Efficient Implementation of Rotation Operations for High Performance QRD-RLS Filtering

Bruno Haller
Communication Technology Laboratory
Swiss Federal Institute of Technology (ETH)
CH–8092 Zurich, Switzerland
e-mail: haller@nari.ee.ethz.ch

Jürgen Götze
Institute of Network Theory and Circuit Design
Technical University of Munich
D–80290 Munich, Germany
e-mail: Juergen.Goetze@nws.e-technik.tu-muenchen.de

Joseph R. Cavallaro
Department of Electrical and Computer Engineering
Rice University
Houston, TX 77251-1892, U.S.A.
e-mail: cavallar@ece.rice.edu

Published in *Proc. ASAP '97*, Zurich, Switzerland, July 14–16, 1997, pp. 162–174.

(Received ASAP '97 Best Paper Award)

Published in Proc. ASAP '97, Zurich, Switzerland, July 14–16, 1997, pp. 162–174.

Efficient Implementation of Rotation Operations for High Performance QRD-RLS Filtering

B. Haller
Swiss Federal Institute of Technology (ETH)
CH–8092 Zurich, Switzerland

J. Götze
Technical University of Munich
D–80290 Munich, Germany

J. R. Cavallaro *
Rice University
Houston, TX 77251-1892, U.S.A.

Abstract

In this contribution we present practical techniques for implementing Givens rotations based on the well-known CORDIC algorithm. Rotations are the basic operation in many high performance adaptive filtering schemes as well as numerous other advanced signal processing algorithms relying on matrix decompositions. To improve the efficiency of these methods, we propose to use "approximate rotations", whereby only a few (i.e. $r \ll b$, where b is the operand word length) elementary angles of the original CORDIC sequence are applied, so as to reduce the total number of required shift-add operations. This seemingly rather ad hoc and heuristic procedure constitutes a representative example of a very useful design concept termed "approximate signal processing" recently introduced and formally exposed by Nawab et al. in [1], concerning the trade-off between system performance and implementation complexity, i.e. between accuracy and resources. This is a subject of increasing importance with respect to the efficient realization of demanding signal processing tasks. We present the application of the described rotation schemes to QRD-RLS filtering in wireless communications, specifically high speed channel equalization and beamforming, i.e. for intersymbol and co-channel/interuser interference suppression, respectively. It is shown via computer simulations that the convergence behavior of the scheme using approximate Givens rotations is insensitive to the value of r, and that the misadjustment error decreases as r is increased, opening up possibilities for "incremental refinement" strategies.

1: Introduction

There is a growing number of important real-time applications, especially in the area of wireless communications, where systems are required to operate reliably in the presence of strong interference. The capacity of radio networks can be greatly improved by exploiting spatial filtering techniques such as adaptive digital beamforming [2]. To combat the rapidly time-varying multipath propagation encountered in mobile radio advanced channel equalization concepts [3] are essential to obtain satisfactory performance. Code division multiple access (CDMA) [4], employing spread-spectrum signaling with different signature sequences to distinguish between users, is a media access strategy that has gained considerable interest lately for wireless transmission. Since hereby all active users concurrently share the entire allocated frequency band, sophisticated signal processing is required to reliably extract each user's data from the "mixture" of signals present at the receiver input. Several authors advocate the use of MMSE- (min. mean square error) type detectors to cope with this hostile interference situation [5]–[7]. Furthermore, interference rejection techniques are also a

* Supported by the U.S. National Science Foundation under grant NCR-9506681.

prerequisite in wireless overlay systems which have to coexist, i.e. operate simultaneously within the same frequency band and geographical area together with other services without the need for any explicit "mutual coordination" [8]. It is within this context that the authors are investigating the methods discussed in the sequel, but it must be emphasized that the presented schemes are of very general interest and applicable in all situations that call for high performance adaptive filtering to extract the information of interest from signals heavily corrupted by time-varying distortions as well as nonstationary disturbances.

The recursive least squares (RLS) algorithm is preferred over the popular least mean square (LMS) gradient descent technique in such instances (i.e., when the input data correlation matrix exhibits a large eigenvalue spread) due to its superior convergence properties. A vast amount of research has been invested into reducing the demanding computational requirements and overcoming the poor numerical properties of the RLS algorithm in order to increase its practical applicability. The method of choice for fast and numerically robust RLS filtering is based upon orthogonal triangularization of the input data matrix via **QR** decomposition (QRD) [9]. The triangularization procedure can be realized using either Givens, Householder or (modified) Gram-Schmidt transformations [10]. In situations where the data is to be processed on a sample by sample basis (as opposed to block-wise) **QR**-updating through a sequence of Givens rotations is commonly employed. The resulting QRD-RLS algorithm requires $O(M^2)$ operations per updating step, where M is the problem order.[1] Unfortunately, this time complexity does not comply with the capabilities of many current systems based on general-purpose, programmable DSPs. With the tremendous advances of microelectronics in the past few years the idea of parallel processing using systolic/wavefront arrays has become practically feasible since their properties, such as regularity, modularity and locality, take maximum advantage of VLSI technology [11]. A highly efficient parallel and pipelined triangular systolic processor array (tri-array) realization of the QRD-RLS algorithm based on Givens rotations was first explored by Gentleman & Kung [12] and later refined by McWhirter [13].[2] The main tasks of the processing elements are the evaluation and execution of plane rotations to annihilate specific matrix elements. With MAC- (multiply-and-accumulate) type arithmetic units the computation of the rotation angles requires either square-roots and divisions or some trigonometric functions, whereas the vector rotations themselves only need multiplications and additions. This means that the determination of the rotation parameters takes many more clock/instruction cycles than it does to apply them, leading to an unequal distribution of the work-load within the array. Due to the computational expense of the square-root operation, a number of authors have published square-root free methods for performing Givens rotations [10], [15]–[18]. A unified approach to the different square-root free techniques is presented in [19], generalizing all known algorithms. Recently, algorithms that avoid both square-roots as well as divisions have also been introduced [20], [21]. An alternative, new type of rotation termed the "scaled tangent rotation" (STAR), allowing the QRD-RLS algorithm to be pipelined to arbitrary levels, is described in [22]. The single chip, bit-level pipelined implementation of a 100 MHz fourth-order adaptive filter employing this technique is presented in [23]. In view of a fixed-point implementation, McWhirter et al. have proposed the so-called "normalized Givens rotation" [24], whereby all involved parameters are reformulated such that their magnitude never exceeds the value of one. An often used alternative method of performing two-dimensional vector rotations relies on the well-known CORDIC (COordinate Rotation DIgital Computer) algorithm introduced by Volder [25] and extended by Walther [26]. The fundamental idea behind the CORDIC scheme is to carry out the evaluation and execution of ("macro"-)rotations as a series of "micro-rotations" using a fixed set of elementary rotation angles. Through a clever choice of the elementary angles all computations can be implemented efficiently in VLSI using a sequence of shift and add/subtract operations. Instead of performing the complete series of micro-rotations, requiring a number of iterations roughly equal to the word length b of the operands, we propose to use a small subset of r rotation angles ($r = 1$ in the extreme case), appropriately selected for each computation so as to achieve a good approximation of the exact result. This leads to a

[1] The computational complexity can be reduced to $O(M)$ in applications involving time series input by exploiting the shifting property $\rightarrow$ fast RLS algorithms [9].

