorithms are of the same order ($\sim 10^{-31}$); this allows for a fair comparison of their convergence speeds from the weight trajectories.

The ensemble-averaged MSE learning curves of the two algorithms are shown in Fig. 6. These curves plot the squared error, averaged over all the trials, as a function of algorithm iterations. As these curves indicate, the NLMS-type algorithm converges in about 60 iterations, whereas the LMS-type algorithm appears to converge in about 300 iterations. In fact, the LMS-type algorithm takes much longer than this to converge; the MSE's of the two algorithms approach the same order of magnitude ($\sim 10^{-31}$) only after 5000 iterations. As mentioned previously, a meaningful comparison of the two algorithms is possible from these curves since their final MSE's are of the same order. Thus, the NLMS-type algorithm is seen to converge much faster than the LMS-type algorithm at comparable steady-state MSE's.

VI. CONCLUSION

In this paper, we generalized the normalized LMS algorithm (proposed for linear adaptive filtering) by developing a new class of *nonlinear normalized LMS-type (NLMS-type)* adaptive algorithms that are applicable to a wide variety of nonlinear filter structures. These algorithms were derived from the class of nonlinear stochastic gradient (LMS-type) algorithms by choosing an optimal time-varying step-size parameter that minimizes the next-step mean square error (MSE) at each iteration of the adaptive algorithm. By providing for an automatic choice for the step-size, the NLMS-type algorithms eliminate the difficult problem of step-size design that is inherent in all LMS-type algorithms. We illustrated the application of these algorithms through two computer simulation examples: adaptive nonlinear

