ALGORITHMS AND ARCHITECTURES FOR SPLIT RECURSIVE LEAST SQUARES

K. J. Ray Liu and An-Yeu Wu
Electrical Engineering Department and Institute for Systems Research
University of Maryland
College Park, MD 20742

Abstract - In this paper, a new computationally efficient algorithm for recursive least-squares (RLS) filtering is presented. The proposed *Split RLS* algorithm can perform the approximated RLS with $O(N)$ complexity for signals having no special data structure to be exploited. Our performance analysis shows that the estimation bias will be small when the input data are less correlated. We also show that for highly correlated data, the orthogonal preprocessing scheme can be used to improve the performance of the Split RLS. The systolic implementation of our algorithm based on the QR-decomposition RLS (QRD-RLS) array requires only $O(N)$ hardware complexity and the system latency can be reduced to $O(\log_2 N)$. A major advantage of the Split RLS is its superior tracking capability over the conventional RLS under non-stationary environments.

INTRODUCTION

The family of recursive least-squares (RLS) adaptive algorithms are well known for their superiority to the LMS-type algorithms in both convergence rate and misadjustment [1]. However, the $O(N^2)$ computational complexity becomes the major drawback for their applications as well as for their cost-effective implementation. To alleviate the computational burden of the RLS, the family of fast RLS algorithms such as fast transversal filters, RLS lattice filters, and QR-decomposition based lattice filters (QRD-LSL), have been proposed [1]. By exploiting the special structure of the input data matrix, they can perform RLS estimation with $O(N)$ complexity. One major disadvantage of the fast RLS algorithms is that they work for data with shifting input only. In many applications like multichannel adaptive array processing and image processing, the fast RLS algorithms cannot be applied because no special matrix structure can be exploited. In this paper, we propose an *approximated* RLS algorithm based on the *projection method* [2][3][4][5]. Through multiple decomposition of the signal space and making suitable approximations, we can perform RLS for non-structured data with $O(N)$ complexity. Thus, both the complexity problem in the conventional RLS and the data constraint in the fast RLS can be resolved. We shall call such RLS estimation the *Split RLS*. The systolic implementation of the Split RLS based on the QR-decomposition RLS (QRD-RLS) systolic array in [6] is also proposed. The hardware complexity for the resulting RLS array can be reduced to $O(N)$ and the system latency is only $O(\log_2 N)$.

It is noteworthy that since approximation is made while performing the Split RLS, our approach is not to obtain exact least-squares (LS) solutions. The approximation errors will introduce misadjustment (bias) to the LS errors. In order to know under what circumstances the algorithm will produce small and acceptable

0-7803-2123-5/94 $4.00 © 1994 IEEE

bias, we also provide some basic analyses for the performance of the Split RLS. The analyses together with the simulation results indicate that the Split RLS works well when applied to broad-band/less-correlated signals. Based on this observation, we also propose the *orthogonal preprocessing* scheme to improve the performance of the Split RLS.

THE PROJECTION METHOD

Given an observation data matrix $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_n] \in \mathcal{R}^{m \times n}$ without any exhibited structure and the desired signal vector $\mathbf{y} \in \mathcal{R}^{m \times 1}$, the LS problem is to find the optimal weight coefficients $\hat{\mathbf{w}}$ which minimize the LS errors

$$\|\mathbf{e}\|^2 = \|\mathbf{A}\mathbf{w} - \mathbf{y}\|^2. \tag{1}$$

In general, $\hat{\mathbf{w}}$ is of the form

$$\hat{\mathbf{w}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{y}. \tag{2}$$

We also have $\hat{\mathbf{y}} = \mathbf{A}\hat{\mathbf{w}} = \mathbf{P}\mathbf{y}$ and $\hat{\mathbf{e}} = \mathbf{y} - \hat{\mathbf{y}}$, where $\hat{\mathbf{y}}$ is the optimal projection of $\mathbf{y}$ on the column space of $\mathbf{A}$, $\mathbf{P} = \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$ is the projection matrix, and $\hat{\mathbf{e}}$ is the optimal residual vector. The *principle of orthogonality* ensures that $\hat{\mathbf{e}}$ is orthogonal to the column space of $\mathbf{A}$. For RLS algorithms that calculate exact LS solution, such a direct projection to the N-dimensional space takes $O(N^2)$ complexity. Knowing this, in order to reduce the complexity, we shall try to perform projection onto spaces of smaller dimension.