[2] A corresponding linear array is treated in [14].

significant reduction in the total number of performed shift-add operations.

The rest of this paper is organized as follows. In Section 2 we review the QRD-RLS algorithm, explain the CORDIC technique as applied in the circular coordinate system and describe the design and ASIC implementation of a fixed-point CORDIC processor element (CPE) capable of computing all the elementary functions required for the evaluation and execution of Givens rotations. Hereby, special attention is given to the problem of handling complex valued input signals. To improve the efficiency of the CORDIC technique, "approximate rotations" are introduced in Section 3. A possible method for constructing orthonormal micro-rotations, along with a discussion of such issues as the simple computation of the optimal rotation parameters and scaling factor compensation, are dealt with in detail. Results from computer simulations are presented to illustrate the performance of the described methods. Section 4 concludes the paper.

2: QRD-RLS algorithm based on CORDIC arithmetic

The QRD-RLS algorithm is numerically more robust than the standard RLS algorithm due to the fact that it operates directly on the incoming data matrix via the **QR** decomposition rather than working with a (time-averaged) estimate of the input data correlation matrix. Here one takes advantage of numerically stable unitary transformations on the one hand, and circumvents the need to form an estimate of the correlation matrix, whereby the condition number of the data matrix is squared and the required word length doubled. In the following subsections we briefly review the QRD-RLS algorithm and demonstrate how all the basic operations required to compute the a posteriori error in least squares problems can be performed using CORDIC arithmetic.

2.1: QRD-RLS algorithm

Here we look at the general case of the recursive least squares minimization problem based on an adaptive linear combiner. Let $\mathbf{x}[t] \in \mathbb{C}^M$ be a vector of observations taken from M data signals at sample time t. Using a linear combination of the signals $x_m[t]$ $(m = 1 \ldots M)$, a desired signal $y[t]$ is to be estimated for the same time instant. Thereby, the goal is to minimize the sum of exponentially weighted squared errors,

$$\min_{\mathbf{w}[t]} \sum_{i=1}^{t} \beta^{t-i} |y[i] - \mathbf{x}^{\mathrm{T}}[i]\mathbf{w}[t]|^2. \tag{1}$$

The so-called "forgetting factor" $0 \ll \beta \leq 1$ is commonly used to discount old data from the computations (exponential downdating), in order to provide a certain tracking capability when the system operates in a nonstationary environment. This is equivalent to determining the weight vector $\mathbf{w}[t]$ which minimizes the ℓ_2-norm of the vector of error residuals $\mathbf{e}[t]$,

$$\|\mathbf{e}[t]\| = \sqrt{\mathbf{e}^{\mathrm{H}}[t]\mathbf{e}[t]}, \tag{2}$$

where

$$\mathbf{e}[t] = \mathbf{B}[t](\mathbf{y}[t] - \mathbf{X}[t]\mathbf{w}[t]) \in \mathbb{C}^t, \tag{3}$$

with the data matrix

$$\mathbf{X}[t] \triangleq \begin{pmatrix} \mathbf{x}^{\mathrm{T}}[1] \\ \vdots \\ \mathbf{x}^{\mathrm{T}}[t] \end{pmatrix} \in \mathbb{C}^{t \times M},$$

and the weighting matrix

$$\mathbf{B}[t] \triangleq \mathrm{diag}(\sqrt{\beta}^{t-1}, \sqrt{\beta}^{t-2}, \cdots, 1) \in \mathbb{R}^{t \times t}.$$

Initialization:

$\mathbf{R}[0] = \sqrt{\delta}\mathbf{I}_M \in \mathbb{R}^{M\times M}$ with $0 \leq \delta \ll 1, \quad \mathbf{u}[0] = 0_M$

for $t = 1, 2, \ldots$ do:

$$\hat{\mathbf{Q}}[t] \begin{pmatrix} \sqrt{\beta}\mathbf{R}[t-1] & \sqrt{\beta}\mathbf{u}[t-1] \\ \mathbf{x}^{\mathrm{T}}[t] & y[t] \end{pmatrix} = \begin{pmatrix} \mathbf{R}[t] & \mathbf{u}[t] \\ 0_M^{\mathrm{T}} & \tilde{e}[t] \end{pmatrix}$$

where $\hat{\mathbf{Q}}[t] = \hat{\mathbf{Q}}_M[t] \cdots \hat{\mathbf{Q}}_1[t] \in \mathbb{C}^{(M+1)\times(M+1)}$

$$\text{with } \hat{\mathbf{Q}}_m[t] = \begin{pmatrix} \mathbf{I}_{m-1} & \vdots & & \vdots \\ \cdots & \cos\theta_m[t] & \cdots & \sin\theta_m[t]\exp(-\jmath\phi_m[t]) \\ & \vdots & \mathbf{I}_{M-m} & \vdots \\ \cdots & -\sin\theta_m[t] & \cdots & \cos\theta_m[t]\exp(-\jmath\phi_m[t]) \end{pmatrix}$$

$e[t] = \tilde{e}[t] \cdot \tilde{\gamma}[t]$ with $\tilde{\gamma}[t] = \prod_{m=1}^{M} \cos\theta_m[t]\exp(-\jmath\phi_m[t])$

Table 1. QRD-RLS algorithm based on complex Givens rotations.

Since the Euclidean vector norm is invariant with respect to unitary (orthogonal) transformations $\mathbf{Q}[t]$, we apply the QRD to transform the weighted input data matrix $\mathbf{B}[t]\mathbf{X}[t]$ into an upper triangular matrix $\mathbf{R}[t] \in \mathbb{C}^{M\times M}$:

$$\mathbf{Q}[t]\mathbf{e}[t] = \begin{pmatrix} \mathbf{u}[t] \\ \mathbf{v}[t] \end{pmatrix} - \begin{pmatrix} \mathbf{R}[t] \\ \mathbf{0} \end{pmatrix} \mathbf{w}[t]. \tag{4}$$

As can be seen from this equation, the minimum norm condition for the error residual $\mathbf{e}[t]$ is obtained when

$$\mathbf{R}[t]\mathbf{w}[t] = \mathbf{u}[t]. \tag{5}$$

This equation defines the least squares weight solution for the adaptive linear combiner. Since the matrix $\mathbf{R}[t]$ is upper triangular, the weight vector $\mathbf{w}[t]$ may be derived very simply by a process of back-substitution.

