Fast Adaptive Algorithms using Eigenspace Projections

N.Gopalan Nair[†] and Andreas S. Spanias[‡]

[†]Intel Corporation, Chandler, Arizona 85226.

[‡]Dept of Electrical Engineering, Telecommunications Research Center,
Arizona State University, Tempe, Arizona 85287-7206

Abstract

Although adaptive gradient algorithms are simple and relatively robust, they generally have poor performance in the absence of "rich" excitation. In particular, it is well known that the convergence speed of the LMS algorithm deteriorates when the condition number of the input autocorrelation matrix is large. This problem has been previously addressed using weighted RLS or normalized frequency-domain algorithms. In this paper, we present a new approach that employs gradient projections in selected eigenvector sub-spaces to improve the convergence properties of LMS algorithms for colored inputs. We also introduce an efficient method to iteratively update an "eigen subspace" of the autocorrelation matrix. The proposed algorithm is more efficient, in terms of computational complexity, than the WRLS and its convergence speed approaches that of the WRLS even for highly correlated inputs.

Introduction

In this paper, we present a fast adaptive algorithm that employs gradient projections in selected eigenspaces. The projections are used to improve the convergence speed and numerical behavior of the adaptive algorithm when the excitation is highly colored. We will refer to this algorithm as the Eigenspace Projection Algorithm (EPA). The EPA performs partial eigen-decomposition of the autocorrelation matrix, to iteratively estimate a small set of eigenvalue/eigenvector pairs associated with the weakest modes (smallest eigenvalues) of the adaptive system. We will refer to the subspace spanned by these eigenvectors as the "low subspace". The convergence rate of the tap vector components associated with this low subspace is improved by projecting the gradient on the subspace basis vectors. The projected gradient components are then associated with individual time-varying step sizes. It can be shown that the convergence rate in this subspace approaches that of the RLS algorithm while adaptation speed in the "high subspace" is similar to that of the LMS but with a much smaller eigenvalue spread. Thus, the performance of the proposed EPA approaches that of the RLS algorithm, without inverting the autocorrelation matrix.

The EPA is developed here for pth order sequential FIR system identification scheme. Extension to IIR system or linear prediction is straight forward. The choice of the rank of the low subspace is based on signal statistics that are usually known a priori in many practical applications. This algorithm, however, is most practical when the rank of the low subspace is much smaller than the system order p. The proposed EPA is related to self-orthogonalizing algorithms [1] [4] in the sense that it performs orthogonalization, but only in the low subspace. In the following, we describe the EPA along with an efficient eigenspace updating scheme. We also give preliminary results and remarks.

The Eigenspace Projection Algorithm

Consider an adaptive linear combiner with p taps and with the input vector at time k defined as, $\mathbf{x}(k) = [x(k),x(k-1),...,x(k-p+1)]^T$. Assume that at most m ($m<p$) eigenvalues of the autocorrelation matrix, $\mathbf{r}=E[\mathbf{x}(k)\mathbf{x}^T(k)]$ are small, thereby causing the algorithm to converge slowly. Also assume that the p eigenvalues of $\mathbf{r}$ are arranged such that, $0<\lambda_1<\lambda_2<..<\lambda_p$, and $\mathbf{q}=[\mathbf{q}_1,\mathbf{q}_2,...,\mathbf{q}_p]$ is the unitary matrix that diagonalizes the autocorrelation matrix, $\mathbf{r}$. We then define $\mathbf{q}_L = [\mathbf{q}_1,\mathbf{q}_2,..,\mathbf{q}_m]$ as the rank-m matrix spanned by the 'm' eigenvectors associated with the smallest eigenvalues, and $\mathbf{q}_H = [\mathbf{q}_{m+1},\mathbf{q}_{m+2},..,\mathbf{q}_p]$. We call the subspace spanned by $\mathbf{q}_L$ and $\mathbf{q}_H$ as "low subspace" and "high subspace" respectively. At iteration k, the gradient vector $\nabla_\xi(k)$ is written as,