To motivate the idea, let us consider the LS problem with the partition $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2]$, where $\mathbf{A}_1, \mathbf{A}_2 \in \mathcal{R}^{n \times (m/2)}$. Now instead of projecting $\mathbf{y}$ directly onto the space spanned by $\mathbf{A}$ (denoted as *span*$\{\mathbf{A}\}$), we project $\mathbf{y}$ onto the two smaller subspaces, *span*$\{\mathbf{A}_1\}$ and *span*$\{\mathbf{A}_2\}$, and obtain the optimal projections $\tilde{\mathbf{y}}_1$ and $\tilde{\mathbf{y}}_2$ on each subspace (see Fig.1). The next step is to find a "good" estimation of the optimal projection $\hat{\mathbf{y}}$, say $\hat{\mathbf{y}}_{approx}$. If we can estimate a 1-D or 2-D subspace from $\tilde{\mathbf{y}}_1$ and $\tilde{\mathbf{y}}_2$ and project the desired signal $\mathbf{y}$ directly on it to obtain $\hat{\mathbf{y}}_{approx}$, the projection spaces become smaller and the computational complexity is reduced as well. In the following, we propose two estimation methods based on their geometric relationship in the Hilbert space.

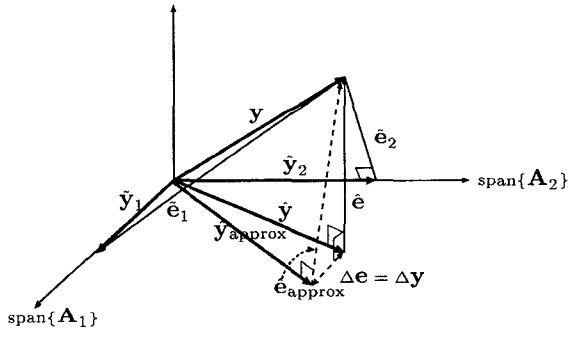

Figure 1: Geometric interpretation of the projection method.

Estimation Method I (Split RLS I)

The first approach is simply to add the two subspace projections $\tilde{\mathbf{y}}_1$ and $\tilde{\mathbf{y}}_2$ together, *i.e.*,

$$\hat{\mathbf{y}}_{approx} = \tilde{\mathbf{y}}_1 + \tilde{\mathbf{y}}_2. \tag{3}$$

This provides the most intuitive and simplest way to estimate $\hat{\mathbf{y}}_{approx}$. Let Fig.2(a) represent one of the existing RLS algorithms that project $\mathbf{y}$ onto the N-dimensional space of $\mathbf{A}$ and compute the optimal projection $\hat{\mathbf{e}}$ (or $\hat{\mathbf{y}}$, depending on the requirements) for the current iteration. The complexity is $O(N^2)$ per time iteration for the data matrix of size N. Now using Fig.2(a) as a basic building block, we can construct the block diagram for estimation method I as shown in Fig.2(b). Because the whole projection space is first split into two equal but smaller subspaces to perform the RLS estimation, we shall call this approach the *Split-RLS* (SP-RLS). It can be easily shown that the complexity is reduced by nearly half through such a decomposition.

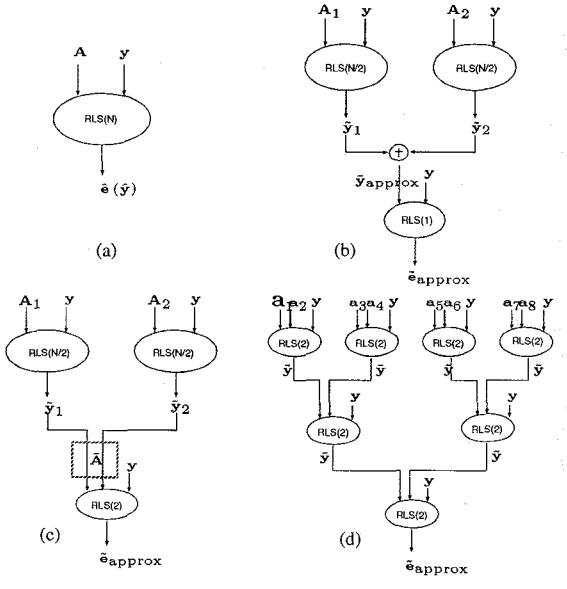

Figure 2: Block diagram for (a) a N-input RLS algorithm, (b) the SP-RLS I algorithm, (c) the SP-RLS II algorithm, (d) the TSP-RLS II algorithm.

