INCREASING THE PERFORMANCE OF THE LMS ALGORITHM USING AN ADAPTIVE PRECONDITIONER

I. K. Proudler, I.D. Skidmore, and J.G. McWhirter.
Rm. E506, DRA, St. Andrews Road, Malvern, Worcestershire, WR14 3PS, UK.
Tel. +44 1684 894228 Fax. +44 1684 896502
e-mail: proudler@signal.dra.hmg.gb

ABSTRACT

In this paper we outline a technique for increasing the convergence rate of the LMS algorithm by means of a preconditioning filter which reduces the eigenvalue spread of the input signal. Specifically we use a low order linear prediction lattice filter followed by a tapped-delay-line as the preconditioner. Some computer simulations are provided to demonstrate the increased convergence rate of the new algorithm.

1 INTRODUCTION

Adaptive digital filters have many potential applications including interference suppression, echo cancellation, and channel equalisation. Traditionally [1] there have been two classes of algorithms: those based on the LMS algorithm and those on the RLS algorithm. The properties of these algorithms are well known. Two of the most important characteristics are convergence rate and computational load. Clearly, the former determines whether or not the algorithm is suitable for a given application whereas the latter determines if the algorithm can be implemented in a cost effective manner. The LMS algorithm requires very few operations but its convergence rate is a function of the input signal and can be very slow. The RLS algorithm, on the other hand, converges very quickly but has a much higher computational load.

Recently there has been much interest in the fast Newton class [2] of adaptive filtering algorithms. Under the assumption that the input signal is produced by an AR process of order less that the size of the adaptive filter, these algorithms provide a fast convergence rate, typical of the RLS algorithm, for a computational load more similar to that of the LMS algorithm. Since speech is known to modelled quite well as the output of an AR(10) process, the fast Newton algorithm could be used to provide a cost effective adaptive filtering algorithm for voice communications applications.

The fast Newton algorithm is based on the standard RLS algorithm [1] which requires an estimate of the data covariance matrix. If the filter is of order N, the data consists of the input signal and its (N-1) lags. In the case that this signal is produced by a AR(M) process, it is possible to theoretically predict this N×N matrix given its leading M×M submatrix. By taking advantage of this result, part of the computation can be circumvented and the fast Newton algorithm results. A similar idea was used by Gardiner et al. [3] to derive a LS ladder/lattice algorithm with reduced computational load but RLS-like performance. Here the reflection coefficients for order N>M can be shown theoretically to be zero. The Nth order LS lattice part of the algorithm thus reduces to a Mth order lattice and a (N-M) tapped-delay-line (TDL).

In this paper an alternative approach to the problem is presented. A low order, non-adapting lattice filter followed by a tapped delay line is used as a preconditioner in conjunction with the LMS algorithm. The concept of a preconditioner is briefly described in section 2. The idea of the 'clamped' lattice preconditioner is developed in section 3. Some computer simulations that demonstrate the increased convergence rate of the new algorithm are presented in section 4.

2 PRECONDITIONING

The LMS algorithm is a stochastic gradient decent algorithm. The dependence of the convergence rate of such algorithms with the eigenvalue spread of the data covariance matrix is well known [1]. One solution to this problem is the use of a preconditioner [4]. This is an invertible transformation of the data such that the transformed data covariance matrix has a much smaller eigenvalue spread. Consider the optimum filter coefficients for the Wiener filtering problem $\underline{w}^T(n)\underline{x}(n) \approx y(n)$:

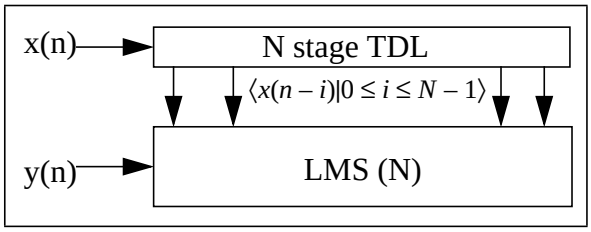

Figure 1 LMS Adaptive FIR Filter

$$\underline{w}(n) = (X^H(n)X(n))^{-1}(X^H(n)\underline{y}(n)) \quad (1)$$

where $X(n)$ is the data matrix and $\underline{y}(n)$ the desired signal vector:

$$X(n) = \left[\underline{x}(1), \ldots, \underline{x}(n)\right]^T \quad \underline{y}(n) = \left[y(1), \ldots, y(n)\right]^T \quad (2)$$

Now let $\Phi(n)$ be a fixed, non-singular matrix. If the original data is transformed using $\Phi(n)$ so that $X'(n) = X(n)\Phi(n)$ then it is easy to show that:

$$\underline{w}(n) = \Phi(n)\underline{w}'(n) \quad (3)$$

where $\underline{w}'(n)$ is the solution to the preconditioned filtering problem:

$$\underline{w}'^T(n)\underline{x}'(n) \approx y(n) \quad (4)$$

If the transformation $\Phi(n)$ is chosen correctly, the eigenvalue spread of the new covariance matrix $X'^H(n)X'(n)$ can be much smaller than for the original problem. This means that it is possible to solve the new problem (equation (4)) faster, using the LMS algorithm, than it would the original one. The solution to the original problem can be found via equation (3).