The triangular system of equations can be updated on a sample by sample basis according to the algorithm summarized in **Tab. 1**. The unitary update transformation $\hat{\mathbf{Q}}[t]$ represents a sequence of M complex Givens rotations, consisting of a phase compensation term $\mathbf{G}(\jmath\phi_m)$ times a real Givens rotation $\mathbf{G}(\theta_m)$, which operate on two rows of the matrix at a time as follows:

$$\underbrace{\begin{pmatrix} \cos\theta_m & \sin\theta_m \\ -\sin\theta_m & \cos\theta_m \end{pmatrix}}_{\mathbf{G}(\theta_m)} \cdot \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & \exp(-\jmath\phi_m) \end{pmatrix}}_{\mathbf{G}(\jmath\phi_m)} \cdot \begin{pmatrix} \ldots & 0 & r_{m,m} & r_{m,m+1} & \ldots \\ \ldots & 0 & x_m & x_{m+1} & \ldots \end{pmatrix}$$

$$= \begin{pmatrix} \ldots & 0 & r'_{m,m} & r'_{m,m+1} & \ldots \\ \ldots & 0 & 0 & x'_{m+1} & \ldots \end{pmatrix}, \tag{6}$$

where the rotation angles ϕ_m and θ_m are chosen to cancel the complex value x_m:

$$\phi_m = \arctan\frac{\mathrm{Im}(x_m)}{\mathrm{Re}(x_m)}, \tag{7}$$

$$\theta_m = \arctan\frac{|x_m|}{r_{m,m}}. \tag{8}$$

This form of complex Givens rotation differs from the general case found in [27] for the singular value decomposition (SVD) of complex matrices, because the diagonal elements of $\mathbf{R}$, i.e. $r_{m,m}$ ($m = 1 \ldots M$) are positive real numbers in our case ($\rightarrow$ only a single phase compensation term).

The application of M such complex Givens rotations successively eliminates the elements of the input data vector $\mathbf{x}^{\mathrm{T}}[t]$ whilst updating $\mathbf{R}[t]$ and $\mathbf{u}[t]$. Obviously, the a posteriori error $e[t]$, which is the quantity sought in many applications such as adaptive beamforming [28], can be calculated without explicitly computing the weight vector and applying the result to Eq. (3), saving hardware and further improving numerical robustness [13].

2.2: Implementation of Givens rotations employing CORDIC

Since the QRD-RLS algorithm mainly performs orthogonal rotations, the CORDIC algorithm is especially well-suited to carry out the necessary computations. An outstanding feature of the CORDIC scheme is that the determination of the rotation parameters takes the same number of clock/instruction cycles as does the actual rotation itself. This permits a truly systolic implementation of the tri-array for QRD-RLS filtering, since the work-load is equal within both the boundary and internal cells.[3] Additionally, the complete array can be constructed by repeated use of a single type of cell, providing the required functionality with a minimum of control signals and interconnection. These features are highly advantageous for an efficient VLSI implementation, because it allows "tiling" of a highly optimized macro-cell building block, enabling very compact, high speed (and/or low power) designs.

CORDIC algorithm:

The basic idea underlying the CORDIC scheme is to carry out vector ("macro"-)rotations by an arbitrary rotation angle θ via a series of $b+1$ "micro-rotations" using a fixed set of predefined elementary angles α_j

$$\theta = \sum_{j=0}^{b} \sigma_j \alpha_j, \quad \sigma_j \in \{-1, +1\}. \tag{9}$$

This leads to a representation of the rotation angle θ in terms of the rotation coefficients σ_j. If the elementary angles are defined as

$$\alpha_j \stackrel{\triangle}{=} \arctan 2^{-j}, \quad j \in \mathcal{J} = \{0, 1, 2, \ldots, b\}, \tag{10}$$

it follows that, an unscaled μ-rotation $\mathbf{G}_{\mathrm{u}}(\alpha_j)$ can be performed via two shift-add operations, which are easily realized in hardware:

$$\begin{pmatrix} x_{j+1} \\ y_{j+1} \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & \tan\alpha_j \\ -\tan\alpha_j & 1 \end{pmatrix}}_{\mathbf{G}_{\mathrm{u}(\alpha_j)}} \cdot \begin{pmatrix} x_j \\ y_j \end{pmatrix} = \begin{pmatrix} 1 & \sigma_j 2^{-j} \\ -\sigma_j 2^{-j} & 1 \end{pmatrix} \cdot \begin{pmatrix} x_j \\ y_j \end{pmatrix}. \tag{11}$$

The final result is obtained with a precision of b bits ($\stackrel{b}{=}$) after the execution of $b+1$ unscaled μ-rotations (CORDIC iterations) and a multiplication with the scaling factor $K = \prod_{j=0}^{b} 1/\sqrt{1+2^{-2j}}$ ($\rightarrow$ scaled rotation $\mathbf{G}_{\mathrm{s}}(\theta)$):

$$\begin{pmatrix} x_{\mathrm{out}} \\ y_{\mathrm{out}} \end{pmatrix} = K \cdot \prod_{j=0}^{b} \mathbf{G}_{\mathrm{u}}(\alpha_j) \cdot \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \stackrel{b}{=} \mathbf{G}(\theta) \cdot \begin{pmatrix} x_0 \\ y_0 \end{pmatrix}. \tag{12}$$

The multiplication with the constant factor K can also be decomposed into a sequence of simple shift-add operations which are often performed in a series of additional scaling iterations.

The CORDIC has two modes of operation called "vectoring", to compute the magnitude and phase of a vector

$$\begin{pmatrix} x_{\mathrm{out}} \\ y_{\mathrm{out}} \end{pmatrix} = \begin{pmatrix} \mathrm{sign}(x_{\mathrm{in}}) \cdot \sqrt{x_{\mathrm{in}}^2 + y_{\mathrm{in}}^2} \\ 0 \end{pmatrix}, \tag{13}$$

[3] Due to the lack of this property, an asynchronous technique based on a wavefront array was chosen to build such a system in the past [29].

Initialization:

$x_0 = x_{\text{in}}/2$, $y_0 = y_{\text{in}}/2$
add all-zero guard bit extension

for $j = 0 \ldots 18$ do:

$x_{j+1} = x_j - \sigma_j y_j 2^{-s_j} + \gamma_j x_j 2^{-s_j}$
$y_{j+1} = y_j + \sigma_j x_j 2^{-s_j} + \gamma_j y_j 2^{-s_j}$
where
$\sigma_j = -\text{sign}(x_j) \cdot \text{sign}(y_j) = \sigma_{\text{out}}$ in Vec. mode and
$\sigma_j = \sigma_{\text{in}}$ in Rot. mode
$\{s_j | j = 0, 1, \ldots, 18\} = \{0, 1, 2, 3, 4, 5, 6, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17\}$
$\{\gamma_j | j = 0, 1, \ldots, 18\} = \{0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1\}$

Correction cycle:

if $x_{19} < 0$ then $x_{20} = x_{19} + 1$ else $x_{20} = x_{19}$
if $y_{19} < 0$ then $y_{20} = y_{19} + 1$ else $y_{20} = y_{19}$
erase guard bits

$x_{\text{in}} = x_{20}$, $x_{\text{out}} = x_{20}$, $y_{\text{out}} = y_{20}$

Table 2. CORDIC algorithm based on [33] as implemented in the CPE.

$$\theta_{\text{out}} = -\arctan\frac{y_{\text{in}}}{x_{\text{in}}}, \tag{14}$$

whereby the vector $(x_{\text{in}} \;\; y_{\text{in}})^{\text{T}}$ is rotated to the x-axis, and "rotation"

$$\begin{pmatrix} x_{\text{out}} \\ y_{\text{out}} \end{pmatrix} = \begin{pmatrix} \cos\theta_{\text{in}} & -\sin\theta_{\text{in}} \\ \sin\theta_{\text{in}} & \cos\theta_{\text{in}} \end{pmatrix} \cdot \begin{pmatrix} x_{\text{in}} \\ y_{\text{in}} \end{pmatrix}, \tag{15}$$

$$\theta_{\text{out}} = \theta_{\text{in}}, \tag{16}$$

in which case the vector $(x_{\text{in}} \;\; y_{\text{in}})^{\text{T}}$ is rotated by the angle θ_{in}.

