Exploiting Sparsity in Adaptive Filters

Richard K. Martin, William A. Sethares, Robert C. Williamson, and C. Richard Johnson, Jr., *Fellow, IEEE*

***Abstract*—This paper studies a class of algorithms called natural gradient (NG) algorithms. The least mean square (LMS) algorithm is derived within the NG framework, and a family of LMS variants that exploit sparsity is derived. This procedure is repeated for other algorithm families, such as the constant modulus algorithm (CMA) and decision-directed (DD) LMS.**

Mean squared error analysis, stability analysis, and convergence analysis of the family of sparse LMS algorithms are provided, and it is shown that if the system is sparse, then the new algorithms will converge faster for a given total asymptotic MSE. Simulations are provided to confirm the analysis. In addition, Bayesian priors matching the statistics of a database of real channels are given, and algorithms are derived that exploit these priors. Simulations using measured channels are used to show a realistic application of these algorithms.

***Index Terms*—Adaptive filters, Bayesian priors, measured channels, natural gradient, sparse channels.**

I. Introduction

TRANSMISSION channels are often sparse, that is, most of the taps are small, and only a few taps are large. This includes the traditional notion of sparsity as a few large taps separated by many negligible taps. Optimal equalizers often reflect this sparsity, yet equalization methods such as the least mean square (LMS) algorithm, the constant modulus algorithm (CMA), and decision directed (DD) algorithms do not exploit this *a priori* information. Typical approaches to exploiting sparsity are motivated by complexity reduction (at the expense of a small performance loss), which is often accomplished by only updating a subset of the channel model or equalizer taps [1]–[5]. For example, [4] uses a least-squares based technique performed on the filter input and output to determine if each tap is active (nonzero). If an "activity measure" exceeds a threshold, the tap is considered active and is updated; otherwise, it is not updated. In [5], Sugiyama *et al.* keep a queue of indices of inactive taps and allow an inactive tap to become active once it exits the queue. However, if the newly active tap remains small, it will shortly be returned to the end of the queue. In this way, the taps that are in the queue are not updated. In contrast to these approaches, the exponentiated gradient (EG) algorithm [6] does not try to reduce computational complexity, yet it has been shown to have better performance than typical gradient methods when the target weight vector is sparse [7], [8].

This paper rederives the EG algorithm as an approximate "natural" gradient (NG) descent algorithm [9] and interprets the improved performance under sparsity in terms of a reparameterization of the underlying linear process. The MSE and convergence properties of EG-like algorithms are studied, and algorithms are derived that make use of the CMA and DD cost functions.

Section II introduces the EG algorithm. Section III provides a theoretical analysis of the excess MSE and convergence properties of EG-like algorithms. Section IV gives a framework for deriving algorithms that exploit prior knowledge of the parameters to be estimated, and Section V applies this framework to a variety of traditional algorithms. Sections VI and VII demonstrate the performance of these algorithms for synthetic and real data, respectively, and Section VIII concludes the paper.

II. Problem Setup and Motivation

A data sequence y_k is assumed to be generated from input data x_k in a linear manner. The simplest of the EG algorithms [6, eq. (3.5)] estimates y_k by

$$\hat{y}_k = \sum_i w_k^i x_k^i. \tag{1}$$

In the EG framework, the weights are typically assumed positive. The weight vector $\mathbf{w}_k = [w_k^1, \ldots, w_k^N]^T$ is updated at each iteration k by

$$w_{k+1}^i = w_k^i + \mu w_k^i (y_k - \hat{y}_k) x_k^i \tag{2}$$

where μ is a small positive stepsize. This was derived by an approximation (which is discussed in [6, Sec. 4.4]) of the general update strategy

$$w_{k+1}^i = w_k^i \exp\left(\mu \frac{\partial L(y_k, \hat{y}_k)}{\partial w_k^i}\right) \tag{3}$$

where the cost (or loss) function is

$$L(y_k, \hat{y}_k) = (y_k - \hat{y}_k)^2. \tag{4}$$

Alternatively, (2) can be derived by estimating y_k by (1) but with

$$w_k^i = \gamma(z_k^i) = \tfrac{1}{4}(z_k^i)^2 \tag{5}$$

Manuscript received May 16, 2001; revised April 2, 2002. This work was supported in part by a research contract with Fox Digital. The associate editor coordinating the review of this paper and approving it for publication was Prof. Jose Carlos M. Bermudez.

R. K. Martin and C. R. Johnson, Jr. are with the School of Electrical and Computer Engineering, Cornell University, Ithaca, NY 14853 USA (e-mail: frodo@ece.cornell.edu; johnson@ece.cornell.edu).

W. A. Sethares is with the Department of Electrical and Computer Engineering, University of Wisconsin, Madison, WI 53706 USA (e-mail: sethares@ece.wisc.edu).

R. C. Williamson is with the Department of Telecommunications Engineering, Australian National University, Canberra, Australia (e-mail: Bob.Williamson@anu.edu.au).

Publisher Item Identifier 10.1109/TSP.2002.800414.

for some parameter vector $\mathbf{z} = (z_k^1, \ldots, z_k^N)^T$. If we think of the algorithm as adapting over the underlying z-space, we can use the Euclidean gradient descent

$$z_{k+1}^i = z_k^i - \mu \frac{\partial L(y_k, \hat{y}_k)}{\partial z_k^i}. \tag{6}$$

If we absorb constants into the stepsize, this yields

$$z_{k+1}^i = z_k^i + \underbrace{\mu \dot{\gamma}(z_k^i)(y_k - \hat{y}_k)x_k^i}_{\Delta}. \tag{7}$$

What we are truly interested in is the effective update rule for $\mathbf{w}$ and not $\mathbf{z}$ since that is what is used to generate our estimate $\hat{y}_k$ in (1). The effective update to $\mathbf{w}$ is approximately

$$\begin{aligned} w_{k+1}^i &= \gamma(z_{k+1}^i) = \gamma(z_k^i + \Delta) \cong \gamma(z_k^i) + \dot{\gamma}(z_k^i)\Delta \quad (8)\\ &= w_k^i + \mu \dot{\gamma}^2(z_k^i)(y_k - \hat{y}_k)x_k^i. \quad (9)\end{aligned}$$

Since Δ is $o(\mu)$, the approximation in (8) leading to (9) becomes increasingly accurate for small μ. For the γ given by (5), $\dot{\gamma}^2(z_k^i) = (1/4)(z_k^i)^2 = w_k^i$, which gives us (2) once again. Thus, the EG algorithm can also be thought of as a Euclidean gradient descent algorithm, but the gradient descent is taking place in z-space.

The algorithm (2) can be contrasted with the standard LMS algorithm

$$w_{k+1}^i = w_k^i + \mu(y_k - \hat{y}_k)x_k^i \tag{10}$$

which can be formally derived using the Euclidean gradient descent strategy of (6). The only difference between (2) and (10) is the presence of the w_k^i multiplying the stepsize. In essence, when the current estimate of w_k^i is small, its update is small, and when the current estimate of w_k^i is large, its update is large.

Duttweiler [10] has proposed an algorithm called proportionate normalized LMS (PNLMS) that has similar features. That is, PNLMS modifies NLMS by multiplying the update terms by the tap weights, resulting in faster convergence. The difference between [10] and our work is that we provide a framework for deriving many variations of these algorithms with a means for fitting the algorithm to the environment. However, the results and intuition in [10] should be useful when implementing the variants found in our paper.

Some insight into the effects of the nonlinear modifications we have added to LMS can be obtained by theoretically analyzing the steady-state mean square error (MSE). In [7], the steady-state MSE of (3) when driven with a white input sequence (x_k) with variance σ_x^2 in the presence of white noise with variance σ_n^2 was shown to be

$$\begin{aligned}\lim_{k\to\infty} E(y_k - \hat{y}_k)^2 \\ = \xi = \left(1 + \mu\left(\|\mathbf{w}^*\|_1 - \frac{\|\mathbf{w}^*\|_2}{\|\mathbf{w}^*\|_1}\right)\sigma_x^2\right)\sigma_n^2 \end{aligned} \tag{11}$$

where $\mathbf{w}^*$ is the target weight vector, and $\|\mathbf{w}\|_p = \sqrt[p]{\sum_{i=1}^N |w_i|^p}$, whereas the MSE for LMS is

$$\xi = (1 + \mu N \sigma_x^2)\sigma_n^2. \tag{12}$$

The key term is $(\|\mathbf{w}^*\|_1 - (\|\mathbf{w}^*\|_2/\|\mathbf{w}^*\|_1))$, which provides a measure of the sparsity of the target weight vector. When there is considerable sparsity, this term is small, and the step size can be made larger for faster convergence without adversely affecting the excess MSE. When this term is large there is not much sparsity, and the step size must remain small in order not to increase the MSE. For example, consider the channel $[1, a, a, a, a, 0.5]$. For $|a| = 0.05$, the "measure of sparsity" equals 1.04, whereas for $|a| = 0.5$, it equals 3.07.

In Section V, we will relax the assumption that the weights are positive, allowing for more practical algorithms. First, however, we will consider the MSE and stability behavior of EG-like algorithms (algorithms with component-wise modifications to the stepsize).

III. Analysis

This section derives theoretical expressions for the MSE and stability criteria of the class of algorithms introduced in Section II in a fashion similar to that in [11] and [12].

A. Asymptotic Mean Squared Error

Consider the vector update rule of

$$\begin{aligned}\mathbf{w}_{k+1} &= \mathbf{w}_k - \mu D\hat{\nabla}_k \\ &= \mathbf{w}_k + \mu D(-\nabla_k + \mathbf{\Gamma}_k)\end{aligned} \tag{13}$$

where

- ∇_k gradient of the cost function at time k with respect to $\mathbf{w}_k$;
- $\hat{\nabla}_k$ estimate of this gradient;
- $\mathbf{\Gamma}_k$ gradient noise, as in [11];
- D diagonal matrix that depends on the current value of $\mathbf{w}$.