Estimation Method II (Split RLS II)

In estimation method I, we try to project $\mathbf{y}$ onto the estimated optimal projection vector $\hat{\mathbf{y}}_{approx}$. In this approach, we will project $\mathbf{y}$ directly onto the 2-D subspace $\tilde{\mathbf{A}} \triangleq span\{\tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2\}$. As a result, the estimation shall be more accurate with slightly increase in complexity.

As with estimation method I, we can construct the block diagram for estimation method II (see Fig.2(c)) which is similar to Fig.2(b) except for the post-processing

part. The projection residual on $span\{\tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2\}$ is computed through a 2-input RLS block with $\tilde{\mathbf{y}}_1$ and $\tilde{\mathbf{y}}_2$ as the inputs.

Tree-Split RLS based on Estimation Method I and II

In estimation method I and II, we try to reduce the complexity by making one approximation at the last stage. Now consider the block diagram in Fig.2(c). If we repeatedly expand the two building blocks on the top by applying the same decomposition and approximation, we will obtain the block diagram in Fig.2(d). We shall call this new algorithm the *Tree-Split RLS algorithm* (TSP-RLS) due to its resemblance to a binary tree. Likewise, we can derive the TSP-RLS algorithm from estimation method I (TSP-RLS I) by using the block diagram in Fig.2(b).

Systolic Implementation

Now we will consider the systolic implementation of the above algorithms. First of all, we should note that each RLS building block in Fig.2 is independent of choices of RLS algorithms. Because the QRD-RLS array in [6] can compute the RLS estimation in a fully-pipelined way, it is a good candidate for our purpose. However, the original array computes only the optimal residual. In order to obtain the two optimal subspace projections $\tilde{\mathbf{y}}_1$ and $\tilde{\mathbf{y}}_2$, we need to modify the QRD-RLS array by keeping the delayed version of $y(n)$ (the desired signal at time n) in the rightmost column of the array. Once the residual is computed, we can use $\tilde{y}_1(n) = y(n) - \tilde{e}_1(n)$ and $\tilde{y}_2(n) = y(n) - \tilde{e}_2(n)$ to obtain the two subspace projections.

Now based on the block diagram in Fig.2, we can implement the Split RLS algorithms in the following way: For those RLS blocks which need to compute the optimal projection, the modified array is used for their implementations, while for those RLS blocks which need to compute the optimal residual (usually in the last stage), the QRD-RLS array in [6] is used. As an example, the resulting systolic implementations of the SP-RLS II and the TSP-RLS II are depicted in Fig.3. A comparison of hardware cost for the full-size QRD-RLS in [6] (denoted as FULL-RLS), SP-RLS, TSP-RLS, and QRD-LSL [1, chap.18], is listed in Table 1. As we can see, the complexity of the TSP-RLS is comparable with the QRD-LSL which requires shift data structure.

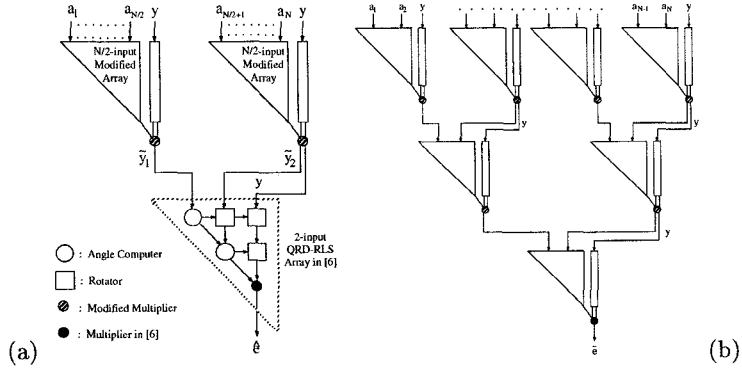

Figure 3: Systolic implementations of (a) the SP-RLS II and (b) the TSP-RLS II.

	No. of Angle Computers	No. of Rotators	System latency
FULL-RLS	N	$N(N+1)/2$	$N+1$
SP-RLS I	$N+1$	$N^2/4+N/2+1$	$N/2+3$
SP-RLS II	$N+2$	$N^2/4+N/2+3$	$N/2+4$
TSP-RLS I	$2N-1$	$2N-1$	$2(\log_2 N+1)$
TSP-RLS II	$2(N-1)$	$3(N-1)$	$3\log_2 N$
QRD-LSL	$2N+1$	$3N+1$	$N+1$

Table 1: Comparison of hardware cost for the FULL-RLS, SP-RLS, TSP-RLS, and QRD-LSL, where the QRD-LSL requires shift data structure.

