ELSEVIER

Signal Processing 52 (1996) 35–47

SIGNAL PROCESSING

On inverse factorization adaptive least-squares algorithms

Athanasios A. Rontogiannis[1], Sergios Theodoridis*

Department of Informatics, Division of Communications and Signal Processing, Panepistimioupolis, TYPA Buildings, University of Athens, GR-157 71 Zografou, Greece

Received 27 April 1995; revised 11 October 1995 and 13 March 1996

Abstract

This paper presents an adaptive exponentially weighted algorithm for least-squares (LS) system identification. The algorithm updates an inverse 'square root' factor of the input data correlation matrix, by applying numerically robust orthogonal Householder transformations. The scheme avoids, almost entirely, costly square roots and divisions (present in other numerically well-behaved adaptive LS schemes) and provides directly the estimates of the unknown system coefficients. Furthermore, it offers enhanced parallelism, which leads to efficient implementations. A square array architecture for implementing the new algorithm, which comprises simple operating blocks, is described. The numerically robust behavior of the algorithm is demonstrated through simulations. The algorithm is compared to the recently developed inverse factorization QR scheme (Alexander and Ghirnikar, 1993), in terms of computational complexity, parallel potential and numerical properties.

Zusammenfassung

Ein adaptiver, exponentiell gewichteter Algorithmus zur Systemidentifikation auf der Basis kleinster Quadrate (least squares, LS) wird vorgestellt. Er führt einen inversen "Quadratwurzel"-Faktor der Eingangsdaten-Korrelationsmatrix nach, indem er numerisch robuste orthogonale Householder-Transformationen anwendet. Das Verfahren vermeidet nahezu vollkommen aufwendige Wurzel- und Divisionsschritte (wie sie in anderen numerisch gutmütigen adaptiven LS-Methoden auftreten) und liefert unmittelbar die Schätzwerte der unbekannten Systemkoeffizienten. Darüber hinaus bietet es einen verbesserten Parallelismus, der zu effizienten Realisierungen führt. Es wird eine quadratische Array-Architektur zur Realisierung des neuen Algorithmus beschrieben, welche einfache Funktionsblöcke umfaßt. Das numerisch robuste Verhalten des Algorithmus wird anhand von Simulationen demonstriert. Der Algorithmus wird mit dem kürzlich entwickelten inversen Faktorisierungs-QR-Schema (Alexander und Ghirnikar, 1993) im Hinblick auf Rechenaufwand, Parallelität und numerische Eigenschaften verglichen.

Résumé

Cet article présente un algorithme adaptatif avec pondération exponentielle pour l'identification de systèmes au sens des moindres carrés (LS). L'algorithme met à jour un facteur en 'racine carrée' inverse de la matrice de corrélation des données d'entrée, en appliquant des transformations orthogonales de Householder numériquement robustes. La méthode évite presque entièrement l'emploi de coûteuses divisions et racines carrées (que l'on rencontre dans d'autres

*Corresponding author. Fax: 301 72 28 981; e-mail: stheodor@zeus.di.uoa.ariadne-t.gr.

[1]E-mail: tronto@zeus.di.uoa.ariadne-t.gr.

PII S0165-1684(96)00060-6

méthodes LS adaptatives et numériquement robustes) et fournit directement les estimées des coefficients du système à identifier. De plus, il offre un parallélisme avancé menant à des implémentations efficaces. On décrit une architecture de type matrice carrée comprenant de simples blocs opérateurs pour implémenter le nouvel algorithme. La robustesse numérique de l'algorithme est mise en évidence par des simulations. L'algorithme est comparé à la méthode QR de factorisation inverse développée récemment (Alexander et Ghirnikar, 1993), en terme de complexité de calcul, de potentiel de parallélisme et de propriétés numériques.

Keywords: Inverse factorization algorithms; Orthogonal Householder transformations; Parallel algorithms; Numerical stability

1. Introduction

Adaptive least-squares algorithms for system identification [9, 10] are popular due to their fast converging properties and are used in a variety of applications, such as channel equalization, echo cancelation, spectral analysis, control, to name but a few. Among the various efficiency issues characterizing the performance of an algorithm, those of parallelism and numerical robustness are of particular importance, especially in applications where medium to long filter lengths are required. Sometimes it may be preferable to use an algorithm of higher computational complexity, but with good numerical error properties and high parallelism, since this may allow its implementation with shorter word lengths and fixed point arithmetic on an array architecture. This has led to the development of a class of adaptive algorithms based on the QR factorization of the input data matrix.

Three methods can be used for the QR decomposition (QRD) of the input data matrix, namely the Givens method, the modified Gram–Schmidt and the Householder methods [8]. All three of them are known to be numerically robust, although several authors have claimed the superiority of the Householder method in limiting the accumulation of round-off errors [20, 15, 27]. Recursive QRD algorithms that are based on the modified Gram–Schmidt and the Givens approach have already been developed [6, 16, 13]. These algorithms are of $O(p^2)$ complexity per time iteration, with p being the order of the system. They are 'square-root' schemes, which update the Cholesky factor of the input data correlation matrix, and can be efficiently implemented on triangular systolic arrays. Furthermore, these schemes can provide the modeling error directly without it being necessary to compute explicitly the estimates of the transversal parameters of the unknown FIR system. If, however, these estimates are required, a highly serial backsubstitution step must be employed. This step cannot be pipelined with the update step, thus limiting the performance of the algorithm. Schemes which overcome the backsubstitution step have been suggested, but at the expense of realization complexity [24, 26].

An alternative $O(p^2)$ RLS scheme was recently introduced and it is based on the update of the inverse Cholesky factor of the data correlation matrix via the Givens rotations approach [1, 19]. This inverse QR scheme circumvents the backsubstitution problem and can provide directly the unknown coefficient estimates. The algorithm can also be implemented very efficiently on triangular systolic arrays [7, 17]. Fast schemes which compute the modeling error with $O(p)$ computational complexity, based on Givens rotations, have also been derived for the above algorithms [5, 12, 21, 22].

The use of Householder transformations in the LS framework has so far been restricted in block type problems [20, 25, 15]. This is a direct consequence of the nature of a Householder matrix, which is usually applied to annul block of elements in vectors or matrices. Thus, Householder transformations have been used either in simple block LS updates (downdates) [20, 25] or in block RLS schemes [15]. Simple 2×2 Householder reflections have also appeared in place of Givens rotations in the QRD–RLS problem [14].

This paper presents an $O(p^2)$ RLS algorithm which springs from an inverse square root factor of the data correlation matrix and incorporates numerically robust block orthogonal[1] Householder transformations. The algorithm updates a square factor, instead of a triangular one, and computes the filter parameters directly, without involving matrix inversions or backsubstitution steps. The proposed scheme avoids almost entirely divisions and square roots (present in algorithms based on Givens rotations) and employs low-cost additions and multiplications. Furthermore, it exhibits high degree of parallelism, which makes it amenable to efficient implementation. An array architecture which efficiently implements the new scheme is described. This architecture is designed to take full advantage of the algorithm's parallelism, and it comprises very simple operating blocks. We must state that the derived algorithm consists of matrix–vector and vector–vector operations, which make it suitable for efficient implementations on vector processors.