(In this section, we will assume that $\mathbf{w} \cong \mathbf{w}^*$ since we are considering asymptotic MSE.) This is a more general form of the LMS update rule (in which $D = I$). The EG algorithm (2) is an example of an algorithm of this form, and a more general form is given in Section IV by (33).

The reason that we want to determine the asymptotic MSE is that the result of the next theorem provides a basis for a fair comparison between LMS and algorithms of the form of (13). For a given algorithm and system, it is possible to compute the asymptotic MSE as a function of μ. The step sizes for the different algorithms can then be adjusted such that at convergence, all have the same MSE. Then simulations can be run, and the convergence time can be compared fairly.

Theorem III.1: The asymptotic MSE of (13) for a small step size μ is given by

$$\xi = \xi_{\min}\left(1 + \tfrac{1}{2}\mu \sum_{p=1}^{n} \lambda_p [Q^{-1} D Q]_{p,p}\right) \tag{14}$$

where $Q^{-1}RQ = \Lambda$ (diagonalization), $R = E[\mathbf{X}\mathbf{X}^T]$, and $\lambda_p = \Lambda_{pp}$ (the pth diagonal element of Λ, where p is simply the index for the summation).

Proof: Using the definition of the error system $\mathbf{v}_k = \mathbf{w}_k - \mathbf{w}^*$ and the fact that the gradient can be expressed as $2R\mathbf{v}_k$, where $R = E[\mathbf{X}\mathbf{X}^T]$ (see [12])

$$\begin{aligned}\mathbf{v}_{k+1} &= \mathbf{v}_k + \mu D(-2R\mathbf{v}_k + \mathbf{\Gamma}_k) \\ &= (I - 2\mu DR)\mathbf{v}_k + \mu D\mathbf{\Gamma}_k. \end{aligned} \tag{15}$$

At this point, [11] rotates the coordinate system by diagonalizing R. That is not feasible here because if we diagonalize DR via $Q^{-1}DRQ$, then $Q^{-1}DQ$ may not be diagonal.

To determine the MSE, we need to find the covariance matrix of $\mathbf{v}_k$. Define $C = E[\mathbf{v}_k\ \mathbf{v}_k^T]$, and assume that the system is near convergence [hence, $\text{cov}(\mathbf{v}_{k+1}) \cong \text{cov}(\mathbf{v}_k)$]. Then

$$C = (I - 2\mu DR)C(I - 2\mu DR)^T + \mu^2 D\,\text{cov}(\mathbf{\Gamma}_k)D.$$

From [11], $\text{cov}(\mathbf{\Gamma}_k) = 4\xi_{\min}R$, where $\xi_{\min}$ is the minimum MSE not counting misadjustment (which is defined in [12]). Inserting this and absorbing the 2s and the 4 into μ and μ^2 gives

$$C = (I - \mu DR)C(I - \mu DR)^T + \mu^2\xi_{\min}DRD. \tag{16}$$

The approximate solution for C (which is obtained by ignoring the $\mu^2 DRCRD$ term) is

$$C = \tfrac{1}{2}\,\mu\xi_{\min}D \tag{17}$$

and the exact solution, which is derived in the Appendix, is

$$C = \frac{1}{2}\,\mu\xi_{\min}D\left(I - \frac{\mu}{2}\,RD\right)^{-1} \tag{18}$$

which holds provided that all eigenvalues of $(\mu/2)RD$ are less than one in magnitude. For simplicity, when μ is small, we prefer (17) over (18).

Widrow *et al.* [12] show that for an FIR filter with weight error vector $\mathbf{v}_k$, the MSE is

$$\xi = \xi_{\min} + \sum_{p=1}^{n} \lambda_p E[(\acute{v}_{p,k})^2]$$

where $\acute{\mathbf{v}}_k = Q^{-1}\mathbf{v}_k$ (so that $\acute{v}_{p,k}$ is the pth element of this vector at time k), and Q and λ_p are as defined in Theorem III.1. In our case, $\text{cov}(\acute{\mathbf{v}}) = Q^{-1}\,\text{cov}(\mathbf{v})Q = Q^{-1}CQ$. From (17), the theoretical MSE ξ is then

$$\xi = \xi_{\min}\left(1 + \tfrac{1}{2}\,\mu\sum_{p=1}^{n}\lambda_p[Q^{-1}DQ]_{p,p}\right) \tag{19}$$

where $[\,.\,]_{p,p}$ indicates the pth element on the main diagonal. ■

If both D and R are diagonal, $Q = I$; therefore

$$\xi = \xi_{\min}\left(1 + \tfrac{1}{2}\,\mu\,\text{tr}(DR)\right) \tag{20}$$

where $\text{tr}(A)$ denotes the trace of A. When D is just the Jacobian used by [7], this simplifies to the result derived in [7]. In this paper, D is always diagonal. Although [7] shows that D is nondiagonal for the regular EG algorithm, it *is* diagonal for the approximate algorithm studied in Section II. However, R is often not diagonal, such as in an equalization setting.

B. Stability Criteria

Using methods similar to those in [12], one can derive stability criteria for EG-like algorithms. This analysis is standard; therefore, only the highlights will be given here. Since (13) is very similar to LMS, one of the criteria for stability of the average system is that

$$0 < \mu < \frac{2}{\hat{\lambda}_{\max,\,k}} \tag{21}$$

for all k, where $\hat{\lambda}_{\max,\,k}$ is the largest eigenvalue of D_kR (for $D_k = I$, this reduces to the LMS stability criterion). It should be emphasized that (21) refers to the more general time-varying system, rather than the steady-state system, hence, the inclusion of the time index k in the eigenvalues and in D_k. Equation (21) can be enforced by applying a clipping function to the weights to bound the elements of D_k, ensuring that $\hat{\lambda}_{\max,\,k}$ is bounded as well. However, since we now have a time-varying system (characterized by $A_k = D_kR$), we must invoke the "slow time variation lemma" of [13]. This says that if (21) is met, and if the rate of time variation of the system is slow (quantitatively, $\sup_{p>p_0}||A_{p+1} - A_p||$ is small for some finite time p_0), then the system is stable. Since we have been assuming that the step size μ is small, this guarantees stability.

C. Directionality of Convergence

In this section, we will see that the speed of convergence (in particular, near the true weight vector) depends heavily on the direction of the parameter error vector; in fact, it is more so than in LMS, which only exhibits directionality if the input is correlated.

Again, consider the average error system, with update rule

$$\mathbf{v}_{k+1} = (I - \mu DR)\mathbf{v}_k = \mathbf{v}_k + \Delta\mathbf{v}_k$$

where $\mathbf{v}_k = \mathbf{w}_k - \mathbf{w}^*$, and $\Delta\mathbf{v}_k = -\mu DR\mathbf{v}_k$. The MSEs at times k and $k+1$ are

$$\begin{aligned}\xi_k &= \mathbf{v}_k^T R\mathbf{v}_k, \\ \xi_{k+1} &= (\mathbf{v}_k + \Delta\mathbf{v}_k)^T R(\mathbf{v}_k + \Delta\mathbf{v}_k).\end{aligned}$$

This means that if we are at $\mathbf{w}_k$ at time k, then the change in the MSE will be

$$(\Delta\xi)_k = \xi_{k+1} - \xi_k = -\mu\mathbf{v}_k^T R(2D - \mu D^2)R\mathbf{v}_k \tag{22}$$

$$\cong -2\mu\mathbf{v}_k^T RDR\mathbf{v}_k \tag{23}$$

assuming μ is small. To proceed further, we must make some simplifying assumptions. First, assume that $R = I$. Second, assume $N = 2$ so that we can visualize our results. For concreteness, we will assume $D_{ii} = |w_i| + \epsilon$ [this leads to an algorithm similar to (2), which is called signed sparse LMS (SSLMS), which will be elaborated on in Section V]. We would like to compare values of (23) for LMS and SSLMS. To do that, we must choose the step sizes to equate the asymptotic MSE. Using (20), we have

$$\mu_{LMS}N = \mu_{SSLMS}\,\text{tr}(D^*).$$

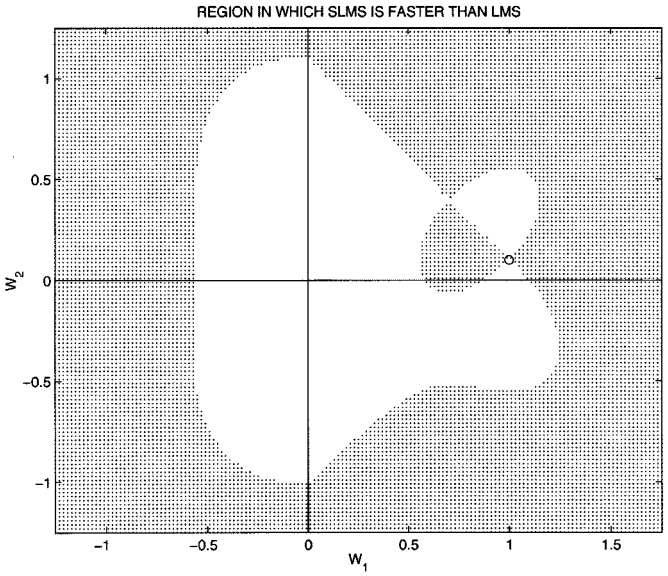

Fig. 1. Region (shaded) in which SSLMS outperforms LMS. The circle denotes $\mathbf{w}^*$.