PERFORMANCE ANALYSIS AND SIMULATIONS

Estimation Error for SP-RLS I

Consider the LS problem in (1) and decompose the column space of $\mathbf{A}$ into two equal-dimensional subspaces, *i.e.*, $\mathbf{A}=[\mathbf{A}_1\ ,\mathbf{A}_2]$. Let $\hat{\mathbf{w}}^T=[\hat{\mathbf{w}}_1^T\ ,\hat{\mathbf{w}}_2^T]$, the optimal projection vector $\hat{\mathbf{y}}$ can be represented as

$$\hat{\mathbf{y}}=\mathbf{A}\hat{\mathbf{w}}=\hat{\mathbf{y}}_1+\hat{\mathbf{y}}_2 \tag{4}$$

where $\hat{\mathbf{y}}_1=\mathbf{A}_1\hat{\mathbf{w}}_1$ and $\hat{\mathbf{y}}_2=\mathbf{A}_2\hat{\mathbf{w}}_2$. From the *normal equations*

$$\mathbf{A}^T\mathbf{A}\hat{\mathbf{w}}=\mathbf{A}^T\mathbf{y}, \tag{5}$$

we have

$$\mathbf{A}_1^T\mathbf{A}_1\hat{\mathbf{w}}_1+\mathbf{A}_1^T\mathbf{A}_2\hat{\mathbf{w}}_2=\mathbf{A}_1^T\mathbf{y}, \tag{6}$$

$$\mathbf{A}_2^T\mathbf{A}_1\hat{\mathbf{w}}_1+\mathbf{A}_2^T\mathbf{A}_2\hat{\mathbf{w}}_2=\mathbf{A}_2^T\mathbf{y}. \tag{7}$$

Let $\tilde{\mathbf{w}}_i,\tilde{\mathbf{y}}_i$, $i=1,2$, be the optimal weight vectors and the optimal projection vectors when considering two subspaces *span*$\{\mathbf{A}_1\}$ and *span*$\{\mathbf{A}_2\}$ separately. From (4) and (5), we have

$$\tilde{\mathbf{w}}_i=(\mathbf{A}_i^T\mathbf{A}_i)^{-1}\mathbf{A}_i^T\mathbf{y},\quad \tilde{\mathbf{y}}_i=\mathbf{A}_i\tilde{\mathbf{w}}_i,\quad i=1,2. \tag{8}$$

Premultiplying $\mathbf{A}_1(\mathbf{A}_1^T\mathbf{A}_1)^{-1}$ on (6) and using the definitions of $\hat{\mathbf{y}}_1,\hat{\mathbf{y}}_2,\tilde{\mathbf{y}}_1,\tilde{\mathbf{y}}_2$, (6) can be simplified as

$$\hat{\mathbf{y}}_1+\mathbf{P}_1\hat{\mathbf{y}}_2=\tilde{\mathbf{y}}_1. \tag{9}$$

Similarly, from (7) we can obtain

$$\mathbf{P}_2\hat{\mathbf{y}}_1+\hat{\mathbf{y}}_2=\tilde{\mathbf{y}}_2 \tag{10}$$

where $\mathbf{P}_i=\mathbf{A}_i(\mathbf{A}_i^T\mathbf{A}_i)^{-1}\mathbf{A}_i^T$, $i=1,2$ are the projection operators.

In SP-RLS I, we estimate the optimal projection by

$$\hat{\mathbf{y}}_{approx}=\tilde{\mathbf{y}}_1+\tilde{\mathbf{y}}_2, \tag{11}$$

and the estimation error (bias) is given by

$$\|\Delta\mathbf{e}_1\|^2=\|\hat{\mathbf{e}}_{approx}-\hat{\mathbf{e}}\|^2=\|\hat{\mathbf{y}}-\hat{\mathbf{y}}_{approx}\|^2. \tag{12}$$

Substituting (9)-(11) into (12) yields

$$\|\Delta\mathbf{e}_1\|^2=\|\hat{\mathbf{y}}-\tilde{\mathbf{y}}_1-\tilde{\mathbf{y}}_2\|^2=\|\mathbf{P}_1\hat{\mathbf{y}}_2+\mathbf{P}_2\hat{\mathbf{y}}_1\|^2. \tag{13}$$

