New Fast QR Decomposition Least Squares Adaptive Algorithms

Athanasios A. Rontogiannis and Sergios Theodoridis

***Abstract*—This paper presents two new, closely related adaptive algorithms for LS system identification. The starting point for the derivation of the algorithms is the inverse Cholesky factor of the data correlation matrix, obtained via a QR decomposition (QRD). Both algorithms are of $O(p)$ computational complexity, with p being the order of the system. The first algorithm is a fixed order QRD scheme with enhanced parallelism. The second is an order recursive lattice type algorithm based exclusively on orthogonal Givens rotations, with lower complexity compared to previously derived ones. Both algorithms are derived following a new approach, which exploits efficient time and order updates of a specific state vector quantity.**

***Index Terms*—Adaptive algorithms, fast algorithms.**

I. Introduction

ADAPTIVE least squares algorithms for system identification [1]–[7] are popular due to their fast converging properties and are used in a variety of applications, such as channel equalization, echo cancellation, spectral analysis, and control, to name but a few. Among the various efficiency issues characterizing the performance of an algorithm, those of computational complexity, parallelism, and numerical robustness are of particular importance, especially in applications where medium to long filter lengths are required. It may sometimes be preferable to use an algorithm of higher complexity but with good numerical error robustness since this may allow its implementation with shorter wordlenghts and fixed point arithmetic. This has led to the development of a class of adaptive algorithms, based on the numerically robust QR factorization of the input data matrix via the Givens rotation approach [23].

The development of Givens rotations-based QR decomposition algorithms has evolved along three basic directions. Schemes of $O(p^2)$ complexity per time iteration were the first to be derived, with p being the order of the system [8], [9]. These schemes update the Cholesky factor of the input data correlation matrix and can efficiently be implemented on two-dimensional (2-D) systolic arrays. Furthermore, as it is shown in [9], the modeling error can be extracted directly without it being necessary to compute explicitly the estimates of the transversal parameters of the unknown FIR system. Square-root free forms of the above algorithms related to the modified Gram–Schmidt factorization approach [10] were also proposed. An alternative $O(p^2)$ RLS scheme, based on the update of the inverse Cholesky factor of the data correlation matrix, was also recently developed [11], [12].

The other category of Givens rotations-based algorithms is of the lattice, order recursive type, exhibiting $O(p)$ complexity per time iteration [13]–[16]. As with all LS lattice structures [6], [7], these algorithms compute the modeling LS error for all intermediate orders in a pipelined fashion. A third class consists of algorithms that compute the modeling error directly, although they lack the pipelining property of the lattice-type algorithms [17]–[21]. On the other hand, they have lower complexity, compared with their lattice counterparts, and they are appropriate for fixed-order modeling. This is basically due to the fact that a set of rotation parameters (corresponding to the lattice reflection coefficients) are generated backward in order, starting from the one with the maximum order [1].

This paper presents two closely related yet different Givens rotations-based QR decomposition, $O(p)$ algorithms. One is of the latter type, i.e., fixed-order, direct error computing algorithm. It has similar complexity, but it offers enhanced parallelism compared with previously derived ones of the same category. Thus, if two processors are used, the computation time is almost halved. A modification of this algorithm leads to an order recursive lattice-type scheme involving orthogonal Givens rotations only. The complexity of the lattice-type algorithm is the same to that of the fixed-order one. Therefore, a substantial saving is accomplished compared to already known QR lattice schemes.

In this work, a novel approach is used for deriving the algorithms. Specifically, we concentrate on the (inverse) Cholesky factor of the input data matrix and investigate its order and time update properties. Then, a particular vector quantity, which provides all the necessary for the LS error update rotation parameters, is efficiently updated. This method is different from the approach followed so far for the derivation of fast QRD-based schemes [1], [17]–[20], where update expressions of the orthogonal factor Q are formulated, and then, a pinning vector is applied in order to extract the necessary quantities. The new method is simpler, more direct, and provides insight into all the internal quantities appearing in the new algorithms.

The paper is organized as follows. Section II reviews the application of the QR decomposition method to the RLS problem. The new fast QRD algorithms are then derived in Section III. Section IV highlights the connection of the new algorithms with already known fast QRD schemes. Simula-

Manuscript received July 27, 1994; revised January 5, 1998. The associate editor coordinating the review of this paper and approving it for publication was Dr. Pierre Duhammel.

The authors are with the Department of Informatics, University of Athens, Athens, Greece (e-mail: stheodor@di.uoa.gr).

Publisher Item Identifier S 1053-587X(98)05219-2.

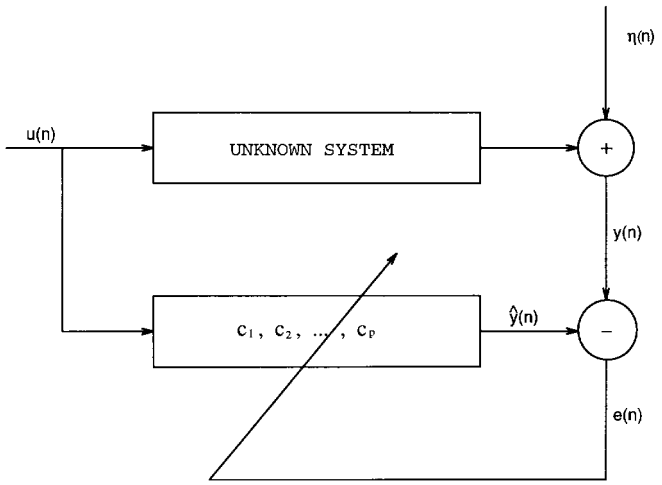

Fig. 1. System identification problem.

tion results are provided in Section V, whereas Section VI concludes this work. For clarity of presentation real signals are considered throughout this paper. We mostly adopt the notation that appears in [1].

II. Formulation of the Problem

Fig. 1 illustrates the typical system identification task, which is our main concern in this paper. Given an unknown FIR system excited by an input signal $u(n)$, we seek the estimates of the p unknown tap coefficients so that the error $\varepsilon(N)$ between the measured output of the system $y(N)$ and the output of an associated model $\hat{y}(N)$ is minimum in the least squares sense. That is, the sum

$$||\boldsymbol{\varepsilon}_p(N)||^2 = \sum_{n=1}^{N} \lambda^{N-n}[y(n) - \boldsymbol{c}_p^T(N)\boldsymbol{u}_p(n)]^2$$

is minimum, where λ is the usual forgetting factor with $0 \ll \lambda \leq 1$

$$\boldsymbol{\varepsilon}_p^T(N) = [\varepsilon_p(1), \varepsilon_p(2), \cdots, \varepsilon_p(N)]$$
$$\boldsymbol{c}_p^T(N) = [c_1(N), c_2(N), \cdots, c_p(N)]$$

and

$$\boldsymbol{u}_p^T(n) = [u(n), u(n-1), \cdots, u(n-p+1)].$$

The quantity $\eta(n)$ in the figure stands for the measurement noise. From the above definitions

$$\boldsymbol{\varepsilon}_p(N) = \boldsymbol{y}(N) - U_p(N)\boldsymbol{c}_p(N)$$

is readily understood, where

$$\boldsymbol{y}(N) = \Lambda(N)[y(1), y(2), \cdots, y(N)]^T$$
$$\Lambda(N) = \mathrm{diag}[\lambda^{N-1/2}, \lambda^{N-2/2}, \cdots, 1]$$

and $U_p(N)$ is the $N \times p$ input data matrix given by

$$U_p(N) = \Lambda(N)\begin{bmatrix} \boldsymbol{u}_p^T(1) \\ \boldsymbol{u}_p^T(2) \\ \vdots \\ \boldsymbol{u}_p^T(N) \end{bmatrix}$$
$$= \Lambda(N)\begin{bmatrix} u(1) & 0 & \cdots & 0 \\ u(2) & u(1) & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ u(N) & u(N-1) & \cdots & u(N-p+1) \end{bmatrix}.$$

