An Algorithm for Calculating the QR and Singular Value Decompositions of Polynomial Matrices

Joanne A. Foster, *Member, IEEE*, John G. McWhirter, Martin R. Davies, *Student Member, IEEE*, and Jonathon A. Chambers, *Senior Member, IEEE*

***Abstract*—In this paper, a new algorithm for calculating the QR decomposition (QRD) of a polynomial matrix is introduced. This algorithm amounts to transforming a polynomial matrix to upper triangular form by application of a series of paraunitary matrices such as elementary delay and rotation matrices. It is shown that this algorithm can also be used to formulate the singular value decomposition (SVD) of a polynomial matrix, which essentially amounts to diagonalizing a polynomial matrix again by application of a series of paraunitary matrices. Example matrices are used to demonstrate both types of decomposition. Mathematical proofs of convergence of both decompositions are also outlined. Finally, a possible application of such decompositions in multichannel signal processing is discussed.**

***Index Terms*—Convolutive mixing, multiple-input–multiple-output (MIMO) channel equalization, paraunitary matrix, polynomial matrix QR decomposition (QRD), polynomial matrix singular value decomposition (SVD).**

I. Introduction

POLYNOMIAL matrices have many potential applications in the field of control, but in recent years they have also been used extensively in the areas of digital signal processing and communications. Examples of their applications include broadband adaptive sensor array processing, the description of multiple-input–multiple-output (MIMO) communication channels, broadband subspace decomposition, and also digital filter banks for subband coding or data compression [1], [2]. In the context of this paper, polynomial matrices are used to describe a convolutive mixing process, which occurs, for example, when a set of signals arrive at an array of sensors via multiple paths. This will result in the received signals consisting of weighted and delayed versions of the transmitted signals and the channel matrix required to express this mixing process takes the form of a polynomial matrix, where each element is a finite impulse response (FIR) filter. In this situation, the indeterminate variable of each polynomial element of the channel matrix is z^{-1}, as this is often used to represent a unit delay. A $p \times q$ polynomial matrix $\underline{\mathbf{A}}(z)$ can therefore be expressed as

$$\underline{\mathbf{A}}(z) = \sum_{\tau=t_1}^{t_2} \mathbf{A}(\tau) z^{-\tau} = \begin{bmatrix} \underline{a}_{11}(z) & \underline{a}_{12}(z) & \cdots & \underline{a}_{1q}(z) \\ \underline{a}_{21}(z) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \underline{a}_{p1}(z) & \cdots & \cdots & \underline{a}_{pq}(z) \end{bmatrix} \tag{1}$$

where $\tau \in \mathbb{Z}$, $\mathbf{A}(\tau) \in \mathbb{C}^{p \times q}$ is the matrix of coefficients of $z^{-\tau}$ and $t_1 \leq t_2$, where the values of the parameters t_1 and t_2 are not necessarily positive. From (1), it can be seen that a polynomial matrix can be expressed as either a matrix with polynomial elements or alternatively as a polynomial with matrix coefficients. The polynomial coefficient in the (j,k)th polynomial element of $\underline{\mathbf{A}}(z)$ corresponding to a delay of $z^{-\tau}$ will be denoted as $a_{jk}(\tau)$ and so the polynomial element $\underline{a}_{jk}(z)$ can be expressed as

$$\underline{a}_{jk}(z) = \sum_{\tau=t_1}^{t_2} a_{jk}(\tau) z^{-\tau}. \tag{2}$$

In the context of this paper, this represents the impulse response of the propagation channel from the kth transmitter to the jth sensor.

In the simpler case of a narrowband MIMO transmission, the signal propagation can be modeled as an instantaneous mixture and so a matrix of complex scalar entries is sufficient to describe the mixing process. One approach for communicating over a channel of this form is to estimate initially the channel matrix and then calculate its SVD [3], thus transforming the channel matrix into a diagonal matrix by means of a unitary transformation. This then enables the transmitted signals to be easily retrieved from the received signals by exploiting the diagonal structure of the transformed channel matrix [4]. Alternatively, the channel matrix could be transformed into an upper triangular matrix by calculating its QRD [3]. Using a process of back substitution the transmitted signals can then be recovered. This process is known as MIMO channel equalization and could easily be extended from instantaneous to convolutive signal processing, where polynomial matrices are now observed, but would require a suitable algorithm for calculating either the QRD or SVD of a polynomial matrix. Note that the conventional approach to communicating over polynomial channels is to use orthogonal frequency division multiplexing (OFDM) and thereby exploit a circulant form of the channel matrix [5]. However, the decompositions proposed in this paper facilitate

Manuscript received February 27, 2009; accepted September 03, 2009. First published October 13, 2009; current version published February 10, 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Milica Stojanovic.

J. A. Foster, M. R. Davies and J. A. Chambers are with the Advanced Signal Processing Group, Department of Electronic and Electrical Engineering, Loughborough University, Loughborough LE11 3TU, U.K. (e-mail: J.A.Foster@lboro.ac.uk).

J. G. McWhirter is with the Centre of Digital Signal Processing, School of Engineering, Cardiff University, Cardiff CF10 3XQ, U.K.

Digital Object Identifier 10.1109/TSP.2009.2034325

potentially new time-domain strategies for MIMO communications.