Clearly, an optimum preconditioner is a transformation that results in a new data matrix that has an eigenvalue spread of unity i.e. produces an orthonormal matrix. Such a transformation is well known in adaptive filtering: Gram-Schmidt orthogonalisation (GSO). Here the preconditioner is a (full rank) upper triangular matrix $R^{-1}(n)$ and the transformed matrix is orthonormal:

$$X(n)R^{-1}(n) = Q(n) \quad (5)$$

In a Nth order adaptive filtering problem (see figure 1), the columns of the data matrix $X(n)$ consist of the vector:

$$\underline{x}(n) = \left[x(1) \ \ldots \ x(n)\right]^T \quad (6)$$

and the (N-1) lags $\langle \underline{x}(n-i)|1 \le i \le N-1 \rangle$. In this case, it is well known that the columns of the matrix $Q(n)$ consists of the N backward prediction residuals normalised to unit norm. The convergence rate of an LMS adaptive filter can thus be increased by first transforming the tapped-delay-line data into the normalised backward prediction residuals. Clearly, the manner in which the backward residuals are produced

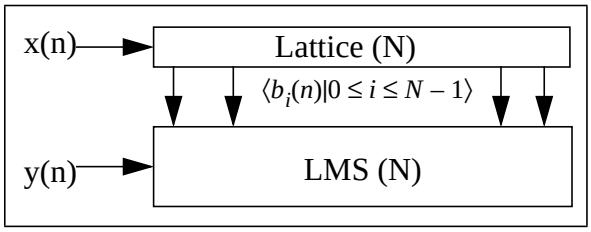

Figure 2 LMS Algorithm with Lattice Preconditioner

is not important. Thus an alternative to GSO, is to use an Nth order linear prediction lattice filter to generate the backward prediction residuals - with appropriate normalisation (figure 2). The advantage in doing this is that a lattice algorithm requires less computation than GSO. The drawback is that the preconditioning transformation, $R^{-1}(n)$ of equation (5), is not explicitly available. It is therefore not possible to determine the original optimum filter coefficients (equation (3)). Note, however, if the quantity of interest is the adaptive filter residual, as in echo cancellation for example, then the filter coefficients are not explicitly required.

The reflection coefficients in the lattice filter will clearly need to be calculated in some manner. The obvious solution is to use an adaptive LS lattice algorithm to estimate the correct coefficients from a short section of the data. If the adaptive filter order is large this could require a large amount of data (typically 10N); however assuming that the input signal is produced by a low order AR process, a clamped lattice [3] can be used.

3 CLAMPED PRECONDITIONER

If the input signal $x(n)$ is assumed to be from an AR(M) process (M<N), then the Nth order LP lattice filter can be shortened [3] to a Mth order one followed by a (N-M) stage tapped delay line (figure 3). This simplification follows from the fact that the Mth order backward residual time series is, theoretically, white. Alternatively, from [2] we have (equation 6c):

$$\underline{\omega}_{b,\,p}(n) = \begin{bmatrix} 0 \\ \underline{\omega}_{b,\,p-1}(n-1) \end{bmatrix} \quad (7)$$

where $\omega_{b,\,p}(n)$ (p>M) is the backward prediction coefficients for order p at time n. Thus

$$b_p(n) = \underline{\omega}^T_{b,\,p}(n)\underline{x}_p(n) \quad (8)$$

$$= \left[0 \ \ \underline{\omega}^T_{b,\,p-1}(n-1)\right] \begin{bmatrix} x(n) \\ \underline{x}_{p-1}(n-1) \end{bmatrix} = b_{p-1}(n-1) \quad (9)$$

Provided the input signal is generated by an AR(p) process ($p \le M$), the clamped lattice preconditioner is still optimum (if the signals are normalised). When the

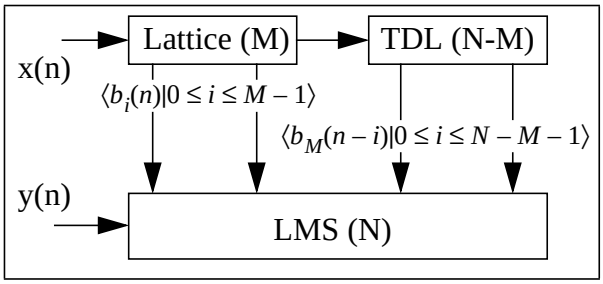

Figure 3 Preconditioned LMS Algorithm

input signal generating process is AR(p) ($p > M$), or even not AR at all, the Mth order backward residual will not be white as expected. The transformed data covariance matrix will not be a scaled unit matrix and the preconditioner will not be optimum. Note, however, that the data from this sub-optimum preconditioner still spans the original data space. Thus the LMS algorithm can still find a suitable linear combination of the signals to correctly minimise the output error. This can be seen in the extreme case of M=0 i.e. when there is no lattice filter just a tapped-delay-line. The preconditioned LMS algorithm clearly reduces to an ordinary LMS adaptive filtering algorithm (figure 1).