The paper is organized as follows. The QR decomposition approach to the RLS problem via Givens rotations is described in Section 2. Both the QR and the inverse QR algorithms are discussed. Orthogonal Householder transformations are then briefly reviewed and a Householder RLS (HRLS) algorithm is developed in Section 3. Section 4 discusses parallel implications of the scheme derived and proposes a square array for its efficient implementation. The performance of the algorithm is illustrated with simulations in Section 5 while Section 6 concludes this work. For clarity of presentation real signals are considered throughout the paper. We mostly adopt the notation that appears in [10].

2. QR decomposition and the RLS 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

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

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

$$\boldsymbol{\varepsilon}^{\mathrm{T}}(N) = [\varepsilon(1), \varepsilon(2), \ldots, \varepsilon(N)],$$

$$\boldsymbol{c}^{\mathrm{T}}(N) = [c_1(N), c_2(N), \ldots, c_p(N)],$$

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

The quantity $\eta(n)$ in the figure stands for the measurement noise. It is well known that the LS solution $\boldsymbol{c}(N)$ is obtained from the equation

$$\tilde{R}(N)\boldsymbol{c}(N) = \boldsymbol{p}(N), \tag{2}$$

where the upper triangular matrix $\tilde{R}(N)$ stands for the Cholesky factor of the input data correlation matrix and $\boldsymbol{p}(N)$ corresponds to the rotated reference vector [10]. In most real time applications a recursive least-squares algorithm is required, in which the solution of the least-squares problem at time N is obtained from the solution of the previous time instant. Indeed, it has been shown that [10]

$$\hat{Q}(N)\begin{bmatrix} \lambda^{1/2}\tilde{R}(N-1) \\ \boldsymbol{u}^{\mathrm{T}}(N) \end{bmatrix} = \begin{bmatrix} \tilde{R}(N) \\ \boldsymbol{0}^{\mathrm{T}} \end{bmatrix}. \tag{3}$$

$\hat{Q}(N)$ results from a sequence of basic Givens rotations which successively annihilate the elements of $\boldsymbol{u}^{\mathrm{T}}(N)$ against $\lambda^{1/2}\tilde{R}(N-1)$. At the same time it is most interesting that [21, p. 882]

$$\hat{Q}(N)\begin{bmatrix} \lambda^{1/2}\boldsymbol{p}(N-1) \\ y(N) \end{bmatrix} = \begin{bmatrix} \boldsymbol{p}(N) \\ \tilde{e}(N) \end{bmatrix}, \tag{4}$$

where

$$\tilde{e}(N) = \operatorname{sgn}(\varepsilon(N))\sqrt{\varepsilon(N)e(N)}; \tag{5}$$

$e(N)$ is the a priori error expressed as

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

After updating $\tilde{R}(N-1)$ and $\boldsymbol{p}(N-1)$ from (3) and (4), the coefficients' vector $\boldsymbol{c}(N)$ can be computed from (2) with backsubstitution. This highly serial

[1] According to [8, p. 70], a matrix $Q \in \mathbb{R}^{m \times m}$ is said to be orthogonal when $Q^{\mathrm{T}}Q = I$. That is, the columns of Q form an orthonormal basis for $\mathbb{R}^m$.

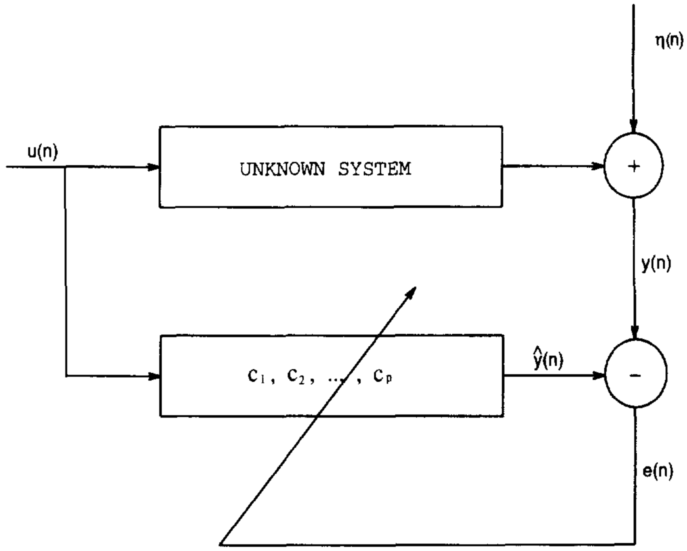

Fig. 1. System identification problem.

backsubstitution step has always been a drawback in the performance of the QRD–RLS algorithm, especially when the algorithm is implemented on an upper triangular systolic architecture [1]. One way to overcome this problem is to update $\tilde{R}^{-1}$ instead of $\tilde{R}$. It can be shown that the same rotations used to update $\tilde{R}$ can also be applied for updating $\tilde{R}^{-\mathrm{T}}$ [18, 19],

$$\hat{Q}(N)\begin{bmatrix}\lambda^{-1/2}\tilde{R}^{-\mathrm{T}}(N-1)\\ \mathbf{0}^{\mathrm{T}}\end{bmatrix}=\begin{bmatrix}\tilde{R}^{-\mathrm{T}}(N)\\ \boldsymbol{w}^{\mathrm{T}}(N)\end{bmatrix}. \tag{7}$$

$\boldsymbol{w}(N)$ is a scaled version of the Kalman gain vector. Indeed, if we multiply the transpose of the matrix on the left-hand side (LHS) of (3) with the matrix on the LHS of (7) the identity matrix results. The update of $\tilde{R}^{-\mathrm{T}}$ in (7) is easily verified since the same must hold for the matrices in the right-hand side of the corresponding expressions. If now

$$\boldsymbol{g}(N)=\frac{\tilde{R}^{-\mathrm{T}}(N-1)\boldsymbol{u}(N)}{\sqrt{\lambda}}, \tag{8}$$

then the rotation parameters of $\hat{Q}(N)$ required in (7) can be obtained by successively annulling the first p elements of the vector $[-\boldsymbol{g}^{\mathrm{T}}(N), 1]^{\mathrm{T}}$ as follows:

$$\hat{Q}(N)\begin{bmatrix}-\boldsymbol{g}(N)\\ 1\end{bmatrix}=\begin{bmatrix}\mathbf{0}\\ \delta(N)\end{bmatrix}. \tag{9}$$