In other words, the prewindowed assumption is adopted. The solution to the above problem is provided by the well-known normal equations

$$\boldsymbol{c}_p(N) = R_p^{-1}(N)\boldsymbol{d}_p(N)$$

where

$$R_p(N) = U_p^T(N)U_p(N)$$
$$\boldsymbol{d}_p(N) = U_p^T(N)\boldsymbol{y}(N).$$

The task of the current paper is to develop new algorithms for the efficient computation of $\varepsilon_p(N)$. The approach is via the QR decomposition [23] of the input data matrix $U_p(N)$ that is

$$Q_p(N)U_p(N) = \begin{bmatrix} \tilde{R}_p(N) \\ \bigcirc \end{bmatrix}$$

with $Q_p(N)Q_p^T(N) = Q_p^T(N)Q_p(N) = \boldsymbol{I}$. $\tilde{R}_p(N)$ is a $p \times p$ upper triangular factor. Obviously

$$R_p(N) = \tilde{R}_p^T(N)\tilde{R}_p(N).$$

By premultiplying $\boldsymbol{y}(N)$ with $Q_p(N)$, we obtain

$$Q_p(N)\boldsymbol{y}(N) = \begin{bmatrix} \boldsymbol{p}_p(N) \\ \boldsymbol{v}_p(N) \end{bmatrix}$$

where $\boldsymbol{p}_p(N)$ is the upper $p \times 1$ part and $\boldsymbol{v}_p(N)$ the lower $(N-p) \times 1$ part of the resulting (transformed) vector. It is by now well known [1] that the LS solution is given by

$$\tilde{R}_p(N)\boldsymbol{c}_p(N) = \boldsymbol{p}_p(N). \tag{1}$$

The efficient update of the factor $\tilde{R}_p(N)$ is at the heart of our problem. It has been shown that ([1])

$$\hat{Q}_p(N)\begin{bmatrix} \lambda^{1/2}\tilde{R}_p(N-1) \\ \boldsymbol{u}_p^T(N) \end{bmatrix} = \begin{bmatrix} \tilde{R}_p(N) \\ \boldsymbol{0}^T \end{bmatrix} \tag{2}$$

and $\hat{Q}_p(N)$ consists of a sequence of basic Givens rotations, which successively annihilate the elements of $\boldsymbol{u}_p^T(N)$ against $\lambda^{1/2}\tilde{R}_p(N-1)$, resulting in the update $\tilde{R}_p(N)$. That is

$$\hat{Q}_p(N) = Q_{\theta_p}(N)Q_{\theta_{p-1}}(N)\cdots Q_{\theta_1}(N) \tag{3}$$

where

$$Q_{\theta_i}(N) = \begin{bmatrix} \boldsymbol{I}_{i-1} & & \bigcirc & & \\ & \cos\theta_i(N) & 0 \cdots 0 & \sin\theta_i(N) & \\ \bigcirc & & \boldsymbol{I}_{p-i} & & \\ & -\sin\theta_i(N) & 0 \cdots 0 & \cos\theta_i(N) & \end{bmatrix}. \tag{4}$$

At the same time, it is most interesting that ([1])

$$\hat{Q}_p(N)\begin{bmatrix}\lambda^{1/2}\boldsymbol{p}_p(N-1)\\ y(N)\end{bmatrix}=\begin{bmatrix}\boldsymbol{p}_p(N)\\ \tilde{e}_p(N)\end{bmatrix} \tag{5}$$

where

$$\tilde{e}_p(N)=\operatorname{sgn}(\varepsilon_p(N))\sqrt{\varepsilon_p(N)e_p(N)}$$

and $e_p(N)$ is the *a priori* error expressed as

$$e_p(N)=y(N)-\boldsymbol{c}_p^T(N-1)\boldsymbol{u}_p(N). \tag{6}$$

The so-called angle normalized error $\tilde{e}_p(N)$ is related to the rotation angles θ_i by [1, pp. 271–272]

$$\tilde{e}_p(N)=e_p(N)\tilde{a}_p(N)=e_p(N)\prod_{i=1}^{p}\cos\theta_i(N) \tag{7}$$

and $\tilde{a}_p(N)$ is the square root of the likelihood related variable $a_p(N)$ defined as ([2])

$$\begin{aligned}a_p(N)=&(1+\lambda^{-1}\boldsymbol{u}_p^T(N)\tilde{R}_p^{-1}(N-1)\\ &\tilde{R}_p^{-T}(N-1)\boldsymbol{u}_p(N))^{-1}.\end{aligned} \tag{8}$$

Two types of algorithms will be presented in the following sections. The first is of the fixed-order type, computing directly the error $e_p(N)$. A modification of this leads to an order-recursive scheme (lattice type) for direct error computation.

III. The New Fast QRD Algorithms

In contrast to previously derived fast QRD algorithms [1], [17]–[21], our starting point is the vector

$$\boldsymbol{g}_p(N)=\frac{\tilde{R}_p^{-T}(N-1)\boldsymbol{u}_p(N)}{\sqrt{\lambda}}. \tag{9}$$

The essence behind any fast fixed-order $O(p)$ scheme is that the time shift property of the input data offers the possibility to circumvent the time update of a matrix by updating a vector quantity instead. Different algorithms are built on different vectors (the state variables of the equivalent algorithmic system [1]). In this paper, the algorithms evolve around $\boldsymbol{g}_p(N)$, whose time update provides all necessary rotation angles.

A. Time Update of $\boldsymbol{g}_p(N)$

From the definition of the factor $\tilde{R}_p(N)$, it is easy to see that

$$Q_p(N)U_{p+1}(N)=\begin{bmatrix}\tilde{R}_p(N) & \boldsymbol{p}_p^b(N)\\ \bigcirc & \boldsymbol{v}_p^b(N)\end{bmatrix} \tag{10}$$

where $\boldsymbol{p}_p^b(N)$ is the upper $p\times 1$ part of the vector $Q_p(N)\boldsymbol{y}^b(N)$, and

$$\boldsymbol{y}^b(N)=\Lambda(N)[0,\cdots,0,u(1),\cdots,u(N-p)]$$

that is, the last column of the data matrix $U_{p+1}(N)$. $\boldsymbol{p}_p^b(N)$ is a quantity related to the backward prediction QR problem [1]. Multiplying both sides of (10) by an orthogonal matrix $Q_p^b(N)$, which annihilates the elements of $\boldsymbol{v}_p^b(N)$ by rotating them against its first element, we obtain

$$Q_p^b(N)Q_p(N)U_{p+1}(N)=\begin{bmatrix}\tilde{R}_p(N) & \boldsymbol{p}_p^b(N)\\ \boldsymbol{0}^T & \tilde{a}_p^b(N)\\ \bigcirc & \boldsymbol{0}\end{bmatrix}$$

with

$$\tilde{a}_p^b(N)=\sqrt{a_p^b(N)}=\|\boldsymbol{v}_p^b(N)\|$$

being the square root of the backward prediction error power [1]. It is now straightforward that

$$\tilde{R}_{p+1}(N)=\begin{bmatrix}\tilde{R}_p(N) & \boldsymbol{p}_p^b(N)\\ \boldsymbol{0}^T & \tilde{a}_p^b(N)\end{bmatrix} \tag{11}$$

holds. Assuming persistency of excitation, the inverse of the factor $\tilde{R}_{p+1}(N)$ exists and can easily be obtained from (11) as

$$\tilde{R}_{p+1}^{-1}(N)=\begin{bmatrix}\tilde{R}_p^{-1}(N) & -\dfrac{1}{\tilde{a}_p^b(N)}\tilde{R}_p^{-1}(N)\boldsymbol{p}_p^b(N)\\ \boldsymbol{0}^T & \dfrac{1}{\tilde{a}_p^b(N)}\end{bmatrix}. \tag{12}$$