Although the preconditioner may be sub-optimal when ($p > M$), it is possible that the preconditioned data is still better conditioned than the original data. This is because even with a smaller lattice than required, some of the correlation in the data will have been removed. As a result the LMS algorithm may still converge faster than if the preconditioner was not used.

Thus far we have assumed that the backward residual time series are normalised. This is required to ensure that the transformed data covariance matrix is a scaled unit matrix. Without this normalisation the transformed data covariance matrix will probably not have a reduced eigenvalue spread, however, it will be diagonal. Recall that the LMS algorithm can be viewed as a RLS algorithm with the covariance matrix estimate replaced by a scaled unit matrix. Thus it is likely to perform better if the input signals have a diagonal covariance matrix rather than a dense one. Thus the normalisation of the residuals may not be necessary (see figure 5).

It should be noted that the use of backward linear prediction to improve the convergence rate of gradient descent algorithms in not new. Griffiths [5], for example, proposed the use of the orthogonality of the backward residuals to decouple the update of the filter coefficients and hence increase the convergence rate. This resulted in what became known as the Gradient Adaptive Lattice algorithm and whilst it has a higher

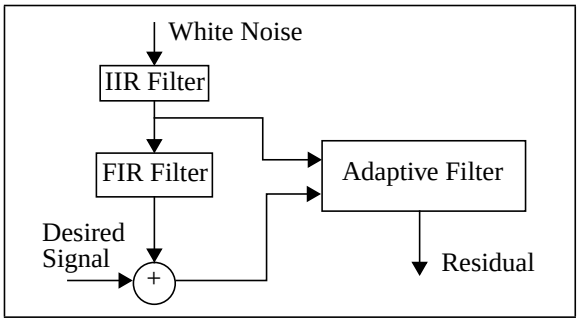

Figure 4 Computer Simulation Experiment

convergence rate than the LMS algorithm it is not as fast as the RLS algorithm.

An Mth order (non-adapting) lattice filter requires 2M operations per time step whilst the LMS algorithm requires 2N operations and normalisation requires an extra M operations. Thus the clamped lattice preconditioner algorithm requires 2(M+N) or (3M+2N) operations per time step depending on whether or not the backward residuals are normalised. Note that these figures ignore the extra operations need to estimate the lattice reflection coefficients. The fast Newton algorithm requires (5M+2N) operation.

4 SIMULATIONS

To test the new algorithm, a computer simulation experiment was performed based on an echo cancellation problem. A 3rd order AR process was constructed such that data from a 10 stage tapped-delay-line had a covariance matrix with a condition number of approximately 3000. This signal was used as the input to the adaptive filter. It was also passed through a 10th order FIR filter which represents the echo path. Various 10th order adaptive filters were used to cancel the 'echo' signal (figure 4). The behaviour of the adaptive filter was monitored by means of the mean-square a-priori error. Each of the following plots was obtained form a single run with the averaging being performed over 20 adjacent samples. In all cases, 50 data samples were used to estimate the lattice filter coefficients in the preconditioned LMS algorithm.

Figure 5 shows the performance of the preconditioned LMS algorithm (with and without normalisation) when the order of the AR (M) process is known a-priori. The standard (normalised) LMS algorithm, a RLS lattice and the fast Newton algorithm (with exact M) are also shown. The NLMS algorithm did converge to a solution but the data had a large condition number and the algorithm converges very slowly. Clearly, the preconditioned LMS algorithm (with and without normalisation) performs well converging faster than normalised LMS algorithm and

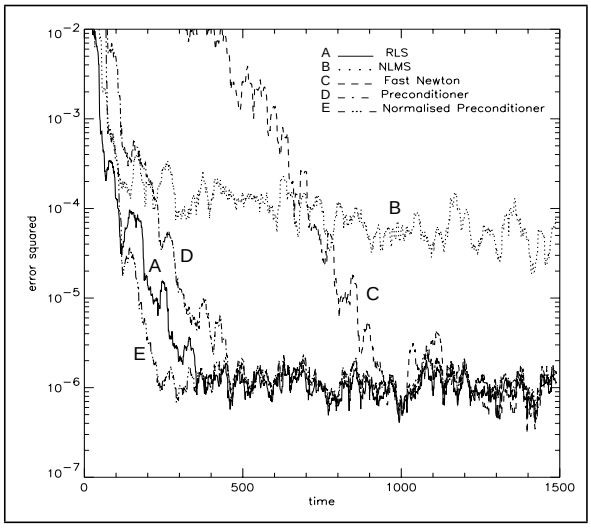

Figure 5 Convergence Rates (Optimum M).

being comparable with the RLS lattice. During this initialisation phase, the preconditioned LMS algorithm appears to out perform the fast Newton algorithm (but see figure 7).