Obviously, the real Givens rotation $\mathbf{G}(\theta_m)$ in Eq. (6) can be carried out using the CORDIC algorithm in rotation mode, whereas the determination of the rotation angle according to Eq. (8) is accomplished using the CORDIC in vectoring mode.

CORDIC processor element:

The required computations, i.e. the CORDIC equations, may be implemented in a number of ways, such as bit-serial or word-parallel with the iterations being carried out in either a serial or parallel manner [30]. By choosing a recursive, pipelined, or array architecture a trade-off between area, throughput, and latency is possible, depending on the application in mind. A CORDIC processor element (CPE) was recently implemented at ETH as an application-specific integrated circuit (ASIC) [31]. Due to area limitations a recursive, word-parallel scheme using fixed-point arithmetic with 16-bit data words (22-bit internal precision) was chosen.[4] The main design target was to come up with a small circuit in terms of silicon area, thus making it feasible to integrate several CPEs on a single chip in a future project. In order to keep the throughput as high and latency as low as possible, the special shift sequence $\{s_j\}$ proposed by Despain [33] resulting in the least number of CORDIC iterations for 16-bit precision was chosen. The implemented CORDIC algorithm is summarized in **Tab. 2**. In this scheme the scaling operation is incorporated into each iteration through the additional terms containing the scaling coefficient γ_j. Due to the restrictions that apply to student IC design projects at ETH, only two CPEs were integrated on the manufactured 1.2 μm CMOS

[4] The implementation of a floating-point chip is presented in [32].

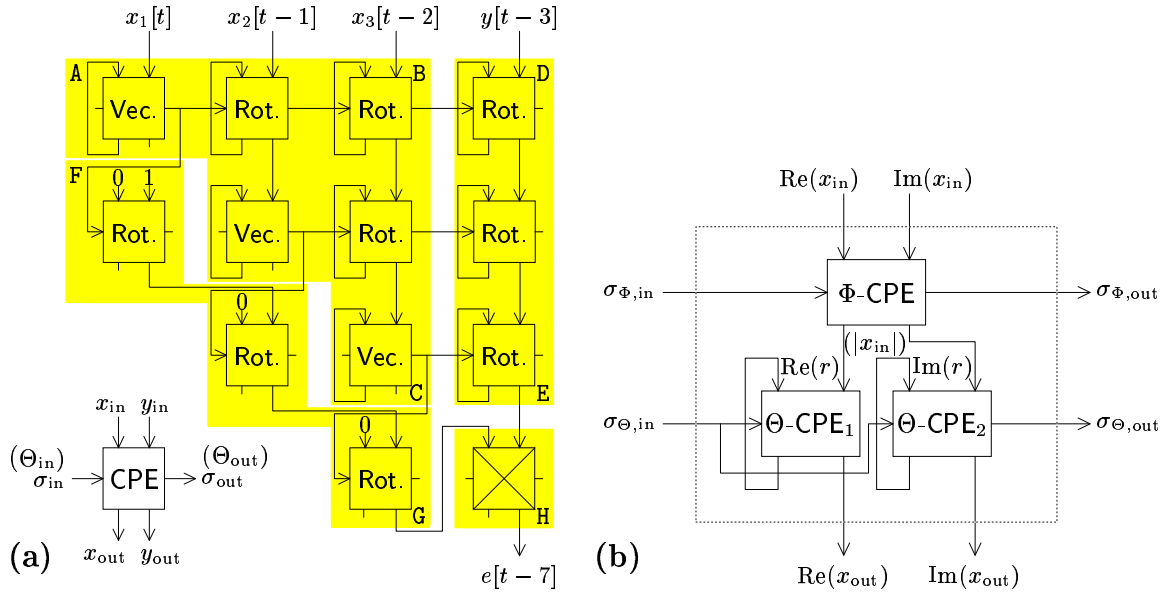

Figure 1. (a) Systolic CPE array. (b) CPE-based "super-cell" for complex Givens rotations.

chip. It comprises approximately 32 k transistors on a core area of 9.2 mm^2. The incorporation of several units in a future version of the component should be quite simple due to the highly modular and parameterized design of the CPE using VHDL. The device achieves a typical cycle time of 25 ns with a 40 MHz system clock resulting in a performance of 2×10^6 rotations per second.

CPE-based systolic tri-array:

The application of the CPE in a systolic antenna array signal processor is presented in [34]. The implementation of a systolic architecture for adaptive linear combining is illustrated schematically in **Fig. 1a** for the case of $M = 3$ real valued input signals. This array entirely consists of CPEs which operate synchronously driven by a single global master clock. Since all the CPEs need the same amount of time to perform their computations they can never be flooded with data. Thereby, the CPEs designated with Vec. are configured to the "vectoring" mode of operation and those labeled with Rot. operate in the "rotation" mode. Each row performs a Givens rotation, whereby the rotation angle is determined by the CPE in vectoring mode at the beginning of the row. The rotation angle is passed to the rotation CPEs to the right with one clock cycle delay, thus requiring the elements of the data vector to be applied to the array in a time staggered fashion as indicated by the indices in Fig. 1a. If complex input data has to be handled we replace each CPE through a "super-cell" comprising three CPEs interconnected according to **Fig. 1b**. In vectoring mode ($\mathrm{Im}(r_{m,m}) = 0$), the imaginary part of the complex value x_m is annihilated by the Φ-CPE and subsequently $|x_m|$ is zeroed in Θ-CPE$_1$.[5] The complex Givens rotation is then coded by the two sequences of rotation coefficients $\{\sigma_{\Phi,j}\}$ and $\{\sigma_{\Theta,j}\}$. By applying these rotation coefficients to a super-cell configured to operate in the rotation mode the incoming vector $(\mathrm{Re}(x_{in})\ \ \mathrm{Im}(y_{in}))^T$ is rotated by ϕ_m in the Φ-CPE and subsequently the real and imaginary parts of $r_{m,n}$ and $x_{m,n}$ $(m < n \leq M)$ are each rotated by θ_m in Θ-CPE$_1$ and Θ-CPE$_2$, respectively.