Since $\hat{Q}(N)$ is orthogonal the last equation implies that

$$\delta(N)=\sqrt{1+\boldsymbol{g}^{\mathrm{T}}(N)\boldsymbol{g}(N)}. \tag{10}$$

The backsubstitution step can now be circumvented if we adopt the following formula for the update of the coefficients vector $\boldsymbol{c}(N)$ [19]:

$$\boldsymbol{c}(N)=\boldsymbol{c}(N-1)-\boldsymbol{w}(N)\frac{e(N)}{\delta(N)}. \tag{11}$$

Finally, we must note that if we express $\hat{Q}(N)$ in block form (see also below), then from (3), (7) and the orthogonality of $\hat{Q}(N)$, $\boldsymbol{w}(N)$ can be written as

$$\boldsymbol{w}(N)=-\frac{\tilde{R}^{-1}(N-1)\boldsymbol{g}(N)}{\sqrt{\lambda}\delta(N)}. \tag{12}$$

2.1. The structure of $\hat{Q}(N)$

Some aspects of the structure of $\hat{Q}(N)$ will be useful in the analysis to follow. Let us rewrite $\hat{Q}(N)$ in the block form

$$\hat{Q}(N)=\begin{bmatrix}\Sigma(N) & \boldsymbol{q}(N)\\ \boldsymbol{\sigma}^{\mathrm{T}}(N) & \tilde{a}(N)\end{bmatrix}, \tag{13}$$

where $\Sigma(N)$ is the upper left $p \times p$ part of $\hat{Q}(N)$. If we substitute Eq. (13) in (3) and (7) and take into consideration the fact that $\hat{Q}(N)$ is orthogonal, the blocks of $\hat{Q}(N)$ can easily be expressed as (see also [10])

$$\Sigma(N) = \lambda^{1/2}\tilde{R}^{-\mathrm{T}}(N)\tilde{R}^{\mathrm{T}}(N-1), \tag{14}$$

$$\boldsymbol{\sigma}(N) = -\frac{\boldsymbol{g}(N)}{\delta(N)}, \tag{15}$$

$$\boldsymbol{q}(N) = \tilde{R}^{-\mathrm{T}}(N)\boldsymbol{u}(N), \tag{16}$$

$$\tilde{a}(N) = 1/\delta(N). \tag{17}$$

In the following sections we develop a 'square root' algorithm that solves the RLS problem. The algorithm updates and applies a square factor, in contrast to already known schemes [6, 1] which update triangular factors. This fact necessitates the use of numerically robust orthogonal Householder transformations. The proposed scheme computes the filter taps directly and avoids almost entirely divisions and square roots. It includes two divisions and one square root, in other words the number of these costly operations is independent of p, the system's order. Moreover, it offers enhanced parallelism, which makes it amenable to efficient implementation on an array architecture or a vector processor machine.

3. Derivation of the Householder RLS (HRLS) algorithm

We begin this section with a brief review of the definition and main properties of orthogonal Householder matrices. We then proceed with the derivation of the HRLS algorithm.

3.1. Orthogonal Householder transformations

An orthogonal Householder matrix has the following special form [8, 25]:

$$P = I - 2\frac{\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}}{\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v}}. \tag{18}$$

It is obvious from (18) that matrix P is also symmetric. Householder transformations are often used to annul block of elements in matrices or vectors by appropriately selecting the Householder vector $\boldsymbol{v}$ in (18). More specifically, if $\boldsymbol{x}$ is a nonzero vector and $\boldsymbol{e}_i$ stands for the unit vector with 1 in the ith position, then it can be shown [8] that when

$$\boldsymbol{v} = \boldsymbol{x} \pm \|\boldsymbol{x}\|\boldsymbol{e}_i, \tag{19}$$

then

$$P\boldsymbol{x} = \mp \|\boldsymbol{x}\|\boldsymbol{e}_i. \tag{20}$$

Note the sign difference in Eqs. (19) and (20). It is worth emphasizing that vectors $\boldsymbol{v}$ and $\boldsymbol{x}$ are identical except for the ith element. In most cases, explicit formation of the Householder matrix from (18) is not required. Instead, we usually aim to take advantage of the matrice's special structure, which is also the case in our analysis.

3.2. The HRLS algorithm

Let us define the square matrix $A(N)$ as the matrix product

$$A(N) = T(N)\tilde{R}(N), \tag{21}$$

where $T(N)$ corresponds to an orthogonal $p \times p$ matrix for every N. The specific recursive expression of $T(N)$ will be derived as we proceed. It is recognized from (21) that $A(N)$ represents a square root factor (not triangular) of the input data correlation matrix. Indeed,

$$\tilde{R}^{\mathrm{T}}(N)\tilde{R}(N) = A^{\mathrm{T}}(N)A(N). \tag{22}$$

Due to the orthogonality of $T(N)$, matrices $A(N)$ and $\tilde{R}(N)$ have the same condition number. Therefore, the dynamic range of the RLS problem does not change if we update and use A ($A^{-\mathrm{T}}$) instead of $\tilde{R}$ ($\tilde{R}^{-\mathrm{T}}$). More specifically, in the new scheme the update of $A^{-\mathrm{T}}$ is carried out by applying orthogonal Householder transformations. We begin by defining the vector

$$\boldsymbol{k}(N) = \frac{A^{-\mathrm{T}}(N-1)\boldsymbol{u}(N)}{\sqrt{\lambda}} = T(N-1)\boldsymbol{g}(N). \tag{23}$$

Obviously, vectors $\boldsymbol{k}(N)$ and $\boldsymbol{g}(N)$ have identical Euclidean norms, which implies that

$$\delta(N) = \sqrt{1 + \boldsymbol{g}^{\mathrm{T}}(N)\boldsymbol{g}(N)} = \sqrt{1 + \boldsymbol{k}^{\mathrm{T}}(N)\boldsymbol{k}(N)}. \tag{24}$$

Let us now assume that $P(N)$ is a $(p+1)\times(p+1)$ orthogonal Householder matrix which when operates on $[\boldsymbol{k}(N), 1]^{\mathrm{T}}$ makes its first p elements vanish, i.e.

$$P(N)\begin{bmatrix} \boldsymbol{k}(N) \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ -\delta(N) \end{bmatrix}. \tag{25}$$

We will prove that $P(N)$ also satisfies the equation

$$P(N)\begin{bmatrix} \lambda^{-1/2}A^{-\mathrm{T}}(N-1) \\ \mathbf{0}^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} A^{-\mathrm{T}}(N) \\ \boldsymbol{w}^{\mathrm{T}}(N) \end{bmatrix}. \tag{26}$$