Thus

$$(\Delta\xi)_{LMS} = -2\mu_{LMS}[(w_1 - w_1^*)^2 + (w_2 - w_2^*)^2] \quad (24)$$

$$(\Delta\xi)_{SSLMS} = -2\mu_{LMS}\left[(w_1 - w_1^*)^2\left(\frac{2|w_1| + 2\epsilon}{|w_1^*| + |w_2^*| + 2\epsilon}\right) + (w_2 - w_2^*)^2\left(\frac{2|w_2| + 2\epsilon}{|w_1^*| + |w_2^*| + 2\epsilon}\right)\right]. \quad (25)$$

When $(\Delta\xi)_{SSLMS} < (\Delta\xi)_{LMS}$, SSLMS will have a larger decrease in MSE (since $\Delta\xi < 0$). Even for this simple case, it is difficult to compute a simple expression for the region in which this is true. However, we can compute it numerically. A plot is given in Fig. 1, with $\mathbf{w}^* = [1, 0.1]$. In the shaded region, SSLMS is faster than LMS in terms of $\Delta\xi$.

Fig. 1 shows that in the two-tap case, if we initialize at the origin, LMS will have an advantage at first, but as we approach $\mathbf{w}^*$, SSLMS will go faster. Since LMS is faster over most of the route, it could do better than SSLMS in this particular case. However, in Section VI, we will examine an example of a system that is more dramatically sparse. In that case, LMS still moves faster at first, but SSLMS gains an advantage more quickly than Fig. 1 implies. The qualitative effects are similar, however.

Of more importance is the fact that near the right answer, the speed of convergence depends on which parameter is in error. If more of the error is in w_1, then SSLMS will go faster, and if more of the error is in w_2, then SSLMS will be slower than LMS. This shows that there is a directionality to this algorithm, even though the input is uncorrelated. This is in contrast to LMS, which has isotropic convergence when the input is uncorrelated.

These results imply that algorithms of the form (13) are not globally faster than standard algorithms. Rather, it depends on in what region of the parameter space the comparison is performed. The critical region will vary with the algorithm and with the true weight vector $\mathbf{w}^*$. The next section offers an interpretation of EG-like algorithms that shows how to exploit prior information to derive algorithms (such as SSLMS) that are logically more suited to certain situations than LMS.

IV. Exploiting Prior Information

This section derives a general form of reparameterized gradient algorithms that can be understood in terms of prior knowledge or in terms of an underlying cost function. Each algorithm will be characterized by a cost function $L(y_k, \hat{y}_k)$, a reparameterization function γ, and by a prior distribution of the unknown parameters. Specifically, let

$$\hat{y}_k = \sum_i w_k^i x_k^i = \sum_i \gamma(z_k^i) x_k^i \quad (26)$$

as in [7, Eq. 8], and $\gamma(z_k^i)$ is assumed to be invertible and differentiable except possibly at a finite number of points. We will consider algorithms that update the entries of $\mathbf{z}$ and, thereby, $\mathbf{w}$.

Mahony and Williamson [14] provide a general discussion of how to encode prior knowledge into learning algorithms using a geometric "preferential structure." The essence of this is to define a metric so that the algorithm evolves over an error surface shaped to incorporate the known prior information. For instance, if the ith component is known to be reliable, whereas the jth component is not, then the algorithm should take larger steps in the jth direction. This is accomplished by warping the underlying space.

Mathematically, the preferential metric is a family of functions $\phi_i(z^i)$ that represent the *a priori* knowledge (the Bayesian prior) of the ith parameter. The idea of a Bayesian prior [15] is that an unknown parameter (in our case, z^i) is viewed as a random variable with a known probability density function [in our case, $\phi_i(z^i)$]. Note that in common with most Bayesian reasoning [15], the prior can be *improper*, i.e., it is not required that ϕ integrated over the whole domain is unity; in fact, the integral can even be infinite.

Using this concept of priors, Mahony and Williamson [14] show that when the standard parameterization is used ($\gamma(z_k^i) = z_k^i$), the NG algorithm of [9] is

$$z_{k+1}^i = \Phi_i^{-1}\left(\Phi_i(z_k^i) - \mu\frac{\partial L}{\partial z_k^i}\frac{1}{\phi_i(z_k^i)}\right) \quad (27)$$

where Φ is the indefinite integral of ϕ. If we interpret ϕ as a (possibly improper) probability density function that is always positive, then its integral will be strictly monotonically increasing, and Φ^{-1} will always exist. However, if ϕ is zero over any range of w, then Φ^{-1} will not exist.

Unfortunately, the updates of the NG algorithm (27) can be quite complicated due to the presence of the nonlinearities Φ and Φ^{-1}. A kinder, gentler algorithm can be derived as a first-order approximation to (27) by rewriting the update as

$$\Phi_i(z_{k+1}^i) = \Phi_i(z_k^i) - \mu\frac{\partial L}{\partial z_k^i}\frac{1}{\phi_i(z_k^i)}. \quad (28)$$

Expanding $\Phi_i(z_{k+1}^i)$ in a Taylor series about z_k^i gives

$$\Phi_i(z_{k+1}^i) = \Phi_i(z_k^i) + \phi(z_k^i)(z_{k+1}^i - z_k^i) + o((z_{k+1}^i - z_k^i)^2)$$

which uses the fact that $\partial\Phi_i(z_k^i)/\partial z_k^i = \phi_i(z_k^i)$. When μ is small and we are reasonably close to convergence, $(z_{k+1}^i - z_k^i)$ will also be small. In that case, the higher order terms may be neglected. Substituting the remainder into the left-hand side of (28) gives

$$\Phi_i(z_k^i) + \phi(z_k^i)(z_{k+1}^i - z_k^i) = \Phi_i(z_k^i) - \mu \frac{\partial L}{\partial z_k^i} \frac{1}{\phi_i(z_k^i)}.$$

Finally, dividing both sides by $\phi(z_k^i)$ and rearranging gives the algorithm

$$\boxed{z_{k+1}^i = z_k^i - \mu \frac{\partial L}{\partial z_k^i} \frac{1}{\phi_i^2(z_k^i)}.} \tag{29}$$

This approximate natural gradient (ANG) algorithm may be preferred to (27) in applications since the updates are simpler, and the resulting algorithms can be more readily analyzed using standard techniques such as averaging theory [16] and local stability [17]. Furthermore, the differences between the NG and ANG algorithms are generally minor.

When prior information is available about the desired target weight vector, it is desired to design an algorithm exploiting that knowledge. For example, the notion of sparsity can be captured by the supposition that with high probability, the tap will have a small value, whereas with low probability, the tap will have a large value. One such prior is $\phi(z) = 1/\sqrt{z}$. As will be shown in Section V-B, this prior leads to algorithm (2).

Suppose ϕ is a (possibly improper) probability density function representing prior beliefs about a parameter. Then, (27) is the NG algorithm that incorporates this prior, and (29) is the first order ANG algorithm, both assuming that the parameterization function γ of (26) is the identity. The same algorithm can also be derived (or interpreted) as a Euclidean gradient descent $(\phi(z) = 1)$ on a modified cost function [i.e., $\gamma(z)$ not necessarily the identity]. The next proposition makes this precise.

Proposition IV.1: Let ϕ and γ represent the priors and parameterizations of an ANG algorithm (29) with $\hat{y}$ parameterized as in (26), and let the cost function be given by $L(y, \hat{y})$. If there are functions $\overline{\gamma}$ and $\overline{\phi}$ with

$$\frac{\dot{\gamma}^2}{\phi^2} = \frac{\dot{\overline{\gamma}}^2}{\overline{\phi}^2} \tag{30}$$

then $\overline{\gamma}$ and $\overline{\phi}$ are an alternate set of priors and parameterizations that yield the same effective update rule for $\mathbf{w}$, as long as μ is sufficiently small.

Proof: From (29), the ANG corresponding to ϕ, γ, and $L(y, \hat{y})$ is

$$z_{k+1}^i = z_k^i - \mu \frac{\partial L}{\partial z_k^i} \frac{1}{\phi_i^2(z_k^i)}.$$

Applying the chain rule yields

$$z_{k+1}^i = z_k^i - \mu \frac{\partial L}{\partial \hat{y}_k} \frac{\partial \hat{y}_k}{\partial z_k^i} \frac{1}{\phi_i^2(z_k^i)}.$$

Since $\partial \hat{y}_k / \partial z_k^i = \dot{\gamma}(z_k^i) x_k^i$, this becomes

$$z_{k+1}^i = z_k^i \underbrace{- \mu \frac{\partial L}{\partial \hat{y}_k} \frac{\dot{\gamma}(z_k^i)}{\phi_i^2(z_k^i)} x_k^i}_{\Delta}.$$

Since y is generated according to $\hat{y}_k = \sum_i w_k^i x_k^i$, what we ought to compare is the effective change of $\mathbf{w}$ as $\mathbf{z}$ changes. For small μ, $\Delta = o(\mu)$, and therefore

$$w_{k+1}^i = \gamma(z_{k+1}^i) = \gamma(z_k^i + \Delta) \cong \gamma(z_k^i) + \dot{\gamma}(z_k^i)\Delta \tag{31}$$

assuming we drop terms in the Taylor series expansion of order μ^2 or higher. Thus, the proof requires μ to be sufficiently small, which is a standard assumption. The resulting effective update to $\mathbf{w}$ is

$$w_{k+1}^i = w_k^i - \mu \frac{\partial L}{\partial \hat{y}_k} x_k^i \left(\frac{\dot{\gamma}^2(z_k^i)}{\phi_i^2(z_k^i)} \right). \tag{32}$$

Similarly, the ANG corresponding to $\overline{\gamma}$ and $\overline{\phi}$ is

$$w_{k+1}^i = w_k^i - \mu \frac{\partial L}{\partial \hat{y}_k} x_k^i \left(\frac{\dot{\overline{\gamma}}^2(z_k^i)}{\overline{\phi}_i^2(z_k^i)} \right).$$

Since $\dot{\gamma}^2/\phi^2 = \dot{\overline{\gamma}}^2/\overline{\phi}^2$ by assumption, these are the same algorithm. ∎

The left and right sides of (30) are both functions of z, yet z is different for each side since the parameterization is different. Thus, both sides of the equation must be separately represented as functions of w and then compared. For example, consider the motivating example of Section II. The algorithm as initially presented in (2) can be thought of as having the standard parameterization $w = \gamma_1(z) = z$ and prior $\phi_1(z) = 1/\sqrt{z}$. Thus

$$\frac{\dot{\gamma}_1^2}{\phi_1^2} = \frac{1}{\phi_1^2(z)} = \frac{1}{\phi_1^2(w)} = w.$$

On the other hand, if we reparameterize w via $\gamma_2(z) = (1/4)z^2$ and use the uniform prior of $\phi_2(z) = 1$, we have

$$\frac{\dot{\gamma}_2^2}{\phi_2^2} = \dot{\gamma}_2^2(z) = \frac{1}{4} z^2 = w.$$

These two ratios used z differently, but in terms of w, the ratios are the same. Consequently, the update rules given by (2) and (9) were the same.