Exponential weighting could also very easily be included within the CORDIC processors, by multiplying the value fed-back from the x_{out} to the x_{in} port with the square-root of the forgetting factor β:

$$x_{out} = \sqrt{\beta} \cdot K \cdot x_b. \tag{17}$$

A scheme combining both scaling factor compensation and weighting based on shift-add operations is employed in the MUSE (Matrix Update Systolic Experiment) system built at MIT Lincoln Labo-

[5] Note that Θ-CPE$_2$ is not required for the vectoring operation!

ratories [35], which is based on a linear systolic array and was realized using restructurable VLSI, i.e. wafer-scale integration (WSI) technology. Unfortunately, this type of downdating procedure could not be included in the CPE because the scaling factor compensation is integrated into the actual CORDIC iterations, which is not possible for the weighting factor, since it is only allowed to affect the value of x_{out}.

Information about other interesting CORDIC processor implementations specifically tailored towards the problem discussed here has been published in [36]–[38].

3: Approximate rotations for QRD-RLS adaptive filtering

For small rotation angles the CORDIC algorithm can be very inefficient, since many unnecessary rotations by large angles are performed in the first few steps which gradually have to be compensated for in the later CORDIC iterations. An intuitively more appropriate strategy would be to pick only those elementary rotation angles α_j which are essentially needed to approximately represent the given angle to within the precision required by the application at hand. Such a technique is especially favorable in situations where numerical accuracy is not a prime concern, which is often the case in DSP applications since processing is carried out on noisy input signals, represented by a finite number of digits. The DSP system designer's main challenge consists of finding a good balance between accuracy, dictated by the required level of performance, and resource requirements, which are limited by the allowed cost, size, weight, and power consumption. The demand for a more systematic approach to this problem has led to the development of a new design methodology called "approximate signal processing" [1]. Along the lines of this novel concept, we propose to apply "approximate rotations" to the problem of QRD-RLS adaptive filtering. This idea has previously been employed in conjunction with other algorithms based on orthogonal transformations, such as the Jacobi algorithm for eigenvalue decomposition (EVD) [39], [40], Kogbetliantz's algorithm for updating the singular value decomposition (SVD) [41], the discrete cosine transform (DCT) [42], and wavelet transforms [43].

3.1: Approximate Givens rotation

The purpose of the Givens rotations in the QRD-RLS algorithm is to annihilate the M components x_m of each new input data vector $\mathbf{x}$. Using the CORDIC algorithm, $b+1$ μ-rotations must be performed to zero an entry x_m of $\mathbf{x}$. Instead of using the exact rotation angle θ_m as specified in Eq. (8), we suggest to apply an approximate rotation by an angle $\tilde{\theta}_m$ to reduce the value x_m such that

$$|x'_m| = |d| \cdot |x_m| \quad \text{with } 0 \leq |d| < 1. \tag{18}$$

The reduction factor d depends on $\tau = r_{m,m}/x_m$ and $\tilde{t} = \tan\tilde{\theta}_m$ as follows:

$$d(\tau, \tilde{t}) = \frac{1 - \tau \cdot \tilde{t}}{\sqrt{1+\tilde{t}^2}}. \tag{19}$$

Obviously, for the exact rotation where $\tilde{t} = \tan\theta_m = x_m/r_{m,m}$ one obtains $d = 0$. Regarding the possible approximate rotation angles $\tilde{\theta}$ we restrict ourselves to the set of elementary CORDIC angles, i.e. $\tilde{\theta} \in \{\alpha_j\}$, that is we rely on a single μ-rotation according to Eq. (11). An orthonormal, i.e. scaled μ-rotation $\mathbf{G}_{\text{s}}(\alpha_j) = K_j \cdot \mathbf{G}_{\text{u}}(\alpha_j)$ is obtained by multiplying the result of Eq. (11) with the scaling factor

$$K_j = \frac{1}{\sqrt{1+2^{-2j}}}, \tag{20}$$

which depends on j. In virtue of an easy scaling factor compensation the approximate rotation is slightly modified by using a "double rotation" with $j = j+1$ in order to avoid the square-root in the scaling factor [44]. Therefore, the unscaled approximate double μ-rotation $\mathbf{G}_{\text{du}}(j)$ becomes

$$\mathbf{G}_{\text{du}}(j) = \begin{pmatrix} 1-2^{-2j} & \sigma 2^{-j+1} \\ -\sigma 2^{-j+1} & 1-2^{-2j} \end{pmatrix}. \tag{21}$$

The scaled μ-rotation is $\mathbf{G}_{\mathrm{ds}}(j) = K_j^2 \cdot \mathbf{G}_{\mathrm{du}}(j)$, where $K_j^2 = 1/(1+2^{-2j})$ can be recursively computed by shift-add operations as follows [39]:

$$K_0^2 = \frac{1}{2}, \quad K_j^2 = (1-2^{-2j}) \prod_{k=1}^{w} (1+2^{-2^{k+1}j}) \text{ for } j > 0 \text{ with } w = \log_2 \lfloor \frac{b}{2j} \rfloor. \tag{22}$$

The number of shift and add operations for scaling decreases for increasing value of j (i.e. for small angles), e.g. for $j > b/2$ no scaling is requires at all.

The optimal CORDIC-based approximate rotation given by the optimal angle α_ℓ is defined by the CORDIC angle α_j which is closest to the exact rotation angle θ, i.e.

$$||\theta| - \alpha_\ell| = \min_{j \in \mathcal{J}} ||\theta| - \alpha_j|. \tag{23}$$

The optimal shift value ℓ, representing α_ℓ, can be determined from the input values x and y as follows:

- If $|y| > |x|$ then rotate $(x \;\; y)^{\mathrm{T}}$ by $-\pi/2$, so that $\tan\theta \leq 1$ holds. This preparatory step also has to be taken into account later when applying the approximate Givens rotations.
- Subsequently, the rotation coefficient σ, i.e. the direction of rotation is determined according to
$$\sigma = \mathrm{sign}(x) \cdot \mathrm{sign}(y). \tag{24}$$
- Since $\tan\theta = |y|/|x| \approx 2^{-\ell}$, it follows that $|x| \approx 2^{\ell} \cdot |y|$ so an estimate ℓ_{e} of ℓ can be derived through a simple binary "scaling operation". E.g. if x and y are represented in fixed-point format, ℓ_{e} is equal to the number of binary left shifts which need to be applied to y in order to normalize its value relative to x, i.e. to align their most significant digits. This leaves us with two possible values for ℓ:
$$\ell \in \{\ell_{\mathrm{e}} - 1, \ell_{\mathrm{e}}\}. \tag{25}$$
- We can then identify which "double rotation angle" (either $\tilde{\theta}_1 = 2\alpha_{\ell_{\mathrm{e}}}$ or $\tilde{\theta}_2 = 2\alpha_{\ell_{\mathrm{e}}+1}$) is closer to θ by comparing their values with the angle $\alpha_{\ell_{\mathrm{e}}} + \alpha_{\ell_{\mathrm{e}}+1}$ which lies exactly between the two. This can be done by computing $(\breve{x} \;\; \breve{y})^{\mathrm{T}} = \mathbf{G}_{\mathrm{u}}(\ell_{\mathrm{e}})\mathbf{G}_{\mathrm{u}}(\ell_{\mathrm{e}}+1)(x \;\; y)^{\mathrm{T}}$ and checking the sign of $\breve{y}$:
$$\ell = \begin{cases} \ell_{\mathrm{e}} - 1 & \text{if } \breve{y} > 0, \\ \ell_{\mathrm{e}} & \text{otherwise.} \end{cases} \tag{26}$$
Note that here only unscaled rotations are necessary.