In other words, it will be shown that $P(N)$ not only updates $A^{-\mathrm{T}}(N-1)$, but also produces the gain vector $\boldsymbol{w}(N)$, which is needed in the calculation of the new filter coefficients. From the definition of the Householder matrix we see that, in order for $P(N)$ to satisfy (25), it must be of the form

$$P(N) = I - 2\frac{\boldsymbol{v}(N)\boldsymbol{v}^{\mathrm{T}}(N)}{\boldsymbol{v}^{\mathrm{T}}(N)\boldsymbol{v}(N)}, \tag{27}$$

where the Householder vector $\boldsymbol{v}(N)$ is given by

$$\boldsymbol{v}(N) = \begin{bmatrix} \boldsymbol{k}(N) \\ 1 + \delta(N) \end{bmatrix}. \tag{28}$$

The selection of the positive sign in the last element of $\boldsymbol{v}(N)$ aims to prevent from numerical problems, which can arise by division with a very small number in (27). If we now set

$$\beta(N) = \frac{2}{\boldsymbol{v}^{\mathrm{T}}(N)\boldsymbol{v}(N)}, \tag{29}$$

then combination of (24) and (28) easily results in

$$\beta(N) = \frac{1}{\delta(N)[1 + \delta(N)]}. \tag{30}$$

Furthermore, (27) is rewritten as

$$P(N) = I - \beta(N)\boldsymbol{v}(N)\boldsymbol{v}^{\mathrm{T}}(N). \tag{31}$$

Let us now return to (26) and compute the effect of the application of $P(N)$. We have

$$\begin{aligned} C(N) &= P(N)\begin{bmatrix} \lambda^{-1/2}A^{-\mathrm{T}}(N-1) \\ \mathbf{0}^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \lambda^{-1/2}A^{-\mathrm{T}}(N-1) \\ \mathbf{0}^{\mathrm{T}} \end{bmatrix} \\ &\quad - \beta(N)\begin{bmatrix} \boldsymbol{k}(N) \\ 1 + \delta(N) \end{bmatrix}\boldsymbol{v}^{\mathrm{T}}(N)\begin{bmatrix} \lambda^{-1/2}A^{-\mathrm{T}}(N-1) \\ \mathbf{0}^{\mathrm{T}} \end{bmatrix}. \end{aligned}$$

From (12), (21), (23) and (28) we obtain

$$\begin{aligned} C(N) &= \begin{bmatrix} \lambda^{-1/2}A^{-\mathrm{T}}(N-1) \\ \mathbf{0}^{\mathrm{T}} \end{bmatrix} \\ &\quad + \beta(N)\begin{bmatrix} \boldsymbol{k}(N) \\ 1 + \delta(N) \end{bmatrix}\boldsymbol{w}^{\mathrm{T}}(N)\delta(N). \end{aligned}$$

From (30) it is obvious that the last row of the resulting matrix equals $\boldsymbol{w}^{\mathrm{T}}(N)$. In order to complete our derivation we must show that the upper $p\times p$ block of the matrix, say $X(N)$, has the form $T(N)\tilde{R}^{-\mathrm{T}}(N)$, where $T(N)$ is an orthogonal matrix. Indeed from (12) $X(N)$ is written as

$$\begin{aligned} X(N) &= \lambda^{-1/2}A^{-\mathrm{T}}(N-1) \\ &\quad - \lambda^{-1/2}\beta(N)\boldsymbol{k}(N)\boldsymbol{g}^{\mathrm{T}}(N)\tilde{R}^{-\mathrm{T}}(N-1), \end{aligned}$$

and from (21) and (23)

$$\begin{aligned} X(N) &= T(N-1)[I - \beta(N)\boldsymbol{g}(N)\boldsymbol{g}^{\mathrm{T}}(N)]\lambda^{-1/2} \\ &\quad \times \tilde{R}^{-\mathrm{T}}(N-1) \\ &= T(N-1)[I - \beta(N)\boldsymbol{g}(N)\boldsymbol{g}^{\mathrm{T}}(N)]\lambda^{-1/2} \\ &\quad \times \tilde{R}^{-\mathrm{T}}(N-1)\tilde{R}^{\mathrm{T}}(N)\tilde{R}^{-\mathrm{T}}(N), \end{aligned}$$

and from (14)

$$\begin{aligned} X(N) &= T(N-1)[I - \beta(N)\boldsymbol{g}(N)\boldsymbol{g}^{\mathrm{T}}(N)] \\ &\quad \times \Sigma^{-1}(N)\tilde{R}^{-\mathrm{T}}(N). \end{aligned}$$

Proposition. *The $p\times p$ matrix*

$$\tilde{T}(N) = [I - \beta(N)\boldsymbol{g}(N)\boldsymbol{g}^{\mathrm{T}}(N)]\Sigma^{-1}(N) \tag{32}$$

is orthogonal.

Proof. It suffices to show that $\tilde{T}^{\mathrm{T}}(N)\tilde{T}(N) = I$. We have

$$\begin{aligned}\tilde{T}^{\mathrm{T}}(N)\tilde{T}(N) &= \Sigma^{-\mathrm{T}}(N)[I - \beta(N)g(N)g^{\mathrm{T}}(N)]^2\Sigma^{-1}(N)\\ &= \Sigma^{-\mathrm{T}}(N)[I - 2\beta(N)g(N)g^{\mathrm{T}}(N)\\ &\quad + \beta^2(N)\|g(N)\|^2 g(N)g^{\mathrm{T}}(N)]\Sigma^{-1}(N).\end{aligned}$$

From Eqs. (24) and (30), the last equation can be simplified as follows:

$$\tilde{T}^{\mathrm{T}}(N)\tilde{T}(N) = \Sigma^{-\mathrm{T}}(N)\left[I - \frac{g(N)g^{\mathrm{T}}(N)}{\delta^2(N)}\right]\Sigma^{-1}(N). \tag{33}$$

Eqs. (13) and (15) and the orthogonality of $\hat{Q}(N)$ imply that

$$\Sigma^{\mathrm{T}}(N)\Sigma(N) + \sigma(N)\sigma^{\mathrm{T}}(N) = I$$

$$\Leftrightarrow \quad \Sigma^{\mathrm{T}}(N)\Sigma(N) = I - \frac{g(N)g^{\mathrm{T}}(N)}{\delta^2(N)}. \tag{34}$$

The completion of the proof is now straightforward by substitution of (34) in (33). □

The validity of Eq. (26) is clear from the above derivation. Furthermore, the orthogonal matrix $T(N)$ is expressed according to the recursive relation

$$T(N) = T(N-1)\tilde{T}(N), \quad T(0) = I. \tag{35}$$

It is really interesting that matrix $P(N)$ also updates $A(N-1)$ as well as an orthogonal transformed version of vector $\boldsymbol{p}(N-1)$. This is proved in Appendix A, where expressions equivalent to (3) and (4) are derived.

