On Systolic Arrays for Recursive Complex Householder Transformations with Applications to Array Processing

C.F. T. Tang, K.J. R. Liu, and S. A. Tretter

Department of Electrical Engineering and Systems Research Center,
University of Maryland, College Park, MD 20742

Abstract

In order to achieve computational efficiency with robust numerical stability, the Householder transformation has been shown to be one of the best orthogonal factorizations. It is also known that the Householder transformation outperforms the Givens rotation in numerical stability under finite-precision implementation, and that it requires fewer arithmetic operations than the modified Gram-Schmidt does. As a result, the QR decomposition using the Householder transformation is very promising for VLSI implementation and real-time high throughput modern signal processing. A recursive complex Householder transformation with a fast initializing algorithm is presented and its associated parallel/pipelined architecture is also discussed.

1 Introduction

Mapping QR decomposition algorithms onto systolic arrays has received considerable attention recently. There are several reasons. One of the reasons is that recent developments in VLSI technology make it possible to build a multiprocessor system on a chip. Another reason is that many real-time signal processing applications require both a high throughput rate and superior numerical accuracy. Therefore, the combination of numerical analysis problems and VLSI designs has played an important role in real-time applications of modern signal processing due to the increasing use of numerical analysis in these areas. Recently, many issues such as systolic Cholesky decomposition [11,12], systolic Givens rotation [4,9,12,14], systolic Modified Givens rotation [8,9,14], systolic-like Modified Gram-Schmidt [15], and vectorized systolic block Householder transformation [6,7] have been reported elsewhere in the literature.

In adaptive array and multichannel applications, the approaches are generally known as direct sample matrix inversion (SMI). It is well known that the classical method, the sample matrix inversion method (SMI), may sometimes lead to undesired numerical characteristics due to ill-conditioned matrices. This means that an extremely high arithmetic precision is required when employing the sample matrix inversion method. To alleviate such roundoff sensitivity caused by the SMI method, the QR decomposition deserves serious consideration. A family of algorithms based on the QR decomposition can be employed. These include the Givens, modified Givens, Householder and modified Gram-Schmidt orthogonalization techniques.

Recently, the problem of designing algorithms based on the Householder transformation (HT) and its associated systolic architectures has been of great interest [1,3,6,7,10,13]. This is because the HT generally outperforms the Givens rotation under finite precision computations [3,6]. It is also true that the Householder method requires less computation than the Givens and modified Gram-Schmidt methods. Recently, a systolic architecture for the recursive block Householder transformation has been presented in [6,7]. Compared to the recursive block HT, the newly developed programmable complex HT systolic architecture saves $O((M-1) \times N)$ in computation time to form the upper triangular matrix during the initialization procedure, where N is the number of sensors and M is the number of data blocks. Furthermore, rather than employing the real HT reported, the complex HT is considered and developed since in many signal processing areas we often deal with complex data. Therefore, it is very important for real-time high throughput modern signal processing applications to develop a programmable complex HT systolic architecture.

In this paper, we first develop a systolic fast complex Householder algorithm which is programmable to handle both the initialization procedure and the recursive procedure. The fast complex Householder algorithm requires N snapshots of data for the initialization procedure to compute the upper triangular matrix while the recursive Householder algorithm in [6,7] needs $N \times M$ snapshots of data where N is the number of sensors and M is the number of data blocks. Then we introduce the new systolic architectures onto which the parallel complex Householder algorithms are mapped. The dataflow in these architectures is sample-by-sample in horizontal junction cells. Therefore, the proposed architectures in this paper are suitable for VLSI implementation and real-time signal processing applications.

2 Complex Householder Algorithm

In many signal processing applications, the QR decomposition provides a method to avoid computing the inverse of a matrix to achieve numerical stability and VLSI systolic implementation. There are many schemes to perform such a decomposition, e.g. Givens/modified Givens, Householder, Gram-Schmidt/modified Gram-Schmidt. Of particular interest in this paper is a sample-by-sample form of the Householder orthogonalization technique. Since in many applications of signal processing, the observed data matrix is complex, it is necessary to consider the complex case of the Householder transformation. Therefore, we assume X is a complex K by N observed data matrix where K is the number of snapshots and N is the number of sensors. The initialization

CH2977-7/91/0000-1033 $1.00 © 1991 IEEE

procedure is used when k the number of data snapshots, is less than or equal to N, the number of sensors, and the recursive procedure is applied when there are more than N data snapshots.