In order to lower the bias value, $\mathbf{P}_1\hat{\mathbf{y}}_2$ and $\mathbf{P}_2\hat{\mathbf{y}}_1$ should be as small as possible. Note that

$$\mathbf{P}_1\hat{\mathbf{y}}_2 = \mathbf{A}_1(\mathbf{A}_1^T\mathbf{A}_1)^{-1}\mathbf{A}_1^T\mathbf{A}_2\hat{\mathbf{w}}_2 = \mathbf{A}_1\tilde{\mathbf{\Phi}_{11}^{-1}}\mathbf{\Phi}_{12}\hat{\mathbf{w}}_2, \tag{14}$$

$$\mathbf{P}_2\hat{\mathbf{y}}_1 = \mathbf{A}_2(\mathbf{A}_2^T\mathbf{A}_2)^{-1}\mathbf{A}_2^T\mathbf{A}_1\hat{\mathbf{w}}_1 = \mathbf{A}_2\tilde{\mathbf{\Phi}_{22}^{-1}}\mathbf{\Phi}_{21}\hat{\mathbf{w}}_1 \tag{15}$$

where $\mathbf{\Phi}_{ij} = \mathbf{A}_i^T\mathbf{A}_j$ is the deterministic correlation matrix. When the column vectors of $\mathbf{A}_1$ and $\mathbf{A}_2$ are more orthogonal to each other, $\mathbf{\Phi}_{12}$ and $\mathbf{\Phi}_{21}$ will approach to zero and the bias is reduced accordingly.

Estimation Error for SP-RLS II

Consider the block diagram of the SP-RLS II in Fig.2(c). The optimal projection of $\mathbf{y}$ onto the space $span\{\tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2\}$ can be written as

$$\hat{\mathbf{y}}_{approx} = \hat{k}_1\tilde{\mathbf{y}}_1 + \hat{k}_2\tilde{\mathbf{y}}_2 \tag{16}$$

where $\hat{\mathbf{k}} = [\hat{k}_1, \hat{k}_2]^T$ is the optimal weight vector. From the *normal equations*, the optimal weight vector can be solved as

$$\hat{\mathbf{k}} = [\hat{k}_1,\ \hat{k}_2]^T = \left[\alpha\,\frac{\tilde{\mathbf{y}}_1^T\tilde{\mathbf{e}}_2}{\|\tilde{\mathbf{y}}_1\|^2},\ \alpha\,\frac{\tilde{\mathbf{y}}_2^T\tilde{\mathbf{e}}_1}{\|\tilde{\mathbf{y}}_2\|^2}\right]^T \tag{17}$$

where

$$\alpha = \left(1 - \frac{\tilde{\mathbf{y}}_1^T\tilde{\mathbf{y}}_2}{\|\tilde{\mathbf{y}}_1\|^2}\frac{\tilde{\mathbf{y}}_2^T\tilde{\mathbf{y}}_1}{\|\tilde{\mathbf{y}}_2\|^2}\right)^{-1} = \csc^2\theta, \tag{18}$$

and θ denotes the angle between $\tilde{\mathbf{y}}_1$ and $\tilde{\mathbf{y}}_2$. From Fig.1, we have

$$\|\hat{\mathbf{e}}_{approx}\|^2 = \|\mathbf{y}\|^2 - \|\mathbf{y}_{approx}\|^2 = \|\mathbf{y}\|^2 - \mathbf{y}^T\mathbf{y}_{approx} = \|\mathbf{y}\|^2 - \hat{k}_1\|\tilde{\mathbf{y}}_1\|^2 - \hat{k}_2\|\tilde{\mathbf{y}}_2\|^2. \tag{19}$$

Substituting (17) into (19) yields

$$\|\hat{\mathbf{e}}_{approx}\|^2 = \|\mathbf{y}\|^2 - \csc^2\theta\|\tilde{\mathbf{y}}_1 - \tilde{\mathbf{y}}_2\|^2. \tag{20}$$

Thus, the bias of SP-RLS II is given by

$$\|\Delta\mathbf{e}_2\|^2 = \|\hat{\mathbf{e}}_{approx}\|^2 - \|\hat{\mathbf{e}}\|^2 = \|\hat{\mathbf{y}}\|^2 - \csc^2\theta\|\tilde{\mathbf{y}}_1 - \tilde{\mathbf{y}}_2\|^2. \tag{21}$$

For any given θ, it can be shown that $\|\Delta\mathbf{e}_2\|^2$ is bounded by

$$\|\Delta\mathbf{e}_2\|^2 \leq \|\Delta\mathbf{e}_1\|^2. \tag{22}$$

This implies that the performance of SP-RLS II is better than that of SP-RLS I in terms of estimation error.