The main steps of the HRLS algorithm are illustrated in Fig. 2. The figure depicts the generation of the Householder matrix $P(N)$ and the resulting recursion for $A^{-\mathrm{T}}$ and $\boldsymbol{w}$. As it was mentioned earlier, matrix $P(N)$ need not be computed explicitly. In such a case the HRLS algorithm is implemented as shown in Fig. 3. The soft-constrained approach [10] is adopted for the algorithm's initialization. Note that the filter taps are directly computed without the use of matrix inversions or backsubstitution steps. This is also a characteristic of the inverse QR algorithm, which is based on Givens rotations and is described in [1]. The complexities of these two algorithms are shown in Table 1. The complexity of the HRLS algorithm is higher with respect to additions and slightly higher with respect to multiplications. This was expected since the new scheme manipulates a square matrix instead of a triangular one which is the case in [1].

- $\mathbf{k}(N) = \lambda^{-1/2}A^{-T}(N-1)\mathbf{u}(N)$
- $P(N)\begin{bmatrix}\mathbf{k}(N)\\ 1\end{bmatrix} = \begin{bmatrix}\mathbf{0}\\ -\delta(N)\end{bmatrix}$
- $P(N)\begin{bmatrix}\lambda^{-1/2}A^{-T}(N-1)\\ \mathbf{0}^T\end{bmatrix} = \begin{bmatrix}A^{-T}(N)\\ \mathbf{w}^T(N)\end{bmatrix}$
- $\epsilon(N) = y(N) - \mathbf{c}^T(N-1)\mathbf{u}(N)$
- $\mathbf{c}(N) = \mathbf{c}(N-1) - \frac{\epsilon(N)}{\delta(N)}\mathbf{w}(N)$

Fig. 2. The main steps of the HRLS algorithm.

For $N = 1\ldots$ do,

1. $\mathbf{k}(N) = \lambda^{-1/2}A^{-T}(N-1)\mathbf{u}(N)$;
2. $\delta(N) = \sqrt{1 + \mathbf{k}^T(N)\mathbf{k}(N)}$;
3. $\beta(N) = \frac{1}{\delta(N)[1+\delta(N)]}$;
4. $\tilde{\mathbf{w}}(N) = A^{-1}(N-1)\mathbf{k}(N)$ $(= -\sqrt{\lambda}\delta(N)\mathbf{w}(N))$;
5. $\hat{\mathbf{w}}(N) = \lambda^{-1/2}\beta(N)\tilde{\mathbf{w}}(N)$;
6. $A^{-T}(N) = \lambda^{-1/2}A^{-T}(N-1) - \mathbf{k}(N)\hat{\mathbf{w}}^T(N)$;
7. $\epsilon(N) = y(N) - \mathbf{c}^T(N-1)\mathbf{u}(N)$;
8. $\mathbf{c}(N) = \mathbf{c}(N-1) + \frac{\epsilon(N)}{\sqrt{\lambda}\delta^2(N)}\tilde{\mathbf{w}}(N)$;

Initialization (Soft-constrained)

$\mathbf{c}(0) = \mathbf{0}$, $A^{-T}(0) = \frac{1}{\mu\lambda^p}diag[1, \lambda, \ldots, \lambda^{p-1}]$

Fig. 3. The new HRLS algorithm.

Table 1
Comparison of complexities of inverse QR [1] and HRLS

Algorithm	Add.'s	Mult.'s	Div.'s/ SQRT's
Inverse QR	$\frac{3}{2}p^2 + O(p)$	$\frac{7}{2}p^2 + O(p)$	$3p + 1$
HRLS	$3p^2 + O(p)$	$4p^2 + O(p)$	3

However, note that the HRLS algorithm avoids almost exclusively divisions and square roots while employing low-cost additions and multiplications, in a multiply–add fashion, which is desirable for efficient DSP implementations. On the other hand, the inverse QR scheme requires $O(p)$ divisions and square roots. The numerical properties of the HRLS algorithm are expected to be very favourable because of the use of numerical robust Householder reflections. Moreover, the derived scheme offers enhanced parallelism, as compared to [1], a notion which is further analyzed in the next section.

It is well known that the exponentially weighted QR and inverse QR algorithms are directly related to the square root information and covariance filters. The latter appear in the Kalman filtering and the unweighted linear LS estimation literature [18, 11, 2, 4, 3]. It turns out that the HRLS algorithm is, in the same way, related to the Potter's square root covariance scheme that was developed in the early 1960s [3]. Indeed, the QR, inverse QR and HRLS algorithms can also arise by adopting the equivalence relation between Kalman and RLS schemes as was recently presented in [23].

In our analysis, however, a different approach is followed. More specifically, the procedure we develop establishes and highlights the connection between the HRLS and the inverse QR schemes (both being covariance square root algorithms). In this way, insight can be gained concerning the internal quantities which appear in these two algorithms. Furthermore, this approach reveals the existence of the numerically robust orthogonal block Householder transformation that updates the algorithm derived (correspondingly a sequence of orthogonal Givens rotations is used in the inverse QR algorithm). A direct conclusion from the above observations is that both the HRLS and the inverse QR algorithms share two important properties as far as their numerical behavior is concerned: (a) they are square root type algorithms which results in reduced dynamic range of data and (b) they are both based on orthogonal transformations a fact which limits the accumulation of round-off errors. Therefore, both algorithms are appropriate candidates for numerically stable implementations.

4. Parallel potential of the algorithm

An array architecture for implementing the HRLS algorithm is depicted in Fig. 4 (for the case $p = 3$). At time N, the square cells of the array store and update the elements of matrix $A^{-\mathrm{T}}(N-1)$ (step 6). They also perform the inner product computations, required in the derivation of the vectors $\boldsymbol{k}(N)$ (step 1) and $\tilde{\boldsymbol{w}}(N)$ (step 4). The cyclic cells, at the bottom of the array, store and update the filter taps (step 8) and produce the a priori error $e(N)$ (step 7). The cyclic cells on the right of the array compute $\delta^2(N)$ and $\beta(N)$ (steps 2 and 3 of the algorithm). The emboldened cyclic cells simply transform the gain $\tilde{\boldsymbol{w}}(N)$ in the form $\lambda^{-1/2}\beta(N)\tilde{\boldsymbol{w}}(N)$, while the bottom right cell produces the quantity $e(N)/(\sqrt{\lambda}\delta^2(N))$, which is required in the calculation of $\boldsymbol{c}(N)$. The functionality of all the individual cells is shown in Fig. 4.

We must note that the elements of both vectors $\boldsymbol{k}(N)$ and $\tilde{\boldsymbol{w}}(N)$ are generated in parallel. As is illustrated in Fig. 4, the new input vector $\boldsymbol{u}(N)$ is applied in parallel to the row cells of $A^{-\mathrm{T}}(N-1)$ allowing for the simultaneous calculation of $k_i(N)$'s. This is accomplished by performing inner product computations in a left-to-right procedure. The thus computed vector $\boldsymbol{k}(N)$ concurrently excites the column cells of $A^{-\mathrm{T}}(N-1)$ and is stored there. Then the elements of $\tilde{\boldsymbol{w}}(N)$ can be produced simultaneously as inner products of $\boldsymbol{k}(N)$ with the columns of $A^{-\mathrm{T}}(N-1)$ in a top-to-bottom procedure.