2.1 Fast Initialization Complex Householder Algorithm

In [6,7], the recursive Householder algorithm proposed needs $N \times M$ snapshots of data for the initialization procedure where N is the number of sensors and M is the number of data blocks. However, the fast initialization Householder algorithm described in this section requires only N snapshots of data. The HT initialization procedure for forming the upper triangular matrix requires only N data snapshots where N is the number of sensors. The factorization of a data matrix $X \in \mathbf{C}^{N \times N}$ can be achieved by a succession of Householder transformations [6] which produces a unitary N by N matrix Q and an upper triangular matrix R_{upper} such that

$$X = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$$

where $X = \begin{bmatrix} \underline{x}_1 & \underline{x}_2 & \cdots & \underline{x}_N \end{bmatrix}$. The algorithm for applying successive Householder transformations to zero out a given complex observed data N by N matrix X can be described as follows. More precisely, let

$$Q^H = Q_{N-1} \cdots Q_2 Q_1 \tag{1}$$

be a sequence of Householder transformations applied to X where Q_i is an N by N complex Householder Transformation matrix with the form

$$Q_i = \begin{bmatrix} I_{i-1 \times i-1} & \vdots & 0 \\ \cdots & \cdots & \cdots \\ 0 & \vdots & H_i \end{bmatrix} \qquad for \;\; i = 1, \cdots, N-1 \tag{2}$$

where $H_i \in \mathbf{C}^{(N-i+1)\times(N-i+1)}$ is a unitary matrix given by [2]

$$H_i = I - \frac{2}{\underline{u}_i^H \underline{u}_i} \underline{u}_i \underline{u}_i^H, \tag{3}$$

and $\underline{u}_i$ is defined as

$$\underline{u}_i^T = \begin{bmatrix} x_i(t_i) + e^{j\theta_i(t_i)} \|\underline{x}_i\|_2 & x_i(t_{i+1}) & \cdots & x_i(t_N) \end{bmatrix} \tag{4}$$

for $\underline{x}_i^T = \begin{bmatrix} x_i(t_i) & \cdots & x_i(t_N) \end{bmatrix}$ and the real number, $\theta_i(t_i)$, given by

$$\theta_i(t_i) = -j \log_e \frac{x_i(t_i)}{|x_i(t_i)|}$$

As a result, the algorithm for applying a sequence of Householder transformations Q to data matrix X can be described in two parts. First, for each $i = 1, \cdots, N-1$, we apply a Householder transformation to $\underline{x}_i$ and obtain

$$(H_i \underline{x}_i)^T = \begin{bmatrix} r_{ii} & 0 & \cdots & 0 \end{bmatrix} \qquad for \;\; i = 1, 2, \cdots, N-1 \tag{5}$$

where $r_{ii} = -e^{j\theta_i(t_i)} \|\underline{x}_i\|_2$ and a scalar parameter λ is defined by

$$\lambda = \frac{2}{\underline{u}_i^H \underline{u}_i} = (\|\underline{x}_i\|_2 (\|\underline{x}_i\|_2 + |x_i(t_i)|))^{-1} \tag{6}$$

Then, the same Householder algorithm H_i is repeatly applied to the remaining $N-i$ column vectors $\underline{x}_k^T = \begin{bmatrix} x_k(t_i) & \cdots & x_k(t_N) \end{bmatrix}$ $\in \mathbf{C}^{N-i+1}$, for $k = i+1, \cdots, N$. Thus, the new set of column vectors can be obtained as follows.

$$H_i \underline{x}_k = \underline{x}_k - \lambda \underline{u}_i \underline{u}_i^H \underline{x}_k = \underline{x}_k - \alpha \underline{u}_i \tag{7}$$

where α is a scalar parameter given by

$$\alpha = \lambda \underline{u}_i^H \underline{x}_k = \lambda (\underline{x}_i^H \underline{x}_k + \underline{x}_k(t_i) e^{-j\theta_i(t_i)} \|\underline{x}_i\|_2)$$

and

$$H_i \underline{x}_k = \begin{bmatrix} x_k(t_i) - \alpha x_i(t_i) + \alpha r_{ii} \\ x_k(t_{i+1}) - \alpha x_i(t_{i+1}) \\ \vdots \\ x_k(t_N) - \alpha x_i(t_M) \end{bmatrix} \qquad for \;\; k = i+1, \cdots, N \tag{8}$$

According to the above procedure, after applying Q_1 to X the first column of $Q_1 X$ is zeroed except for the first element when a chosen N by N unitary matrix H_I is applied to the N by N data matrix X. The second column of $Q_2 Q_1 X$ is zeroed in components 3 through M when a chosen $N-1$ by $N-1$ unitary matrix H_2 is applied to the $N-1$ by $N-1$ submatrix of $Q_1 X$. It is obvious that $Q_2 Q_1 X$ has zeros below the diagonal in both the two columns. Continuing in this way, the data matrix X can be transformed into upper triangular form by applying a product of at most $N-1$ unitary transformations to it. It is well known that the number of arithmetic operations gradually decreases in each of $N-1$ Householder transformations.