As can be seen in **Fig. 2** this procedure guarantees that $|d(|\tau|, \ell)| < 0.51$ (dash-dotted curve). The solid line shows that $|d(|\tau|, \ell)| \leq 1/3$ for the original approximation using a single μ-rotation. Note however the fast convergence of the two schemes with increasing ℓ.

The CORDIC-based approximate Givens rotation for rotating $(x \;\; y)^{\mathrm{T}}$ is summarized as follows:

- Determine the optimal shift value ℓ via the procedure described above.
- Set $\ell = \ell + 1$ and perform the unscaled double μ-rotation $(\bar{x} \;\; \bar{y})^{\mathrm{T}} = \mathbf{G}_{\mathrm{du}}(\ell) \cdot (x \;\; y)^{\mathrm{T}}$.
- Finally, compute $(x' \;\; y')^{\mathrm{T}} = K_\ell^2 \cdot (\bar{x} \;\; \bar{y})^{\mathrm{T}}$ according to Eq. (22).

The CORDIC-based approximate rotation can also be used to generate accurate rotations by applying the described procedure iteratively. Thereby, only those CORDIC angles, which are really necessary are actually executed. The QRD-RLS algorithm can thus be performed with a chosen accuracy, depending upon the number r of optimal CORDIC angles employed to perform the Givens rotations.

3.2: Simulation results

The adaptive channel equalization computer experiment as described in [9] and a time reference beamformer (TRB) [2] example are used to illustrate the performance of the QRD-RLS algorithm

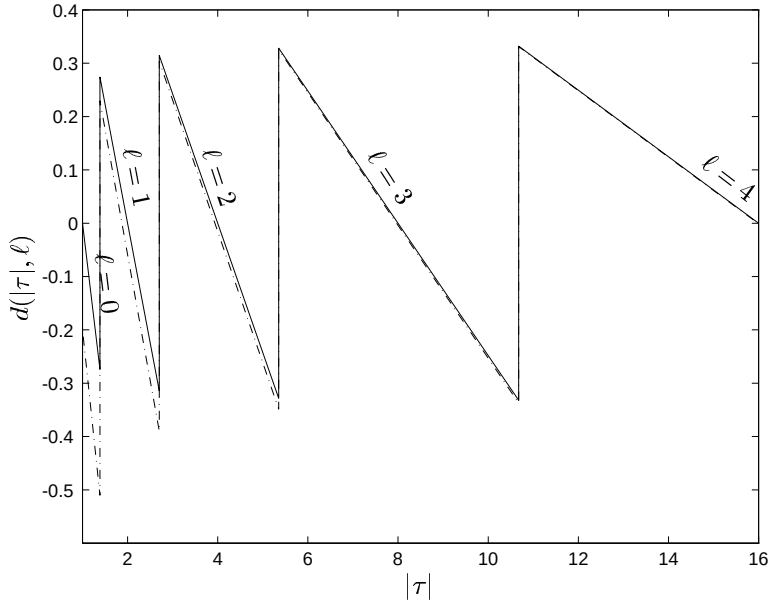

Figure 2. Reduction factor $d(|\tau|, \ell)$ for the CORDIC-based approximate Givens rotation: original approximation (solid), double rotation method (dash-dotted).

based on approximate rotations employing different values for r. Hereby, the reduced data vector $\mathbf{x}'[t]$ is excluded from the further computations and replaced with a new input vector $\mathbf{x}[t+1]$. Nevertheless, it can be shown that covergence is guaranteed, since the reduction factor d satisfies $0 \leq |d| < 1$.

In **Fig. 3a** the ensamble average of the squared a priori error $\overline{|e[t, t-1]|^2}$ is shown for the QRD-RLS algorithm using different approximation levels r as well as exact rotations together with the LMS algorithm applied to a situation with strong distortion (i.e. large eigenvalue spread $\chi(\mathbf{C}) = 46.8216$ for $W = 3.5$). Obviously, using $r = 3$ optimal CORDIC angles per rotation results in the same performance as attained with exact rotations. The misadjustment error decreases as r increases. On the other hand, the convergence of the approximate QRD-RLS does not depend on r.

Instead of working with a fixed number of CORDIC angles per rotation during the whole QRD-RLS algorithm, it is also possible to adapt r during the course of the algorithm, thereby reducing the total amount of required μ-rotations. In stationary situations, the QRD-RLS algorithm usually requires roughly $2M$ steps for convergence. In **Fig. 3b** the ensamble averaged squared a priori error is shown for the case of using $r = 3$ during the whole algorithm and for a scheme employing the following adaptation strategy:

$$\begin{aligned} r &= 1 \quad \text{for } t \leq M, \\ r &= 2 \quad \text{for } M < t \leq 2M, \\ r &= 3 \quad \text{for } t > 2M. \end{aligned}$$

Obviously, this adaptive scheme (dotted line) works just as well as when using $r = 3$ (solid line) throughout.

Fig. 4 shows the beampattern of an $M = 3$ element TRB, the weights of which are adapted via the QRD-RLS algorithm employing different approximation levels r. As can clearly be seen, again $r = 3$ optimal rotation angles are sufficient to achieve comparable performance to a scheme based on exact rotations.

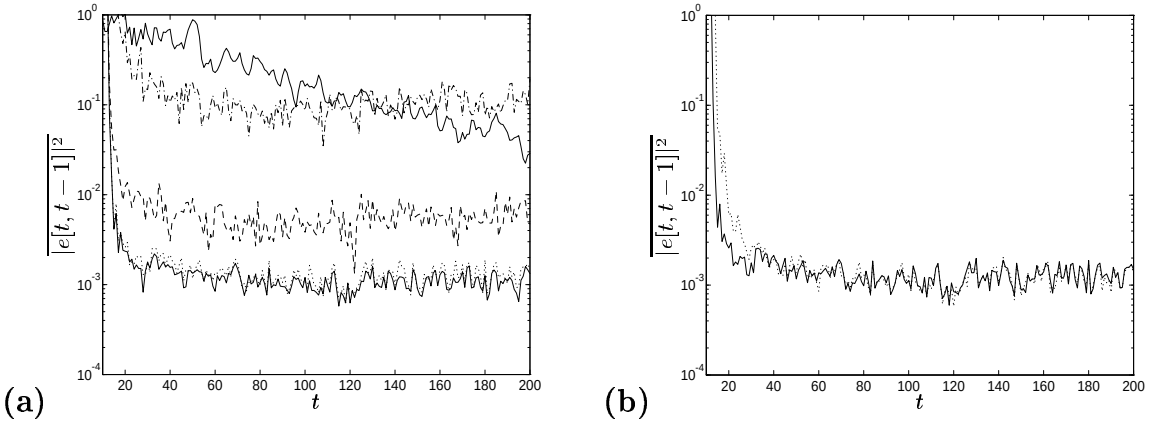

Figure 3. **(a)** Ensamble averaged squared a priori error of 30 independent runs of the channel equalization experiment (filter length $M = 10$, $W = 3.5$, $\mathrm{SNR} = 30\,\mathrm{dB}$) for the LMS algorithm ($\mu = 0.075$) and the QRD-RLS algorithm ($\beta = 1$, $\delta = 0.004$) at different levels of approximation: LMS (solid), QRD-RLS using exact rotations (solid), with $r = 1$ (dash-dotted), $r = 2$ (dashed), and $r = 3$ (dotted) optimal CORDIC angles per rotation. **(b)** QRD-RLS algorithm with $r = 3$ (solid) and the adaptive scheme (dotted).

