Proceedings of the American Control Conference Seattle, Washington • June 1995

THE INVERSE QR DECOMPOSITION IN ORDER RECURSIVE CALCULATION OF LEAST SQUARES COEFFICIENTS

Daniel W. Apley and Jianjun Shi
Dept. of Mechanical Engineering and Applied Mechanics
The University of Michigan
Ann Arbor, MI. 48109-2125
phone: 313-763-7687 fax: 313-936-0363
e-mail: shihang@engin.umich.edu

ABSTRACT

In this paper we examine the properties of QR and inverse QR factorizations in the general linear least squares (LS) problem. By exploiting a straightforward geometric interpretation of the factorization, an efficient algorithm is derived that provides, order recursively, the LS coefficient vector, projection error vector, and residual error energy (i.e. the sum of the squares of the elements of the error vector) for all of the LS problems as the order varies from one to n, n being a prespecified maximum order. Using existing algorithms for time updating the inverse QR factorization, the method applies to the time recursive situation also. Given only R^{-1} and the last row of Q in the inverse QR factorization of the data covariance matrix, all order updates of the LS coefficient vectors and residual error energies are carried out. Application to multichannel adaptive LS filtering is presented.

I. INTRODUCTION

In this paper we consider the general linear LS problem of projecting an N-length "output" vector onto the span of a set of N-length "input" vectors. A solution to the LS problem using a QR factorization of the input data matrix has been known for decades [1]–[3] to provide a numerically stable solution. Moreover, the QR factorization is parallel in the sense that it can be implemented on a two-dimensional systolic array, and easily extends to the time-recursive case [1], [2]. By time-recursive, it is meant that at "time" N+1 an additional element is appended to the input and output vectors, and the resulting QR factorization and LS solution is updated. A time recursive solution is especially important in signal processing applications like adaptive LS filtering and recursive system identification. Also becoming popular in the time-recursive case is the inverse QR factorization method, in which R^{-1}, rather than R is updated, because it avoids the backsubstitution step required in the QR factorization for calculating the LS coefficient vector [4]–[6].

Another desirable property of the QR factorization method is its inherent order recursiveness. To illustrate, consider the situation in which the input data is a time series, possibly multichannel. In this case there is a strong analogy, discussed thoroughly in [7] and [17], between the QR LS implementation and the LS lattice filter implementation. The elements of Q are exactly the normalized backward prediction errors of the LS lattice filter, and the reflection coefficients of the lattice filter are very closely related to the rotation angles in a Givens rotation based QR implementation. Consequently, the QR method inherits all the well known order recursive properties of the LS lattice method. However, in many situations, for example system identification, it is the transversal filter coefficients that are desired, order recursively if the appropriate filter order is unknown *a priori*. Calculating the transversal filter coefficients from the lattice filter coefficients is straightforward in the single channel case, but becomes complicated and inefficient in more general sitiations.

By investigating the geometric interpretation of the QR and inverse QR factorizations in LS and exploiting this interpretation, a very straightforward and computationally efficient method of obtaining order recursions is developed in this paper. The results are derived for the most general linear LS problem described in the first paragraph. The order recursiveness is in the sense that the LS coefficient vectors, the error vectors, and the residual error energies are available in the full order LS projection, as well as the lower order LS projections. By "full order" it is meant that the output vector is projected onto the span of all the input vectors, whereas in the "lower order" problems the output vector is projected onto the span of fewer and fewer of the input vectors. In the specific case that the input data is a time series, the solution to the full order and lower order LS problems represent order recursive calculation of the filter coefficients, prediction errors, and residual error energies for the transversal filters described in the preceeding paragraph.

Much of the previous work on order recursive estimation of the transversal filter coefficients has been developed as batch methods [8] – [10], and is not directly applicable to the time recursive case. In [11] - [13], algorithms that are both time and order recursive were developed. [11] applies to scalar AR filtering, and in [12] the method was extended to multi-input multi-output FIR filtering. In both [11] and [12] the time recursiveness is achieved using either a growing memory rectangular window or sliding rectangular window. In [13] the concepts in [12] were applied to scalar FIR filtering and simplified somewhat by using a pre-windowed data assumption.

The algorithm of this paper possesses many advantages over the algorithms of [8]-[13]. Firstly,it can be applied to the most general linear LS problem. The generality of the method and straightforward geometric interpretation allow it to be easily applied to a wide variety of LS applications. Of [8]-[13], only [10] can be applied order recursively in situations where the input data is not a time series. If implemented in a time recursive situation, a sliding rectangular window, a growing length rectangular window, or an exponentially decaying window can be used. None of the algorithms of [8]-[13] can accomodate the use of exponential windows. Secondly, this algorithm is based on the QR factorization, which is widely known to possess excellent numerical properties. Thirdly, the algorithm is parallel in the sense that it can be implemented on a two-dimensional systolic array for increased computational speed. It should be noted that the algorithms of [9] and [10] also possess this feature. Fourthly, this algorithm has considerably better computational efficiency than the previously developed time and order recursive algorithms. A comparison of the computational expenses for adaptive multichannel LS filtering is provided later in [16].