2.2 Recursive Complex Householder Algorithm

The triangular matrix R can be updated by employing unitary Householder transformations P which have the form

$$P^H \begin{bmatrix} \beta R \\ X \end{bmatrix} = \begin{bmatrix} R' \\ 0 \end{bmatrix} \tag{9}$$

where $R = \begin{bmatrix} r_{11} & r_{21} & \cdots & r_{N1} \\ 0 & r_{22} & \cdots & r_{N2} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & r_{NN} \end{bmatrix}$, $X = \begin{bmatrix} \underline{x}_1 & \underline{x}_2 & \cdots & \underline{x}_N \end{bmatrix}$, and P^H represents a sequence of complex Householder transformations used to zero out each column vector of a new complex M by N data matrix X in turn.

The algorithm for applying successive Householder transformations to update the triangular matrix R to which a new M by N complex data block is added is given by

$$P^H = P_N P_{N-1} \cdots P_2 P_1 \tag{10}$$

where an $M+N$ by $M+N$ complex Householder transformation P_i has the form

$$P_i = \begin{bmatrix} I_{i-1 \times i-1} & \vdots & 0 \\ \cdots & \cdots & \cdots \\ 0 & \vdots & H_i \end{bmatrix} \qquad for \;\; i = 1, 2, \cdots, N \tag{11}$$

where $H_i \in \mathbf{C}^{(M+N-i+1)\times(M+N-i+1)}$, a unitary matrix, is given by

$$H_i = I - \frac{2}{\underline{u}_i^H \underline{u}_i} \underline{u}_i \underline{u}_i^H \tag{12}$$

When given a column vector $\underline{x}_i'$ of the form

$$\underline{x}_i'^T = \begin{bmatrix} \beta r_{ii} & 0 & \cdots & 0 & x_i(t_1) & \cdots & x_i(t_M) \end{bmatrix}, \tag{13}$$

$\underline{u}_i$ can be defined by

$$\underline{u}_i^T = \left[\beta r_{ii} + e^{j\theta_{r_{ii}}} ||\underline{x}_i'||_2 \quad 0 \quad \cdots \quad 0 \quad x_i(t_1) \quad \cdots \quad x_i(t_M) \right] \tag{14}$$

where $\theta_{r_{ii}}$, a real parameter, is given by $\theta_{r_{ii}} = -j \log_e \frac{r_{ii}}{|r_{ii}|}$ Therefore, a Householder transformation can be represented by

$$H_i = \begin{bmatrix} H_{1\times 1} & \vdots & 0 & \vdots & H_{M\times 1} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \vdots & I_{N-i\times N-i} & \vdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ H_{1\times M} & \vdots & 0 & \vdots & H_{M\times M} \end{bmatrix} \tag{15}$$

where $H_{1\times 1} = 1 - \lambda(\beta^2 r_{ii}^* r_{ii} + ||\underline{x}_i'||_2^2 + 2\beta|r_{ii}|||\underline{x}_i'||_2)$
$H_{M\times 1} = -\lambda(\beta r_{ii} + e^{j\theta_{r_{ii}}}||\underline{x}_i'||_2)\underline{x}_i^H$
$H_{1\times M} = -\lambda(\beta r_{ii}^* + e^{-j\theta_{r_{ii}}}||\underline{x}_i'||_2)\underline{x}_i$
$H_{M\times M} = I - \lambda \underline{x}_i \underline{x}_i^H$
and

$$\lambda = \frac{2}{\underline{u}_i^H \underline{u}_i} = (||\underline{x}_i'||_2(||\underline{x}_i'||_2 + \beta|r_{ii}|))^{-1} \tag{16}$$

The algorithm used to update R to which a new M by N data matrix has been added can be described in two parts. The first part of the algorithm is to zero out a column vector of the new data matrix below the diagonal and the second part is similarly to apply the same algorithm to the rest of the column vectors. The following equations are obtained by applying a Householder transformation to an $M+N$ by N matrix $\begin{bmatrix} \beta R \\ X \end{bmatrix}$. Given a column vector $\underline{x}_i'$ described in equation (13), a Householder transformation applied to this matrix gives

$$(H_i \underline{x}_i')^T = \left[r_{ii}' \quad 0 \quad \cdots \quad 0 \right] \quad for \quad i = 1, 2, \cdots, N \tag{17}$$

where $r_{ii}' = -e^{j\theta_{r_{ii}}}||\underline{x}_i'||_2$.