Bandwidth, Eigenvalue Spread, and Bias

From (13) and (21) we know that the orthogonality between the two subspaces $span\{\mathbf{A}_1\}$ and $span\{\mathbf{A}_2\}$ will significantly affect the bias value. However, in practice, the evaluation of degree of orthogonality for multidimensional spaces is nontrivial and computationally intensive (e.g., CS-decomposition [7, pp. 75–78]). Without loss of generality, we will only focus our discussion on single-channel case, where the data matrix $\mathbf{A}$ consists of only shifted data and the degree of orthogonality can be easily measured. In such a case, the degree of orthogonality can be measured

through two indices: the bandwidth and the eigenvalue spread of the data. If the signal is less correlated (orthogonal), the autocorrelation function has smaller duration and thus larger bandwidth. Noise processes are examples. On the other hand, narrow-band processes such as sinusoidal signals are highly correlated. If the data matrix is completely orthogonal, all the eigenvalues are the same and the condition number is one. This implies that if the data matrix is more orthogonal, it will have less eigenvalue spread. It is clear from our previous discussion that the SP-RLS will render less bias for the broad-band signals than for the narrow-band signals.

As to the TSP-RLS, note that the output optimal projection is a linear combination of the input column vectors. If the inputs to one stage of the TSP-RLS array are less correlated, the outputs of this stage will still be less correlated. Therefore, the signal property at the first stage such as bandwidth plays an important role in the overall performance of the TSP-RLS.

Simulation Results

In the following simulations, we will use the autoregressive (AR) process of order p (AR(p)) to generate the simulation data. Besides, the pole locations of the AR processes are used to control the bandwidth property. In the first experiment, we try to perform fourth-order linear prediction (LP) with four AR(4) processes using the SP-RLS and TSP-RLS systolic arrays. The simulation results are shown in Fig.4, in which the x-axis represents the location of the variable poles, and y-axis represents the average output noise power after convergence. Ideally the output should be the noise process with power equal to 0.1. As we can see, when the bandwidth of input signal becomes wider, the bias is reduced. This agrees perfectly with what we expected.

Beside the bias values, we also plot the square root of the *spectral dynamic range* D associated with each AR process. It is known that the eigenvalue spread of the data signal is bounded by the spectral dynamic range [8]

$$1 \leq \frac{\lambda_{\max}}{\lambda_{\min}} \leq \frac{\max\{|U(e^{j\omega})|^2\}}{\min\{|U(e^{j\omega})|^2\}} \triangleq D, \tag{23}$$

where $U(e^{j\omega})$ is the spectrum of the signal. From the simulation results, we see the consistency between the bias value and the spectral dynamic range. This indicates that the performance of the Split RLS algorithms is also affected by the eigenvalue spread of the input signal. This phenomenon is similar to what we have seen in the LMS-type algorithms. Besides, two observations can be made from the experimental results: 1) The SP-RLS performs better than the TSP-RLS. This is pretty much due to the number of approximation stages in each algorithm. 2) The overall performance of SP-RLS II is better than that of SP-RLS I. This agrees with our analysis in (22).

Next we want to examine the convergence rate of our algorithm. Fig.5 shows the convergence curve for the 8-input FULL-RLS and the TSP-RLS II after some initial perturbation. It is interesting to note that although the TSP-RLS II has some bias after it converges, its convergence rate is faster than that of the FULL-RLS. This is due to the fact that the $O(\log_2 N)$ system latency of the TSP-RLS is less than the $O(N)$ latency of the FULL-RLS. Also, to initialize an 8-input full-size array takes more time than to initialize the three small cascaded 2-input arrays. The property of faster convergence rate is especially preferred for the tracking of parameters in non-stationary environments such as multichannel adaptive filtering [9].