$$\nabla_\xi(k) = \nabla_{\xi,L}(k) + \nabla_{\xi,H}(k) \qquad (1)$$

where

$$\nabla_{\xi,L}(k)=\mathcal{P}_{q,L}(\nabla_\xi(k)) \text{ and } \nabla_{\xi,H}(k)=\mathcal{P}_{q,H}(\nabla_\xi(k)) \qquad (2)$$

are projections of the gradient vector on the low subspace and high subspace, respectively. The low and high subspaces are orthogonal, i.e.,

$$\mathbf{q}_L^T \nabla_{\xi,H}(k) = \mathbf{0}; \quad \mathbf{q}_H^T \nabla_{\xi,L}(k) = \mathbf{0} \tag{3}$$

In many practical applications some a priori understanding of the input signal statistics is available. For example, we usually have knowledge of the approximate bounds on the condition number (ratio of maximum to minimum eigenvalue) of the autocorrelation matrix. The EPA is most useful when the eigenvalues of the input autocorrelation matrix are minimally dispersed except for a few eigenvalues that are small relative to the maximum eigenvalue. It is well known that the overall convergence performance of the gradient algorithm is heavily dependent on the smallest eigenvalue of the input autocorrelation. The tap vector convergence dynamics for the gradient algorithm can be conveniently represented using its *natural modes* [1], by first defining the new set of coordinates as:

$$\mathbf{v}(k) = \mathbf{q}^T(\mathbf{w}(k) - \mathbf{w}_0) \tag{4}$$

where $\mathbf{w}_0$ is the optimum tap vector, and $\mathbf{v}(k)$ is the tap error vector at iteration k. $\mathbf{v}$ is represented in the primary coordinate system, with the eigenvectors of the

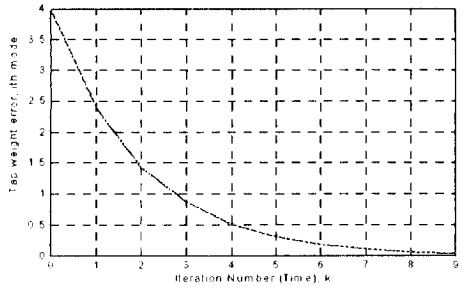

Figure 1. Exponential convergence of primary modes of taps

input autocorrelation matrix as the coordinates. The trajectory of the taps can then be written as

$$\mathbf{v}(k) = (\mathbf{I} - \mu\Lambda)^k \mathbf{v}(0) \tag{5}$$

where Λ is the p-dimensional diagonal matrix with the eigenvalues of $\mathbf{r}$ as its diagonal elements, and $\mathbf{v}(0)$ is the initial tap error vector. Rewriting the equation in scalar form, tap error trajectory for the ith natural mode of the adaptive system is given by [1]

$$v_i(k) = (1 - \mu\lambda_i)^k v_i(0) \tag{6}$$

For asymptotic convergence and stability, it is clear that the step size parameter μ of the adaptive system should be such that $0 < \mu < 2/\lambda_{max}$. An exponential envelope with time constant τ_i can be fitted to the geometric series of the tap vector error. Convergence for the ith component can be approximated as,

$$\tau_i \cong \frac{1}{\mu\lambda_i}; \quad \mu << 1. \tag{7}$$

Equation (7) shows that the tap trajectory in each of the primary modes have inverse relationship with the corresponding eigenvalues. The tap error converges fastest when the eigenvalues are equal.

For the development of the Eigenspace Projection Algorithm, it is assumed that the autocorrelation matrix, $\mathbf{r}$, has *p-m* sufficiently large and minimally dispersed eigenvalues. Thus, if $\mathbf{w}(0)$ lies in the subspace spanned by the eigenvectors associated with these *p-m* eigenvalues, then a gradient algorithm with a single step size converges sufficiently fast [5][6]. However, for the tap components in the *low subspace,* the rate of convergence can be very slow. Eigenspace projection allows use of larger independent step sizes for the *m* weaker modes thereby improving adaptation speed. Figure 2. shows a simplified structure of the EPA, as applied to system identification. The algorithm can be written as,