The format of the remainder of the paper is as follows. In section II. the QR and inverse QR factorization in the general linear LS problem is reviewed and the geometric interpretation of Q and R^{-1} investigated. In section III an efficient method for obtaining the desired order recursions is derived, and in section IV application of the results to multichannel adaptive LS filtering is presented. Throughout the paper, matrices will be represented in upper case bold type and vectors in lower case bold.

II. THE QR AND INVERSE QR FACTORIZATION IN LEAST SQUARES

Consider the following general linear least squares (LS) problem. Let $\{\mathbf{x}_i(N)\}_{i=1}^{n} \mathbf{y}(N)$ be N-dimensional column vectors, $\mathbf{x}_i(N) := [x_i(1), x_i(2), \ldots, x_i(N)]^T$, $\mathbf{y}(N) := [y(1), y(2), \ldots, y(N)]^T$, and let $\mathbf{X}^i(N) = [(\mathbf{x}_1(N), \mathbf{x}_2(N), \ldots, \mathbf{x}_i(N))$ $(i = 1, 2, \ldots, n)$. Here, the superscript T indicates transpose. Assume $\mathbf{X}^n(N)$ has rank n. The LS problem is to find the n-dimensional column vector $\mathbf{w}$ that minimizes

$$\|\mathbf{y}(N) - \mathbf{X}^n(N)\mathbf{w}\|_2, \tag{2.1}$$

where $\|\bullet\|$ is the standard Euclidean norm.

For the purpose of minimizing (2.1), consider the QR factorization of $\mathbf{X}^n(N)$, i.e.

$$\mathbf{X}^n(N) = \mathbf{Q}(N)\mathbf{R}(N), \tag{2.2}$$

where $\mathbf{Q}(N)$ is an N×n orthogonal matrix, and $\mathbf{R}(N)$ is an n×n upper triangular matrix with positive diagonal elements. A proof that this factorization is unique can be found in [3]. It can be easily shown [14] that the solution for the $\mathbf{w}$ that minimizes (2.1) can be obtained by using back substitution in the triangular system of equations

$$\mathbf{R}(N)\mathbf{w} = \mathbf{y}_q(N), \text{ where } \mathbf{y}_q(N) := \mathbf{Q}^T(N)\mathbf{y}(N). \tag{2.3}$$

The QR factorization itself can be achieved using modified Gramm-Schmidt (MGS), Householder transform, or Givens rotation techniques, providing a numerically robust solution to the LS problem [3]. The latter two techniques require minor alterations in (2.2) so that $\mathbf{Q}(N)$ is N×N and $\mathbf{R}(N)$ is N×n with the first n rows an upper triangular matrix and the remaining N-n rows all zero. In (2.3), then, $\mathbf{R}(N)$ would be replaced by its first n rows, and $\mathbf{Q}(N)$ by its first n columns. Another desirable property of the QR method is that it can be easilily extended to the time recursive case, if we interpret N as a time index. That is, given $\mathbf{X}^n(N) = \mathbf{Q}(N)\mathbf{R}(N)$, find the QR factorization of $\mathbf{X}^n(N+1)$ with

$$\mathbf{X}^n(N+1) := \begin{bmatrix} \lambda\mathbf{X}^n(N) \\ x_1(N+1), x_2(N+1), \ldots, x_n(N+1) \end{bmatrix} = \mathbf{Q}(N+1)\mathbf{R}(N+1),$$

$$\mathbf{y}(N+1) := \left[\lambda\mathbf{y}^T(N), y(N+1)\right]^T, \tag{2.4}$$

and $\lambda \in (0,1]$ a forgetting factor.

Efficient, numerically stable algorithms for implementing QR based time-recursive least squares (RLS) have been developed in [2] using Givens rotations, and in [15] using the recursive modified Gramm-Schmidt (RMGS) technique. Both methods require $O(n^2)$ (computational expense proportional to n^2 for large n) operations per time update and have a structure that allows them to be implemented on a two dimensional systolic array. In actuality, it was shown that the two algorithms are algebraically equivalent, any implementation discrepancies the result of numerical roundoff errors [7].