It is worth stating that different steps of the HRLS algorithm can be executed in parallel, thus reducing the overall computation time. More specifically:

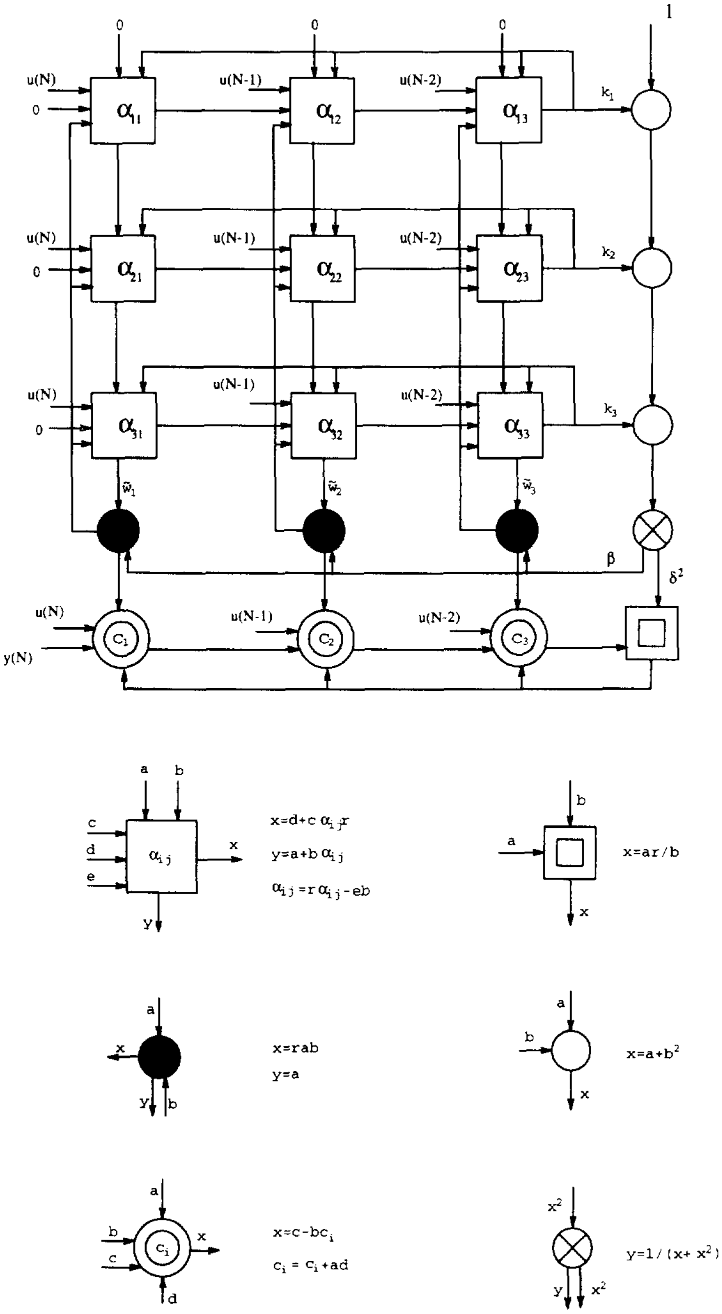

Fig. 4. An array architecture that implements the HRLS algorithm ($r = \lambda^{-1/2}$).

1. Having computed $\boldsymbol{c}(N-1)$ in the previous time instant, vector $\boldsymbol{k}(N)$ can be calculated in parallel with the a priori error $e(N)$ at the bottom of the array.
2. The computation of $\tilde{\boldsymbol{w}}(N)$ and that of $\delta^2(N)$ on the right of the array can be simultaneously performed.
3. All the elements of $A^{-\mathrm{T}}(N-1)$ are simulta-

neously updated according to the formula

$$a_{ij}(N) = \lambda^{-1/2} a_{ij}(N-1) - k_i(N)\hat{w}_j(N).$$

This can be accomplished in parallel with the update of the filter taps at the bottom of the array.

4. The vector $\lambda^{-1/2}\beta(N)\tilde{w}(N)$ in the intermediate cells and the quantity $e(N)/(\sqrt{\lambda}\delta^2(N))$ at the bottom right cell can be calculated concurrently.

It is clear from the above discussion that the computation time per time update iteration is that required for the completion of $2p+1$ multiplications/additions (MADs) and 2 divisions/square roots. Moreover, if we exploit the inherent pipelining of inner products, the above measure can be reduced from $(2p+1)$ to $(2\log_2 p + 1)$ MADs. In any case, a substantial improvement is offered with respect to the algorithm of [1, 7] where the computation time per time update corresponds to $8p$ multiplications, $3p$ additions, $2p$ divisions and p square roots. Thus, a considerable reduction in the throughput rate is achieved. Moreover, the processing units are of the simple multiply–add type, and the number of divisions is independent of p. In contrast, $O(p)$ dividers are needed in [7], which is usually not a desirable feature if VLSI implementation is considered.

Recently, an alternative systolic implementation of the inverse QR algorithm has been proposed [17]. In the new architecture, one processor is assigned to each 2×2 block of neighboring matrix elements by applying an odd–even partitioning technique. Such an approach permits a fully pipelineable implementation of the inverse QR algorithm. This is achieved, however, at the expense of a more complex structure with respect to arithmetic building blocks and communication overheads.

The selection of an appropriate implementation depends heavily on the applications' requirements and specifications. We must state, though, that the algorithm, which is presented in this paper, is well suited for implementation on a vector processor machine. We observe from Fig. 3 that the HRLS algorithm consists, almost entirely, of vector inner and outer products, matrix–vector multiplications, as well as other matrix and vector operations. All these operations employ vectors and matrices of constant dimension p. This is a direct consequence of the use of Householder transformations and makes a vector processor a suitable architecture for the algorithm's efficient implementation. This notion does not seem to be a characteristic of RLS algorithms which are based on Givens rotations, because of the property of Givens rotations to be more selective compared to Householder reflections.

5. Simulations

In order to verify the correctness of the HRLS algorithm a system identification problem is considered. The unknown FIR system is time invariant of order 6, the SNR is 30 dB, the forgetting factor $\lambda = 0.98$ and the initialization parameter $\mu = 0.01$. The input signal and the noise are chosen to be Gaussian white noise processes. In Fig. 5 two initial convergence curves are overlaid although they are not distinguished. One corresponds to the HRLS scheme and the other to the inverse QR algorithm of [1]. The curves are the average of 100 realizations, in which the FIR model remains the same but the input and noise signals are different. Note that long-term simulations were run with no indication of numerical stability problems for the proposed algorithm. A theoretical study of the stability properties of the algorithm is currently under investigation.