Figure 6 shows the effect of an incorrect estimate of the AR order (M). The Fast Newton and the preconditioned algorithm will both converge to a solution even if M is incorrect. The rate of convergence can, however, be adversely affected. Figure 6 shows the variation in the MSE measured over the time interval [1000,1200] for the two algorithms when the AR order estimate (M) is allowed the vary. Here the correct AR order is three. The figure shows that both algorithms have converged by the time n=1000 as long as M≥3. Below this value the algorithms convergence rate is being adversely affected; although the preconditioned algorithms appears to be the better of the two.

Figure 7 shows the MSE for the preconditioned LMS and the fast Newton algorithms when the order of the AR process changes abruptly. Unlike the fast Newton algorithm, the preconditioned LMS algorithm needs to re-estimate periodically the coefficients of the preconditioner. In this experiment, it was assumed that the algorithm 'detected' the need to readapt 10 samples after the change in statistics. It is interesting to note that whilst the preconditioned algorithm was better than the fast Newton algorithm during initialisation, after an abrupt change there is little to choose between them.

5 DISCUSSION

An alternative approach to the construction of cost effective, high performance adaptive filtering algorithms has been presented. It is based on the well established idea of a data preconditioner. The algorithm

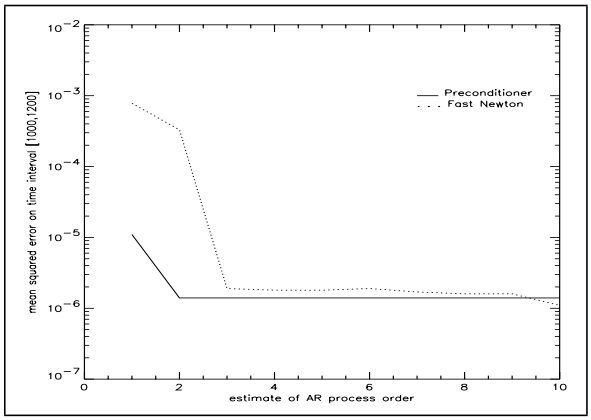

Figure 6 Convergence vs. Assumed AR Order

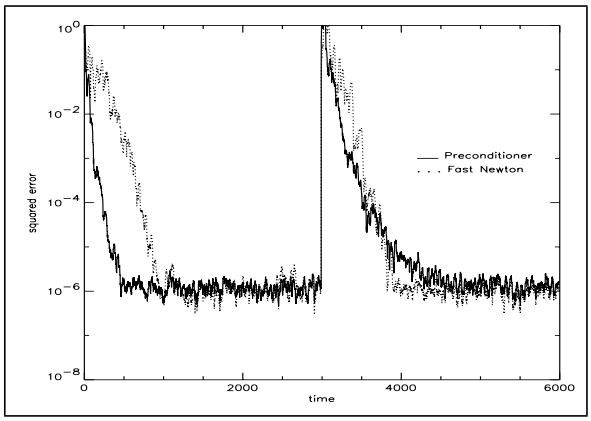

Figure 7 Convergence Rate: Change Of Statistics

performs well and requires marginally less operations than the fast Newton algorithm although suffers from the need to periodically re-estimate the preconditioner coefficients. It would clearly be advantageous to extend this algorithm to allow the preconditioner to adapt continuously.

6 REFERENCES

[1] S Haykin, "Adaptive Filter Theory", 2nd Edition, Prentice-Hall, New Jersey, USA, 1991.

[2] "Fast Newton Transversal Filters - A New Class of Adaptive Estimation Algorithms", G.V. Moustakides & S. Theodoridis, IEEE trans. SP-39(6), pp. 2184-2193.

[3] "Noise Cancellation Studies using a Least-Squares Lattice Filter", T. Gardiner, J.G. McWhirter & T.J. Shepherd, Proc. ICASSP'85, Florida, USA.

[4] O. Axelsson, Iterative Solution Methods, Cambridge Press, 1994.

[5] L.J. Griffiths, "A Continuously Adaptive Filter Implemented as a Lattice Structure", Proc. IEEE ICASSP'77, Hartford, CT, USA, pp. 683-686.