In order to further investigate the properties of $\mathbf{R}(N)$ and $\mathbf{Q}(N)$, we make the following definitions. Let

$$r_{i,j}(N) := [\mathbf{R}(N)]_{i,j}, \ 1 \le i \le j \le n \tag{2.5a}$$

$$\mathbf{J}(N) := \mathrm{diag}\Big(\{r_{i,i}(N)\}_{i=1}^{n}\Big), \tag{2.5b}$$

$$\mathbf{E}(N) := [\mathbf{e}_1(N), \mathbf{e}_2(N), \ldots, \mathbf{e}_n(N)] := \mathbf{Q}(N)\mathbf{J}(N), \tag{2.5c}$$

$$\mathbf{K}(N) := \mathbf{J}^{-1}(N)\mathbf{R}(N), \text{ and} \tag{2.5d}$$

$$\mathbf{W}(N) := \mathbf{K}^{-1}(N). \tag{2.5e}$$

It follows easily that

$$\mathbf{X}^n(N) = \mathbf{Q}(N)\mathbf{R}(N) = \mathbf{E}(N)\mathbf{K}(N), \text{ and} \tag{2.6a}$$

$$\mathbf{E}^T(N)\mathbf{E}(N) = \mathbf{J}^2(N) \tag{2.6b}$$

$\mathbf{K}(N)$ and $\mathbf{W}(N)$ are both upper triangular matrices with ones on the diagonal and the columns of $\mathbf{E}(N)$ are orthogonal by (2.6b). Thus, the $\mathbf{E}(N)\mathbf{K}(N)$ factorization is identical to the $\mathbf{Q}(N)\mathbf{R}(N)$ factorization, except that the columns of $\mathbf{E}(N)$ are scaled so that $\mathbf{K}(N)$ has ones on the diagonal. Writing $\mathbf{W}(N)$ as

$$\mathbf{W}(N) := \begin{bmatrix} 1 & \mathbf{w}_2(N) & \mathbf{w}_3(N) & \cdots & \mathbf{w}_n(N) \\ & 1 & & & \\ & & 1 & & \\ & 0 & & \ddots & \\ & & & & 1 \end{bmatrix}, \tag{2.7}$$

where $\mathbf{w}_i(N)$ is a column vector of length i-1, we state the following Lemma.

Lemma 2.1:

$\mathbf{w}_i(N)$ is the negative of the LS coefficient vector in projecting $\mathbf{x}_i(N)$ onto the column space of $\mathbf{X}^{i-1}(N)$, i.e.

$$\mathbf{w}_i(N) = -\operatorname*{argmin}_{\mathbf{w}} \|\mathbf{x}_i(N) - \mathbf{X}^{i-1}(N)\mathbf{w}\|_2 \tag{2.8}$$

Moreover, $\mathbf{e}_i(N) = \mathbf{x}_i(N) + \mathbf{X}^{i-1}(N)\mathbf{w}_i(N)$, and is the error vector associated with the LS projection, and the i^{th} diagonal element of $\mathbf{R}(N)$ is the square root of the residual error energy, i.e. $r_{i,i}(N) = \sqrt{\mathbf{e}_i^T(N)\mathbf{e}_i(N)}$. The proof of lemma 2.1 can be found in [16].

Define $\mathbf{A}(N) := [\mathbf{X}^n(N), \mathbf{y}(N)]$ and suppose that $\mathbf{Q}(N)$ and $\mathbf{R}^{-1}(N)$ (or, equivalently, $\mathbf{E}(N)$ and $\mathbf{W}(N)$) from the QR factorization of $\mathbf{A}(N)$ are available. This will be referred to as the inverse QR factorization. From lemma 2.1 the normalized error vectors for the various order LS problems are available in $\mathbf{Q}(N)$. Moreover, no back substitution is necessary to obtain the LS coefficient vector in the projection of $\mathbf{y}(N)$ onto the column space of $\mathbf{X}^n(N)$, since, according to Lemma 2.1, they are contained in the last column of $\mathbf{W}(N)$. Methods similar to those for time-recursively updating $\mathbf{Q}(N)$ and $\mathbf{R}(N)$ have been developed for time-recursively updating $\mathbf{Q}(N)$, $\mathbf{R}^{-1}(N)$, and $\mathbf{W}(N)$ in [4]–[6]. [4] assumes either a sliding or growing memory rectangular on the data, while [5] and [6] assume exponentially weighted data. All methods are $O(n^2)$ and require roughly the same computational expense as the methods for updating $\mathbf{Q}(N)$ and $\mathbf{R}(N)$ in [7] and [2]. In addition, they involve the same Givens rotation or MGS concepts and, therefore, enjoy the same numerical stability as the methods for updating $\mathbf{Q}(N)$ and $\mathbf{R}(N)$. Finally, all have parallel capabilities in the sense that they can be implemented on a two dimensional systolic array architecture.