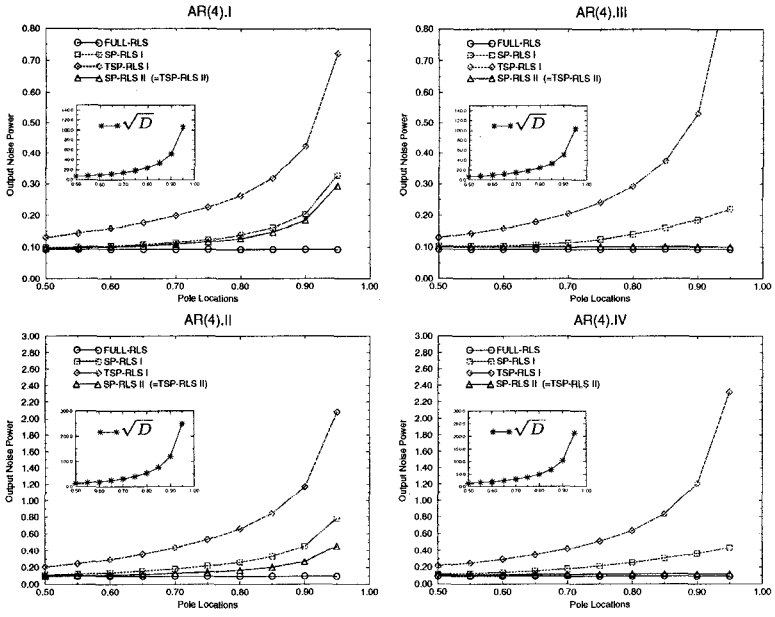

Figure 4: Simulation results of AR(4) I-IV, where the square root of the spectral dynamic range (D) is also plotted for comparison.

SPLIT RLS WITH ORTHOGONAL PREPROCESSING

From the analyses in the previous section, we know that the estimated optimal projection will approach to the real optimal projection when all subspaces are more orthogonal to each other. Therefore, if we can preprocess the data matrix such that the column spaces become more orthogonal (less correlated) to each other, a better performance is expected. The operation for the Split RLS with orthogonal preprocessing is as follows: First perform the orthogonal transform on the current data vector, then use the transformed data as the inputs of the Split RLS. In our approach, the Discrete Cosine Transform (DCT) and the Discrete Hartley Transform (DHT) are used as the preprocessing kernels. As to the hardware implementation, we can employ the time-recursive DCT/DHT lattice structure in [10] to continuously

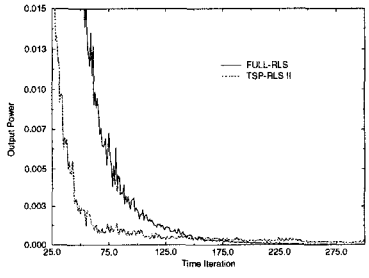

Figure 5: Learning curve of the FULL-RLS and TSP-RLS II after some initial perturbation.

generate the transformed data.

In addition to the two aforementioned transforms, we also propose a new preprocessing scheme called the *Swapped DCT* (SWAP-DCT). Suppose $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N]$ is the DCT-domain data. In the DCT preprocessing, the input data is partitioned as

$$\begin{aligned} \mathbf{A}_1 &= [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{N/2}], \\ \mathbf{A}_2 &= [\mathbf{z}_{N/2+1}, \mathbf{z}_{N/2+2}, \ldots, \mathbf{z}_N]. \end{aligned} \tag{24}$$

To make the input data more uncorrelated, we permute the transformed data column as

$$\begin{aligned} \mathbf{A}_1 &= [\mathbf{z}_1, \mathbf{z}_3, \ldots, \mathbf{z}_{2k-1}, \ldots, \mathbf{z}_{N-1}], \\ \mathbf{A}_2 &= [\mathbf{z}_2, \mathbf{z}_4, \ldots, \mathbf{z}_{2k}, \ldots, \mathbf{z}_N] \end{aligned} \tag{25}$$

in the SWAP-DCT preprocessing scheme. Fig.6 shows the spectrum of the normal DCT partitioning and the SWAP-DCT partitioning. Recall that the eigenvalue spread will affect the bias value, and the eigenvalue spread is bounded by the spectral dynamic range. It is obvious that the SWAP-DCT preprocessing scheme will have better performance due to the smaller eigenvalue spread in both $\mathbf{A}_1$ and $\mathbf{A}_2$.

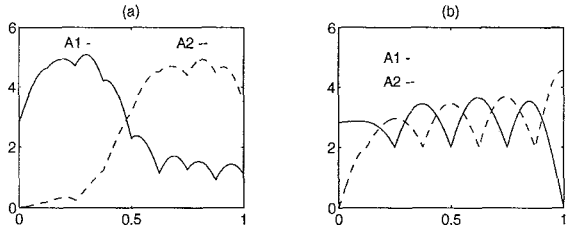

Figure 6: Spectrum of (a) the Normal DCT domain and (b) the SWAP-DCT domain.

To validate our arguments for the orthogonal preprocessing, we will repeat the first experiment in the previous section for the TSP-RLS II with different preprocessing schemes (Fig.7). In general, the TSP-RLS with DCT preprocessing gives a fairly significant improvement in the bias value over the TSP-RLS without any preprocessing (Normal TSP-RLS). Nevertheless, some exceptions can be found in AR(4).III. As to the DHT, it does not perform well in most cases. It is as expected that the SWAP-DCT performs better than the DCT. This supports our assertion for the effect of the SWAP-DCT.