4: Summary

In this paper practical schemes for performing Givens rotations based on the CORDIC technique where discussed in some detail. CORDIC arithmetic is identified as being highly suited for performing all the operations encountered in QRD-RLS filtering as well as many other signal processing tasks that can be formulated in terms of rotations. The concrete example of a CORDIC processor element (CPE) implementation was presented together with its application in McWhirter's QRD tri-array. In an effort to reduce the computational complexity of CORDIC-based adaptive filtering, we introduced the concept of approximate rotations, requiring only a few elementary angles of the original CORDIC sequence. It was demonstrated that an approximation level of $r = 3$ CORDIC angles per rotation results in the same performance as when employing exact rotations. A reduction of r leads to an increase of the misadjustment error, but does not affect the convergence properties of the original QRD-RLS filtering algorithm. Thus, the proposed approximate QRD-updating scheme offers a great degree of freedom for optimizing the "performance vs. complexity" trade-off as well as opening up the possibility to incorporate incremental refinement strategies.

References

[1] S. H. Nawab, A. V. Oppenheim, A. P. Chandrakasan, J. M. Winograd, and J. T. Ludwig, "Approximate Signal Processing," *J. VLSI Signal Processing Systems*, vol. 15, no. 1/2, pp. 177–200, Jan. 1997.

[2] J. Litva and T. K.-Y. Lo, *Digital Beamforming in Wireless Communications*, Boston, MA: Artech House, 1996.

[3] J. G. Proakis, "Adaptive Equalization for TDMA Digital Mobile Radio," *IEEE Trans. Veh. Technol.*, vol. VT-40, no. 2, pp. 333–341, May 1991.

[4] P. W. Baier, "A Critical Review of CDMA," in *Proc. VTC '96*, Atlanta, GA, Apr. 1996, vol. 1, pp. 6–10.

[5] P. B. Rapajic and B. S. Vucetic, "Adaptive Receiver Structures for Asynchronous CDMA Systems," *IEEE J. Select. Areas Commun.*, vol. SAC-12, no. 4, pp. 685–697, May 1994.

[6] U. Madhow and M. L. Honig, "MMSE Interference Suppression for Direct-Sequence Spread-Spectrum CDMA," *IEEE Trans. Commun.*, vol. COM-42, no. 12, pp. 3178–3188, Dec. 1994.

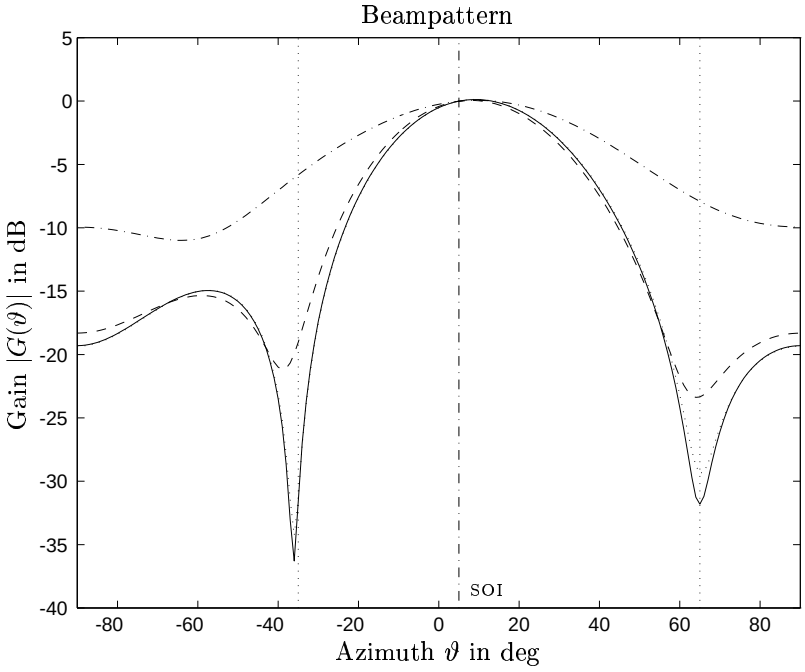

Figure 4. Example beampattern of a TRB with $M = 3$ antenna elements after processing 30 data snapshots ($\mathrm{SNR} = 30\,\mathrm{dB}$, $\mathrm{INR}_1 = 10\,\mathrm{dB}$, $\mathrm{INR}_2 = 10\,\mathrm{dB}$) for the QRD-RLS algorithm ($\beta = 1$, $\delta = 0.004$) at different levels of approximation: QRD-RLS using exact rotations (solid), with $r = 1$ (dash-dotted), $r = 2$ (dashed), and $r = 3$ (dotted) optimal CORDIC angles per rotation.

[7] S. L. Miller, "An Adaptive Direct-Sequence Code-Division Multiple-Access Receiver for Multiuser Interference Rejection," *IEEE Trans. Commun.*, vol. COM-43, no. 2/3/4, pt. III, pp. 1746–1755, Feb./Mar./Apr. 1995.

[8] D. L. Schilling, S. Ghassemzadeh, K. Parsa, and Z. Hadad, "Broadband-CDMA Overlay," *Int. J. Wireless Information Networks*, vol. 2, no. 4, pp. 197–221, Oct. 1995.

[9] S. Haykin, *Adaptive Filter Theory*, Upper Saddle River, NJ: Prentice Hall, 3rd Ed., 1996.

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

[11] S. Y. Kung, "VLSI Array Processors," *IEEE Acoust., Speech, Signal Processing Magazine*, vol. 2, no. 3, pp. 4–22, July 1985.

[12] W. M. Gentleman and H. T. Kung, "Matrix Triangularization by Systolic Arrays," in *Real-Time Signal Processing IV*, Proc. SPIE vol. 298, San Diego, CA, Aug. 1981, pp. 19–26.

[13] J. G. McWhirter, "Recursive Least-Squares Minimization Using a Systolic Array," in *Real Time Signal Processing VI*, Proc. SPIE vol. 431, San Diego, CA, Aug. 1983, pp. 105–112.

[14] F. Lorenzelli and K. Yao, "A Linear Systolic Array for Recursive Least Squares," *IEEE Trans. Signal Processing*, vol. SP-43, no. 12, pp. 3014–3021, Dec. 1995.

[15] W. M. Gentleman, "Least Squares Computations by Givens Transformations without Square Roots," *J. Inst. Math. Appl.*, vol. 12, no. 3, pp. 329–336, Dec. 1973.