III. EFFICIENT ORDER RECURSIONS IN THE INVERSE QR FACTORIZATION

Consider again the inverse QR factorization of $\mathbf{A}(N) := [\mathbf{X}^n(N), \mathbf{y}(N)]$. The notation in this section is the same as that in section II, except that all quantities are, unless otherwise noted, with respect to the factorization of $\mathbf{A}(N)$ and not $\mathbf{X}^n(N)$.

According to Lemma 2.1 and the subsequent discussion, the order recursiveness is in the sense that the error vector, LS coefficient vector, and residual error energy in projecting $\mathbf{x}_i(N)$ onto the column space of $\mathbf{X}^{i-1}(N)$ $(i = 1, 2, \ldots, n)$, and in projecting $\mathbf{y}(N)$ onto the column space of $\mathbf{X}^n(N)$, are readily available from $\mathbf{E}(N)$ and $\mathbf{R}^{-1}(N)$. Here, $\mathbf{X}^0(N)$ will be defined as the zero vector, so that the error vector in projecting $\mathbf{y}(N)$ onto the span of $\mathbf{X}^0(N)$ will be $\mathbf{y}(N)$ itself. In many applications, for example system identification and spectral estimation, this is not the order recursiveness that is desired. Rather, what are desired are the error vectors, LS coefficient vectors, and residual error energies in projecting $\mathbf{y}(N)$ onto the column space of $\mathbf{X}^i(N)$ $(i = 0, 1, \ldots, n)$. In this section, a computationally efficient, conceptually straightforward method for accomplishing this is presented. The method utilizes the inverse QR factorization of $\mathbf{A}(N)$ after its columns have been rearranged in the manner described in the following paragraphs.

Define, for $1 \le i \le n$, $\mathbf{A}^i(N) := \mathbf{A}^{i+1}(N)\mathbf{P}^{i+1}$ with

$$\mathbf{A}^{n+1}(N) := \mathbf{A}(N) := [\mathbf{X}^n(N), \mathbf{y}(N)], \tag{3.1}$$

where $\mathbf{P}^{i+1}$ is the permutation matrix exchanging the i^{th} and $(i+1)^{st}$ columns of the matrix it operates on, i.e.

$$\mathbf{P}^{i+1} := \begin{bmatrix} \mathbf{I}_{i-1} & & \mathbf{0} \\ & \begin{matrix} 0 & 1 \\ 1 & 0 \end{matrix} & \\ \mathbf{0} & & \mathbf{I}_{n-i} \end{bmatrix}. \tag{3.2}$$

$\mathbf{I}_j$ denotes the identity matrix of dimension j. Then,

$$\mathbf{A}^i(N) = [\mathbf{X}^{i-1}(N), \mathbf{y}(N), \mathbf{x}_i(N), \mathbf{x}_{i+1}(N), \ldots, \mathbf{x}_n(N)].$$

Let the following variables be defined as in section II, except that the superscript i indicates it is with respect to the (inverse) QR factorization of $\mathbf{A}^i(N)$ $(1 \le i \le n+1)$.

$$\mathbf{A}^i(N) = \mathbf{Q}^i(N)\mathbf{R}^i(N). \tag{3.3}$$

$$r^i_{k,j}(N) := [\mathbf{R}^i(N)]_{k,j},$$

$$\underline{r}^i_{k,j}(N) := [(\mathbf{R}^i(N))^{-1}]_{k,j}, \quad 1 \le k \le j \le n+1 \tag{3.4a}$$

$$\mathbf{J}^i(N) := \operatorname{diag}\left(\{r^i_{j,j}(N)\}_{j=1}^{n+1}\right), \tag{3.4b}$$

$$\mathbf{E}^i(N) := [\mathbf{e}^i_1(N), \mathbf{e}^i_2(N), \ldots, \mathbf{e}^i_{n+1}(N)] := \mathbf{Q}^i(N)\mathbf{J}^i(N), \tag{3.4c}$$

$$\mathbf{K}^i(N) := (\mathbf{J}^i(N))^{-1}\mathbf{R}^i(N), \text{ and} \tag{3.4d}$$

$$\mathbf{W}^i(N) := \begin{bmatrix} 1 & \mathbf{w}^i_2(N) & \mathbf{w}^i_3(N) & \cdots & \mathbf{w}^i_{n+1}(N) \\ & 1 & & & \\ & & 1 & & \\ & \mathbf{0} & & \ddots & \\ & & & & 1 \end{bmatrix} := (\mathbf{K}^i(N))^{-1}, \tag{3.4e}$$

where $\mathbf{w}^i_j(N)$ is a column vector of length j-1. Then, it follows that