Moreover, by establishing a relation between $\tilde{R}_{p+1}(N)$ and $\tilde{R}_p(N-1)$, we will end up with a step-up/step-down update procedure for $\boldsymbol{g}_p(N)$. Indeed, if $Q_p(N-1)$ triangularizes $U_p(N-1)$, then

$$\begin{aligned}&\begin{bmatrix}1 & \boldsymbol{0}^T\\ \boldsymbol{0} & Q_p(N-1)\end{bmatrix}U_{p+1}(N)\\ &\quad=\begin{bmatrix}\lambda^{N-1/2}u(1) & \boldsymbol{0}^T\\ \boldsymbol{p}_p^f(N) & \tilde{R}_p(N-1)\\ \boldsymbol{v}_p^f(N) & \bigcirc\end{bmatrix}\end{aligned} \tag{13}$$

where $\boldsymbol{v}_p^f(N)$ is related to the forward error power $a_p^f(N)$ as

$$\tilde{a}_p^f(N)=\sqrt{a_p^f(N)}=\sqrt{\|\boldsymbol{v}_p^f(N)\|^2+\lambda^{N-1}|u(1)|^2}.$$

If now $\tilde{Q}_p^f(N)$ is the orthogonal matrix that annihilates the elements of $\boldsymbol{v}_p^f(N)$ against the first element of the matrix in (13), which initially is $\lambda^{N-1/2}u(1)$, we get

$$\begin{aligned}&\tilde{Q}_p^f(N)\begin{bmatrix}\lambda^{N-1/2}u(1) & \boldsymbol{0}^T\\ \boldsymbol{p}_p^f(N) & \tilde{R}_p(N-1)\\ \boldsymbol{v}_p^f(N) & \bigcirc\end{bmatrix}\\ &\quad=\begin{bmatrix}\tilde{a}_p^f(N) & \boldsymbol{0}^T\\ \boldsymbol{p}_p^f(N) & \tilde{R}_p(N-1)\\ \boldsymbol{0} & \bigcirc\end{bmatrix}.\end{aligned} \tag{14}$$

Consider now the upper $(p+1)\times(p+1)$ part of the matrix in the right-hand side of (14), say, $\hat{R}_{p+1}(N)$. In order to obtain the Cholesky factor $\tilde{R}_{p+1}(N)$ of U_{p+1}, $\hat{R}_{p+1}(N)$ must be multiplied with an appropriate orthogonal matrix, say, $\hat{Q}_p^f(N)$, as described by

$$\begin{aligned}\hat{Q}_p^f(N)\hat{R}_{p+1}(N)&=\hat{Q}_p^f(N)\begin{bmatrix}\tilde{a}_p^f(N) & \boldsymbol{0}^T\\ \boldsymbol{p}_p^f(N) & \tilde{R}_p(N-1)\end{bmatrix}\\ &=\tilde{R}_{p+1}(N).\end{aligned} \tag{15}$$

The matrix $\hat{Q}_p^f(N)$, which fulfils the required triangularization in (15), can be contructed as a sequence of p Givens rotations,

which annihilate in a bottom-up procedure the elements of $\boldsymbol{p}_p^f(N)$ against the element in the top row of $\hat{R}_{p+1}(N)$ [initially, $\tilde{a}_p^f(N)$]. As a result, the $(p+1)\times(p+1)$ orthogonal matrix $\hat{Q}_p^f(N)$ can be written as the product of p rotation matrices, that is

$$\hat{Q}_p^f(N) = Q_{\phi_1}(N)Q_{\phi_2}(N)\cdots Q_{\phi_p}(N) \tag{16}$$

where

$$Q_{\phi_i}(N) = \begin{bmatrix} \cos\phi_i(N) & 0 & \cdots & 0 & \sin\phi_i(N) & \\ & & \boldsymbol{I}_{i-1} & & & \bigcirc \\ -\sin\phi_i(N) & 0 & \cdots & 0 & \cos\phi_i(N) & \\ & & \bigcirc & & & \boldsymbol{I}_{p-i} \end{bmatrix}. \tag{17}$$

Indeed, it is easily verified that such a $\hat{Q}_p^f(N)$ preserves the triangular structure of the matrix $\tilde{R}_p(N-1)$ in (15), although its value will be changed as a result. In addition to this, $\hat{Q}_p^f(N)$ "fills" the zero elements of the first row of $\hat{R}_{p+1}(N)$ and, thus, transforms $\hat{R}_{p+1}(N)$ into a $(p+1)\times(p+1)$ upper triangular matrix (with positive diagonal elements).

A byproduct of the triangularization in (15) is that all lower order forward energies $\tilde{a}_i^f(N)$ are generated. These quantities appear as top left elements of the resulting matrices after each multiplication with $Q_{\phi_{i+1}}(N)$ in (15) for $i = p-1, p-2, \cdots, 0$. Indeed, if the initial data matrix in (13) was $U_{i+1}(N)$ instead of $U_{p+1}(N)$, then the corresponding blocks of the matrix in the right-hand side of (13) would be $\boldsymbol{p}_i^f(N), \tilde{R}_i(N-1)$ and $\boldsymbol{v}_i^f(N)$, respectively. Since $U_{i+1}(N)$ coincides with the first $(i+1)$ columns of $U_{p+1}(N)$, completion of the procedure in (13), for the remaining $p-i$ columns of $U_{p+1}(N)$, could be accomplished in such a way that $\boldsymbol{p}_i^f(N)$ remains unaffected. As a result, $\boldsymbol{p}_i^f(N)$ is essentially the upper $i\times 1$ part of $\boldsymbol{p}_p^f(N)$. Taking the procedure for the ith-order problem further on, the nonzero blocks of the matrix in the right-hand side of (14) turn out to be $\tilde{a}_i^f(N), \boldsymbol{p}_i^f(N)$ and $\tilde{R}_i(N-1)$. In other words, the upper $(i+1)\times(i+1)$ block of such a matrix has exactly the same form as the $(i+1)\times(i+1)$ upper left block of the matrix that results after the application of $Q_{\phi_{i+1}}(N)$ in (15). Consequently, the top left element of this last matrix must be $\tilde{a}_i^f(N)$. Moreover, due to the relation between $\boldsymbol{p}_p^f(N)$ and $\boldsymbol{p}_i^f(N)$, the rotation matrix $\hat{Q}_i^f(N)$ of the ith-order problem is expressed as

$$\hat{Q}_i^f(N) = Q_{\phi_1}(N)Q_{\phi_2}(N)\cdots Q_{\phi_i}(N).$$

From (15), we can formulate an alternative to (12) expression for the inverse of $\tilde{R}_{p+1}(N)$ as

$$\tilde{R}_{p+1}^{-1}(N) = \begin{bmatrix} \dfrac{1}{\tilde{a}_p^f(N)} & \boldsymbol{0}^T \\ -\dfrac{1}{\tilde{a}_p^f(N)}\tilde{R}_p^{-1}(N-1)\boldsymbol{p}_p^f(N) & \tilde{R}_p^{-1}(N-1) \end{bmatrix} \cdot (\hat{Q}_p^f(N))^T. \tag{18}$$

We now have available all necessary relations to obtain the time update of $\boldsymbol{g}_p(N)$ in $O(p)$.