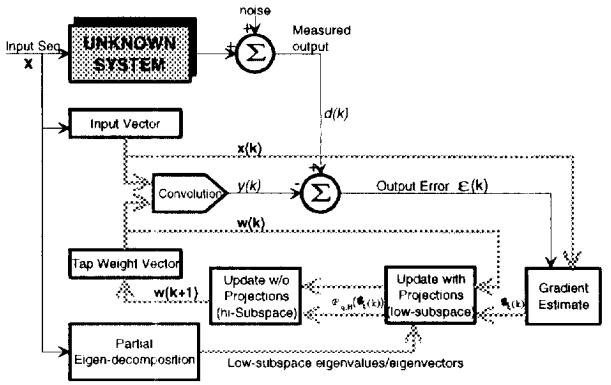

Figure 2. The Eigenspace Projection Algorithm

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \sum_{i=1}^{m} \mu_i \mathcal{P}_{q,i}\left(\nabla_\xi(k)\right) - \mu_H \nabla_{\xi,H}(k) \tag{8}$$

where,

$$\mathcal{P}_{q,i}\left(\nabla_\xi(k)\right) = \left(q_i^T \nabla_\xi(k)\right) q_i \tag{9}$$

is the projection of the gradient vector on the eigenvector $\mathbf{q}_i$ and,

$$\nabla_{\xi,H}(k) = \nabla_{\xi}(k) - \sum_{i=1}^{m} \mathcal{P}_{q,i}\left(\nabla_{\xi}(k)\right) \qquad (10)$$

The step sizes are chosen as follows.

$$\mu_i = \frac{\beta_{\mu i}}{2\lambda_i}; \quad 0 < \beta_{\mu} < 1 \; ; \quad \mu_H = \frac{\beta_{\mu H}}{2\lambda_{max}} \qquad (11)$$

where independent step sizes, μ_i, i=1,..m, are chosen for the gradient projections on the low subspace. The low subspace eigenvalues are iteratively estimated and the step-size scale factors, $\beta_{\mu i}$ are chosen to provide the desired convergence time constant τ_i, such that,

$$\beta_{\mu i} \cong \frac{2}{\tau_i}. \qquad (12)$$

The algorithm can be structured in either sequential or block mode. In a typical sequential implementation, $\mathcal{P}_{q,i}\left(\nabla_{\xi}(k)\right)$, and $\nabla_{\xi,H}(k)$ can be iteratively computed by:

i. using projections of the gradient estimate on successive eigenvectors:

$$\mathcal{P}_{q,i}\left(\nabla_{\xi}(k)\right) = \left(q_i^{T}\nabla_{\xi}(k)^{(i)}\right) q_i \qquad (13)$$

ii. updating the component in the *i*th primary direction of the tap vector by,

$$\mathbf{w}(k+1)^{(i)} = \mathbf{w}(k) - \mu_i \mathcal{P}_{q,i}\left(\nabla_{\xi}(k)\right) \qquad (14)$$

iii. computing the residual gradient estimate, orthogonal to the previously projected eigenvector, i.e.,

$$\nabla_{\xi}(k)^{(i+1)} = \nabla_{\xi}(k)^{(i)} - \mathcal{P}_{q,i}\left(\nabla_{\xi}(k)\right) \qquad (15)$$

with $$\nabla_{\xi}(k)^{(1)} = \nabla_{\xi}(k) \qquad (16)$$

The superscript $(.)^{(i)}$ indicates operation on the *i*th eigenvector projection at iteration k. Use of instantaneous gradient estimates in the projections associated with large step sizes can result in poor transient performance due to excessive gradient noise [3]. Therefore we introduced a filter for the gradient estimates, that is

$$\hat{\nabla}_{\xi}(k) = \gamma_{\nabla}\hat{\nabla}_{\xi}(k-1) + (1-\gamma_{\nabla})\mathbf{x}(k)\varepsilon(k) \qquad (17)$$

where $0<\gamma_{\nabla}<1$, is the forgetting factor, and $\varepsilon(k)$ is the output error.