[16] S. Hammarling, "A Note on Modifications to the Givens Plane Rotation," *J. Inst. Math. Appl.*, vol. 13, pp. 215–218, 1974.

[17] W. Rath, "Fast Givens Rotations for Orthogonal Similarity Transformations," *Numerische Mathematik*, vol. 40, pp. 47–56, 1982.

[18] J. L. Barlow and I. C. F. Ipsen, "Scaled Givens Rotations for the Solution of Linear Least Squares Problems on Systolic Arrays," *SIAM J. Sci. Stat. Comput.*, vol. 8, no. 5, pp. 716–733, Sept. 1987.

[19] S. F. Hsieh, K. J. R. Liu, and K. Yao, "A Unified Square-Root-Free Approach for QRD-Based Recursive Least Squares Estimation," *IEEE Trans. Signal Processing*, vol. SP-41, no. 3, pp. 1405–1409, Mar. 1993.

[20] J. Götze and U. Schwiegelshohn, "A Square Root and Division Free Givens Rotation for Solving Least Squares Problems on Systolic Arrays," *SIAM J. Sci. Stat. Comput.*, vol. 12, pp. 800–807, Dec. 1991.

[21] E. N. Frantzeskakis, K. J. R. Liu, "A Class of Square Root and Division Free Algorithms and Architectures for QRD-Based Adaptive Signal Processing," *IEEE Trans. Signal Processing*, vol. SP-42, no. 9, pp. 2455–2469, Sept. 1994.

[22] K. J. Raghunath and K. K. Parhi, "Pipelined RLS Adaptive Filtering Using Scaled Tangent Rotations (STAR)," *IEEE Trans. Signal Processing*, vol. SP-44, no. 10, pp. 2591–2604, Oct. 1996.

[23] K. J. Raghunath and K. K. Parhi, "A 100 MHz Pipelined RLS Adaptive Filter," in *Proc. ICASSP '95*, Detroit, MI, May 1995, vol. 5, pp. 3187–3190.

[24] J. G. McWhirter, R. L. Walke, and J. Kadlec, "Normalised Givens Rotations for Recursive Least Squares Processing," in *VLSI Signal Processing VIII*, IEEE Press, NJ, 1995, pp. 323–332.

[25] J. E. Volder, "The CORDIC Trigonometric Computing Technique," *IRE Trans. Electronic Computers*, vol. EC-8, no. 3, pp. 330–334, Sept. 1959.

[26] J. S. Walther, "A Unified Algorithm for Elementary Functions," in *Proc. Spring Joint Computer Conf.*, Atlantic City, NJ, May 1971, pp. 379–385.

[27] J. R. Cavallaro and A. C. Elster, "A CORDIC Processor Array for the SVD of a Complex Matrix," in *SVD and Signal Processing II – Algorithms, Analysis and Applications*, Elsevier Science Publishers B.V., Amsterdam, 1991, pp. 227–239.

[28] C. R. Ward, P. J. Hargrave, and J. G. McWhirter, "A Novel Algorithm and Architecture for Adaptive Digital Beamforming," *IEEE Trans. Antennas Propagat.*, vol. AP-34, no. 3, pp. 338–346, Mar. 1986.

[29] J. V. McCanny and J. G. McWhirter, "Some Systolic Array Developments in the United Kingdom," *IEEE Computer*, vol. C-20, no. 7, pp. 51–63, July 1987.

[30] Y. H. Hu, "CORDIC-Based VLSI Architectures for Digital Signal Processing," *IEEE Signal Processing Magazine*, vol. 9, no. 3, pp. 16–35, July 1992.

[31] U. Fleisch and M. Streiff, *CORDIC Processor Element for a Systolic/Wavefront Array*, Student Project Report no. IKT-NT 780, ETH Zurich, Winter Semester 1995/96.

[32] G. J. Hekstra and E. F. A. Deprettere, "Floating Point Cordic," in *Proc. 11th Symp. on Computer Arithmetic ARITH11*, Windsor, Ontario, June 1993, pp. 130–137.

[33] A. M. Despain, "Fourier Transform Computers Using CORDIC Iterations," *IEEE Trans. Computers*, vol. C-23, no. 10, pp. 993–1001, Oct. 1974.

[34] B. Haller, M. Streiff, U. Fleisch, and R. Zimmermann, "Hardware Implementation of a Systolic Antenna Array Signal Processor Based on CORDIC Arithmetic," in *Proc. ICASSP '97*, Munich, Germany, Apr. 1997, vol. 5, pp. 4141–4144.

[35] C. M. Rader, "VLSI Systolic Arrays for Adaptive Nulling," *IEEE Signal Processing Magazine*, vol. 13, no. 4, pp. 29–49, July 1996.

[36] G. D. Bolstad and K. B. Neeld, "A CORDIC Based DSP Element for Adaptive Signal Processing," in *Digital Signal Processing Technology*, Proc. SPIE vol. CR57, Orlando, FL, Apr. 1995, pp. 291–313.

[37] P. Kapteijn, E. Deprettere, L. Timmoneri, and A. Farina, "Implementation of the Recursive QR Algorithm on a 2×2 CORDIC Testboard: A Case Study for Radar Applications," in *Proc. EuMC '95*, Bologna, Italy, Sept. 1995, vol. 1, pp. 500–505.

[38] R. Hamill, J. V. McCanny, and R. L. Walke, "Constant Scale Factor, On-Line CORDIC Algorithm in the Circular Coordinate System," in *VLSI Signal Processing VIII*, IEEE Press, NJ, 1995, pp. 562–571.

[39] J. Götze, S. Paul, and M. Sauer, "An Efficient Jacobi-Like Algorithm for Parallel Eigenvalue Computation," *IEEE Trans. Computers*, vol. C-42, no. 9, pp. 1058–1065, Sept. 1993.

[40] N. D. Hemkumar and J. R. Cavallaro, "Jacobi-Like Matrix Factorizations with CORDIC-Based Inexact Diagonalizations," in *Proc. 5th SIAM Conf. on Applied Linear Algebra*, Snowbird, UT, June 1994, pp. 295–299.

[41] J. Götze and P. Rieder, "SVD-Updating Using Orthonormal μ-Rotations," *J. VLSI Signal Processing Systems*, vol. 14, no. 1, pp. 7–17, Oct. 1996.

[42] P. Rieder, J. Götze, M. Sauer, and J. A. Nossek, "Orthogonal Approximation of the Discrete Cosine Transform," in *Proc. ECCTD '95*, Istanbul, Turkey, Aug. 1995, pp. 1003–1006.

[43] P. Rieder, J. Götze, J. A. Nossek, and C. S. Burrus, "Parameterization of Orthogonal Wavelet Transforms and Their Implementation," to appear in *IEEE Trans. Circuits Syst. II*, 1997.

[44] J.-M. Delosme, "CORDIC Algorithms: Theory and Extensions," in *Advanced Algorithms and Architectures for Signal Processing IV*, Proc. SPIE vol. 1152, San Diego, CA, Aug. 1989, pp. 131–145.