1) Step Down: From (9), (12), and the input vector partition

$$\boldsymbol{u}_{p+1}(N+1) = [\boldsymbol{u}_p^T(N+1), u(N-p+1)]^T$$

we have (19), shown at the bottom of the page, where $g^{(p)}(N+1)$ is the last element of $\boldsymbol{g}_{p+1}(N+1)$, assuming that the numbering of the elements of $\boldsymbol{g}_{p+1}(N+1)$ starts from 0. More specifically, we have

$$g^{(p)}(N+1) = \frac{1}{\sqrt{\lambda}\tilde{a}_p^b(N)}\,[u(N-p+1) - (\tilde{R}_p^{-1}(N)\boldsymbol{p}_p^b(N))^T\boldsymbol{u}_p(N+1)]. \tag{20}$$

According to (1), however, $\tilde{R}_p^{-1}(N)\boldsymbol{p}_p^b(N)$ stands for the coefficients vector of the backward problem at time N. This, combined with (6) and (20), gives

$$g^{(p)}(N+1) = \frac{e_p^b(N+1)}{\sqrt{\lambda}\tilde{a}_p^b(N)}$$

where $e_p^b(N+1)$ corresponds to the *a priori* backward error of order p at time $N+1$.

From (19), we conclude that the vector $\boldsymbol{g}_p(N+1)$ is identical to the first p elements of $\boldsymbol{g}_{p+1}(N+1)$. Straightforward extension of this nesting property results in the expression for the elements of $\boldsymbol{g}_{p+1}(N+1)$ as

$$g^{(i)}(N+1) = \frac{e_i^b(N+1)}{\sqrt{\lambda}\tilde{a}_i^b(N)} \qquad i = 0, 1, \cdots, p. \tag{21}$$

$$\begin{aligned} \boldsymbol{g}_{p+1}(N+1) &= \frac{1}{\sqrt{\lambda}}\begin{bmatrix} \tilde{R}_p^{-T}(N) & \boldsymbol{0} \\ -\dfrac{1}{\tilde{a}_p^b(N)}(\boldsymbol{p}_p^b(N))^T\tilde{R}_p^{-T}(N) & \dfrac{1}{\tilde{a}_p^b(N)} \end{bmatrix}\begin{bmatrix} \boldsymbol{u}_p(N+1) \\ u(N-p+1) \end{bmatrix} \Leftrightarrow \\ \boldsymbol{g}_{p+1}(N+1) &= \frac{1}{\sqrt{\lambda}}\begin{bmatrix} \tilde{R}_p^{-T}(N)\boldsymbol{u}_p(N+1) \\ -\dfrac{1}{\tilde{a}_p^b(N)}(\boldsymbol{p}_p^b(N))^T\tilde{R}_p^{-T}(N)\boldsymbol{u}_p(N+1) + \dfrac{u(N-p+1)}{\tilde{a}_p^b(N)} \end{bmatrix} \Rightarrow \\ \boldsymbol{g}_{p+1}(N+1) &= \begin{bmatrix} \boldsymbol{g}_p(N+1) \\ g^{(p)}(N+1) \end{bmatrix} \end{aligned} \tag{19}$$

$$\boldsymbol{g}_{p+1}(N+1)=\frac{\hat{Q}_p^f(N)}{\sqrt{\lambda}}\begin{bmatrix}\frac{1}{\tilde{a}_p^f(N)} & -\frac{1}{\tilde{a}_p^f(N)}(\tilde{R}_p^{-1}(N-1)\boldsymbol{p}_p^f(N))^T\\ \mathbf{0} & \tilde{R}_p^{-T}(N-1)\end{bmatrix}\begin{bmatrix}u(N+1)\\ \boldsymbol{u}_p(N)\end{bmatrix}\Leftrightarrow$$

$$\boldsymbol{g}_{p+1}(N+1)=\frac{\hat{Q}_p^f(N)}{\sqrt{\lambda}}\begin{bmatrix}\frac{u(N+1)}{\tilde{a}_p^f(N)}-\frac{1}{\tilde{a}_p^f(N)}(\tilde{R}_p^{-1}(N-1)\boldsymbol{p}_p^f(N))^T\boldsymbol{u}_p(N)\\ \tilde{R}_p^{-T}(N-1)\boldsymbol{u}_p(N)\end{bmatrix}$$

2) Step Up: Combining (9), (18), and the input vector partition

$$\boldsymbol{u}_{p+1}(N+1)=[u(N+1),\boldsymbol{u}_p^T(N)]^T$$

we obtain the expression at the top of the page or

$$\begin{bmatrix}r_p(N+1)\\ \boldsymbol{g}_p(N)\end{bmatrix}=(\hat{Q}_p^f(N))^T\boldsymbol{g}_{p+1}(N+1) \tag{22}$$

where

$$r_p(N+1)=\frac{1}{\sqrt{\lambda}\tilde{a}_p^f(N)}[u(N+1)-(\tilde{R}_p^{-1}(N-1)\boldsymbol{p}_p^f(N))^T\boldsymbol{u}_p(N)].$$

Since $\tilde{R}_p^{-1}(N-1)\boldsymbol{p}_p^f(N)$ stands for the forward coefficients vector at time N, $r_p(N+1)$ is related to the *a priori* forward error $e_p^f(N+1)$ as

$$r_p(N+1)=\frac{e_p^f(N+1)}{\sqrt{\lambda}\tilde{a}_p^f(N).}$$

Due to the (order recursive) form of $\hat{Q}_p^f(N)$ and the nesting property of $\boldsymbol{g}_p(N)$ and $\boldsymbol{g}_{p+1}(N+1)$, the following quantities appear at the top of the vector in the right-hand side of (22), after the application of the rotation matrices $Q_{\phi_i}^T(N), i=1,2,\cdots,p$[1]

$$r_i(N+1)=\frac{e_i^f(N+1)}{\sqrt{\lambda}\tilde{a}_i^f(N)}\qquad i=1,\cdots,p. \tag{23}$$

Note, however, that the first element of $\boldsymbol{g}_{p+1}(N+1), r_0(N+1)=(u(N+1)/\tilde{a}_0^f(N)\sqrt{\lambda})$ is known at time $N+1$. Furthermore, given the rotation angles of $\hat{Q}_p^f(N)$ and the vector $\boldsymbol{g}_p(N)$, we can calculate from (22) the last p elements of $\boldsymbol{g}_{p+1}(N+1)$ in $O(p)$. For example, when the first rotation matrix $Q_{\phi_1}^T(N)$ multiplies the vector $\boldsymbol{g}_{p+1}(N+1)$, we get

$$\begin{bmatrix}\cos\phi_1(N) & -\sin\phi_1(N)\\ \sin\phi_1(N) & \cos\phi_1(N)\end{bmatrix}\begin{bmatrix}r_0(N+1)\\ g^{(1)}(N+1)\end{bmatrix}=\begin{bmatrix}r_1(N+1)\\ g^{(1)'}(N+1)\end{bmatrix}\Leftrightarrow$$

$$g^{(1)'}(N+1)=\sin(\phi_1(N))r_0(N+1)+\cos(\phi_1(N))g^{(1)}(N+1)$$

$$r_1(N+1)=\cos(\phi_1(N))r_0(N+1)-\sin(\phi_1(N))g^{(1)}(N+1).$$

[1]This outcome becomes more evident by describing the effect of $\hat{Q}_p^f(N)$ as in (22). Note that the errors $r_i(N+1)$ could also be generated in a backward manner by letting $\hat{Q}_p^f(N)$ premultiply the vector on the left-hand side of (22). This would lead to an alternative form for this step of the algorithm.