In the course of the proof, we have shown that for sufficiently small μ, algorithms incorporating priors and reparameterizations can be written in the general form of

$$w_{k+1}^i = w_k^i - \mu \frac{\partial L}{\partial \hat{y}_k} x_k^i [D_k]_{ii}$$

where D_k is given by

$$[D_k]_{i,j} = \left(\frac{\partial \gamma_i}{\partial (z_k^j)} \right)^2 \left(\frac{1}{\phi_i(z_k^j)} \right)^2. \tag{33}$$

This shows how ANG algorithms can be analyzed using the framework of Section III. Note the similarity between D_k and the Jacobian in [7, equation (15)]. The difference is that the Jacobian in [7] only includes the first factor from (33), whereas our use here is intended to capture the effects of the prior as well.

The ANG may be derivable from an alternative prior ϕ using the standard parameterization $\gamma(z) = z$. This prior will be called the "true" prior because it represents the prior beliefs without the confounding influence of the reparameterization function. Alternatively, the ANG may be derivable from a reparameterization using the standard prior $\phi = 1$ (which corresponds to a belief that all parameter values are equally

TABLE I
ALGORITHMS USING THE STANDARD COST FUNCTION OF $(y_k - \hat{y}_k)^2$ AS IN (4). NOTE THAT WHEN $\phi(z) = 1$, THE NG AND THE ANG UPDATES ARE IDENTICAL. NOTE THAT THESE ALGORITHMS ARE IN z-SPACE, BUT WHEN $\gamma(z) = z$, THE w-SPACE UPDATES ARE THE SAME

Algorithm Name	$\gamma(z)$, $\phi(z)$ **pairs**	**NG update**	**ANG update**
LMS	z, 1	$z^i_{k+1} = z^i_k + \mu\,(y_k - \hat{y}_k)\,x^i_k$	$z^i_{k+1} = z^i_k + \mu\,(y_k - \hat{y}_k)\,x^i_k$
(see note in text)	$\ln(z)$, $\frac{1}{z}$	$z^i_{k+1} = z^i_k \exp(\mu(y_k - \hat{y}_k)x^i_k)$	$z^i_{k+1} = z^i_k + \mu\, z^i_k\,(y_k - \hat{y}_k)\,x^i_k$
EG natural [8]	z, $\frac{1}{z}$	$z^i_{k+1} = z^i_k \exp(\mu\, z^i_k\,(y_k - \hat{y}_k)\,x^i_k)$	$z^i_{k+1} = z^i_k + \mu\,(z^i_k)^2(y_k - \hat{y}_k)\,x^i_k$
	$\exp(z)$, 1	$z^i_{k+1} = z^i_k + \mu \exp(z^i_k)\,(y_k - \hat{y}_k)\,x^i_k$	$z^i_{k+1} = z^i_k + \mu \exp(z^i_k)\,(y_k - \hat{y}_k)\,x^i_k$
Sparse LMS	z, $\frac{1}{\sqrt{z}}$	$z^i_{k+1} = \left(\sqrt{z^i_k} + \frac{1}{2}\mu\sqrt{z^i_k}(y_k - \hat{y}_k)\,x^i_k\right)^2$	$z^i_{k+1} = z^i_k + \mu\, z^i_k\,(y_k - \hat{y}_k)\,x^i_k$
	$\frac{1}{4}z^2$, 1	$z^i_{k+1} = z^i_k + \mu\,(\frac{1}{2}z^i_k)\,(y_k - \hat{y}_k)\,x^i_k$	$z^i_{k+1} = z^i_k + \mu\,(\frac{1}{2}z^i_k)\,(y_k - \hat{y}_k)\,x^i_k$
Signed Sparse LMS	z, $\frac{1}{\sqrt{\lvert z\rvert+\epsilon}}$	too complicated for a table	$z^i_{k+1} = z^i_k + \mu(\lvert z^i_k\rvert + \epsilon)(y_k - \hat{y}_k)\,x^i_k$
	$\frac{1}{4}\mathrm{sgn}(z)z^2 + \sqrt{\epsilon}z$, 1	$z^i_{k+1} = z^i_k + \mu(\frac{1}{2}\lvert z^i_k\rvert + \sqrt{\epsilon})(y_k - \hat{y}_k)x^i_k$	$z^i_{k+1} = z^i_k + \mu(\frac{1}{2}\lvert z^i_k\rvert + \sqrt{\epsilon})(y_k - \hat{y}_k)x^i_k$
Fractional LMS	z, $\frac{1}{z^\alpha}$ $(\alpha \neq 1)$	refer to Equation (45)	$z^i_{k+1} = z^i_k + \mu(z^i_k)^{2\alpha}(y_k - \hat{y}_k)x^i_k$
	$((1-\alpha)z)^{\frac{1}{1-\alpha}}$, 1	$z^i_{k+1} = z^i_k + \mu((1-\alpha)z)^{\frac{\alpha}{1-\alpha}}(y_k - \hat{y}_k)x^i_k$	$z^i_{k+1} = z^i_k + \mu((1-\alpha)z)^{\frac{\alpha}{1-\alpha}}(y_k - \hat{y}_k)x^i_k$
Signed Frac. LMS	see Eq. (48), 1	refer to Equation (49)	refer to Equation (49)
IPL LMS	z, $\frac{c}{\lvert z\rvert^\alpha+\epsilon}$	too complicated for a table	$z^i_{k+1} = z^i_k + \mu(\lvert z^i_k\rvert^\alpha + \epsilon)^2(y_k - \hat{y}_k)x^i_k$
Exponential LMS	z, $c\,e^{-a\lvert z\rvert}$	too complicated for a table	$z^i_{k+1} = z^i_k + \mu\, e^{2\alpha\lvert z^i_k\rvert}(y_k - \hat{y}_k)x^i_k$

likely). In this case, γ can be used to give the cost function over which the algorithm is evolving under the standard Euclidean gradient. These multiple interpretations are useful because sometimes it is easier to understand the behavior of an algorithm from the point of view of priors, whereas sometimes it is easier from the perspective of the cost function. Imposition of the requirement that $\dot{\gamma}^2/\phi^2$ remain invariant allows translation of an algorithm to both frameworks. This is an important feature of the ANG algorithms since this translation is not possible with the nonlinear NG algorithms.

A third possible interpretation (following Amari and Douglas [9], [18]) considers the function $G^{-1}(z)$ [which is mathematically equivalent to $1/\phi^2(z)$] to be a Riemannian metric of distance measure imposed on the underlying space. This can be used, for instance, when signal measurements are corrupted by additive noise (and the probability distribution function of the noise is known) to create a Fisher-efficient learning algorithm. The main difference between this paper and [9] and [18] is that they explicitly assume they know the likelihood function of the noise (and, hence, a noise model) and then effectively use the uniform prior ($\phi = 1$) and standard parameterization ($\gamma = z$), although we make no assumptions about the noise, and we use nonstandard priors and/or parameterizations.

V. ALGORITHMS

This section derives a number of NG and ANG algorithms. The first two examples show how special choices of γ and ϕ result in familiar algorithms, whereas the remaining subsections derive new algorithms specifically for signal processing and communications problems when prior information (such as sparsity) is available. The new algorithms include blind CMA and DD algorithms and an ARMA model algorithm that corresponds to the "equation error" [19] form of standard ARMA model adaptive filters. The parameterizations, priors, costs, and update rules for these algorithms are displayed in Tables I and II.

A. Least Mean Squares

The prior belief corresponding to the LMS algorithm is that all parameters are equally likely to achieve any value. Hence, $\phi_i(w) = 1$, $\Phi_i(w) = w$, and $\Phi_i^{-1}(v) = v$. The appropriate cost function is the mean square cost $L(y_k, \hat{y}_k) = (y_k - \hat{y}_k)^2$, and the parameterization within $\hat{y}_k$ is the standard one (1) with $\gamma(w^i) = w^i$. Hence, $\partial L/\partial w^i_k = 2(y_k - \hat{y}_k)x^i_k$. Substituting into either (27) or (29) yields the LMS algorithm of (10). Note that because $\phi(w) = 1$, there is no difference between the NG and the ANG algorithms.