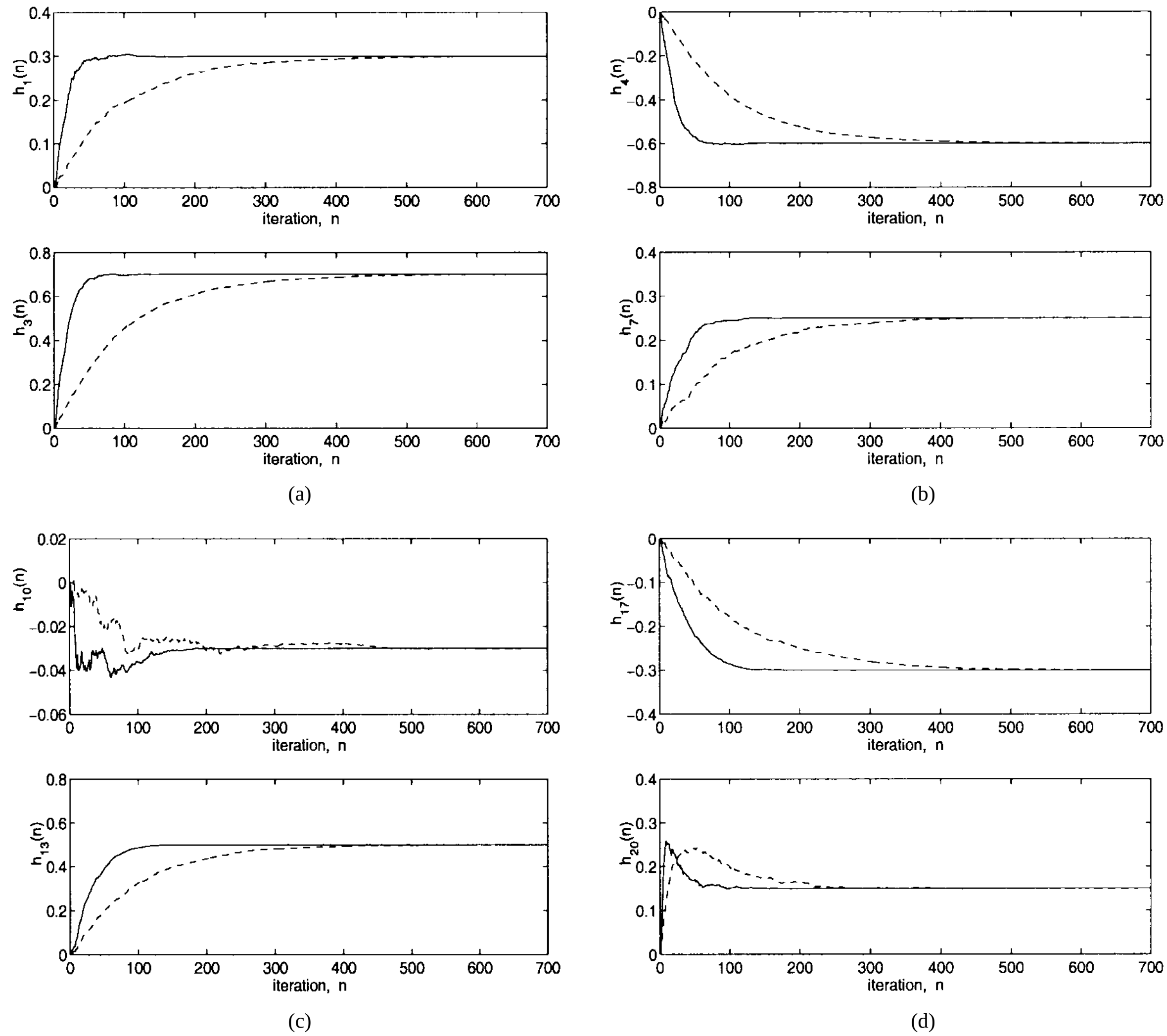


Fig. 5. Ensemble-averaged tap weight trajectories $h_i(n)$ for adaptive Volterra system identification using LMS-type (dashed) and NLMS-type (solid) algorithms. (a) $h_1(n)$ and $h_3(n)$. (b) $h_4(n)$ and $h_7(n)$. (c) $h_{10}(n)$ and $h_{13}(n)$. (d) $h_{17}(n)$ and $h_{20}(n)$.

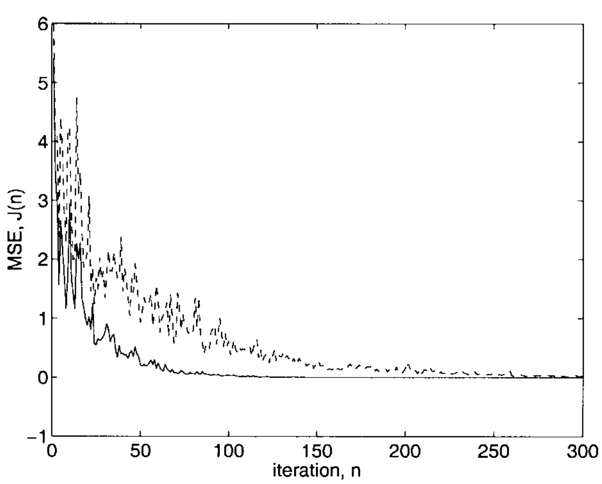


Fig. 6. Ensemble-averaged MSE learning curves for adaptive Volterra system identification using LMS-type (dashed) and NLMS-type (solid) algorithms.

highpass filtering of a sum of sinusoids in impulsive noise using a weighted myriad filter and adaptive identification of a nonlinear system modeled as a second-order Volterra filter. Our experimental results, including ensemble-averaged MSE learning curves and filter weight trajectories, indicate that the NLMS-type algorithms, in general, converge faster than their LMS-type counterparts at comparable steady-state MSE's. These normalized algorithms need to be investigated further by applying them to various nonlinear filters and studying the convergence behavior as well as the steady-state performance in each case.