However, $g^{(1)'}(N+1)=g^{(0)}(N)$, $g^{(0)}(N)$ being the first element of $\boldsymbol{g}_p(N)$. This is the case because $g^{(1)'}(N+1)$ does not change after the application of the remaining rotations. We can therefore rewrite the above equations as

$$g^{(1)}(N+1)=\frac{g^{(0)}(N)-\sin(\phi_1(N))r_0(N+1)}{\cos(\phi_1(N))}$$

$$r_1(N+1)=\cos(\phi_1(N))r_0(N+1)-\sin(\phi_1(N))g^{(1)}(N+1).$$

Proceeding in the same way, we obtain the general expressions

$$g^{(i)}(N+1)=\frac{g^{(i-1)}(N)-\sin(\phi_i(N))r_{i-1}(N+1)}{\cos(\phi_i(N))}$$

$$r_i(N+1)=\cos(\phi_i(N))r_{i-1}(N+1)-\sin(\phi_i(N))g^{(i)}(N+1)$$

for $i=1,2,\cdots,p$. Thus, combining (22) and (19), we achieve the time update of $\boldsymbol{g}_p(N)$ in $O(p)$, which was our initial purpose.

B. Rotation Angles Update

Having completed the time update of $\boldsymbol{g}_p(N)$ in $O(p)$, we have all information necessary to obtain the rotation angle parameters $(\hat{Q}_p)$ that provide the *a priori* error (e_p). Indeed, by considering the effect of $\hat{Q}_p^f(N+1)$ on the first column of $\hat{R}_{p+1}(N+1)$ [see (15)], we have

$$\hat{Q}_p^f(N+1)\begin{bmatrix}\tilde{a}_p^f(N+1)\\ \boldsymbol{p}_p^f(N+1)\end{bmatrix}=\begin{bmatrix}\tilde{a}_0^f(N+1)\\ \mathbf{0}\end{bmatrix}. \tag{24}$$

The rotation parameters of $\hat{Q}_p^f(N+1)$ can be produced from (24) as

$$\cos\phi_i(N+1)=\frac{\tilde{a}_i^f(N+1)}{\tilde{a}_{i-1}^f(N+1)}$$

$$\sin\phi_i(N+1)=\frac{\boldsymbol{p}_p^{f(i)}(N+1)}{\tilde{a}_{i-1}^f(N+1)}$$

for $i=p,p-1,\cdots,1$. $\boldsymbol{p}_p^{f(i)}(N+1)$ is the ith element of $\boldsymbol{p}_p^f(N+1)$, and the square roots of the lower order forward prediction energies are calculated according to

$$\tilde{a}_{i-1}^f(N+1)=\sqrt{(\tilde{a}_i^f(N+1))^2+(\boldsymbol{p}_p^{f(i)}(N+1))^2}$$
$$i=p,p-1,\cdots,1.$$

TABLE I
NEW FIXED-ORDER FAST QRD ALGORITHM

(**Step 1**)

$r_0(N+1) = g^{(0)}(N+1) = \frac{u(N+1)}{\sqrt{\lambda}\tilde{a}_0^f(N)}$;

for $i = 1:p$,

$g^{(i)}(N+1) = \frac{g^{(i-1)}(N) - \sin(\phi_i(N))r_{i-1}(N+1)}{\cos(\phi_i(N))}$;

$r_i(N+1) = \cos(\phi_i(N))r_{i-1}(N+1) - \sin(\phi_i(N))g^{(i)}(N+1)$;

end;

(**Step 2**)

$\tilde{e}_0^f(N+1) = u(N+1)$;

for $i = 1:p$,

$\mathbf{p}_p^{f(i)}(N+1) = \lambda^{1/2}\cos\theta_i(N)\mathbf{p}_p^{f(i)}(N) + \sin\theta_i(N)\tilde{e}_{i-1}^f(N+1)$;

$\tilde{e}_i^f(N+1) = \cos\theta_i(N)\tilde{e}_{i-1}^f(N+1) - \lambda^{1/2}\sin\theta_i(N)\mathbf{p}_p^{f(i)}(N)$;

end;

(**Step 3**)

$\tilde{a}_p^f(N+1) = \sqrt{\lambda[\tilde{a}_p^f(N)]^2 + [\tilde{e}_p^f(N+1)]^2}$;

for $i = p:1$,

$\tilde{a}_{i-1}^f(N+1) = \sqrt{[\tilde{a}_i^f(N+1)]^2 + [\mathbf{p}_p^{f(i)}(N+1)]^2}$;

$\cos\phi_i(N+1) = \frac{\tilde{a}_i^f(N+1)}{\tilde{a}_{i-1}^f(N+1)}$;

$\sin\phi_i(N+1) = \frac{\mathbf{p}_p^{f(i)}(N+1)}{\tilde{a}_{i-1}^f(N+1)}$;

end;

(**Step 4**)

$\delta_0(N+1) = 1$;

for $i = 1:p$,

$\delta_i(N+1) = \sqrt{\delta_{i-1}^2(N+1) + [g^{(i-1)}(N+1)]^2}$;

$\cos\theta_i(N+1) = \frac{\delta_{i-1}(N+1)}{\delta_i(N+1)}$;

TABLE I *(Continued)*

$\sin\theta_i(N+1) = \frac{g^{(i-1)}(N+1)}{\delta_i(N+1)}$;

end;

(**Step 5**)

$\tilde{e}_0(N+1) = y(N+1)$;

for $i = 1:p$,

$\mathbf{p}_p^{(i)}(N+1) = \lambda^{1/2}\cos\theta_i(N+1)\mathbf{p}_p^{(i)}(N) + \sin\theta_i(N+1)\tilde{e}_{i-1}(N+1)$;

$\tilde{e}_i(N+1) = \cos\theta_i(N+1)\tilde{e}_{i-1}(N+1) - \lambda^{1/2}\sin\theta_i(N+1)\mathbf{p}_p^{(i)}(N)$;

end;

$e_p(N+1) = \tilde{e}_p(N+1)\delta_p(N+1)$;

Initialization

$\tilde{a}_0^f(0) = \sqrt{\lambda^p\mu}$, $\mathbf{g}_p(0) = \mathbf{0}$, $\mathbf{p}_p^f(0) = \mathbf{0}$

$cos\theta_i(0) = 1$, $cos\phi_i(0) = 1$, $i = 1, 2, \ldots, p$

However, in order to use (24), we must know the forward rotated reference vector $\boldsymbol{p}_p^f(N+1)$ and the square root of the forward prediction energy $\tilde{a}_p^f(N+1)$ at time $N+1$. The vector $\boldsymbol{p}_p^f(N+1)$ can be obtained by applying (5) to the forward prediction problem at time N. Indeed

$$\hat{Q}_p(N)\begin{bmatrix}\lambda^{1/2}\boldsymbol{p}_p^f(N)\\ u(N+1)\end{bmatrix} = \begin{bmatrix}\boldsymbol{p}_p^f(N+1)\\ \tilde{e}_p^f(N+1)\end{bmatrix} \tag{25}$$

where $\tilde{e}_p^f(N+1)$ is the angle-normalized forward error of the pth order at time $N+1$. $\tilde{a}_p^f(N+1)$ can then be computed from the well-known formula [1]

$$\tilde{a}_p^f(N+1) = \sqrt{\lambda[\tilde{a}_p^f(N)]^2 + [\tilde{e}_p^f(N+1)]^2}. \tag{26}$$

Note from (3) and (4) that when the rotation matrix $Q_{\theta_i}(N), i = 1, 2, \cdots, p$ is applied in (25), the ith-order angle normalized forward error appears at the bottom of the resulting vector.