Since the autocorrelation matrix, $\mathbf{r}$, is symmetric and toeplitz, we need to estimate and track only the first row. The remaining rows are then derived from the first row. At each input sample, we use a forgetting factor, γ_r, to update $\mathbf{r}_1(k)$ using

$$\mathbf{r}_1(k+1) = \gamma_r \mathbf{r}_1(k) + (1-\gamma_r)x_1(k+1)\mathbf{x}^{T}(k+1) \; ;$$

$$0 < \gamma_r < 1 \qquad (18)$$

where, $x_1(k+1)$ is the first element of the input vector, $\mathbf{x}(k+1)$, and $\mathbf{r}_1(k)$ is the first row of $\mathbf{r}$. When $\mathbf{r}$ is stationary or slowly time varying, the computational load for its partial eigendecomposition can be significantly reduced by updating the eigenspace less frequently than the taps. The updates can be done every N iterations, N>>1, and the computed eigenvalue/eigenvector pairs can be used for the next N iterations.

We propose here, a technique for finding the basis vectors for the m dimensional low subspace. This is based on a combination of the inverse power method for finding the smallest eigenvalue/eigenvector pair of a matrix, and the Gauss-Seidel iteration. We note that in our development of the EPA, we use a partial Gauss-Seidel iteration, i.e., we are not computing the matrix inverse.

The inverse power method for finding the smallest eigenvalue of a square matrix, $\mathbf{r}$, and the associated eigenvector operates as follows. Starting with an initial guess, $\mathbf{u}(0)$, which has a non-zero first element, successively update $\mathbf{u}$, i.e., $\mathbf{u}(1)=\mathbf{r}^{-1}\mathbf{u}(0)$, $\mathbf{u}(2)=\mathbf{r}^{-1}\mathbf{u}(1)$, .. $\mathbf{u}(n)=\mathbf{r}^{-1}\mathbf{u}(n-1)$. After n iterations, $\mathbf{u}(n)=(\mathbf{r}^{-1})^{n}\mathbf{u}(0)$ can be written as:

$$\lambda_1^{n}\mathbf{u}(n) = c_1\mathbf{q}_1 + c_2\left(\frac{\lambda_1}{\lambda_2}\right)^{n}\mathbf{q}_2 + \ldots + c_p\left(\frac{\lambda_1}{\lambda_p}\right)^{n}\mathbf{q}_p \qquad (19)$$

After each iteration, the direction of $\mathbf{u}(n)$ points more and more towards the direction of the eigenvector associated with the smallest eigenvalue of $\mathbf{r}$. By using the first element $u_1(n)$ of $\mathbf{u}(n)$ as the scale factor for each iteration, i.e., $\mathbf{u}(n+1)=\mathbf{r}^{-1}\mathbf{u}(n)/u_1(n)$, we get a scale factor that converges to the reciprocal of the smallest eigenvalue, $1/\lambda_1$ [2]. The inverse power method is a robust technique, and converges rapidly if the ratio of the smallest to all other eigenvalues (λ_1/λ_i) is small. However, the basic inverse power method requires computation of the autocorrelation inverse. Explicitly computing the matrix inverse is computationally intensive - often associated with numerical difficulties, when small eigenvalues are encountered. Instead, we iteratively solve the equation, $\mathbf{ru}(n+1)= \mathbf{u}(n)/u_1(n)$. The

technique can be developed from the basic Gauss-Seidel method for solving, **ax=b** [2],

$$a_{ii}x_i(n+1) = a_{ii}x_i(n) + b_i - \sum_{j=1}^{i-1} a_{ij}x_j(n+1) - \sum_{j=i}^{p} a_{ij}x_j(n) \quad (20)$$

where $\mathbf{a}=\{a_{ij}\}$ is a *pxp* square matrix, and $\mathbf{x}=\{x_i\}$, $\mathbf{b}=\{b_i\}$, are *px1* vectors. Replacing the constant vector, **b**, by the auxiliary vector $\mathbf{u}(n)/u_1(n)$, the iterations for estimating the smallest eigenvalue and the associated eigenvector, $\{\hat{\lambda}_1, \hat{\mathbf{q}}_1\}$ can be written as,