APPENDIX
DERIVATION OF $F'_j(0)$

In this Appendix, we derive the expression for $F'_j(0)$ given in (29), that is, we show that

$$F'_j(0) \triangleq \frac{\partial}{\partial\mu} F_j(\mu)\bigg|_{\mu=0} = -\sum_{k=1}^{N} F_k(0) \cdot E\left\{ e(n)\, \frac{\partial^2 y}{\partial w_k \partial w_j}\,(n) + \frac{\partial y}{\partial w_k}\,(n)\, \frac{\partial y}{\partial w_j}\,(n) \right\}. \tag{49}$$

From (20), we have

$$F_j(\mu) = - E\left\{e(n+1)\,\frac{\partial y}{\partial w_j}\,(n+1)\right\} \quad j = 1, 2, \cdots, N. \tag{50}$$

Using $e(n+1) = y(n+1) - d(n+1)$ in (50), we can write

$$\begin{aligned} F_j'(\mu) &= \frac{\partial}{\partial \mu}\,F_j(\mu) \\ &= -E\left\{e(n+1)\cdot \frac{\partial}{\partial \mu}\left[\frac{\partial y}{\partial w_j}\,(n+1)\right]\right\} \\ &\quad - E\left\{\frac{\partial y(n+1)}{\partial \mu}\cdot\frac{\partial y}{\partial w_j}\,(n+1)\right\} \\ &\triangleq A(\mu) + B(\mu) \end{aligned} \tag{51}$$

where we have introduced the two terms $A(\mu)$ and $B(\mu)$ for convenience. Note that $y(n+1) \equiv y(\boldsymbol{w}(n+1), \boldsymbol{x}(n+1))$, and $\boldsymbol{w}(n+1)$ is in turn a function of μ, whereas the desired signal $d(n+1)$ is independent of μ. We can use the chain rule of differentiation to expand the first term in (51) as

$$\begin{aligned} A(\mu) &= -E\left\{e(n+1)\cdot\frac{\partial}{\partial \mu}\left[\frac{\partial y}{\partial w_j}\,(n+1)\right]\right\} \\ &= -E\left\{e(n+1)\cdot\sum_{k=1}^{N}\frac{\partial}{\partial w_k(n+1)}\left[\frac{\partial y}{\partial w_j}\,(n+1)\right]\right. \\ &\qquad \left.\cdot\,\frac{\partial w_k(n+1)}{\partial \mu}\right\} \\ &= -\sum_{k=1}^{N} E\left\{e(n+1)\,\frac{\partial^2 y}{\partial w_k \partial w_j}\,(n+1)\right\}\cdot\,\Lambda_k(n) \\ &= -\sum_{k=1}^{N} E\{G_1(\boldsymbol{w}(n+1), \boldsymbol{x}(n+1), d(n+1))\} \\ &\qquad \cdot F_k(0) \end{aligned} \tag{52}$$

where we have used (16) and (21) and introduced a new function $G_1(\boldsymbol{w}(n+1), \boldsymbol{x}(n+1), d(n+1))$ for convenience. Similarly, we can expand the second term in (51) as follows:

$$\begin{aligned} B(\mu) &= -E\left\{\frac{\partial y(n+1)}{\partial \mu}\cdot\frac{\partial y}{\partial w_j}\,(n+1)\right\} \\ &= -E\left\{\left[\sum_{k=1}^{N}\frac{\partial y(n+1)}{\partial w_k(n+1)}\cdot\frac{\partial w_k(n+1)}{\partial \mu}\right]\right. \\ &\qquad \left.\cdot\,\frac{\partial y}{\partial w_j}\,(n+1)\right\} \\ &= -\sum_{k=1}^{N} E\left\{\frac{\partial y}{\partial w_k}\,(n+1)\cdot\frac{\partial y}{\partial w_j}\,(n+1)\right\}\cdot\Lambda_k(n) \\ &= -\sum_{k=1}^{N} E\{G_2(\boldsymbol{w}(n+1), \boldsymbol{x}(n+1), d(n+1))\} \\ &\qquad \cdot F_k(0) \end{aligned} \tag{53}$$

where we have introduced another new function $G_2(\boldsymbol{w}(n+1), \boldsymbol{x}(n+1), d(n+1))$ for convenience. Referring to (51), we see that the quantity desired is