CONCLUSION

In this paper, we introduced a new $O(N)$ fast algorithm and architecture for the RLS estimation of nonstructured data. We have shown that the bandwidth and/or the eigenvalue spread of the input signal can be used as a good performance index for these algorithms. Therefore, the users will have small bias when dealing with broad-band/less-correlated signals. For narrow-band signals, we can also employ the orthogonal preprocessing to improve its performance. The low complexity as well as the fast convergence rate of the proposed algorithm makes it suitable for RLS estimation under the non-stationary or fast-changing environments where the data matrix has no structure.

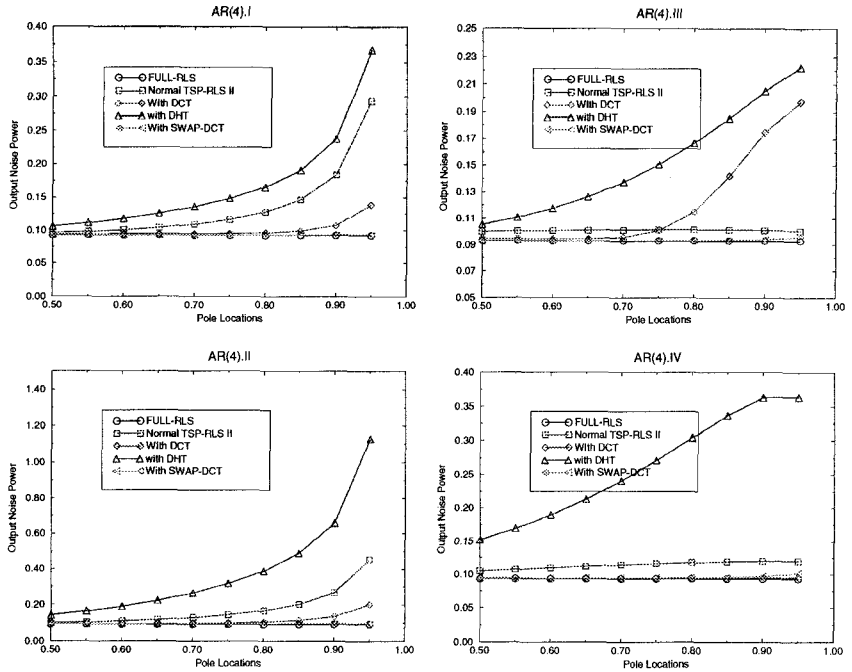

Figure 7: Simulation result of AR(4) I-IV with preprocessing schemes.

References

[1] S. Haykin, *Adaptive Filter Theory.* Prentice-Hall, Englewood Cliffs, N.J., 2nd ed., 1991.

[2] K. Tanabe, "Projection method for solving a singular system of linear equations and its applications," *Numer. Math.*, vol. 17, pp. 203–214, 1971.

[3] A. S. Kydes and R. P. Tewarson, "An iterative methods for solving partitioned linear equations," *Computing*, vol. 15, pp. 357–363, Jan. 1975.

[4] T. Elfving, "Block-iterative methods for consistent and inconsistent linear equations," *Numer. Math.*, vol. 35, pp. 1–12, 1980.

[5] R. Bramley and A. Samem, "Row projection methods for large nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 13, pp. 168–193, Jan. 1992.

[6] J. G. McWhirter, "Recursive least-squares minimization using a systolic array.," *Proc. SPIE, Real-Time Signal Processing VI*, vol. 431, pp. 105–112, 1983.

[7] G. H. Golub and C. F. Van Loan, *Matrix Computations.* The John Hopkins University Press, Baltimore, MD, 2nd ed., 1989.

[8] J. Makhoul, "Linear Prediction: A tutorial review," *Proc. IEEE*, vol. 63, pp. 561–580, April 1975.

[9] K. J. R. Liu and A.-Y. Wu, "A multi-layer 2-D adaptive filtering architecture based on McClellan transformation," in *Proc. IEEE Int. Symp. Circuits and Systems*, (Chicago), pp. 1999–2002, 1993.

[10] K. J. R. Liu and C. T. Chiu, "Unified parallel lattice structures for time-recursive Discrete Cosine/Sine/Hartley transforms," *IEEE Trans. Signal Processing*, vol. 41, pp. 1357–1377, March 1993.