Matrix $\hat{Q}_p(N)$ includes the second set of rotation parameters, which are used in the algorithm. Therefore, in order to complete our derivation, we need a formula for updating these rotation parameters. In the following, it will be shown that such a formula employs the vector $\boldsymbol{g}_p(N+1)$, which is obtained by (22) and (19).

It is already known [22] that the orthogonal matrix, which updates $\tilde{R}_p(N-1)$, also time updates $\tilde{R}_p^{-T}(N-1)$. More specifically, from [22, Th. 4 and Lemma 5] and taking into account the results of [11] (where a forgetting factor $\lambda \neq 1$ is considered), we can write

$$\hat{Q}_p(N)\begin{bmatrix}\lambda^{-1/2}\tilde{R}_p^{-T}(N-1)\\ \mathbf{0}^T\end{bmatrix} = \begin{bmatrix}\tilde{R}_p^{-T}(N)\\ \tilde{\boldsymbol{w}}_p^T(N)\end{bmatrix} \tag{27}$$

$$\hat{Q}_p(N)\begin{bmatrix}-\boldsymbol{g}_p(N)\\ 1\end{bmatrix} = \begin{bmatrix}\mathbf{0}\\ \delta_p(N)\end{bmatrix} \tag{28}$$

where

$$\tilde{\boldsymbol{w}}_p(N) = -\frac{\tilde{R}_p^{-1}(N-1)\boldsymbol{g}_p(N)}{\sqrt{\lambda}\delta_p(N)} \tag{29}$$

and

$$\delta_p(N) = \sqrt{1 + \boldsymbol{g}_p^T(N)\boldsymbol{g}_p(N)}. \tag{30}$$

Note that $\tilde{\boldsymbol{w}}_p(N)$ is a scaled version of the Kalman gain vector. To be more precise, if $\hat{\boldsymbol{w}}_p(N)$ stands for the Kalman gain, then [22]

$$\tilde{\boldsymbol{w}}_p(N) = -\hat{\boldsymbol{w}}_p(N)\delta_p(N). \tag{31}$$

Moreover, from (30) and (9), we get

$$\delta_p(N) = \sqrt{1 + \lambda^{-1}\boldsymbol{u}_p^T(N)\tilde{R}_p^{-1}(N-1)\tilde{R}_p^{-T}(N-1)\boldsymbol{u}_p(N)}$$

TABLE II
COMPARISON OF COMPLEXITIES OF FAST ROTATION-BASED ALGORITHMS

Algorithm	Additions	Multiplications	SQRT/Divisions
QR-Lattice	$8p$	$27p+1$	$6p$
Fast QRD	$8p+2$	$20p+3$	$6p+4$
New	$8p+1$	$19p+3$	$7p+3$

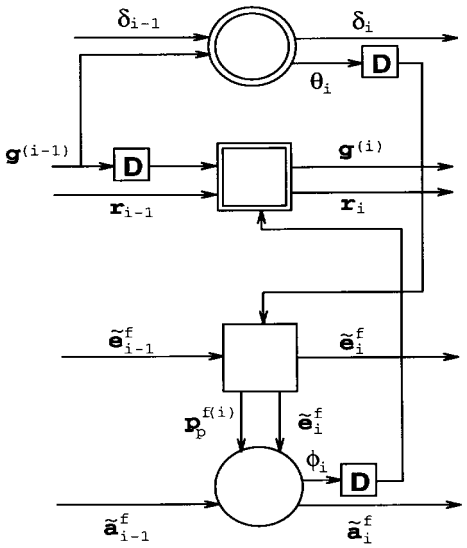

Fig. 2. ith stage of the prediction section of the lattice type algorithm.

or from (8)

$$\delta_p(N) = \frac{1}{\tilde{a}_p(N)} \tag{32}$$

that is, $\delta_p(N)$ equals the inverse of the angle normalized variable.

The rotation angles of $\hat{Q}_p(N+1)$ can be calculated by rewriting (28) at time $N+1$. Indeed

$$\hat{Q}_p(N+1)\begin{bmatrix} -\boldsymbol{g}_p(N+1) \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \delta_p(N+1) \end{bmatrix}. \tag{33}$$

Each rotation matrix of $\hat{Q}_p(N+1)$ annihilates one element of $-\boldsymbol{g}_p(N+1)$ by rotating it against the last element of the vector in (33), which is initially 1. The procedure starts from the first element and proceeds downwards. This is so because the rotations of $\hat{Q}_p(N+1)$ are also used in the time update of $\tilde{R}_p(N)$ [(2) written for time $N+1$]. As a result, the corresponding rotation matrices must be of the form given in (3) and (4). Due to the nesting property of $\boldsymbol{g}_p(N+1)$, it is clear that during the rotation process described in (33), we generate $\delta_n(N+1)$ for $n = 1, 2, \cdots, p$. Thus, according to (32), we essentially obtain the angle normalized variables for all orders at time $N+1$.

Equations (19), (22), (24)–(26), and (33) compose the prediction part of our algorithm. However, the *a priori* error at time $N+1$ must be calculated. This is accomplished in the filtering part of the algorithm. Indeed, (5) at time $N+1$ takes the form

$$\hat{Q}_p(N+1)\begin{bmatrix} \lambda^{1/2}\boldsymbol{p}_p(N) \\ y(N+1) \end{bmatrix} = \begin{bmatrix} \boldsymbol{p}_p(N+1) \\ \tilde{e}_p(N+1) \end{bmatrix}. \tag{34}$$

The *a priori* error is then given according to (7) and (32) as

$$e_p(N+1) = \tilde{e}_p(N+1)\delta_p(N+1). \tag{35}$$

Note that having available $\delta_n(N+1)$ for $n = 1, 2, \cdots, p$, we can similarly calculate the *a priori* errors of all orders.

The algorithm described so far is summarized in Table I. The complexity of the algorithm is shown in Table II. Its complexity is similar to that of the fast QRD algorithm of [1] and [18]–[21] (see Table II). However, there is a distinct advantage. Note that steps 1 and 2 of Table I can be performed

TABLE III
NEW LATTICE TYPE ALGORITHM

$r_0(N+1) = g^{(0)}(N+1) = \frac{u(N+1)}{\sqrt{\lambda}\tilde{a}_0^f(N)}$;

$\tilde{e}_0^f(N+1) = u(N+1)$;

$\tilde{a}_0^f(N+1) = \sqrt{\lambda[\tilde{a}_0^f(N)]^2 + [\tilde{e}_0^f(N+1)]^2}$;

$\delta_0(N+1) = 1$;

$\tilde{e}_0(N+1) = y(N+1)$;

for $i = 1 : p$,

$g^{(i)}(N+1) = \frac{g^{(i-1)}(N) - \sin(\phi_i(N))r_{i-1}(N+1)}{\cos(\phi_i(N))}$;

$r_i(N+1) = \cos(\phi_i(N))r_{i-1}(N+1) - \sin(\phi_i(N))g^{(i)}(N+1)$;