Several methods exist for calculating the decomposition of a polynomial matrix, for example, the Smith–McMillan decomposition [2], [6] and a method developed by Vaidyanathan for factorizing a paraunitary polynomial matrix into a series of elementary rotation and delay matrices [2]. In [1], a polynomial matrix EVD (PEVD) routine, referred to as the second-order sequential best rotation (SBR2) algorithm, is introduced. This algorithm constitutes a direct extension of Jacobi's EVD routine from scalar to polynomial matrices, and can also be used to calculate the SVD of a polynomial channel matrix [1], [7], analogous to how the conventional EVD can be used to obtain the SVD of a matrix with scalar elements. However, there is a problem when using the SBR2 algorithm to formulate this decomposition, resulting in the user being unable to ensure the off-diagonal elements of the diagonal matrix obtained from the decomposition are sufficiently small. This problem is not observed with the polynomial SVD (PSVD) algorithm proposed in this paper.

We extend the ideas developed within the SBR2 algorithm to obtain an algorithm for calculating the QRD of a polynomial matrix (PQRD). The authors have previously proposed a method for calculating this decomposition known as the PQRD by steps (PQRD-BS) algorithm [8]–[10]. However, the algorithm proposed in this paper typically converges faster [11]. Furthermore, the existing PQRD-BS algorithm has demonstrated erratic behavior as it converges [11]. This does not occur with the algorithm proposed in this paper. The new PQRD algorithm is verified by means of a simple numerical example. To the best of our knowledge, there are currently no other techniques for calculating a PQRD. It is then shown that this decomposition can be used to obtain the SVD of a polynomial matrix in Section IV. The previous generalization of the SVD to polynomial matrices, which operates by application of the SBR2 algorithm, is then briefly discussed to show a clear advantage of the proposed algorithm over this approach. A simple example is given to illustrate how the method performs as a decomposition technique. Finally, Section V discusses a potential application of the two decompositions in terms of the equalization of polynomial channel matrices. The purpose of this paper is to introduce fundamentally new polynomial matrix decompositions so this application is only briefly outlined, and more details of possible MIMO communication applications together with bit error rate (BER) evaluations are given in [12]–[14].

A. Choice of Notation

Matrices are denoted as bold upper case characters, vectors by bold lower case characters, and scalars by regular lower case characters. A polynomial matrix, vector, or scalar will also use the qualifier (z) to denote it as a polynomial in the indeterminate variable z^{-1}. Furthermore, to avoid confusion with the notation used for the z-transform of a variable, polynomial quantities will use the additional underline notation, for example, see (1). The superscripts $(\cdot)^H$, $(\cdot)^T$, and $(\cdot)^*$ denote, respectively, the Hermitian conjugate, the transpose, and the complex conjugate of the quantity in brackets. $\mathbf{I}_p$ is used to represent the $p \times p$ identity matrix, and $[\cdot]_{jkt}$ the coefficient of z^{-t} in the (j,k)th element of the polynomial matrix in the square brackets. The set of polynomial matrices, with complex coefficients, will be denoted by $\underline{\mathbb{C}}^{a\times b}$ where a and b are the number of rows and columns in the matrix. Similarly, $\underline{\mathbb{R}}^{a\times b}$ will represent the set of polynomial matrices of dimension a by b whose coefficients are real.

B. Polynomial Matrix Definitions

The order of a polynomial matrix is defined to be the number of coefficient matrices required to express the polynomial matrix, excluding the coefficient matrix of z^0, i.e., for the polynomial matrix $\underline{\mathbf{A}}(z)$ in (1), this is defined by the quantity $(t_2 - t_1)$. The paraconjugate of the polynomial matrix $\underline{\mathbf{A}}(z)$ is defined to be $\widetilde{\underline{\mathbf{A}}}(z) = \underline{\mathbf{A}}_*^T(1/z)$, where $(\cdot)_*$ denotes the complex conjugation of each of the coefficients of the polynomial matrix. The tilde notation will be used to denote paraconjugation throughout the paper. A polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times p}$ is said to be paraunitary if the following statement is true: $\underline{\mathbf{A}}(z)\widetilde{\underline{\mathbf{A}}}(z) = \widetilde{\underline{\mathbf{A}}}(z)\underline{\mathbf{A}}(z) = \mathbf{I}_p$. A polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times p}$ is para-Hermitian if it is equal to its paraconjugate, i.e., if $\widetilde{\underline{\mathbf{A}}}(z) = \underline{\mathbf{A}}(z)$ and so the individual coefficients associated with the polynomial elements satisfy $a_{jk}(t) = a_{kj}^*(-t) \forall t \in \mathbb{Z}$ and for $j,k = 1\ldots,p$. Finally, the Frobenius norm, or F-norm, of the polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times q}$ is defined as

$$\|\underline{\mathbf{A}}(z)\|_F = \sqrt{\sum_{\tau=t_1}^{t_2}\sum_{i=1}^{p}\sum_{j=1}^{q}|a_{ij}(\tau)|^2}. \tag{3}$$

II. The QR Decomposition of a Polynomial Matrix

The PQRD by columns (PQRD-BC) algorithm is a novel technique for factorizing a polynomial matrix into an upper triangular and a paraunitary polynomial matrix. Let $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times q}$, then the objective of the algorithm is to calculate a paraunitary matrix $\underline{\mathbf{Q}}(z) \in \underline{\mathbb{C}}^{p\times p}$ such that

$$\underline{\mathbf{Q}}(z)\underline{\mathbf{A}}(z) \cong \underline{\mathbf{R}}(z) \tag{4}$$

where $\underline{\mathbf{R}}(z) \in \underline{\mathbb{C}}^{p\times q}$ is an upper triangular polynomial matrix. As each element of $\underline{\mathbf{A}}(z)$ is a FIR filter, it is generally not possible to obtain an exactly upper triangular matrix. However, it will be shown that a good approximation is achievable using the PQRD-BC algorithm. The polynomial matrix $\underline{\mathbf{Q}}(z)$ in (4) is computed as a series of elementary delay and rotation matrices [2]. These matrices can be used to formulate a polynomial Givens rotation, which forms a fundamental part of the PQRD-BC algorithm and is therefore introduced first.