$$\mathbf{A}^i(N) = \mathbf{Q}^i(N)\mathbf{R}^i(N) = \mathbf{E}^i(N)\mathbf{K}^i(N), \text{ and} \tag{3.5}$$

$$(\mathbf{E}^i(N))^T\mathbf{E}^i(N) = (\mathbf{J}^i(N))^2. \tag{3.6}$$

Clearly, by Lemma 2.1, the desired order recursions can be obtained from the inverse QR factorization of $\mathbf{A}^i(N)$ $(1 \le i \le n+1)$. Specifically, the error vector, LS coefficient vector, and residual error energy in the projection of $\mathbf{y}(N)$ onto the column space of $\mathbf{X}^{i-1}(N)$ are $\mathbf{e}^i_i(N)$, $-\mathbf{w}^i_i(N)$, and $\frac{1}{(\underline{r}^i_{i,i}(N))^2}$, respectively, in the sense that

$$\mathbf{w}^i_i(N) = -\operatorname*{argmin}_{\mathbf{w}} \|\mathbf{y}(N) - \mathbf{X}^{i-1}(N)\mathbf{w}\|_2, \tag{3.7}$$

$$\mathbf{e}^i_i(N) = \mathbf{y}(N) + \mathbf{X}^{i-1}(N)\mathbf{w}^i_i(N), \text{ and} \tag{3.8}$$

$$(\mathbf{e}^i_i(N))^T\mathbf{e}^i_i(N) = \frac{1}{(\underline{r}^i_{i,i}(N))^2}. \tag{3.9}$$

The following proposition provides an efficient method for determining $\mathbf{Q}^i(N)$ and $(\mathbf{R}^i(N))^{-1}$, given $\mathbf{Q}^{i+1}(N)$ and $(\mathbf{R}^{i+1}(N))^{-1}$.

Proposition 3.1: $\mathbf{Q}^i(N) = \mathbf{Q}^{i+1}(N)\tilde{\mathbf{Q}}^i(N)$ and

$$(\mathbf{R}^i(N))^{-1} = \mathbf{P}^{i+1}(\mathbf{R}^{i+1}(N))^{-1}\tilde{\mathbf{Q}}^i(N), \text{ where} \tag{3.10}$$

$$\tilde{\mathbf{Q}}^i(N) := \begin{bmatrix} \mathbf{I}_{i-1} & & \mathbf{0} \\ & \begin{matrix} c^i & s^i \\ s^i & -c^i \end{matrix} & \\ \mathbf{0} & & \mathbf{I}_{n-i} \end{bmatrix}, \tag{3.11}$$

$$s^i = \frac{\underline{r}^{i+1}_{i,i}(N)}{\sqrt{(\underline{r}^{i+1}_{i,i}(N))^2 + (\underline{r}^{i+1}_{i,i+1}(N))^2}}, \text{ and}$$

$$c^i = \frac{-\underline{r}^{i+1}_{i,i+1}(N)}{\sqrt{(\underline{r}^{i+1}_{i,i}(N))^2 + (\underline{r}^{i+1}_{i,i+1}(N))^2}}.$$

The proof follows by direct substitution and the uniqueness properties of the QR factorization. The full proof is provided in [16].

Based on the preceding paragraphs, the following is an outline of the procedure for obtaining the order recursions for the error vectors, LS coefficient vectors, and residual error energies in projecting $\mathbf{y}(N)$ onto the column space of $\mathbf{X}^i(N)$ $(0 \le i \le n)$. Assume we start with $\mathbf{Q}(N)$ and $\mathbf{R}^{-1}(N)$ in the inverse QR factorization of $\mathbf{A}(N)$. The LS coefficient vector in projecting $\mathbf{y}(N)$ onto the column space of $\mathbf{X}^n(N)$ is available immediately from the $(n+1)^{st}$ column of $\mathbf{R}^{-1}(N)$ as

$$-\mathbf{w}^{n+1}_{n+1}(N) = \frac{-1}{\underline{r}^{n+1}_{n+1,n+1}(N)}\left[\underline{r}^{n+1}_{1,n+1}(N), \underline{r}^{n+1}_{2,n+1}(N), \ldots, \underline{r}^{n+1}_{n,n+1}(N)\right]^T$$

and the associated error vector is the $(n+1)^{st}$ column of $\mathbf{Q}(N)$ divided by $\underline{r}^{n+1}_{n+1,n+1}(N)$. The residual error energy, i.e. the sum of the squares of the elements of the error vector, is also available as$\left[r^{n+1}_{n+1,n+1}(N) \right]^{-2}$. To obtain the order updates for all the lower order problems of projecting $y(N)$ onto the column space of $X^{i-1}(N)$ ($i = 1, 2, \ldots, n$), iteratively update $Q^i(N)$ and $(R^i(N))^{-1}$ using (3.10). Then, at each stage, the LS coefficient vector is

$$-w^i_i(N) = \frac{-1}{r^i_{i,i}(N)} \left[r^i_{1,i}(N), r^i_{2,i}(N), \ldots, r^i_{i-1,i}(N) \right]^T,$$

the associated error vector is the i^{th} column of $Q^i(N)$ divided by $r^i_{i,i}(N)$, and the residual error energy is $\frac{1}{\left(r^i_{i,i}(N)\right)^2}$.