$\mathbf{p}_p^{f(i)}(N+1) = \lambda^{1/2}\cos\theta_i(N)\mathbf{p}_p^{f(i)}(N) + \sin\theta_i(N)\tilde{e}_{i-1}^f(N+1)$;

$\tilde{e}_i^f(N+1) = \cos\theta_i(N)\tilde{e}_{i-1}^f(N+1) - \lambda^{1/2}\sin\theta_i(N)\mathbf{p}_p^{f(i)}(N)$;

$\tilde{a}_i^f(N+1) = \sqrt{\lambda[\tilde{a}_i^f(N)]^2 + [\tilde{e}_i^f(N+1)]^2}$;

$\cos\phi_i(N+1) = \frac{\tilde{a}_i^f(N+1)}{\tilde{a}_{i-1}^f(N+1)}$;

$\sin\phi_i(N+1) = \frac{\mathbf{p}_p^{f(i)}(N+1)}{\tilde{a}_{i-1}^f(N+1)}$;

$\delta_i(N+1) = \sqrt{\delta_{i-1}^2(N+1) + [g^{(i-1)}(N+1)]^2}$;

$\cos\theta_i(N+1) = \frac{\delta_{i-1}(N+1)}{\delta_i(N+1)}$;

$\sin\theta_i(N+1) = \frac{g^{(i-1)}(N+1)}{\delta_i(N+1)}$;

$\mathbf{p}_p^{(i)}(N+1) = \lambda^{1/2}\cos\theta_i(N+1)\mathbf{p}_p^{(i)}(N) + \sin\theta_i(N+1)\tilde{e}_{i-1}(N+1)$;

$\tilde{e}_i(N+1) = \cos\theta_i(N+1)\tilde{e}_{i-1}(N+1) - \lambda^{1/2}\sin\theta_i(N+1)\mathbf{p}_p^{(i)}(N)$;

end;

$e_p(N+1) = \tilde{e}_p(N+1)\delta_p(N+1)$;

Initialization

$\mathbf{g}_p(0) = \mathbf{0}$, $\mathbf{p}_p^f(0) = \mathbf{0}$

$cos\theta_i(0) = 1$, $cos\phi_i(0) = 1$, $i = 1, 2, \ldots, p$

$\tilde{a}_i^f(0) = \sqrt{\lambda^p\mu}$, $i = 0, 1, \ldots, p$

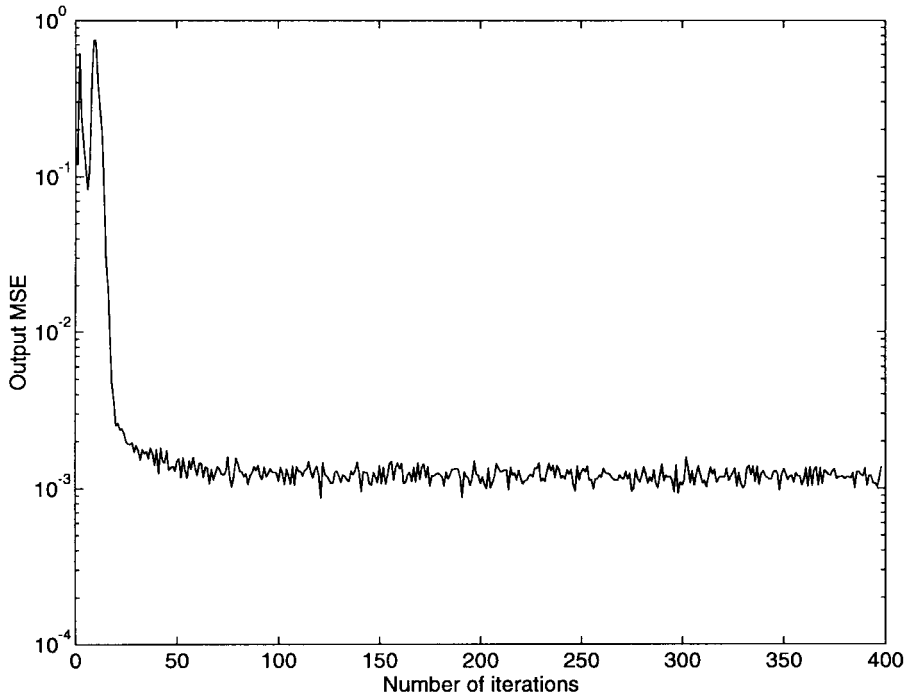

Fig. 3. Initial convergence curves.

concurrently. The same holds for steps 3 and 4 of Table I. Thus, by using two sets of DSP's, the execution time is almost halved. This is not possible with the algorithms of [17]–[21], which are sequential for each time iteration.

The algorithm of Table I is a fixed-order algorithm. As we can easily observe from Table I, the execution of step 3 starts after step 2 has been completed and $\tilde{a}_p^f(N+1)$ has been calculated. Then, the loop of step 3 goes backward in order, and this does not comply with the basic "pipeline" concept of a lattice structure. However, steps 2 and 3 of the algorithm can be combined if (26) is adopted for the calculation of the forward energies of all orders. Such a modification leads to a new lattice-type algorithm with the same complexity as our fixed-order scheme. Furthermore, the new lattice-type scheme includes orthogonal rotations only. The new lattice algorithm is shown in Table III. Compared with its previously derived counterparts [13], [14], [16], the new lattice algorithm has a substantially lower complexity (Table II). One lattice stage is depicted in Fig. 2.

The initialization of the new algorithms is based on the soft-constraint approach. More specifically, we make the following assumption concerning the input signal

$$u(-p) = \mu^{1/2}$$

where μ is a small positive scalar. Under this assumption, all initial conditions that appear at the bottom of Tables I and III are easily obtained.

IV. Relation to Other Fast QRD Algorithms

In this section, we exploit the connection of the new algorithms with previously derived fast QRD algorithms [1], [17]–[21]. We further show that the methodology developed in Section III can be applied for deriving both classes of fast QRD schemes. Let us consider the rotation matrix $\hat{Q}_p(N)$ written in block form [1]

$$\hat{Q}_p(N) = \begin{bmatrix} \Sigma_p(N) & \boldsymbol{q}_p(N) \\ \boldsymbol{\sigma}_p^T(N) & \tilde{a}_p(N) \end{bmatrix}$$

where $\tilde{a}_p(N)$ is the angle-normalized variable, and $\Sigma_p(N)$ stands for the upper left $p \times p$ part of $\hat{Q}_p(N)$. From (2), (27), and the orthogonality of $\hat{Q}_p(N)$, the remaining blocks of $\hat{Q}_p(N)$ can be expressed as

$$\Sigma_p(N) = \lambda^{1/2}\tilde{R}_p^{-T}(N)\tilde{R}_p^T(N-1) \tag{36}$$

$$\boldsymbol{q}_p(N) = \tilde{R}_p^{-T}(N)\boldsymbol{u}_p(N) \tag{37}$$

$$\boldsymbol{\sigma}_p(N) = -\lambda^{-1/2}\tilde{R}_p^{-T}(N-1)\boldsymbol{u}_p(N)\tilde{a}_p(N). \tag{38}$$