$$\hat{\lambda}_1(n) = \frac{1}{|u_1(n)|}; \quad \hat{\mathbf{q}}_1(n) = \frac{\mathbf{u}(n)}{\|\mathbf{u}(n)\|} \quad (21)$$

where $|u_1(n)|$ is the absolute value of the first element of the auxiliary vector, and $\|\mathbf{u}(n)\|$ is the magnitude of the vector. Substituting in equation (20), we get

$$r_{ii}u_i(n+1) = r_{ii}u_i(n) + \hat{\lambda}_1(n)u_i(n) - \sum_{j=1}^{i-1} r_{ij}u_j(n+1) - \sum_{j=i}^{p} r_{ij}u_j(n) \quad (22)$$

We introduce the notation, $\mathbf{u}(n+1,i,n)$ to represent the vector $\mathbf{u}(n)$, with in-place updates of the first (i-1) elements at the (n+1)st iteration. The remaining elements are at iteration n. Using this notation, the above equation becomes,

$$r_{ii}u_i(n+1) = r_{ii}u_i(n) + \hat{\lambda}_1(n)\,u_i(n) - \mathbf{r}_i\,\mathbf{u}(n+1,i,n) \quad (23)$$

Using cyclic shifts of the estimate of the first row of **r** (from equation (18)), the iteration to update the auxiliary vector can be rewritten as:

$$u_i(n+1) = \frac{\left(1+\hat{\lambda}_1(n)\right)u_i(n) - \hat{\mathbf{r}}_i'\mathbf{u}(n+1,i,n)}{\hat{r}_{1,1}} \quad (24)$$

$\hat{\mathbf{r}}_i'$ is the approximation of the *i*th row of the autocorrelation matrix, computed from the estimate of the first row, as,

$$\hat{\mathbf{r}}_i' \equiv \left\{\hat{\mathbf{r}}'_{i,j}\right\} = \hat{\mathbf{r}}_{1,i-j+1}; \quad i > j$$
$$= \hat{\mathbf{r}}_{1,1}; \quad i = j$$
$$= \hat{\mathbf{r}}_{1,j-i+1}; \quad i < j \quad (25)$$

Iterations for the estimation of the *m* lowest order eigenvalue/eigenvector pairs of the autocorrelation matrix, thus rely on the *m* auxiliary vectors, $[\mathbf{u}_1, \mathbf{u}_2, .. \mathbf{u}_m]$. These are initially chosen to be an orthogonal set, with non-zero first elements. The smallest eigenvalue/eigenvector pair and the associated auxiliary vector, $\mathbf{u}_1$, are first updated using equation (24). The second lowest (j=2) and successively higher order eigenspace components are then updated, after orthogonalizing its estimate from the past iteration, against the lower order eigenspace. The iteration for $\{\hat{\lambda}_s, \hat{\mathbf{q}}_s, \mathbf{u}_s\}$, s=2:m can be written as,

$$\mathbf{u}_s(n) = \mathbf{u}_s(n) - \left(\mathbf{u}_s(n)\cdot\hat{\mathbf{q}}_{s-1}\right)\hat{\mathbf{q}}_{s-1} \quad (26)$$

$$\hat{\lambda}_s(n) = \frac{1}{|u_{s,1}(n)|}; \hat{\mathbf{q}}_s(n) = \frac{\mathbf{u}_s(n)}{\|\mathbf{u}_s(n)\|} \quad (27)$$

$$u_{s,i}(n+1) = \frac{\left(1+\hat{\lambda}_{s(n)}\right)u_{s,i}(n) - \hat{\mathbf{r}}_s'\mathbf{u}_s(n+1,s,n)}{\hat{r}_{1,1}} \quad (28)$$

To summarize, The Eigenspace Projection Algorithm consists of the following steps:

i. Iterative estimation and update of the performance error surface gradient, equation (17),
ii. Iterative estimation and update of the first row of the autocorrelation matrix, equation (18),
iii. Iterations to update the low subspace eigenvalues and the associated eigenvectors, equations (26), (27), and (28), and
iv. Tap updates with eigenspace projections, based on equations, (13),(14),(15), and (16)