To save computational expense, the order updating of $Q^i(N)$ can be omitted, or, as would be likely in a time recursive situation, one can order update only the last row of $Q^i(N)$, which represents the normalized backward prediction errors. The time recursive algorithms of [5] and [6] can be easily modified so that the Nth row of $Q(N)$ is calculated in addition to $R^{-1}(N)$.

It should also be noted that the entire matrix operation of (3.12) need not be performed. In the order recursions, only the i^{th} column of $(R^i(N))^{-1}$ needs to be calculated, since the first i-1 columns remain unchanged from $(R^{i+1}(N))^{-1}$, and the last n-i columns are not needed in the subsequent lower order recursions. This significantly reduces the computational expense involved in the order updates. The complete order updating algorithm for calculating the LS coefficient vectors, the residual errors at time N, and the residual error energy for all the lower order problems of projecting $y(N)$ onto the column space of $X^{i-1}(N)$ ($1 \le i \le n$) is given in Table I. Here, the N^{th} row of $Q^i(N)$ is denoted $\left[q^i_1, q^i_2, \ldots, q^i_{n+1}\right]$. Thus, the normalized residual error at the N^{th} timestep in projecting $y(N)$ onto the column space of $X^{i-1}(N)$ is q^i_i. In Table I, the time index N has been dropped for convenience.

TABLE I: The Order Updating Algorithm

Initialization

$r^{n+1}_{k,l} := \left[R^{-1}\right]_{k,l}$,

$\left[q^{n+1}_1, q^{n+1}_2, \ldots, q^{n+1}_{n+1}\right]$ = last row of Q

For $j = 1$ to n do

$w^{n+1}_{j,n+1} = r^{n+1}_{j,n+1}/r^{n+1}_{n+1,n+1}$

end j loop

For $i = n$ down to 1 do

$d = \sqrt{\left(r^{n+1}_{i,i}\right)^2 + \left(r^{i+1}_{i,i+1}\right)^2}$

$s = r^{n+1}_{i,i}/d$

$c = -r^{i+1}_{i,i+1}/d$

$r^i_{i,i} = s\, r^{i+1}_{i+1,i+1}$

$q^i_i = c\, q^{n+1}_i + s\, q^{i+1}_{i+1}$

For $j = 1$ to $i-1$ do

$r^i_{j,i} = c\, r^{n+1}_{j,i} + s\, r^{i+1}_{j,i+1}$

$w^i_{j,i} = r^i_{j,i}/r^i_{i,i}$

end j loop

end i loop

The algorithm in Table I requires that $R^{-1}(N)$ and the last row of $Q(N)$ are available. This can be accomplished using any of the previously mentioned batch or time-recursive algorithms. Given $R^{-1}(N)$ and the last row of $Q(N)$, the computational expense of the algorithm is $1.5n^2+6.5n$ MADs plus n square roots. If implemented in a time recursive framework, then the total computational expense involved at each timestep for time updating $R^{-1}(N)$ and the last row of $Q(N)$ and order updating the LS coefficient vectors, the residual error at time N, and the residual error energy for all order recursions depends on the particular algorithm used to time update $R^{-1}(N)$. The algorithm of [6] time updates $W(N)$, not $R^{-1}(N)$, assuming an exponentially weighted window on the data. After slight modifications, it can be used to update $R^{-1}(N)$ and the last row of $Q(N)$ with a computational expense of $2n^2+9n+2$ MADs per timestep. Thus, the total computational expense for the time and order updates is $3.5n^2+15.5n+2$ MADs plus n square roots.

IV. APPLICATIONS TO ADAPTIVE LEAST SQUARES FILTERING

In this section, we discuss how to apply the results of sections II and III to a multichannel adaptive LS filtering.

Suppose we have a scalar desired output sequence $d(j)$ and an m-channel vector input sequence $\mathbf{u}(j) = [u_1(j)\ u_2(j) \ldots u_m(j)]$ with samples available for $0 \le j \le t$. With minor modifications the results of this section can be extended to the multichannel output situation, but for the sake of notational simplicity we consider here only the single channel output case. Define for $i = 1, 2, \ldots, p$; $j = p-1, p, \ldots, t$

$$\mathbf{u}^i(j) := [\mathbf{u}(j), \mathbf{u}(j-1), \ldots, \mathbf{u}(j-i+1)], \qquad (4.1)$$

where p is some pre-specified maximum filter order.