$$F_j'(0) = A(0) + B(0). \tag{54}$$

Recall from (19) that $\mu = 0$ corresponds to $\boldsymbol{w}(n+1) = \boldsymbol{w}(n)$. Using this fact in the definition of $A(\mu)$ given in (52), we can write

$$\begin{aligned} A(0) &= -\sum_{k=1}^{N} E\{G_1(\boldsymbol{w}(n+1), \boldsymbol{x}(n+1), d(n+1))\}|_{\mu=0} \\ &\qquad \cdot\,F_k(0) \\ &= -\sum_{k=1}^{N} E\{G_1(\boldsymbol{w}(n), \boldsymbol{x}(n+1), d(n+1))\}\cdot F_k(0). \end{aligned} \tag{55}$$

Now, noting that $\boldsymbol{w}(n)$ is a deterministic quantity in the steepest descent algorithm of (9) and assuming that the processes $\boldsymbol{x}(n+1)$ and $d(n+1)$ are strictly stationary, we conclude that

$$\begin{aligned} &E\{G_1(\boldsymbol{w}(n), \boldsymbol{x}(n+1), d(n+1))\} \\ &\qquad = E\{G_1(\boldsymbol{w}(n), \boldsymbol{x}(n), d(n))\}. \end{aligned} \tag{56}$$

Notice that the right-hand side in the above equation involves quantities at time n only. Using (56) in (55) and the definition of $G_1(\cdot,\cdot,\cdot)$ given in (52), we finally obtain

$$\begin{aligned} A(0) &= -\sum_{k=1}^{N} E\{G_1(\boldsymbol{w}(n), \boldsymbol{x}(n), d(n))\}\cdot F_k(0) \\ &= -\sum_{k=1}^{N} F_k(0)\cdot\,E\left\{e(n)\,\frac{\partial^2 y}{\partial w_k \partial w_j}\,(n)\right\}. \end{aligned} \tag{57}$$

Following a similar procedure to derive the quantity $B(0)$ from (53), we obtain

$$\begin{aligned} B(0) &= -\sum_{k=1}^{N} E\{G_2(\boldsymbol{w}(n+1), \boldsymbol{x}(n+1), d(n+1))\}|_{\mu=0} \\ &\qquad \cdot F_k(0) \\ &= -\sum_{k=1}^{N} E\{G_2(\boldsymbol{w}(n), \boldsymbol{x}(n), d(n))\}\cdot F_k(0) \\ &= -\sum_{k=1}^{N} F_k(0)\cdot E\left\{\frac{\partial y}{\partial w_k}\,(n)\,\frac{\partial y}{\partial w_j}\,(n)\right\}. \end{aligned} \tag{58}$$

Using (57) and (58) in (54), we finally have the desired expression for $F_j'(0)$:

$$\begin{aligned} F_j'(0) = &-\sum_{k=1}^{N} F_k(0) \\ &\cdot E\left\{e(n)\,\frac{\partial^2 y}{\partial w_k \partial w_j}\,(n) + \frac{\partial y}{\partial w_k}\,(n)\,\frac{\partial y}{\partial w_j}\,(n)\right\}. \end{aligned} \tag{59}$$

REFERENCES

[1] S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[2] M. Rupp, "The behavior of LMS and NLMS algorithms in the presence of spherically invariant processes," *IEEE Trans. Signal Processing*, vol. 41, pp. 1149–1160, Mar. 1993.

[3] S. Kalluri and G. R. Arce, "A general class of nonlinear normalized LMS-type adaptive algorithms," in *Proc. 1998 CISS*, Princeton, NJ, Mar. 1998.

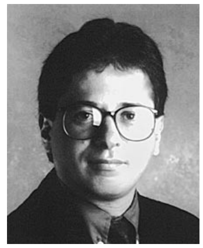
[4] M. Schetzen, *The Volterra and Wiener Theories of Nonlinear Systems*. New York: Wiley, 1980.

[5] S. Benedetto, E. Biglieri, and V. Castellani, *Digital Transmission Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1987.