A. An Elementary Polynomial Givens Rotation

An elementary polynomial Givens rotation (EPGR) is a polynomial matrix that can be applied to either a polynomial vector or matrix to selectively zero one coefficient of a polynomial element. For simplicity, a 2×2 EPGR is introduced first. An EPGR

takes the form of a Givens rotation preceded by an elementary time shift matrix as follows:

$$\underline{\mathbf{G}}^{(t,\alpha,\theta,\phi)}(z) = \begin{bmatrix} ce^{i\alpha} & se^{i\phi} \\ -se^{-i\phi} & ce^{-i\alpha} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & z^t \end{bmatrix} \tag{5}$$

$$= \begin{bmatrix} ce^{i\alpha} & se^{i\phi}z^t \\ -se^{-i\phi} & ce^{-i\alpha}z^t \end{bmatrix} \tag{6}$$

where c and s define the cosine and sine of the angle θ, respectively. The aim of this matrix, when applied to a polynomial vector $\underline{\mathbf{a}}(z) = [\underline{a}_1(z), \underline{a}_2(z)]^T \in \underline{\mathbb{C}}^{2\times 1}$ as demonstrated by

$$\begin{bmatrix} ce^{i\alpha} & se^{i\phi}z^t \\ -se^{-i\phi} & ce^{-i\alpha}z^t \end{bmatrix} \begin{bmatrix} \underline{a}_1(z) \\ \underline{a}_2(z) \end{bmatrix} = \begin{bmatrix} \underline{a}'_1(z) \\ \underline{a}'_2(z) \end{bmatrix} \tag{7}$$

is to drive a specified coefficient from the polynomial element $\underline{a}_2(z)$ to zero. For example, to zero the coefficient of $z^{-\tau}$, i.e., $a_2(\tau)$, then the lag parameter in the EPGR matrix is set as $t = \tau$ and the rotation angles θ, α, and ϕ are chosen such that

$$\theta = \begin{cases} \tan^{-1}\left(\dfrac{|a_2(\tau)|}{|a_1(0)|}\right), & \text{if } a_1(0) \neq 0 \\ \pi/2, & \text{if } a_1(0) = 0 \end{cases} \tag{8}$$

$$\alpha = -\arg(a_1(0)) \quad \text{and} \quad \phi = -\arg(a_2(\tau)) \tag{9}$$

thus resulting in $a'_2(0) = 0$. Furthermore, following the application of the EPGR, the coefficient $a'_1(0)$ has increased in magnitude squared such that $|a'_1(0)|^2 = |a_1(0)|^2 + |a_2(\tau)|^2$ and this coefficient will also be real. Note that the rotation angles in (9) could have alternatively been chosen such that $\alpha = \arg(a_2(\tau))$ and $\phi = \arg(a_1(0))$ and this will still drive the dominant coefficient $a_2(\tau)$ to zero. However, this alternative approach will not ensure that the coefficient $a'_1(0)$ is a positive real scalar, which is required for uniqueness when a scalar matrix is decomposed with the proposed polynomial QRD algorithm.

The polynomial matrix $\underline{\mathbf{G}}^{(t,\alpha,\theta,\phi)}(z)$ of (6) is paraunitary. This matrix represents a multichannel all-pass filter and accordingly preserves the total signal power at every frequency. Consequently, the transformation given by (7) satisfies $\|\underline{\mathbf{a}}'(z)\|_F = \|\underline{\mathbf{a}}(z)\|_F$. Note that the order of the vector $\underline{\mathbf{a}}(z)$ will increase by $|t|$ under this transformation. This is due to the elementary delay matrix incorporated in the EPGR, which will apply a t-fold delay upon $\underline{a}_2(z)$. As a result, the order of the vector $\underline{\mathbf{a}}(z)$ must increase to accommodate these shifted coefficients.

An EPGR can easily be extended so that it can be applied to a polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times q}$ to drive a single coefficient of one of the polynomial elements to zero. For example, suppose we wish to drive the polynomial coefficient $a_{jk}(t)$ to zero by rotating it with $a_{kk}(0)$. The appropriate EPGR required to do this takes the form of a $p \times p$ identity matrix with the exception of the four elements positioned at the intersection of rows j and k with columns j and k. These elements are given by the four elements of the EPGR $\underline{\mathbf{G}}^{(t,\alpha,\theta,\phi)}(z)$ given in (6). This $p\times p$ EPGR matrix will be defined as $\underline{\mathbf{G}}^{(j,k,t,\alpha,\theta,\phi)}(z)$ where the superscripts j and k have been added to denote the position of the coefficient, which will be driven to zero under the application of the EPGR. The coefficients required for calculating the rotation angles in (8) and (9) then correspond to $a_2(\tau) = a_{jk}(t)$ and $a_1(0) = a_{kk}(0)$. Matrices of this type now form the basis of the proposed algorithm for calculating the PQRD.