The goal is to time recursively find the i^{th} order FIR linear filters ($\mathbf{h}_i$, $1 \le i \le p$) that best match the desired output sequence $d(j)$ with the filter output sequences

$$y_i(j,\mathbf{h}_i) := \mathbf{u}^i(j)\mathbf{h}_i, \qquad (4.2)$$

where $\mathbf{h}_i$ is the $m \cdot i$ length column vector of filter coefficients. Suppose we want to select the filter coefficients so that they are optimal in the LS sense. That is, at time t, we want to select the optimal i^{th} order filter, denoted $\mathbf{h}_i(t)$, so as to minimize the loss functions

$$\sum_{j=0}^{t-p+1} \lambda^{2j} e_i^2(t-j,\mathbf{h}_i) \qquad (4.3)$$

for filters of order $1 \le i \le p$. Here, $e_i(j,\mathbf{h}_i)$ is the output error at time j of the i^{th} order filter $\mathbf{h}_i$:

$$e(j,\mathbf{h}_i) := d(j) - y_i(j,\mathbf{h}_i) = d(j) - \mathbf{u}^i(j)\mathbf{h}_i, \qquad (4.4)$$

and $\lambda \in (0,1]$ is a forgetting factor. We can use the results of the previous sections to solve the problem by setting $n = mp$ and defining

$$\mathbf{X}^n(t) := [\mathbf{x}_1(t), \mathbf{x}_2(t), \ldots, \mathbf{x}_n(t)] := \qquad (4.5a)$$

$$\left[\lambda^{t-p+1}\left[\mathbf{u}^p(p-1)\right]^T, \lambda^{t-p}\left[\mathbf{u}^p(p)\right]^T, \ldots, \lambda\left[\mathbf{u}^p(t-1)\right]^T, \left[\mathbf{u}^p(t)\right]^T \right]^T$$

and $\mathbf{y}(t) := \left[\lambda^{t-p+1} d(p-1), \lambda^{t-p} d(p), \ldots, \lambda d(t-1), d(t) \right]^T$,

and $\mathbf{A}(t) := [\mathbf{X}^n(t), \mathbf{y}(t)]$. $\qquad$ (4.5b)

Note that it is not assumed here that the data is pre-windowed. Also note that if we project $\mathbf{y}(t)$ onto the span of the first m•i columns of $\mathbf{X}^n(t)$, the LS coefficient vector is exactly $\mathbf{h}_i(t)$,the i^{th} order filter that minimizes (4.3). Furthermore, the j^{th} component of the associated error vector in the projection is $\lambda^{t-p+2-j}e_i(p-2+j,\mathbf{h}_i(t))$. Consequently, the results of sections II and III can be used to find the optimal filter coefficients, output errors, and minimized loss functions for all order filters (i = 1, 2, . . ., p). Specifically, suppose we are using the algorithm of [6] to time update $\mathbf{R}^{-1}(t)$ and the last row of $\mathbf{Q}(t)$ in the inverse QR factorization of $\mathbf{A}(t)$. Then, using the notation of section III, the optimal filter coefficients $\mathbf{h}_i(t)$ and corresponding loss function of (4.3) are available from the $(mi+1)^{st}$ column of $(\mathbf{R}^{mi+1}(t))^{-1}$. Also, the output error at time t of the optimal i^{th} order filter, $e_i(t,\mathbf{h}_i(t))$, is available from the last row of $\mathbf{Q}^{mi+1}(t)$. The exact relationships are given in the discussion following Proposition 3.1.

In addition to the computational advantages[16], the method of this paper is not restricted to the case that the input data is a time series. The results here apply to general LS problems, where, of course, the transversal filter coefficients would be replaced by the coefficient vector in the LS projection. Since situations where the input data is not a time series are just special cases of multi-channel filtering with p=1, the algorithms of [9], [10], and [12] still apply, whereas the algorithms of the remainder of [8] – [13] do not. However, since [9] and [12] were designed to order update all filters simultaneously, they cannot be applied *order recursively* to situations where the input data is not a time series. There are no such restrictions on the algorithm of this paper.

The generality of the method also allows considerable freedom in channel length and the sequence in which the filter orders are updated ("downdated" would actually be a more appropriate term). It is not necessary that the filter orders for the various input channels be equal. In the case of unequal channel lengths, the number of columns of $\mathbf{X}^n(t)$ is $n=\sum_{i=1}^{m} p_i$, where p_i is the order of the i^{th} channel. In addition, the filters can be order updated in any manner desired by simply defining $\mathbf{X}^n(t)$ so that its columns are arranged appropriately. Of the order recursive transversal filter methods [8] – [13], only [10] possesses these features.