Simulation Result

Figure 3 shows the typical comparison of learning curves of the EPA, and the LMS algorithm. Both algorithms are applied in a system identification. scenario where the adaptive system is FIR with 9 taps excited by colored noise. The EPA uses projections on only two of the nine eigenvectors (i.e., p=9 and m=2). The colored input is generated by passing a random sequence through an 8th order all-pole filter, with pole pairs at [0.8 exp(±j.707), 0.8 exp(±j1.335), 0.8 exp(±j1.964), 0.9 exp(±j2.553)]. The eigenvalues of the input autocorrelation matrix were approximately [2.1, 1.97, 1.39, 1.37, 1.13, .84, .46, .39, .21]. The step-sizes for the simulations were $\mu = 0.09$ for the LMS, and $\beta_{\mu 1} = \beta_{\mu 2} = 0.12$, $\beta_{\mu H} = .2$ for the EPA (as defined in (11)). These parameters were experimentally chosen for the fastest possible convergence rate without introducing instability.

It can be shown that the computational requirement for the tap weight and autocorrelation matrix updates for a p-tap FIR adaptive filter, with projections on the m-dimensional low subspace is approximately $(m+1)3p$

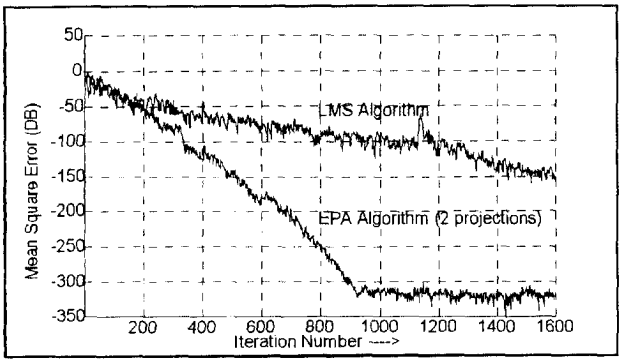

Figure 3. Typical Convergence Performance

operations. In addition, the eigenvector updates for the low subspace require approximately $(3p+p^2)$ multiply-adds and p divides per eigenvector, thus requiring an average of $(p+4)mp/N$ operations per input sample, if updated every N samples. Therefore, the increase in computational load for the EPA compared to the conventional LMS algorithm is modest if $m<<p$, and N is large.

Remarks:

A new adaptive algorithm based on partial eigendecomposition of the autocorrelation matrix and gradient vector projections on this eigenspace is proposed. A iterative method for estimating the eigenvalues and eigenvectors of the low subspace was also introduced. It was shown that the EPA performs better than the LMS when the excitation is colored. This comes with a modest increase in computational complexity.

References:

[1] S.Haykin, *Adaptive Filter Theory*, Prentice Hall Inc., 1991.

[2] G.Strang, *Linear Algebra and its Applications*, Harcourt Brace Jovanovich Inc., 1988.

[3] B.Widrow and S.D.Stearns, *Adaptive Signal Processing*, Prentice Hall Inc., 1985.

[4] R.D.Gitlin and F.R.Magee Jr., "Self-orthogonalizing adaptive equalization algorithms", *IEEE Trans. Communications*, vol. COM-25, no. 7, pp. 666-672, July 1977.

[5] F.M.Boland and J.B.Foley, "Stochastic convergence of the LMS algorithm in adaptive systems", *signal Processing*, vol. 13, no. 4, pp. 339-352, December 1987.

[6] R.D.Gitlin, J.E.Mazo and M.G.Taylor, "On the design of gradient algorithms for digitally implemented adaptive filters", *IEEE Trans. Circuit Theory*, vol. CT-20, no. 3, pp. 125-136, March 1973.

[7] W.B.Mikhael, F.H.Wu, L.G.Kazovsky, G.S.Kang, and L.J.Fransen, "Adaptive filters with individual adaptation of parameters", *IEEE Trans. Circuits and Systems.*, vol. CAS-33, no. 7, pp. 677-686, July 1986.