Note that the new algorithms presented in Section III are based on the update of $\boldsymbol{g}_p(N)$, which is a scaled version of $\boldsymbol{\sigma}_p(N)$. It is really interesting that the already existing fast QRD schemes [17]–[21] essentially stem from the time update of $\boldsymbol{q}_p(N)$. Due to the form of $\boldsymbol{q}_p(N)$ (37), the approach introduced in Section III can also be applied, and expressions similar to (19) and (22) can be obtained for the update of $\boldsymbol{q}_p(N)$. In addition to this, formulae equivalent to (21) and (23) can be directly derived, which now involve the *a posteriori* backward and forward errors, in contrast with (21) and (23), which involve the *a priori* quantities. Specifically, the elements of $\boldsymbol{q}_{p+1}(N+1)$ can be expressed as

$$q^{(i)}(N+1) = \frac{\varepsilon_i^b(N+1)}{\tilde{a}_i^b(N+1)} \qquad i = 0, 1, \cdots, p$$

whereas the new quantities $r_i'(N+1)$ corresponding to $r_i(N+1)$ satisfy

$$r_i'(N+1) = \frac{\varepsilon_i^f(N+1)}{\tilde{a}_i^f(N+1)} \qquad i = 0, 1, \cdots, p.$$

The fast QRD algorithms that appear in [1] and [18]–[21] include steps 2 and 3 [see (24) and (25)], whereas the rotation parameters of $\hat{Q}_p(N+1)$ can also be obtained from $\boldsymbol{q}_p(N+1)$ (although in a backward manner). It must be emphasized that the methodology that appears so far in the literature ([1], [17]–[20]) is quite different. Time and order update formulas of the factor Q_p are initially derived. Since the vector $\boldsymbol{q}_p$ is contained in the last row of Q_p [1], pinning vectors $\boldsymbol{\pi} = [0, 0, \cdots, 0, 1]^T$ are then applied in order to extract the update expressions for $\boldsymbol{q}_p$.

V. Simulations

In order to verify the validity of the derived algorithms, a system identification problem was considered. The unknown FIR system was of order 10, the SNR $= 30$ dB, the forgetting factor $\lambda = 0.98$, and the initialization parameter $\mu = 0.01$. Fig. 3 shows the obtained error convergence curves. Three curves are overlaid, although they are not distinguished. Two correspond to the novel algorithms developed in Section III and the third to the fast QRD algorithm of [21]. The curves are the average of 200 realizations. Note that experiments with up to 500 000 iterations were run with no indication of numerical stability problems for the new direct error computing algorithms.

VI. Conclusions

In this paper, two new fast QRD algorithms are derived following a novel approach. The new approach is based on the efficient time and order updates of a particular vector quantity that is basically the state vector of the equivalent state space description of the algorithmic process. The first algorithm is a fixed-order QRD scheme for direct error computation with enhanced parallelism. A modification of the scheme leads to an order recursive lattice-type QR algorithm with lower complexity compared with previously derived QRD lattice algorithms. It is shown that the methodology proposed in this paper can easily be adopted for the development of already existing fast QRD schemes.

References

[1] N. Kalouptsidis and S. Theodoridis, Eds., *Adaptive System Identification and Signal Processing Algorithms*. Engelwood Cliffs, NJ: Prentice-Hall, 1993.

[2] S. Haykin, *Adaptive Filter Theory*. Engelwood Cliffs, NJ: Prentice-Hall, 1991.

[3] L. Ljung, M. Morf, and D. Falconer, "Fast calculation of gain matrices for recursive estimation sources," *Int. J. Contr.*, vol. 27, pp. 1–19, 1978.

[4] G. Carayiannis, D. G. Manolakis, and N. Kalouptsidis, "A fast sequential algorithm for least squares filtering and prediction," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-31, pp. 1394–1402, Dec. 1983.

[5] J. M. Cioffi and T. Kailath, "Fast RLS transversal filters for adaptive filtering," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 304–337, Apr. 1984.

[6] B. Friedlander, "Lattice filters for adaptive processing," *Proc. IEEE*, vol. 70, pp. 829–867, Aug. 1982.

[7] F. Ling, D. Manolakis, and J. G. Proakis, "Numerically robust least-squares lattice-ladder algorithms with direct updating of the reflection coefficients," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 837–845, Aug. 1986.

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

[9] J. G. McWhirter, "Recursive least squares minimization using a systolic array," *Proc. SPIE Int. Soc. Opt. Eng.*, vol. 431, pp. 105–112, 1983.

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

[11] S. T. Alexander and A. L. Ghirnikar, "A method for recursive least squares adaptive filtering based upon an inverse QR decomposition," *IEEE Trans. Signal Processing*, vol. 41, pp. 20–30, Jan. 1993.

[12] A. L. Ghirnikar, S. T. Alexander, and R. J. Plemmons, "A parallel implementation of the inverse QR adaptive filter," *Comput. Elec. Eng.*, vol. 18, no. 3/4, pp. 291–300, 1992.

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

[14] I. K. Proudler, J. G. McWhirter, and T. J. Shepherd, "Computationally efficient, QR decomposition approach to least squares adaptive filtering," *Proc. Inst. Elect. Eng.*, vol. 138, pt. F, pp. 341–353, Aug. 1991.

[15] P. S. Lewis, "QR-based algorithms for multichannel adaptive least squares lattice filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, pp. 421–432, Mar. 1990.

[16] B. Yang and J. F. Bohme, "Rotation-based RLS algorithms: Unified derivations, numerical properties, and parallel implications," *IEEE Trans. Signal Processing*, vol. 40, pp. 1151–1167, May 1992.

[17] J. M. Cioffi, "The fast adaptive ROTOR's RLS algorithm," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, pp. 631–653, Apr. 1990.

[18] I. K. Proudler, J. G. McWhirter, and T. J. Shepherd, "Fast QRD-based algorithms for least squares linear prediction," in *Proc. IMA Conf. Math. Signal Process.*, Warwick, U.K., Dec. 12–15, 1988.

[19] M. G. Bellanger, "The FLS-QR algorithm for adaptive filtering," *Signal Process.*, vol. 17, pp. 291–304, 1989.

[20] ——, "A survey of QR based fast least squares adaptive filters: From principles to realization," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Toronto, Ont., Canada, 1991, pp. 1833–1836.

[21] 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, Apr. 1991.

[22] C. T. Pan and R. J. Plemmons, "Least squares modifications with inverse factorizations: Parallel implications," *J. Comput. Appl. Math.*, vol. 27, pp. 109–127, 1989.

[23] G. H. Golub and C. Van Loan, *Matrix Computations*, 2nd ed. Baltimore, MD: Johns Hopkins Univ. Press, 1989.

Athanasios A. Rontogiannis was born in Athens, Greece, on June 16, 1968. He received the diploma in electrical and computer engineering from the National Technical University of Athens in 1991, the M.A.Sc degree in electrical and computer engineering from the University of Victoria, Victoria, B.C., Canada, in 1993, and the Ph.D degree in signal processing from the Department of Informatics, University of Athens, in 1997.

Since March 1997, he has been with the Greek Air Force. From November 1994 to March 1997, he was a recipient of a Scholarship from the State Scholarship Foundation for the completion of his Ph.D. degree. His research interests are focused on adaptive filtering algorithms and their application to channel equalization and echo cancellation schemes.

Sergios Theodoridis was born in Piraeus, Greece, on December 17, 1951. He received the B.Sc. degree with honors in physics from the University of Athens, Athens, Greece, in 1973 and the M.Sc. and Ph.D degrees in communications and signal processing, both from the University of Birmingham, Birmingham, U.K., in 1975 and 1978 respectively.

From 1978 to 1981, he was a Research Fellow in the Department of Electronics and Electrical Engineering, University of Birmingham. From 1981 to 1983, he was with the Greek Army. From 1984 to 1995, he was with the Department of Computer Engineering, University of Patras, Patras, Greece. Since 1995, he has been with the Department of Informatics, University of Athens. His current research interests are focused on the fields of digital signal processing, communications, and pattern recognition.