B. Exponentiated Gradient Algorithm

Consider the prior $\phi_i(z) = 1/z$, $z > 0$, which corresponds to a high likelihood that the parameter value is small and a small chance that the parameter is large. Accordingly, $\Phi_i(z) = \ln(z)$, and $\Phi_i^{-1}(v) = \exp(v)$. Using the standard cost $L(y_k, \hat{y}_k)$ of (4) and the standard parameterization $\gamma(z) = z$ gives the NG algorithm

$$z^i_{k+1} = \exp(\ln(z^i_k) + \mu(y_k - \hat{y}_k)x^i_k z^i_k) \tag{34}$$

$$= z^i_k \exp(\mu z^i_k(y_k - \hat{y}_k)x^i_k). \tag{35}$$

This is the exponentiated gradient algorithm of [8]. The associated ANG algorithm is

$$z^i_{k+1} = z^i_k + \mu(z^i_k)^2(y_k - \hat{y}_k)x^i_k. \tag{36}$$

An equivalent way to derive this ANG algorithm is to let $\gamma(z) = \exp(z)$ and $\phi(z) = 1$. In this case, both the NG and ANG algorithms are

$$z^i_{k+1} = z^i_k + \mu \exp(z^i_k)(y_k - \hat{y}_k)x^i_k. \tag{37}$$

To see the equivalence of (36) and (37) as a concrete example of Proposition IV.1, convert them into w-space. Equation (36)

TABLE II
ALGORITHMS USING MISCELLANEOUS COST FUNCTIONS. WHEN $\phi(z) = 1$, THE NG AND ANG UPDATES ARE IDENTICAL

Algorithm Name	$\gamma(z)$, $\phi(z)$ **pairs**	**NG update**	**ANG update**
Algorithms using the CMA cost function: $L = \frac{1}{4}(\hat{y}_k^2 - c)^2$			
CMA	w, 1	$z_{k+1}^i = z_k^i - \mu\ \hat{y}_k(\hat{y}_k^2 - c)\ x_k^i$	$z_{k+1}^i = z_k^i - \mu\ \hat{y}_k(\hat{y}_k^2 - c)\ x_k^i$
Signed Sparse CMA	z, $\frac{1}{\lvert z\rvert+\epsilon}$	$z_{k+1}^i = z_k^i \exp(-\mu(\lvert z_k^i\rvert + \epsilon)\hat{y}_k(\hat{y}_k^2 - c)x_k^i)$	$z_{k+1}^i = z_k^i - \mu(\lvert z_k^i\rvert + \epsilon)^2\hat{y}_k(\hat{y}_k^2 - c)x_k^i$
Fractional CMA	z, $\frac{1}{z^\alpha}$	refer to Equation (52)	$z_{k+1}^i = z_k^i - \mu(z_k^i)^{2\alpha}\hat{y}_k(\hat{y}_k^2 - c)x_k^i$
	$((1-\alpha)z)^{\frac{1}{1-\alpha}}$, 1	$z_{k+1}^i = z_k^i - \mu((1-\alpha)z)^{\frac{\alpha}{1-\alpha}}\hat{y}_k(\hat{y}_k^2 - c)x_k^i$	$z_{k+1}^i = z_k^i - \mu((1-\alpha)z)^{\frac{\alpha}{1-\alpha}}\hat{y}_k(\hat{y}_k^2 - c)x_k^i$
Signed Frac. CMA	see Eq. (48), 1	refer to Equation (53)	refer to Equation (53)
Algorithms using the DD cost function: $L = \frac{1}{2}(Q(\hat{y}_k) - \hat{y}_k)^2$			
DD-LMS	z, 1	$z_{k+1}^i = z_k^i + \mu\ (Q(\hat{y}_k) - \hat{y}_k)\ x_k^i$	$z_{k+1}^i = z_k^i + \mu\ (Q(\hat{y}_k) - \hat{y}_k)\ x_k^i$
Sparse DD-LMS	z, $\frac{1}{\sqrt{\lvert z\rvert+\epsilon}}$	too complicated for a table	$z_{k+1}^i = z_k^i + \mu(\lvert z_k^i\rvert + \epsilon)(Q(\hat{y}_k) - \hat{y}_k)\ x_k^i$
	$\frac{1}{4}\mathrm{sgn}(z)z^2 + \sqrt{\epsilon}z$, 1	$z_{k+1}^i = z_k^i + \mu(\frac{1}{2}\lvert z_k^i\rvert + \sqrt{\epsilon})(Q(\hat{y}_k) - \hat{y}_k)x_k^i$	$z_{k+1}^i = z_k^i + \mu(\frac{1}{2}\lvert z_k^i\rvert + \sqrt{\epsilon})(Q(\hat{y}_k) - \hat{y}_k)x_k^i$
Algorithms using ARMA model parameterization with $L = \frac{1}{2}(y_k - \hat{y}_k)^2$			
Equation Error	z, 1	$z_{k+1}^i = z_k^i + \mu\ (y_k - \hat{y}_k)\ y_{k-i}$	same as NG update
		$z_{k+1}^{j+n} = z_k^{j+n} + \mu\ (y_k - \hat{y}_k)\ x_{k-j}$	same as NG update
Sparse Equation Error	$\frac{1}{4}\mathrm{sgn}(z)z^2 + \sqrt{\epsilon}z$, 1	$z_{k+1}^i = z_k^i + \mu\ (y_k - \hat{y}_k)\ y_{k-i}\ (\lvert z_k^i\rvert + \epsilon)$	same as NG update
		$z_{k+1}^{j+n} = z_k^{j+n} + \mu\ (y_k - \hat{y}_k)\ x_{k-j}\ (\lvert z_k^i\rvert + \epsilon)$	same as NG update

is already in that format (since there we had $\mathbf{w} = \mathbf{z}$); therefore, consider expanding (37) in a Taylor series as in (8). This yields

$$w_{k+1}^i = w_k^i + \mu(\exp(z_k^i))^2(y_k - \hat{y}_k)x_k^i.$$

Since $w = \gamma(z) = \exp(z)$, this is effectively

$$w_{k+1}^i = w_k^i + \mu(w_k^i)^2(y_k - \hat{y}_k)x_k^i$$

which is the same as (36).

Now, consider an algorithm with $\gamma(z) = z$ and a prior $\phi(z) = 1/\sqrt{z}$, which is qualitatively similar to the prior in the previous example (i.e., $1/z$). The corresponding NG update is

$$z_{k+1}^i = \left(\sqrt{z_k^i} + \tfrac{1}{2}\,\mu\sqrt{z_k^i}\,(y_k - \hat{y}_k)x_k^i\right)^2$$

and the simpler ANG update is

$$z_{k+1}^i = z_k^i + \mu z_k^i(y_k - \hat{y}_k)x_k^i. \tag{38}$$

This is exactly the example introduced in Section II since $\mathbf{w} = \mathbf{z}$. Recall that an equivalent way to derive this ANG algorithm comes from setting $\gamma(z) = (1/4)z^2$ [and $\phi(z) = 1$].

In Section V-C, we will modify $\gamma(z)$ to allow both positive and negative weights. Using the duality of priors and parameterizations, we will show how such modifications affect the priors.

C. Practical LMS for Sparsity

Suppose that the "sparse" prior of $\phi(w) = 1/\sqrt{w}$, $w > 0$ appears to fit well (assuming no reparameterization). Then, the discussion in Section V-B suggests that an algorithm of the general form of (2) may be particularly effective. As discussed in Section II, this is equivalent to a Euclidean gradient descent in z-space with $\gamma(z) = (1/4)z^2$. We would like to allow positive and negative coefficients and remove the stationary point that occurs at $\mathbf{w} = \mathbf{0}$. This can be done by modifying the parameterization (as first suggested in [14]) to

$$\gamma(z) = \tfrac{1}{4}\,\mathrm{sgn}(z)(z)^2 + \sqrt{\epsilon}\,z \tag{39}$$

where $\epsilon > 0$. What is the effect of this modification?

Since $\dot{\gamma}(z) = (1/2)|z| + \sqrt{\epsilon} > 0$, equilibrium in the corresponding ANG algorithm can only occur when $y_k = \hat{y}_k$. Specifically, ϵ keeps the update term from vanishing for small z, which would have prevented coefficients from adapting across zero. Now, consider the question of how such *ad hoc* modifications influence the sparsity prior. The parameterization given by (39) and a Euclidean gradient over z is equivalent to $\dot{\overline{\gamma}}(z) = 1$ and $\overline{\phi}(z) = \sqrt{1/(|z| + \epsilon)}$ by Proposition IV.1. This is because

$$(\dot{\gamma}(z))^2 = \tfrac{1}{4}\,|z|^2 + \sqrt{\epsilon}|z| + \epsilon \tag{40}$$

and

$$|w| = |\gamma(z)| = \tfrac{1}{4}\,|z|^2 + \sqrt{\epsilon}|z| \tag{41}$$

which imply

$$\left(\frac{\dot{\overline{\gamma}}}{\overline{\phi}}\right)^2 = \left(\frac{\dot{\gamma}}{\phi}\right)^2 = (\dot{\gamma}(z))^2 = |w| + \epsilon. \tag{42}$$

If we choose $\overline{\gamma}(z) = z$ (thus redefining z), then $\dot{\overline{\gamma}} = 1$ and $w = z$ so that

$$\frac{1}{\left(\overline{\phi}(z)\right)^2} = |z| + \epsilon. \tag{43}$$

This yields $\overline{\phi}(z) = \sqrt{1/(|z| + \epsilon)}$. Compared with the previous prior of $\phi(z) = \sqrt{1/z}$, there is little difference. The ϵ is a small modification that only changes the algorithm near $z = 0$. The

algorithms are listed in Table I as "Sparse LMS" and "Signed Sparse LMS."

Analogous designs result from other choices of priors. The parameterization $\gamma(z) = (1/3)z^3$ leads to the Euclidean descent algorithm

$$z_{k+1}^i = z_k^i + \mu(z_k^i)^2(y_k - \hat{y}_k)x_k^i. \tag{44}$$

As with all Euclidean gradient algorithms, $\phi(z) = 1$. To determine the corresponding sparsity prior, invoke (30) to obtain the equivalent ANG algorithm. This gives $\dot{\overline{\gamma}}(z) = 1$ and $\phi(z) = (3|z|)^{2/3}$.