III. The PQRD by Columns Algorithm

The PQRD-BC algorithm operates as a series of ordered steps where at each step, all coefficients relating to all polynomial elements beneath the diagonal in one column of the matrix are driven sufficiently small by applying a series of EPGR matrices interspersed with paraunitary inverse delay matrices, described below. The step process will be referred to as a column-step to avoid confusion with the terminology used in the PQRD-BS algorithm [9], [10]. The algorithm begins the first column-step with the first column of the matrix. The iterative process to implement this first step is now explained.

A. A Single Column-Step of the Algorithm

The initial iteration of the first column-step begins by locating the coefficient with maximum magnitude from any of the polynomial elements situated beneath the diagonal in the first column, i.e., the coefficient with maximum magnitude from any of the series of polynomial elements $\underline{a}_{21}(z), \ldots, \underline{a}_{p1}(z)$. Suppose this coefficient is found to be $a_{j1}(t)$, which denotes the coefficient of z^{-t} in the polynomial element $\underline{a}_{j1}(z)$. This coefficient will be referred to as the dominant coefficient and if it is not unique then any one of the dominant coefficients may be chosen.

First, the rotation angles θ, α, and ϕ and the EPGR matrix $\underline{\mathbf{G}}^{(j,1,t,\alpha,\theta,\phi)}(z)$ are calculated according to Section II-A, such that when this matrix is applied to the polynomial matrix $\underline{\mathbf{A}}(z)$ as follows:

$$\underline{\mathbf{A}}'(z) = \underline{\mathbf{G}}^{(j,1,t,\alpha,\theta,\phi)}(z)\underline{\mathbf{A}}(z) \tag{10}$$

the dominant coefficient $a_{j1}(t)$ will have been driven to zero. Following the transformation, $a'_{j1}(0) = 0$ and $|a'_{11}(0)|^2 = |a_{11}(0)|^2 + |a_{j1}(t)|^2$.

Next a paraunitary inverse delay matrix $\underline{\mathbf{B}}^{(j,t)}(z)$ is applied to $\underline{\mathbf{A}}'(z)$ to obtain

$$\underline{\mathbf{A}}''(z) = \underline{\mathbf{B}}^{(j,t)}(z)\underline{\mathbf{A}}'(z). \tag{11}$$

The matrix $\underline{\mathbf{B}}^{(j,t)}(z) \in \mathbb{R}^{p\times p}$ takes the form of an identity matrix with the exception of the jth diagonal element which is z^{-t}. The application of this matrix will apply a t-fold delay to all elements in the jth row of $\underline{\mathbf{A}}'(z)$ such that $a''_{jk}(\tau + t) = a'_{jk}(\tau)$ for $k = 1, \ldots, q$ and $\forall \tau \in \mathbb{Z}$. The purpose of this matrix is to ensure that coefficients of z^0 in $\underline{\mathbf{A}}(z)$ are returned to their original positions following the application of the EPGR, in particular $a''_{kk}(0) = a'_{kk}(-t)$. As a result, this will stop any erratic behavior, which has been previously observed in the PQRD-BS algorithm as it converges [11].

This completes the first iteration of the first column-step of the algorithm. Over this iteration the complete transformation performed to zero the polynomial coefficient $a_{j1}(t)$ is of the form

$$\underline{\mathbf{A}}''(z) = \underline{\mathbf{B}}^{(j,t)}(z)\underline{\mathbf{G}}^{(j,1,t,\alpha,\theta,\phi)}(z)\underline{\mathbf{A}}(z). \tag{12}$$

Note that following this transformation, the order of the matrix has increased by $2|t|$ due to the application of the EPGR and the paraunitary inverse delay matrix. This iterative process is now repeated replacing $\underline{\mathbf{A}}(z)$ with $\underline{\mathbf{A}}''(z)$ until all coefficients

associated with polynomial elements beneath the diagonal in the first column of the matrix are sufficiently small in magnitude and therefore satisfy the following stopping condition:

$$|a_{j1}(t)| < \epsilon \tag{13}$$

for $j = 2, \ldots, p$ and $\forall t \in \mathbb{Z}$ where $\epsilon > 0$ is a prespecified small value. Once this stopping condition has been satisfied, the overall transformation performed in the first column-step of the algorithm is of the form

$$\underline{\mathbf{A}}^1(z) = \underline{\mathbf{Q}}^1(z)\underline{\mathbf{A}}(z) \tag{14}$$

where $\underline{\mathbf{Q}}^1(z) \in \underline{\mathbb{C}}^{p\times p}$ is formed from a series of EPGRs interspersed with paraunitary inverse delay matrices and is therefore paraunitary by construction. Furthermore, following this column-step the first column of the matrix $\underline{\mathbf{A}}(z)$ will satisfy

$$\underline{\mathbf{Q}}^1(z)\begin{bmatrix}\underline{a}_{11}(z)\\ \underline{a}_{21}(z)\\ \vdots \\ \underline{a}_{p1}(z)\end{bmatrix} \cong \begin{bmatrix}\underline{a}^1_{11}(z)\\ 0\\ \vdots \\ 0\end{bmatrix} \tag{15}$$

where $\underline{a}^1_{11}(z)$ is the first diagonal polynomial element of $\underline{\mathbf{A}}^1(z)$.