The stable performance of the HRLS algorithm is demonstrated from another example. In this case the order of the unknown time-invariant FIR system is 8 while the input signal is given according to the following linear combination of sinusoids [21]:

$$u(n) = \cos(0.05\pi n) + \sqrt{2}\cos(0.3\pi n) + v(n),$$

where $v(n)$ is white Gaussian noise with variance equal to 10^{-10}. The remaining specifications are the same with those of the first example. Note that the 8×8 autocorrelation matrix of the above input signal is nearly singular [21]. The squared errors obtained after applying the HRLS and the conventional RLS [9] algorithms are plotted in Fig. 6. Notice that the HRLS scheme retains a stable

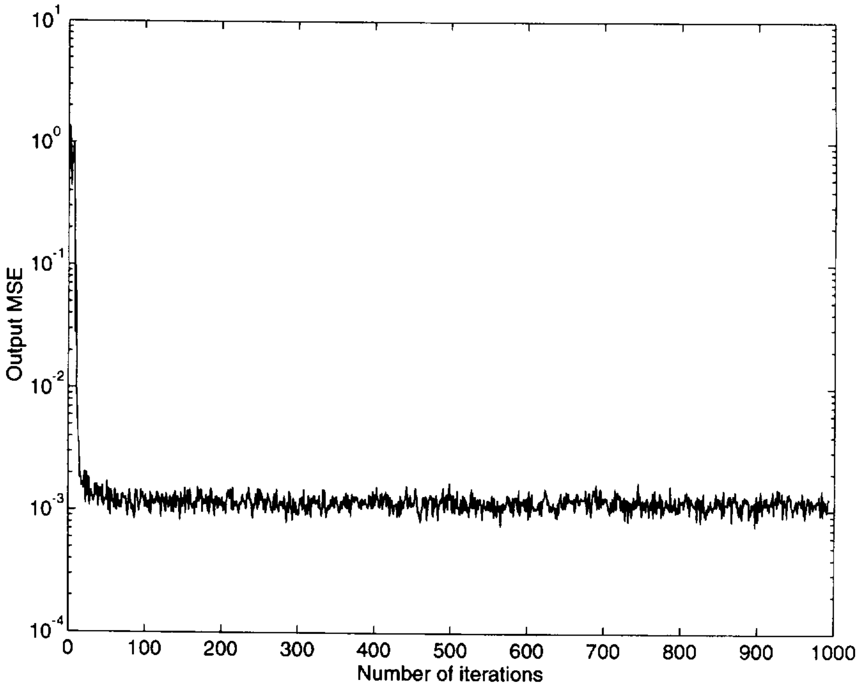

Fig. 5. Initial convergence curves.

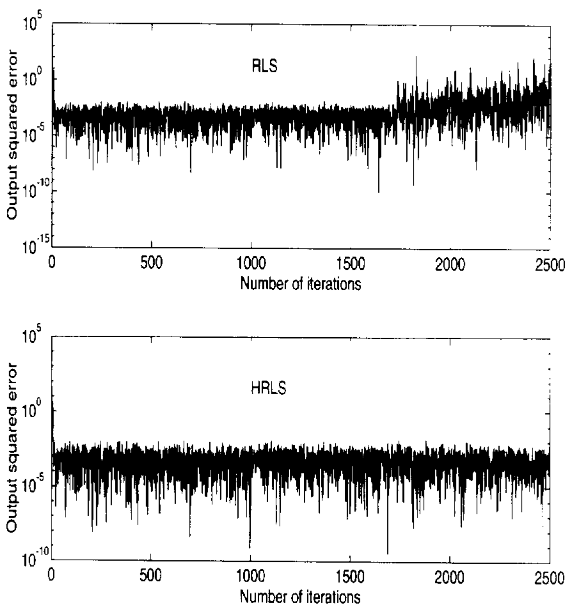

Fig. 6. The squared error of RLS and HRLS.

behavior in contrast to the RLS algorithm that diverges after about 1700 iterations.

6. Conclusions

An adaptive least-squares algorithm based on Householder reflections was developed in this paper. The new scheme employs low-cost additions and multiplications and computes directly the unknown coefficient estimates without the use of backsubstitution. The algorithm's enhanced parallelism combined to its numerically stable behavior can lead to efficient implementation of the new scheme on parallel architectures with short wordlengths and fixed-point arithmetic. The algorithm is favorably compared to the recently developed inverse QR scheme (both being square root inverse factorization schemes) in terms of computational complexity, parallel properties and numerical behavior.

Appendix A

We will prove that the Householder matrix $P(N)$ given in (31) satisfies the following equations:

$$P(N)\begin{bmatrix}\lambda^{1/2}A(N-1)\\ -\boldsymbol{u}^{\mathrm{T}}(N)\end{bmatrix}=\begin{bmatrix}A(N)\\ \boldsymbol{0}^{\mathrm{T}}\end{bmatrix}, \tag{36}$$

$$P(N)\begin{bmatrix}\lambda^{1/2}\hat{\boldsymbol{p}}(N-1)\\ -y(N)\end{bmatrix}=\begin{bmatrix}\hat{\boldsymbol{p}}(N)\\ \tilde{e}(N)\end{bmatrix}, \tag{37}$$

where $\hat{\boldsymbol{p}}(N)=T(N)\boldsymbol{p}(N)$ for every N. Let us suppose that

$$P(N)\begin{bmatrix}\lambda^{1/2}A(N-1)\\ -\boldsymbol{u}^{\mathrm{T}}(N)\end{bmatrix}=\begin{bmatrix}Y(N)\\ \boldsymbol{t}^{\mathrm{T}}(N)\end{bmatrix},$$

where $Y(N)$ corresponds to the upper $p\times p$ part of the resulting matrix and $\boldsymbol{t}^{\mathrm{T}}(N)$ represents its last row. Eqs. (31) and (28) imply that

$$\begin{aligned}\boldsymbol{t}^{\mathrm{T}}(N)&=-\boldsymbol{u}^{\mathrm{T}}(N)-\beta(N)(1+\delta(N))\\&\quad\times[\lambda^{1/2}\boldsymbol{k}^{\mathrm{T}}(N)A(N-1)-(1+\delta(N))\boldsymbol{u}^{\mathrm{T}}(N)]\\&=-\boldsymbol{u}^{\mathrm{T}}(N)-\frac{1}{\delta(N)}\\&\quad\times[\lambda^{1/2}\boldsymbol{g}^{\mathrm{T}}(N)\tilde{R}(N-1)-(1+\delta(N))\boldsymbol{u}^{\mathrm{T}}(N)]\\&=-\boldsymbol{u}^{\mathrm{T}}(N)\\&\quad-\frac{1}{\delta(N)}[\boldsymbol{u}^{\mathrm{T}}(N)-(1+\delta(N))\boldsymbol{u}^{\mathrm{T}}(N)]\\&=\boldsymbol{0}^{\mathrm{T}}.\end{aligned}$$