For the general case of fractional power priors, consider $\phi(z) = 1/z^\alpha$ for $\alpha \neq 1$ and $z > 0$ for which $\Phi(z) = (1/(1-\alpha))z^{1-\alpha}$ and $\Phi^{-1}(v) = ((1-\alpha)v)^{\alpha-1}$. Under the standard parameterization $\gamma(z) = z$, the NG algorithm is

$$z_{k+1}^i = (1-\alpha)\left(\frac{1}{1-\alpha}(z_k^i)^{1-\alpha} + \mu(y_k - \hat{y}_k)x_k^i(z_k^i)^\alpha\right)^{\alpha-1} \tag{45}$$

whereas the ANG algorithm is

$$z_{k+1}^i = z_k^i + \mu(z_k^i)^{2\alpha}(y_k - \hat{y}_k)x_k^i. \tag{46}$$

By Proposition IV.1, this corresponds to a Euclidean gradient descent with $\overline{\phi}(z) = 1$ and some $\overline{\gamma}(z)$. To find $\overline{\gamma}(z)$, note that

$$\frac{\dot{\gamma}}{\phi} = z^\alpha = w^\alpha.$$

Since $\overline{\phi}(z) = 1$

$$\dot{\overline{\gamma}} = w^\alpha = \overline{\gamma}^\alpha$$

$$\frac{d\overline{\gamma}}{\overline{\gamma}^\alpha} = dz$$

$$z = \frac{\overline{\gamma}^{1-\alpha}}{1-\alpha}$$

giving the parameterization function

$$\overline{\gamma}(z) = ((1-\alpha)z)^{1/(1-\alpha)}. \tag{47}$$

Both (45) and (46) are only appropriate for positive target weights w. To create a practical algorithm, the parameterization can be modified to

$$\gamma(z) = \operatorname{sgn}(z)((1-\alpha)|z|)^{1/(1-\alpha)} + \epsilon z \tag{48}$$

which corresponds to the algorithm (both NG and ANG)

$$z_{k+1}^i = z_k^i + \mu(y_k - \hat{y}_k)x_k^i\left(((1-\alpha)|z_k^i|)^{\alpha/(1-\alpha)} + \epsilon\right). \tag{49}$$

Understandably, it might be undesirable to implement an algorithm this complicated.

All the above derivations occur for individual weights. If different priors are available for different weights, then different algorithm updates can be used. This might be useful in an equalizer design problem, where the center taps are likely to be large, whereas the tails of the equalizer are likely to be small and sparse.

D. Blind Adaptation With CMA

The previous sections assumed that the cost function was the standard squared error function (4). However, other cost functions can be treated analogously. For example, the CMA [20] uses the cost function

$$L(y_k, \hat{y}_k) = \tfrac{1}{4}(\hat{y}_k^2 - c)^2 \tag{50}$$

where c is a constant appropriate for the given signal constellation. This leads to algorithms that do not require knowledge of y_k and can be used to adapt the weights even in the absence of a training signal. Such algorithms are called "blind."

To derive the CMA algorithm, use the cost function in (50), the parameterization $\gamma(z) = z$, and the uniform prior $\phi(z) = 1$. This leads to NG and ANG algorithms that are both given by

$$z_{k+1}^i = z_k^i - \mu\hat{y}_k(\hat{y}_k^2 - c)x_k^i \tag{51}$$

which is the standard CMA of [20].

Suppose, however, that prior knowledge suggests a prior of $\phi(z) = 1/\sqrt{z}$ and $z > 0$. Then, the NG algorithm is

$$z_{k+1}^i = \left(\sqrt{z_k^i} + \frac{\mu}{2}\hat{y}_k(\hat{y}_k^2 - c)x_k^i\sqrt{z_k^i}\right)^2$$

whereas the simpler ANG update gives

$$z_{k+1}^i = z_k^i - \mu\hat{y}_k(\hat{y}_k^2 - c)x_k^i z_k^i.$$

As in Section V-C, to be practical, these updates must be modified to allow both positive and negative weights and to remove the stationary point at $z = 0$. Modifying the prior to $\phi(z) = 1/\sqrt{|z| + \epsilon}$ results in the ANG algorithm

$$z_{k+1}^i = z_k^i - \mu\hat{y}_k(\hat{y}_k^2 - c)x_k^i(|z_k^i| + \epsilon).$$

Equivalently, Proposition IV.1 shows that this algorithm results from a Euclidean descent in z-space on the cost function (50) with $\phi(z) = 1$ and the reparameterization $\gamma(z)$ given by (39).

More generally, for fractional power priors, $\phi(z) = 1/z^\alpha$, for $\alpha \neq 1$, $\Phi(z) = (1/(1-\alpha))z^{1-\alpha}$, and $\Phi^{-1}(v) = ((1-\alpha)v)^{\alpha-1}$. Under the standard parameterization $\gamma(z) = z$ and the cost function (50), the NG algorithm is

$$z_{k+1}^i = (1-\alpha)\left(\frac{1}{1-\alpha}(z_k^i)^{1-\alpha} - \mu\hat{y}_k(\hat{y}_k^2 - c)x_k^i(z_k^i)^\alpha\right)^{\alpha-1} \tag{52}$$

whereas the ANG algorithm is

$$z_{k+1}^i = z_k^i - \mu\hat{y}_k(\hat{y}_k^2 - c)x_k^i(z_k^i)^{2\alpha}.$$

By Proposition IV.1, this corresponds to a Euclidean gradient descent with $\overline{\phi}(z) = 1$ and $\overline{\gamma}(z)$ given by (47). Again, to create a practical algorithm, the parameterization can be modified as in (48) (with the uniform prior), producing the NG and ANG algorithms

$$z_{k+1}^i = z_k^i - \mu\hat{y}_k(\hat{y}_k^2 - c)x_k^i\left(((1-\alpha)|z_k^i|)^{\alpha/(1-\alpha)} + \epsilon\right). \tag{53}$$

The true prior is too difficult to compute in this case.

Variations on $L(y_k, \hat{y}_k)$ are also common, such as generalizing (50) to

$$L(y_k, \hat{y}_k) = \frac{1}{pq}\left||\hat{y}_k|^p - c\right|^q$$

where p and q take on various values. This can be combined with prior information leading to a generalized CMA for sparsity. The procedure is similar to that for standard CMA.

E. Other Extensions

Another important class of algorithms are DD blind algorithms designed for use with a finite alphabet. The DD algorithm can be viewed as Euclidean descent over the cost function

$$L(y_k, \hat{y}_k) = \tfrac{1}{2}\left(Q(\hat{y}_k) - \hat{y}_k\right)^2 \tag{54}$$

where the memoryless $Q(\cdot)$ quantizes the argument to the nearest symbol in the alphabet. [In such a derivation, one ignores the fact that $Q(\cdot)$ is discontinuous and formally replaces its derivative with zero.] All of the analysis done for the MSE cost function also applies here. Simply replace y_k with $Q(\hat{y}_k)$ in any of the NG or ANG update rules.

Another use of EG-like algorithms is for systems with ARMA model parameterizations. Many system modeling and signal processing applications require that estimates be made of autoregressive (as well as moving average) parameters. One algorithm that makes use of an ARMA model parameterization is the "equation error" algorithm. Details can be found in [21]. The standard equation error algorithm and its sparse counterpart are listed in Table II.

VI. Simulations

This section gives experimental performance curves in several system identification scenarios that compare the performance of LMS to that of signed sparse LMS (SSLMS in Table I) in terms of MSE, convergence rates, and tracking ability. For the comparisons to be fair, the step sizes are chosen to equate the MSE after convergence, using Theorem III.1.

A. Performance in a Sparse Environment

This set of simulations was run in a sparse environment. The first channel had ten taps, with nonzero taps of values $[0.1, 1, -0.5, 0.1]$ located in positions $[1, 3, 4, 8]$. The second channel had 100 taps, with nonzero taps of the same values located in positions $[1, 30, 35, 85]$. The parameters we used were $\mu = 0.0050$, $\mu_{sslms} = 0.0215$ (for channel 1), $\mu_{sslms} = 0.0629$ (for channel 2), $\sigma_n^2 = 0.025^2$, $\epsilon = 0.0625$ (the parameter in SSLMS), and $\mathbf{w}_0 = \mathbf{0}$.

Fig. 2 shows the MSE versus time for both algorithms. The MSE was computed by taking the ensemble average of e^2 over 100 runs. These experiments suggest that when the environment is sufficiently sparse, the sparse version of LMS converges much faster.

B. Performance in a Nonsparse Environment

The next set of simulations was run in a nonsparse environment. The channels were changed from four out of ten (and four out of 100) to nine out of ten (and nine out of 100) nonzero taps, but the ANG algorithm was run assuming (falsely) that the sparse conditions were still present.

Fig. 3 shows the MSE versus time for both algorithms. In these cases, the performance of sparse LMS is worse than regular LMS. In this example, the performance loss in a nonsparse environment (by a "sparse" algorithm) is comparable with the performance gain in a sparse environment, as compared with LMS.

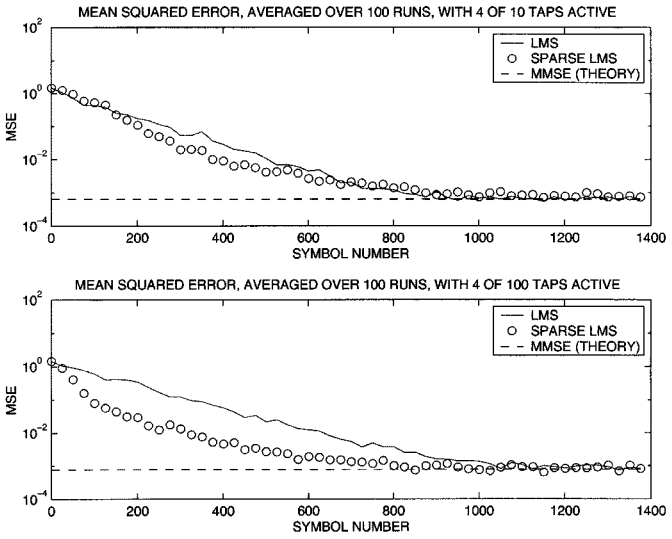

Fig. 2. MSE curves for LMS and sparse LMS in a sparse environment.