When the same Householder transformation is applied to the rest of column vectors $\underline{x}_k'$ with

$$\underline{x}_k'^T = \left[\beta r_{ki} \quad \cdots \quad \beta r_{kk} \quad 0 \quad \cdots \quad 0 \quad x_k(t_1) \quad \cdots \quad x_k(t_M) \right],$$

the new set of column vectors can by obtained by

$$H_i \underline{x}_k' = \begin{bmatrix} \beta r_{ki} - \alpha r_{ii} + \alpha r_{ii}' \\ \beta r_{k\,i+1} \\ \vdots \\ \beta r_{kk} \\ 0 \\ \vdots \\ 0 \\ x_k(t_1) - \alpha x_i(t_1) \\ \vdots \\ x_k(t_M) - \alpha x_i(t_M) \end{bmatrix} \quad for \quad k = i+1, \cdots, N \tag{18}$$

where α, a scalar parameter, is given by $\alpha = \lambda \underline{u}_i^H \underline{x}_k' = \lambda(\underline{x}_i'^H \underline{x}_k' - r_{ii}'^* r_{ki})$.

For the case of recursive Householder transformations, an $M+N$ by $M+N$ unitary matrix H_1 is chosen so that it zeros out M components of the first column of the matrix which is obtained by applying H_1 to the matrix formed by combining the new M by M data matrix and the updated upper triangular matrix. Next the M components of the second column are zeroed when a chosen $M+N-1$ by $M+N-1$ unitary matrix H_2 is applied to the $M+N-1$ by $N-1$ submatrix of $Q_1 X$. It is apparent that the upper triangular matrix can be updated by applying a sequence of N Householder transformations. It is also known that the number of arithmetic operations is the same for each of the N Householder transformations. The initialization Householder transformations and recursive Householder transformations have the same algorithm. However, the number of arithmetic operators is less for initialization Householder transformations than that for recursive Householder transformations.

3 VLSI Array Processors Implementation

The parallel complex Householder algorithms with fast initialization have been presented so far. We now consider the issue of VLSI systolic implementation. Because of the breakthrough of VLSI techniques and the great demands of real-time high throughput applications of modern signal processing, systolic architectures have been proposed for many modern signal processing tasks in recent years. Such an architecture has the important properties of the use of repetition of simple processing elements and of a regular and local communication scheme among them. It is apparent that the simplicity, modularity, and expandability of systolic array processors make them suitable for VLSI/WSI implementation. The Householder transformation algorithm for adaptive filtering and estimation is a systolic algorithm which can be efficiently executed on systolic array processors.

According to equations 5,6, and 17, the boundary cells of the systolic initialization/recursive Householder transformations are described in Figure 1. The internal cells of the systolic recursive Householder transformations are illustrated in Figure 2 according to equations 8,16, and 18. The triangular systolic architecture for the parallel complex Householder algorithm is shown in Figure 3. In adaptive antenna and radar applications, the period of updating the optimum weights is significantly larger than the actual computation time [15], and the two-dimensional systolic array processors can be reduced into one-dimensional systolic array processors by employing a simply feedback configuration as illustrated in Figure 4. The one-dimensional triangular architecture using the feedback configuration and the two-dimensional triangular architecture require their own local memory and some minimal control circuity and programmable capabilities.

4 Conclusion

This paper has presented the recursive complex Householder algorithm with a fast initialization which can be programmed for both initialization and recursive procedures. Compared to the recursive block Householder algorithm described in [6,7], it saves $O((M-1)\times N)$ computation time for the initialization procedure to form the upper triangular matrix and requires less operations per time than the real block Householder algorithm to compute a complex data matrix. Applications based on this complex Householder algorithm will be presented in the future papers. For example, a Householder-based MVDR adaptive beamformer and a Householder-based MVDL adaptive locator with their VLSI systolic architectures will also be presented for real-time high throughput applications in adaptive array processing.

References

[1] J. Cioffi, "The Fast Householder Filters RLS Adaptive Filter." *Proc. ICASSP 1990*, pp. 1619-1621.

[2] Gene H. Golub and Charles F. Van Loan, *Matrix Computation*. The Johns Hopkins University Press 1983.

[3] L.Johnson, "A Computational Array for The QR method," *1982 Conference on advanced Research in VLSI*, MIT, pp. 123-129.