Furthermore,

$$\begin{aligned}Y(N)&=\lambda^{1/2}A(N-1)\\&\quad-\beta(N)\boldsymbol{k}(N)[\lambda^{1/2}\boldsymbol{k}^{\mathrm{T}}(N)A(N-1)\\&\qquad-(1+\delta(N))\boldsymbol{u}^{\mathrm{T}}(N)]\\&=T(N-1)[\lambda^{1/2}\tilde{R}(N-1)\\&\qquad+\beta(N)\delta(N)\boldsymbol{g}(N)\boldsymbol{u}^{\mathrm{T}}(N)]\tilde{R}^{-1}(N)\tilde{R}(N),\end{aligned}$$

or from (8) and (14)

$$\begin{aligned}Y(N)&=T(N-1)[I+\beta(N)\delta(N)\boldsymbol{g}(N)\boldsymbol{g}^{\mathrm{T}}(N)]\\&\quad\times\Sigma^{\mathrm{T}}(N)\tilde{R}(N).\end{aligned}$$

If we now substitute $\Sigma^{\mathrm{T}}(N)$ from (34), then after some manipulations we obtain

$$\begin{aligned}Y(N)&=T(N-1)[I-\beta(N)\boldsymbol{g}(N)\boldsymbol{g}^{\mathrm{T}}(N)]\\&\quad\times\Sigma^{-1}(N)\tilde{R}(N)\\&=T(N-1)\tilde{T}(N)\tilde{R}(N)=T(N)\tilde{R}(N)\\&=A(N),\end{aligned}$$

which ensures the validity of (36).

Eq. (37) is proved by exploiting the relation between the orthogonal matrices $\hat{Q}(N)$ and $P(N)$. Indeed, Eqs. (3), (7), (36), (26) and the uniqueness of the QR decomposition (under the assumption that $\tilde{R}$ has positive diagonal elements) lead to the following expression:

$$\hat{Q}(N)=\begin{bmatrix}T^{\mathrm{T}}(N)&\boldsymbol{0}\\ \boldsymbol{0}^{\mathrm{T}}&1\end{bmatrix}P(N)\begin{bmatrix}T(N-1)&\boldsymbol{0}\\ \boldsymbol{0}^{\mathrm{T}}&-1\end{bmatrix}. \tag{38}$$

The correctness of (37) is now straightforward if we substitute (38) in (4).

References

[1] S.T. Alexander and A.L. Ghirnikar, "A method for recursive least squares adaptive filtering based upon an inverse QR decomposition", *IEEE Trans. Signal Process.*, Vol. SP-41, No. 1, January 1993, pp. 20–30.

[2] G.J. Bierman, "A comparison of discrete linear filtering algorithms", *IEEE Trans. Aerospace Electronic Systems*, Vol. AES-9, No. 1, January 1973, pp. 28–37.

[3] G.J. Bierman, *Factorization Methods for Discrete Sequential Estimation*, Academic Press, New York, 1977.

[4] G.J. Bierman and C.L. Thronton, "Numerical comparison of Kalman filter algorithms: Orbit determination case study", *Automatica*, Vol. 13, 1977, pp. 13–25.

[5] J.M. Cioffi, "The fast adaptive ROTOR's RLS algorithm", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. ASSP-38, No. 4, April 1990, pp. 631–653.

[6] W.M. Gentleman and H.T. Kung, "Matrix triangularization by systolic arrays", *Proc. SPIE, Internat. Soc. Opt. Eng.*, Vol. 298, 1981, pp. 298–303.

[7] A.L. Ghirnikar, S.T. Alexander and R.J. Plemmons, "A parallel implementation of the inverse QR adaptive filter", *Comput. Elec. Eng.*, Vol. 18, Nos. 3/4, 1992, pp. 291–300.

[8] G.H. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, 1989, 2nd Edition.

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

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

[11] P.G. Kaminski, A.E. Bryson and S.F. Schmidt, "Discrete square-root filtering: A survey of current techniques", *IEEE Trans. Automat. Control*, Vol. AC-16, No. 6, December 1971, pp. 727–735.

[12] F. Ling, "Givens rotation based least squares lattice and related algorithms", *IEEE Trans. Signal Process.*, Vol. SP-39, No. 7, July 1991, pp. 1541–1551.

[13] F. Ling, D. Manolakis and J.G. Proakis, "A recursive modified Gram–Schmidt algorithm for least-squares estimation", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. ASSP-34, No. 7, August 1986, pp. 829–836.

[14] Z.S. Liu, "On-line parameter identification algorithms based on Householder transformations", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. SP-41, No. 9, September 1993, pp. 2863–2871.

[15] K.R. Liu, S.F. Hsieh and K. Yao, "Systolic block Householder transformation for RLS algorithm with two-level pipelined implementation", *IEEE Trans. Signal Process.*, Vol. SP-40, No. 4, April 1992, pp. 946–958.

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

[17] M. Moonen and J. Vandewalle, "A systolic array for recursive least squares computations", *IEEE Trans. Signal Process.*, Vol. SP-41, No. 2, February 1993, pp. 906–912.

[18] M. Morf and T. Kailath, "Square root algorithms and least-squares estimation", *IEEE Trans. Automat. Control*, Vol. AC-20, 1975, pp. 487–497.

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

[20] C.M. Rader and A.O. Steinhardt, "Hyperbolic Householder transformations", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. ASSP-34, No. 6, December 1986, pp. 1589–1602.

[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 Process.*, Vol. SP-39, No. 4, April 1991, pp. 879–891.

[22] A.A. Rontogiannis and S. Theodoridis, "New fast inverse QR least squares adaptive algorithms", *Proc. IEEE Internat. Conf. Acoust. Speech Signal Process.*, Detroit, MI, May 1995, pp. 1412–1415.

[23] A.H. Sayed and T. Kailath, "A state-space approach to adaptive filtering", *IEEE Signal Process. Magazine*, Vol. 11, No. 3, July 1994, pp. 18–60.

[24] T.J. Shepherd and J. Hudson, "Parallel weight extraction from a systolic adaptive beamformer", *Proc. IMA Conf. on Math. and Signal Processing*, Warwick, England, December 1988.

[25] A.O. Steinhardt, "Householder transfoms in signal processing", *IEEE ASSP Magazine*, July 1988, pp. 4–12.

[26] A.P. Varvitsiotis and S. Theodoridis, "A pipelined structure for QR adaptive LS system identification" *IEEE Trans. Signal Process.*, Vol. SP-39, August 1991, pp. 1920–23.

[27] J.H. Wilkinson, *The Algebraic Eigenvalue Problem*, Oxford University Press, 1965.