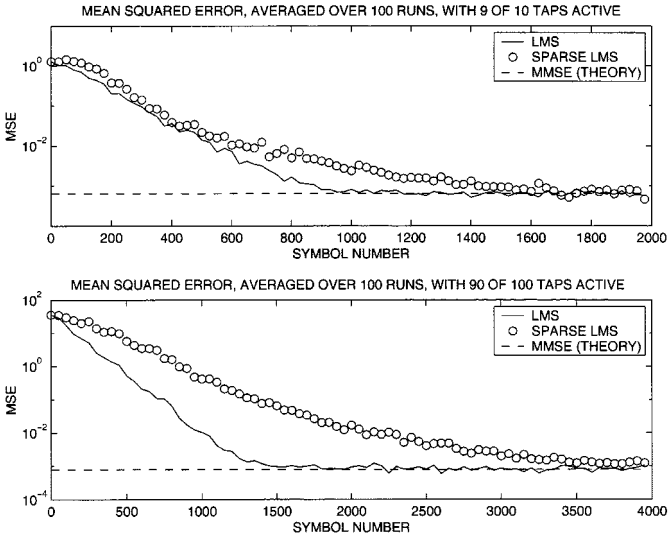

Fig. 3. MSE curves for LMS and sparse LMS in a nonsparse environment. Note the different scales.

C. Tracking Ability

The next simulation was run on a 20–tap channel with two taps set to vary sinusoidally in time. The actual channel consisted of the first channel from the sparse simulation in Section VI-A with ten zeros appended. Then, taps 15 and 16 were set to vary as $1+0.2\sin(k\pi/256)$ and $0.2\sin(k\pi/256)$, respectively, where k is the iteration number. Again, the step sizes were chosen to equate the asymptotic MSE.

Fig. 4 shows the tracking ability of both algorithms. The upper plot shows the values of the actual taps and the estimates as the taps fluctuate. As expected, sparse LMS is better at tracking the change in the large taps but not the small taps. The lower plot shows the MSE when only the larger tap is fluctuating. When only the large tap is fluctuating, the sparse algorithm has a lower MSE than LMS.

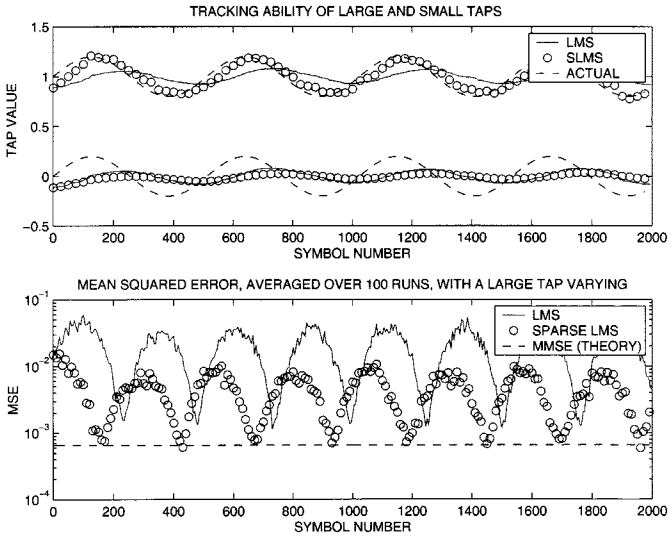

Fig. 4. Comparison of the tracking ability of the two algorithms.

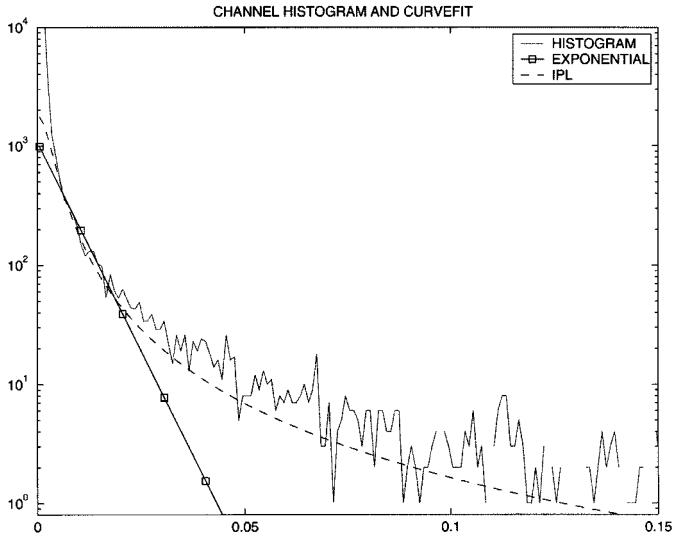

Fig. 6. Histogram and curve fits for the channel.

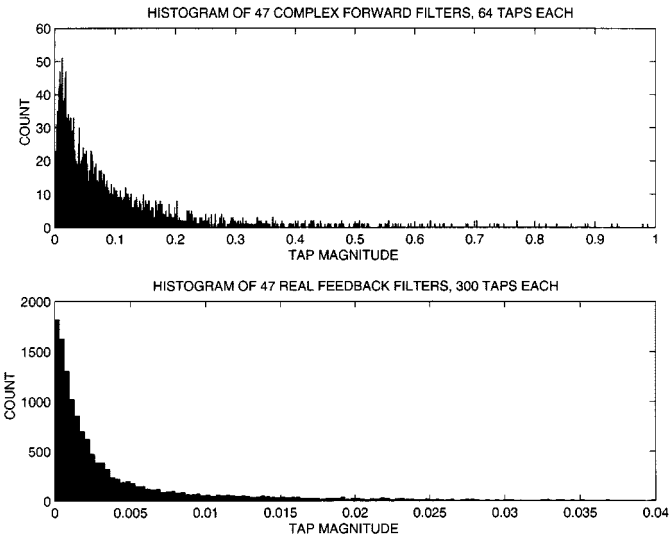

Fig. 5. Histograms of the DFE.

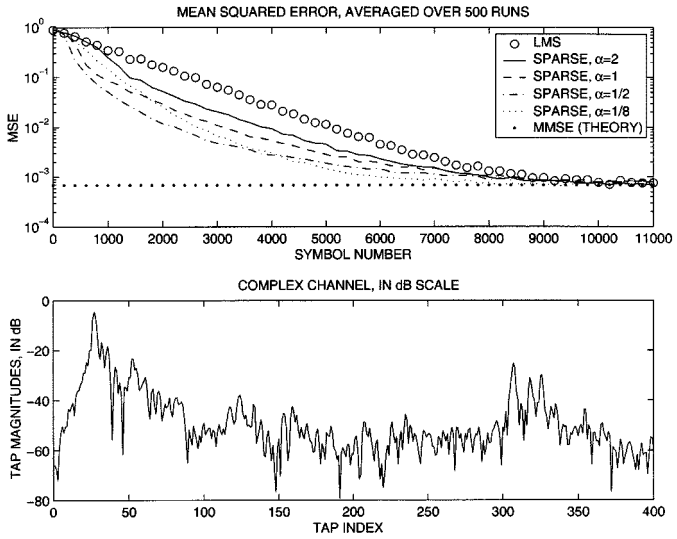

Fig. 7. Measured complex channel and plot of convergence rates for channel identification.

VII. Measured Channels

In March 2000, researchers (including three of the authors) from Cornell University, University of Wisconsin-Madison, Australian National University, Applied Signal Technology, and NxtWave Communications met in Philadelphia for field measurements of digital television channels [22]. The data collected there have been compiled into a large database of identified channels and associated MMSE decision feedback equalizers. The data from 47 of these channels were used to produce histograms of the magnitudes of typical equalizer coefficients. The histograms of the forward filter of the equalizer are well modeled by both exponential priors ($ce^{-\alpha|z|}$) and inverse power law (IPL) priors ($c/(|z|^{\alpha}+\epsilon)$). The feedback filter was well modeled by an exponential prior, and the channel itself was well modeled by an IPL prior. The exponential and IPL update algorithms for channel identification are given in Table I.

The top plot in Fig. 5 is a histogram of the magnitudes of all the complex taps from the forward equalizers for all of the channels. The bottom plot is a histogram of the tap magnitudes of all of the (real) feedback equalizers. Fig. 6 shows the histogram of the magnitudes of the complex channel taps, as well as exponential and IPL curves that have been fitted to the histogram.

The gain constant c appears in the update rule in such a way that it can be absorbed by the step size. The parameter ϵ in the IPL prior and the α that appears in the exponential prior are both subject to changes in the scale of $\mathbf{w}$. Thus, a different receiver with a different automatic gain controller will have different values for these parameters.

This paper has focused on the real case, although it is a simple matter to extend the algorithms to the complex case. Fig. 7 shows an example of identification of one of the complex channels used for the histogram. The top plot shows the MSE versus time for traditional LMS and sparse LMS with a variety of

values of α (for the IPL prior). The bottom plot shows the complex channel in decibel scale. The same simulation was done using ten different channels from the database, and the resulting MSE curves were almost identical to those in Fig. 7.

The value of ϵ that we used was larger than specified by the curve fit to speed initial convergence (since the model is initialized to zero). This is justified because this only affects the algorithm in the vicinity of the origin so locally (near the optimum), the performance should not change much. The actual value of ϵ was chosen because it resulted in faster convergence than the other values we tried.

In practice, values for α and ϵ (as well as the functional form of the curve) will depend somewhat on the application and the physical environment. To choose the parameters, one must measure many channels, form a histogram of the tap magnitudes, and then fit curves to the histograms. Another approach is to simply run algorithms corresponding to different parameter values and observe which parameter values offer fastest convergence (in the mean).