[6] T. Koh and E. J. Powers, "Second-order Volterra filtering and its application to nonlinear system identification," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, pp. 1445–1455, Dec. 1985.
[7] I. Pitas and A. N. Venetsanopoulos, *Non-linear Digital Filters—Principles and Applications*. Boston, MA: Kluwer, 1990.
[8] ——, "Order statistics in digital image processing," *Proc. IEEE*, vol. 80, Dec. 1992.
[9] V. J. Mathews, "Adaptive polynomial filters," *IEEE Signal Processing Mag.*, vol. 8, pp. 10–26, July 1991.
[10] I. Pitas and A. Venetsanopoulos, "Adaptive filters based on order statistics," *IEEE Trans. Signal Processing*, vol. 39, Feb. 1991.
[11] D. W. Griffith and G. R. Arce, "Partially decoupled Volterra filters: Formulation and LMS adaptation," *IEEE Trans. Signal Processing*, vol. 45, pp. 1485–1494, June 1997.
[12] F. Palmieri and C. G. Boncelet, Jr., "Ll filters—A new class of order statistic filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, May 1989.
[13] P. Ghandi and S. A. Kassam, "Design and performance of combination filters," *IEEE Trans. Signal Processing*, vol. 39, July 1991.
[14] Y.-T. Kim and G. R. Arce, "Permutation filter lattices: A general order-statistic filtering framework," *IEEE Trans. Signal Processing*, vol. 42, Sept. 1994.
[15] J. G. Gonzalez and G. R. Arce, "Weighted myriad filters: A robust filtering framework derived from α-stable distributions," in *Proc. IEEE ICASSP*, Atlanta, GA, 1996.
[16] ——, "Weighted myriad filters: A powerful framework for efficient filtering in impulsive environments," submitted for publication.
[17] S. Kalluri and G. R. Arce, "Adaptive weighted myriad filter algorithms for robust signal processing in α-stable noise environments," *IEEE Trans. Signal Processing*, vol. 46, pp. 322–334, Feb. 1998.
[18] ——, "Robust frequency-selective filtering using weighted myriad filters admitting real-valued weights," submitted for publication.
[19] C. L. Nikias and M. Shao, *Signal Processing with Alpha-Stable Distributions and Applications*. New York: Wiley, 1995.

Sudhakar Kalluri (S'90–M'99) was born in Vijayawada, India, in 1968. He received the B.Tech. degree in electronics and communication engineering from the Indian Institute of Technology, Madras, in 1989, the M.S.E. degree in electrical engineering from Princeton University, Princeton, NJ, in 1991, and the Ph.D. degree in electrical engineering from the University of Delaware, Newark, in 1998.

He has worked at the IBM T.J. Watson Research Center, Yorktown Heights, NY, in the Video and Image Technologies Department. Since 1998, he has been a Postdoctoral Research Associate in the Department of Electrical and Computer Engineering, University of Delaware. He has consulted with the Army Research Laboratories in the area of communications. His research interests include robust signal processing and communications, statistical signal processing, communication theory, signal processing for communications, nonlinear adaptive algorithms, and nonlinear filter theory.



Gonzalo R. Arce (SM'93) was born in La Paz, Bolivia. He received the B.S.E.E. degree with the highest honors from the University of Arkansas, Fayetteville, in 1979, and the M.S. and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, IN, in 1980 and 1982, respectively.

Since 1982, he has been with the Department of Electrical and Computer Engineering, University of Delaware, Newark, where he is currently Professor, Associate Chair, and Fellow in the Center for Advanced Studies. He has consulted for several industrial organizations in the general areas of signal and image processing and digital communications. His research interests include robust signal processing and its applications, communication theory, image processing, communications, and electronic imaging. He holds two U.S. patents.

Dr. Arce has served as Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING, as Guest Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING, and Guest Editor for the OPTICAL SOCIETY OF AMERICA'S OPTICS EXPRESS. He is a member of the Digital Signal Processing Technical Committee of the Circuits and Systems Society and is a member of the Board of Nonlinear Signal and Image Processing.