[4] H. T. Kung and W. M. Gentleman, "Matrix triangularization by systolic arrays," *Proc. SPIE*, **298**, Real-Time Signal IV, 1981, pp. 19-26.

[5] H. T. Kung, "Why systolic architectures?" *Computer*, **15**, 37, 1982, pp. 37-46.

[6] K. J. R. Liu, S. F. Heieh, and K. Yao, "Recursive LS Filtering using Block Householder Transformations." *Proc. IEEE ICASSP*, 1990, pp. 1631-1634.

[7] K. J. R. Liu, S. F. Heieh, and K. Yao, "Two Level Pipelined Implementation of Systolic Block Householder Transformations with Application to RLS Algorithm." *Proc. Int'l Conf. on Application-Specific Array Processors*, pp. 748-769, Princeton, Sep. 1990.

[8] J. G. Mcwhirter, "recursive least-squares minimization using a systolic array," *Proc. SPIE*, **431**, Real-Time Signal Processing VI, 1983, pp. 105-112.

[9] J. G. McWhirter and T. J. Shepherd, "Systolic Array processor for MVDR Beamforming," *IEE proceedings*, Vol 136, No. 2 April 1989, pp. 75-80.

[10] Charles M. Rader and Allan O. Steinhardt, "Hyperbolic Householder Transformations." *IEEE Trans. on ASSP*, vol ASSP-34, No. 6 DEC. 1986, pp. 1589-1602.

[11] R. Schreiber and W. P. Tang, "On systolic arrays for updating the Cholesky factorization,"Swedish Roy. Inst.Technol., Sweden, Tech. Rep. **TRITA-NA-8313**, 1983.

[12] R. Schreiber, "Implementation of adaptive array algorithms," *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP-34, NO. 5, pp. 1038-1045, Oct. 1986.

[13] Allan O. Steinhardt, "Householder Transformations in Signal Processing." *IEEE ASSP Magazine*, July 1988, pp. 4-12.

[14] C.R.Ward, P.J. Hargrave, and J.G. McWhirter, "A novel algorithm and architecture for adaptive digital beamforming," *IEEE on AP*, **AP-34**, pp. 338-346, Mar. 1986.

[15] Stanley M. Yuen, "Algorithmic, Architectural, and Beam Pattern Issues of Sidelobe Cancellation," *IEEE on AES*, VOL.25, NO.4, July 1989.

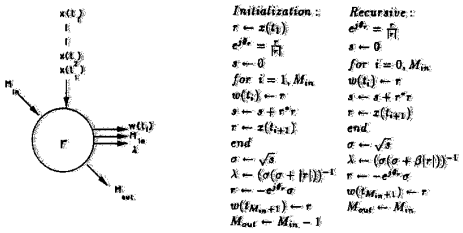

Figure 1: The Boundary Cell of The Systolic Initialization/Recursive Householder Transformations

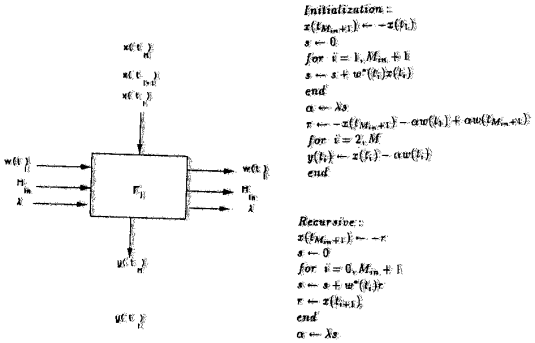

Figure 2: The Internal Cell of The Systolic Initialization/Recursive Householder Transformations

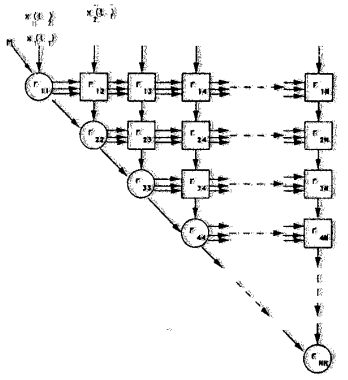

Figure 3: Triangular Systolic Architecture of Householder Transformations

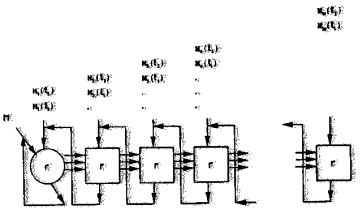

Figure 4: Linear Triangular Systolic Architecture of Householder Transformations with Feedback Configuration