B. The Algorithm

To begin a subsequent column-step, the iterative process outlined in Section III-A is repeated moving to the next column of the matrix. The columns are visited according to the ordering $k = 1, 2 \ldots, \min\{(p-1), q\}$, which is important for convergence of the algorithm. Following i column-steps of the algorithm, the transformation is of the form

$$\underline{\mathbf{A}}^i(z) = \underline{\mathbf{Q}}^i(z)\underline{\mathbf{A}}(z) \tag{16}$$

where $\underline{\mathbf{Q}}^i(z)$ is formed from a series of EPGRs interspersed with paraunitary inverse delay matrices and is therefore paraunitary by construction. Note that over each iteration of each step of the algorithm, the order of the matrices $\underline{\mathbf{A}}^i(z)$ and $\underline{\mathbf{Q}}^i(z)$ will increase due to the application of both the EPGR and inverse delay step. Over a series of column-steps, the orders of both matrices can become unnecessarily large for an accurate decomposition, with many of the coefficients associated with outer lags of the matrix equal to zero or a very small proportion of the F-norm of the matrix. For this reason, the truncation method described in the Appendix is applied at the end of each iteration of each column-step of the algorithm. This will stop the orders of the matrices becoming unnecessarily large and, as confirmed in Section III-E, an accurate PQRD can still be calculated.

Once all columns of the matrix have been visited, this completes one sweep of the algorithm. The PQRD-BC algorithm, although driving the dominant coefficient at each iteration to zero, only ensures that all coefficients of the series of elements beneath the diagonal in one column are suitably small before moving to the next column in the ordering. Therefore, through future column-steps of the algorithm, these small coefficients could be rotated with other suitably small coefficients, allowing them to increase in magnitude and so multiple sweeps of the algorithm may be required. Despite this, the algorithm is guaranteed to converge, as will be shown later, and generally only a couple of sweeps are ever required. Note that this is not a problem when calculating the QRD of a scalar matrix, where all elements beneath the diagonal are driven precisely to zero. A summary of the algorithm is given in Table I.

TABLE I
SUMMARY OF THE PQRD-BC ALGORITHM

Input: polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times q}$.
Specify: the convergence parameter ϵ, the truncation parameter μ, the maximum number of iterations per column-step of the algorithm MaxIter and the max number of sweeps of the algorithm MaxSweeps.
Set $\underline{\mathbf{Q}}(z) \Leftarrow \mathbf{I}_p$, $g_1 \Leftarrow 1 + \epsilon$ and $n \Leftarrow 0$.
while $n <$ MaxSweeps and $g_1 > \epsilon$ **do**
  $n \Leftarrow n + 1$.
  for $k = 1, \ldots, \min\{p-1, q\}$ **do**
    Set $iter \Leftarrow 0$ and $g_2 \Leftarrow 1 + \epsilon$.
    while iter $<$ MaxIter and $g_2 > \epsilon$ **do**
      Find j and t such that $|a_{jk}(t)| \geq |a_{mk}(\tau)|$ for $m = k+1, \ldots, p$ and $\forall \tau \in \mathbb{Z}$. Set $g_2 \Leftarrow |a_{jk}(t)|$.
      if $g_2 > \epsilon$ **then**
        iter $\Leftarrow$ iter $+ 1$.
        Calculate the rotation parameters (θ, α, ϕ) according to Section II-A.
        Set $\underline{\mathbf{A}}(z) \Leftarrow \underline{\mathbf{B}}^{(j,t)}(z)\underline{\mathbf{G}}^{(j,k,\alpha,\theta,\phi,t)}(z)\underline{\mathbf{A}}(z)$.
        Set $\underline{\mathbf{Q}}(z) \Leftarrow \underline{\mathbf{B}}^{(j,t)}(z)\underline{\mathbf{G}}^{(j,k,\alpha,\theta,\phi,t)}(z)\underline{\mathbf{Q}}(z)$.
        Truncate $\underline{\mathbf{A}}(z)$ and $\underline{\mathbf{Q}}(z)$ according to the Appendix.
      end if
    end while
  end for
  Find j, k and t such that $|a_{jk}(t)| \geq |a_{mn}(\tau)|$ holds for $n = 1, \ldots, q$, $m = n+1, \ldots, p$ and $\forall \tau \in \mathbb{Z}$. Set $g_1 = |a_{jk}(t)|$.
end while
Set $\underline{\mathbf{R}}(z) = \underline{\mathbf{A}}(z)$.

C. Nonuniqueness of the Decomposition

When computing the QRD of complex scalar matrices, then uniqueness of the solution is easily enforced by requiring that the diagonal elements of the upper triangular matrix obtained by the QRD be both real and positive. However, when calculating the QRD of a polynomial matrix, enforcing uniqueness is not so simple due to the additional dimension of the problem arising from the lag index of the polynomials. Assume that the matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times q}$ has a PQRD as follows:

$$\underline{\mathbf{Q}}_1(z)\underline{\mathbf{A}}(z) = \underline{\mathbf{R}}_1(z) \tag{17}$$

where $\underline{\mathbf{Q}}_1(z) \in \underline{\mathbb{C}}^{p\times p}$ is paraunitary and $\underline{\mathbf{R}}_1(z) \in \underline{\mathbb{C}}^{p\times q}$ is upper triangular. It is possible to apply a further diagonal paraunitary matrix $\underline{\mathbf{T}}(z) \in \underline{\mathbb{C}}^{p\times p}$ to this decomposition, with nonzero elements of the form

$$\underline{t}_{jj}(z) = e^{i\beta}z^{-\tau} \tag{18}$$

for $j = 1, \ldots, p$, such that $\underline{\mathbf{Q}}_2(z)\underline{\mathbf{A}}(z) = \underline{\mathbf{R}}_2(z)$, where $\underline{\mathbf{Q}}_2(z) = \underline{\mathbf{T}}(z)\underline{\mathbf{Q}}_1(z) \in \underline{\mathbb{C}}^{p\times p}$ is paraunitary and $\underline{\mathbf{R}}_2(z) \in \underline{\mathbb{C}}^{p\times q}$ is still upper triangular. Therefore, the two matrices obtained by the PQRD-BC algorithm are not uniquely

defined. Note that for the proposed application of the PQRD to MIMO communication problems, discussed in Section V, nonuniqueness of the solutions does not affect end-to-end performance. A unique solution could be ensured by requiring the dominant coefficient of each diagonal element in the upper triangular polynomial matrix to be real and lie in the zero plane, i.e., to be the coefficient of z^0.