V. CONCLUSIONS

A geometric interpretation of the QR factorization in the general linear LS problem has been investigated, resulting in a highly efficient algorithm for achieving order recursions in the LS coefficient vectors, error vectors, and error energies.

The generality of the derivation and clear geometric interpretation allow easy application of the results to a wide variety of LS problems, either off-line, or time recursively with exponential weighting, a sliding rectangular window, or a growing length rectangular window. A multichannel adaptive LS filtering example is presented. The results of this paper were used to derive a method for time and order recursively identifying the transversal filter coefficients, residual errors, and residual error energies. The residual error energies can then be used to select the most appropriate filter/model order. In addition to being more versatile, the method presented here is more computationally efficient than the previous methods (namely, those of [9] and [12]) applied to time and order recursive situations.

REFERENCES

1 H. T. Kung and W.M. Gentleman, "Matrix triangularization by systolic arrays," *Proc. SPIE Int. Soc. Opt. Eng.*, vol. 298, 1981.

2 J. G. McWhirter, "Recursive least-squares minimization using a systolic array," *Proc. SPIE Int. Soc. Opt. Eng.*, paper 431-15, 1983.

3 G. H. Golub and C. F. Van Loan, *Matrix Compputations*. Baltimore, MD: The Johns Hopkins University Press, 1983.

4 C. T. Pan and R. J. Plemmons, "Least squares modifications with inverse factorizations: parallel implications," *J. Comp. Appl. Math.*, vol. 27, pp. 109-127, 1989.

5 S. T. Alexander and A. L. Ghirnikar, "A method for Recursive Least Squares Filtering based upon an inverse QR decomposition," *IEEE Trans. Signal Processing*, vol. 41, pp. 20-30, Jan., 1993.

6 H. Sakai, "Recursive least-squares algorithms of modified Gram-Schmidt type for parallel weight extraction," *IEEE Trans. Signal Processing*, vol. 42, pp. 429-433, Feb., 1994.

7 F. Ling, "Givens rotation based least squares lattice and related algorithms," *IEEE Trans. Signal Processing*, vol. 39, pp. 1541-1551, July, 1991.

8 M. Morf, B. Dickinson, T. Kailath, and A. Vieira, "Efficient Solution of Covariance Equations for Linear Prediction," *IEEE Trans. Accoustics, Speech, Signal Processing*, vol. ASSP-25, no. 5, pp. 429-433, Oct., 1977.

9 N. Kalouptsidis, G. Carayannis, D. Manolakis, and E. Koukoutsis, "Efficient recursive in order least squares FIR filtering and prediction," *IEEE Trans. Accoustics, Speech, Signal Processing*, vol. 33, pp. 1175-1187, Oct., 1985.

10 G. Glentis and N. Kalouptsisis, "Efficient multichannel FIR filtering using a single step versatile order recursive algorithm," *Signal Processing*, vol. 37, pp. 437-462, 1994.

11 B. Porat, B. Friedlander, and M. Morf, "Square Root Covariance Ladder Algorithms," *IEEE Trans. Automatic Control*, vol. AC-27, no. 4, pp. 813-829, Aug., 1982.

12 B. Porat and T. Kailath, "Normalized Lattice Algorithms for Least-Squares FIR System Identification," *IEEE Trans. Accoustics, Speech, Signal Processing*, vol. ASSP-31, no. 1, pp. 122-128, Feb., 1983.

13 X. Yu and Z. He, "Efficient Block Implementation of Exact Sequential Least-Squares Problems," *IEEE Trans. Accoustics, Speech, Signal Processing*, vol. 36, no. 3, pp. 392-399, March, 1988.

14 P. Strobach, *Linear Prediction Theory*. N.Y., N.Y.: Springer-Verlag, 1990.

15 F. Ling, D. Manolakis, and J. G. Proakis, "A recursive modified Gram-Schmidt algorithm for least-squares estimation," *IEEE Trans. Accoustics, Speech, Signal Processing*, vol. 34, pp. 829-836, Aug., 1986.

16 D. Apley and J. Shi, "The Inverse QR Decomposition in order recursive calculation of least squares coefficients", submitted to *IEEE Trans. Signal Processing*, 1995.

17 P. A. Regalia and M. G. Bellanger, "On the duality between fast QR methods and lattice methods in least squares adaptive filtering," *IEEE Trans. Signal Processing*, vol. 39, pp. 879-891, April, 1991.