VIII. Conclusions and Future Work

We have derived the LMS algorithm and a wide range of variants through the framework of the natural gradient algorithms. In a similar fashion, other algorithms (such as CMA and DD-LMS) and their variants were derived. Using the concepts of reparameterization and priors, it is possible to exploit prior knowledge of the probability density function of the unknown parameters with particular attention to the case of a sparse distribution of taps. The modifications to the algorithms were shown to be component-wise modifications to the step size, and the ANG algorithm philosophy provides a strategy for designing these modifications.

An analysis of the mean-square-error, convergence rates, and stability has been provided, along with simulations that support the results. It was shown that if one has accurate knowledge of the prior, then substantial performance gains can be achieved. Conversely, if a false prior is assumed, performance degradation occurs. More details can be found in [21].

Future work may involve a more detailed approach to choosing the stepsize when the system is unknown, analyzing the stability and convergence behavior of the algorithms in greater detail, considering equalization applications (a DFE in particular) in more depth and comparing bit error rate curves to the MSE curves, examining sensitivities of the algorithms to incorrect priors, and theoretically determining suitable priors of channels from common statistical propagation models.

Appendix

This appendix derives (18). Starting from (16), postulate that the solution for C is in the form of the approximate solution [which is obtained by ignoring the $\mu^2 DRCRD$ term in (16)], plus an order μ^2 error term E_1:

$$C = \frac{\mu}{2}\,\xi_{\min} D + E_1. \tag{A.1}$$

Substituting into (16) and simplifying yields

$$DRE_1 + E_1 RD = \mu DRE_1 RD + \frac{\mu^2}{2}\,\xi_{\min} DRDRD. \tag{A.2}$$

Now, postulate that the error term E_1 is of the form

$$E_1 = \frac{\mu^2}{4}\,\xi_{\min} DRD + E_2 \tag{A.3}$$

where E_2 is an order-μ^3 error term. Substitution of (A.3) into (A.2) yields

$$DRE_2 + E_2 RD = \mu DRE_2 RD + \frac{\mu^3}{4}\,\xi_{\min} D(RD)^3.$$

Thus, at each step, the remaining error term is of the form of the solution obtained by ignoring the term of highest order of μ, plus a yet higher order error term.

Continuing this process indefinitely produces the infinite series

$$C = \frac{1}{2}\,\mu\xi_{\min} D \sum_{i=0}^{\infty} \left(\frac{\mu}{2}\,RD\right)^i$$

which converges if and only if every eigenvalue of $(\mu/2)RD$ has a magnitude less than one. The infinite series can then be expressed as

$$C = \frac{1}{2}\,\mu\xi_{\min} D \left(I - \frac{\mu}{2}\,RD\right)^{-1} \tag{A.4}$$

which is the desired result. ■

References

[1] T. J. Endres, R. A. Casas, S. N. Hulyalkar, and C. H. Strolle, "On sparse equalization using mean-square-error and constant modulus criteria," in *Proc. 34th Annu. Conf. Inform. Sci. Syst.*, vol. 1, Princeton, NJ, 2000, pp. TA7b-7–12.

[2] T. Aboulnasr and K. Mayyas, "Complexity reduction of the NLMS algorithm via selective coefficient update," *IEEE Trans. Signal Processing*, vol. 47, pp. 1421–1424, May 1999.

[3] S. Ariyavisitakul, N. R. Sollenberger, and L. J. Greenstein, "Tap-selectable decision-feedback equalization," *IEEE Trans. Commun.*, vol. 45, pp. 1497–1500, Dec. 1997.

[4] J. Homer, "Detection guided NLMS estimation of sparsely parameterized channels," *IEEE Trans. Circuits Syst. II*, vol. 47, pp. 1437–1442, Dec. 2000.

[5] A. Sugiyama, H. Sato, A. Hirano, and S. Ikeda, "A fast convergence algorithm for adaptive FIR filters under computational constraint for adaptive tap-position control," *IEEE Trans. Circuits Syst. II*, vol. 43, pp. 629–636, Sept. 1996.

[6] J. Kivinen and M. K. Warmuth, "Exponentiated gradient versus gradient descent for linear predictors," *Inform. Comput.*, vol. 132, no. 1, pp. 1–64, Jan. 1997.

[7] S. I. Hill and R. C. Williamson, "Convergence of exponentiated gradient algorithms," *IEEE Trans. Signal Processing*, vol. 49, pp. 1208–1215, June 2001.

[8] R. E. Mahony and R. C. Williamson, "Riemannian structure of some new gradient descent learning algorithms," in *Proc. Adapt. Syst. Signal Process., Commun., Contr. Symp.*, Lake Louise, AB, Canada, 2000, pp. 197–202.

[9] S. Amari, "Natural gradient works efficiently in learning," *Neural Comput.*, vol. 10, no. 2, pp. 251–276, Feb. 1998.

[10] D. L. Duttweiler, "Proportionate normalized least-mean-squares adaptation in echo cancelers," *IEEE Trans. Speech Audio Processing*, vol. 8, pp. 508–518, Sept. 2000.

[11] B. Widrow, J. R. Glover, Jr., J. M. McCool, J. Kaunitz, C. S. Williams, R. H. Hearn, J. R. Zeidler, E. Dong, Jr., and R. C. Goodlin, "Adaptive noise cancelling: Principles and applications," *Proc. IEEE*, vol. 63, pp. 1692–1716, Dec. 1975.

[12] B. Widrow, J. McCool, M. G. Larimore, and C. R. Johnson, Jr., "Stationary and nonstationary learning characteristics of the LMS adaptive filter," *Proc. IEEE*, vol. 64, pp. 1151–1162, Aug. 1976.
[13] W. A. Sethares, B. D. O. Anderson, and C. R. Johnson, Jr., "Adaptive algorithms with filtered regressor and filtered error," *Math. Contr., Signals, Syst.*, vol. 2, no. 4, pp. 381–403, 1989.
[14] R. E. Mahony and R. C. Williamson, "Prior knowledge and preferential structures in gradient descent learning algorithms," *J. Machine Learn. Res.*, vol. 1, pp. 311–355, 2001.
[15] C. P. Robert, *The Bayesian Choice*. New York: Springer, 1994.
[16] I. Mareels and J. W. Polderman, *Adaptive Systems: An Introduction*. Boston, MA: Birkhauser, 1996.
[17] J. A. Bucklew, T. Kurtz, and W. A. Sethares, "Weak convergence and local stability properties of fixed stepsize recursive algorithms," *IEEE Trans. Inform. Theory*, vol. 39, pp. 966–978, May 1993.
[18] S. C. Douglas and S. Amari, "Natural gradient adaptation," in *Unsupervised Adaptive Filtering*. New York: Wiley, 2000, vol. 1, pp. 13–61.
[19] C. R. Johnson, Jr., *Lectures on Adaptive Parameter Estimation*. Englewood Cliffs, NJ: Prentice-Hall, 1988.
[20] C. R. Johnson, Jr., P. Schniter, T. J. Endres, J. D. Behm, D. R. Brown, and R. A. Casas, "Blind equalization using the constant modulus criterion: A review," *Proc. IEEE*, vol. 86, pp. 1927–1950, Oct. 1998.
[21] R. K. Martin, "Exploiting sparsity in adaptive filters," M.S. thesis, Cornell Univ., Ithaca, NY, 2001.
[22] I. Garrison, R. K. Martin, W. A. Sethares, B. Hart, W. Chung, J. Balakrishnan, R. A. Casas, T. J. Endres, P. Schniter, M. G. Larimore, and C. R. Johnson, Jr., "DTV channel characterization," in *Proc. Conf. Inform. Sci. Syst.*, Baltimore, MD, 2001.

Richard K. Martin received the B.S. degrees in physics and electrical engineering (summa cum laude) from the University of Maryland, College Park, in 1999 and the M.S. degree in electrical engineering from Cornell University, Itahca, NY, in 2001.

His research interests include sparse equalization and adaptive channel shortening/adaptive per tone equalization for multicarrier systems. He plans to pursue the Ph.D. degree from Cornell and then seek an academic position.

William A. Sethares received the B.A. degree in mathematics from Brandeis University, Waltham, MA, and the M.S. and Ph.D. degrees in electrical engineering from Cornell University, Ithaca, NY.

He has worked at the Raytheon Company as a Systems Engineer and is currently Associate Professor with the Department of Electrical and Computer Engineering, University of Wisconsin, Madison. His research interests include adaptation and learning in signal processing, communications, and acoustics. He is the author of *Tuning, Timbre, Spectrum, Scale* (New York: Springer, 1998).

Robert C. Williamson received the B.E. degree from the Queensland Institute of Technology, Brisbane, Australia, in 1984 and the M.Eng.Sc. degree in 1986 and the Ph.D. degree in 1990 from the University of Queensland, Brisbane, all in electrical engineering.

Since 1990, he has been with the Australian National University, Canberra, where he is a Professor with the Department of Telecommunications Engineering, Research School of Information Sciences and Engineering. His scientific interests include signal processing and machine learning.

C. Richard Johnson, Jr. (F'89) was born in Macon, GA, in 1950. He received the B.E.E. degree with high honors from the Georgia Institute of Technology, Atlanta, in 1973 and the M.S.E.E. and Ph.D. degrees in electrical engineering with minors in engineering–economic systems and art history from Stanford University in 1975 and 1977, respectively.

He is currently a Professor with the School of Electrical and Computer Engineering, Cornell University, Ithaca, NY. His current research interests are in adaptive parameter estimation theory that is useful in applications of digital signal processing to communications systems.

Dr. Johnson was selected by Eta Kappa Nu as the Outstanding Young Electrical Engineer in 1982 and as the C. Holmes MacDonald Outstanding Teacher of 1983. In 1991, he was selected a Distinguished Lecturer of the IEEE Signal Processing Society.