D. Proof of Convergence

Convergence of a single column-step is deduced first. The arguments follow directly from those used to prove convergence of the SBR2 algorithm [1]. Suppose the algorithm is at the start of the jth column-step. With every application of an EPGR to zero the dominant coefficient say $a_{kj}(t)$, the quantity $|a_{jj}(0)|^2$ will increase by the magnitude squared of the largest coefficient beneath the diagonal in this column. Furthermore, this quantity is bounded above by the squared F-norm of the polynomial elements on and below the diagonal in the jth column of $\underline{\mathbf{A}}(z)$, i.e., the quantity $\sum_\tau \sum_{n=j}^{p} |a_{nj}(\tau)|^2$, which remains constant throughout all iterations of this column-step. As $|a_{jj}(0)|^2$ is monotonically increasing and bounded above, then it must have a supremum, say s. It follows that for any $\epsilon > 0$ there must be an iteration at which $|s - |a_{jj}(0)|^2| < \epsilon$. Therefore, at a subsequent iteration, the magnitude squared of the maximum coefficient associated with a polynomial element situated beneath the diagonal in this column, denoted as g, must satisfy $g \leq |s - |a_{jj}(0)|^2| < \epsilon$ and so g is bounded by ϵ. Therefore, over a series of EPGRs the stopping condition set by (13) is guaranteed for any prespecified value of $\epsilon > 0$ and the column-step converges in this respect.

Following this column-step, the quantity

$$\sum_{\tau} \sum_{m=1}^{j} \sum_{\substack{n=1 \\ n \geq m}}^{q} |a_{mn}(\tau)|^2 \tag{19}$$

will remain constant through all subsequent column-steps of that particular sweep of the algorithm. However, if the algorithm requires multiple sweeps, then this quantity can decrease. Despite this, the quantity $|a_{11}(0)|^2$ can only ever increase through all column-steps and therefore all sweeps of the algorithm, but this cannot continue indefinitely. As a consequence, a point will be reached where no further rotations are required within the first column of the matrix. Once this has been achieved, the quantity $|a_{22}(0)|^2$ will increase monotonically through all future column-steps until a point is reached where this quantity can no longer increase. Consequently, no further rotations are required in this column. This will continue to happen to all diagonal elements situated on the coefficient plane of order zero, i.e., diagonal elements of the matrix $\mathbf{A}(0)$, working from left to right through the matrix. Consequently, convergence of the algorithm is guaranteed. Note that truncation of the polynomial matrices does not invalidate this proof of convergence, as the magnitude squared of each of the diagonal zero lag coefficients will always be bounded above by the squared F-norm of their column and monotonically increasing in each column-step of the algorithm.

The proof that the order in which the columns of the matrix are visited is important for convergence of the algorithm is clear from this outline. Other approaches for calculating this decomposition have been examined in [8] and have been found to be less efficient than the method described here. For example, an earlier effort of the authors operated as an entirely iterative procedure to formulate this decomposition, which did not require a process of steps. This approach drove the largest off-diagonal coefficient associated with any polynomial element positioned beneath the diagonal of the matrix to zero at each iteration by means of an EPGR. However, although this method will converge, it is generally slower to implement than the PQRD-BC algorithm when applied to matrices of larger dimension than 2×2.[1] This is because the application of an EPGR will modify all elements in two rows of the matrix. To minimize the number of EPGRs required, it is necessary to avoid rotating coefficients that have previously been driven to zero with nonzero coefficients as this could force the coefficients, which are equal to zero, to increase in magnitude. As a result, the coefficients of the polynomial elements beneath the diagonal of the matrix most likely will have to be driven to zero a number of times. This is easily avoided by visiting the polynomial elements of the matrix in a specific order, such as the column ordering used in the PQRD-BC algorithm. Without this structured approach, working through the columns of the matrix from left to right, it will typically require the application of more EPGRs for a polynomial matrix to be transformed into an upper triangular polynomial matrix.

The previous implementation of this decomposition, the PQRD-BS algorithm, also used an ordered approach. In fact, this algorithm was a direct generalization of the conventional scalar matrix QRD and therefore operated by driving each of the polynomial elements situated beneath the diagonal of the matrix approximately to zero in turn. However, the PQRD-BS algorithm required more iterative subroutines within the algorithm and would generally, as a result of this, require more EPGRs to converge. Although this ordered method makes sense when dealing with scalar matrices, this approach is not necessarily the best for polynomial matrix decompositions. The algorithm presented in this paper was developed from the previous approaches in order to address these issues. Note that each of these different methods of formulating a PQRD will not generate the same paraunitary or upper triangular matrices due to the nonuniqueness discussed above. Furthermore, different decompositions will result in matrices of varying orders. The PQRD-BC algorithm is now demonstrated by a couple of simple examples.

E. Worked Examples

Example 1: For the first example, the PQRD-BC algorithm was applied to the polynomial matrix

$$\underline{\mathbf{A}} = \begin{bmatrix} 2 & 0 & 2z^{+1} \\ z^{+1} & 1 & 0 \\ 0 & z^{-1} & 1 \end{bmatrix} \tag{20}$$

which has only two nonzero coefficients associated with the polynomial elements situated beneath the diagonal of the polynomial matrix to eliminate. When applied to this polynomial

[1] In the 2×2 case, the two methods are the same.

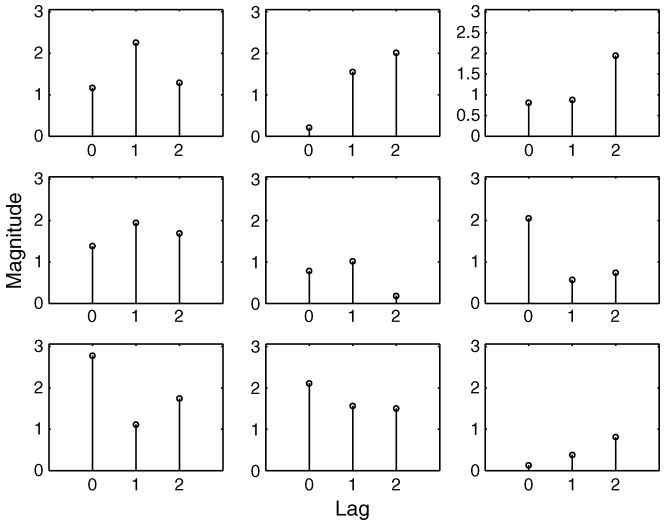

Fig. 1. Stem plot representation of the polynomial matrix $\underline{\mathbf{A}}(z)$ to be used as input to the PQRD-BC algorithm.

matrix, the algorithm required only a single sweep consisting of three iterations, to achieve the decomposition $\underline{\mathbf{Q}}(z)\underline{\mathbf{A}}(z) = \underline{\mathbf{R}}(z)$, where all coefficients associated with polynomial elements beneath the diagonal of the upper triangular polynomial matrix $\underline{\mathbf{R}}(z)$ are equal to zero to computational precision. Furthermore, for this simple example, it was not necessary to truncate the orders of the polynomial matrices at any point. The paraunitary and upper triangular polynomial matrices obtained from the decomposition are

$$\underline{\mathbf{Q}}(z) = \begin{bmatrix} 0.8944 & 0.4472z^{-1} & 0 \\ -0.2981z^{+1} & 0.5963 & 0.7454z^{+1} \\ 0.3333z^{+1} & -0.6667 & 0.6667z^{+1} \end{bmatrix} \quad (21)$$

and

$$\underline{\mathbf{R}}(z) = \begin{bmatrix} 2.2361 & 0.4472z^{-1} & 1.7889z^{+1} \\ 0 & 1.3416 & 0.7454z^{+1} - 0.5963z^{+2} \\ 0 & 0 & 0.6667z^{+1} + 0.6667z^{+2} \end{bmatrix}. \quad (22)$$

The inverse decomposition can be calculated as $\underline{\mathbf{A}}(z) = \underline{\widetilde{\mathbf{Q}}}(z)\underline{\mathbf{R}}(z)$. To ensure that an accurate decomposition has been calculated, the relative error can be calculated as

$$E_{rel} = \left\| \underline{\mathbf{A}}(z) - \underline{\widetilde{\mathbf{Q}}}(z)\underline{\mathbf{R}}(z) \right\|_F / \left\| \underline{\mathbf{A}}(z) \right\|_F . \quad (23)$$

This measure was found to equal zero (to computational precision) for this example. It is worth noting how simple and accurate this decomposition is compared with performing numerous scalar QR decompositions in the frequency domain.

Example 2: A polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{3\times 3}$ was then generated, with each element chosen to be a second-order FIR filter, where both the real and imaginary parts of the coefficients were drawn from a Gaussian distribution with mean zero and unit variance. A graphical representation of this polynomial matrix can be seen in Fig. 1, where a stem plot has been used to show the magnitude of the series of coefficients for each of the polynomial elements. The position of the stem plot in each figure corresponds to the position of the polynomial element, which it represents within the polynomial matrix.

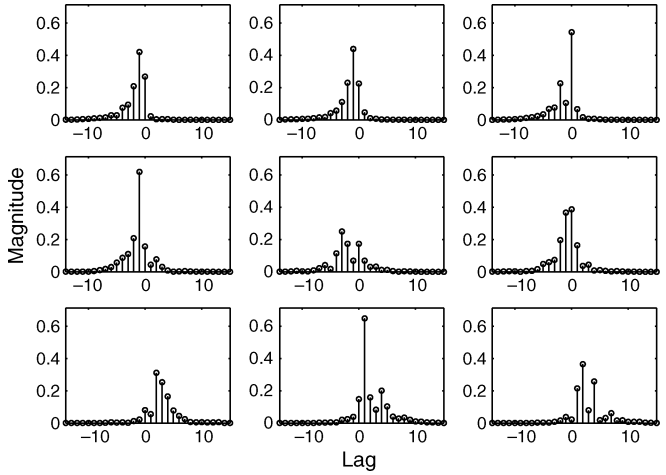

Fig. 2. Stem plot representation of the paraunitary matrix $\underline{\mathbf{Q}}(z)$ obtained by applying the PQRD-BC algorithm to the polynomial matrix $\underline{\mathbf{A}}(z)$.

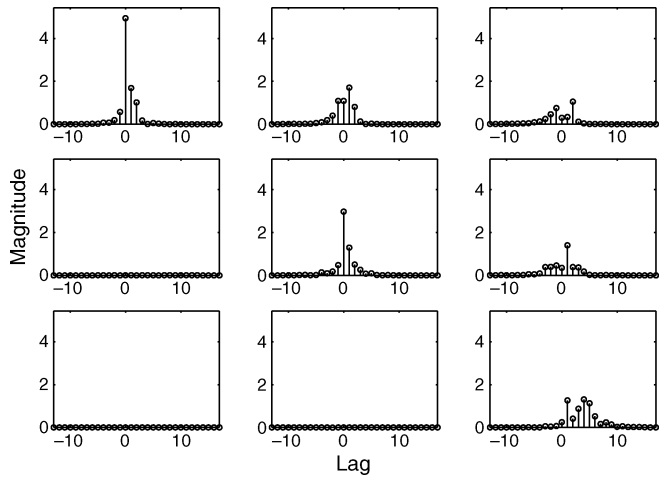

Fig. 3. Stem plot representation of the approximately upper triangular matrix $\underline{\mathbf{R}}(z)$ obtained by applying the PQRD-BC algorithm to the matrix $\underline{\mathbf{A}}(z)$.

The PQRD-BC algorithm was applied to this polynomial matrix with the stopping criterion $\epsilon = 10^{-2}$ and using the polynomial matrix truncation method with μ set equal to 10^{-7}. Only a single sweep of the algorithm was required to ensure that the stopping condition was satisfied, requiring a total of 126 EPGRs. Following the decomposition, the coefficient of the polynomial element with maximum magnitude situated beneath the diagonal of $\underline{\mathbf{R}}(z)$ was found to be 9.3×10^{-3}. The paraunitary polynomial matrix $\underline{\mathbf{Q}}(z)$ (of order 29) and the approximately upper triangular polynomial matrix $\underline{\mathbf{R}}(z)$ (of order 30) obtained from the decomposition can be seen in Figs. 2 and 3, respectively. The relative error for this example can be calculated according to (23) and was found to be 1.2×10^{-3}, confirming that truncating the polynomial matrices has not significantly compromised the decomposition. Note that if this measure is too large, then the value of μ can of course be decreased. For the potential application of the decomposition a strictly upper triangular matrix is required and so the relative error can again be calculated, however, now setting all polynomial elements beneath the diagonal of the matrix $\underline{\mathbf{R}}(z)$ equal to zero. This measure is now found to be 5×10^{-3},

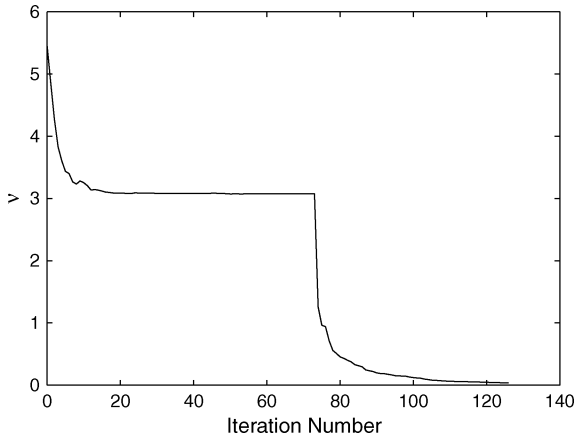

Fig. 4. F-norm of all polynomial elements beneath the diagonal ν over the series of iterations of the PQRD-BC algorithm.

confirming that a good approximation of the decomposition has been performed. Finally, to demonstrate convergence of the algorithm, the F-norm of the polynomial elements beneath the diagonal of the polynomial matrix $\underline{\mathbf{A}}(z)$ was calculated following each iteration of each column-step of the algorithm. This measure ν is illustrated in Fig. 4, where the two stages of convergence, corresponding to the two column-steps, can clearly be seen. The final value of this measure was found to be 3.7×10^{-2}, which accounts for 0.49% of the F-norm of the entire matrix ($||\underline{\mathbf{A}}(z)||_F = 7.56$), confirming that the matrix $\underline{\mathbf{R}}(z)$ is suitably upper triangular.

IV. The SVD of a Polynomial Matrix

The PSVD of a polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times q}$ is given as

$$\underline{\mathbf{U}}(z)\underline{\mathbf{A}}(z)\underline{\widetilde{\mathbf{V}}}(z) \cong \underline{\mathbf{\Lambda}}(z) \tag{24}$$

where $\underline{\mathbf{\Lambda}}(z) = \mathrm{diag}\{\underline{\lambda}_{11}(z), \ldots, \underline{\lambda}_{qq}(z)\} \in \underline{\mathbb{C}}^{p\times q}$ denotes a diagonal polynomial matrix and the matrices $\underline{\mathbf{U}}(z) \in \underline{\mathbb{C}}^{p\times p}$ and $\underline{\mathbf{V}}(z) \in \underline{\mathbb{C}}^{q\times q}$ are paraunitary. A novel algorithm is now proposed for calculating this decomposition, which operates by iteratively calculating the PQRD of the polynomial matrix. As each polynomial element of $\underline{\mathbf{A}}(z)$ constitutes a FIR filter, it is often not possible to achieve exact diagonalization of a polynomial matrix. However, as the results in this paper will confirm, a good approximation can be achieved using the proposed algorithm.

A. Calculating the PSVD Using the PQRD-BC Algorithm

This algorithm will be referred to as the PSVD by PQRD algorithm and operates as an iterative process where at each iteration two paraunitary matrices, formulated using the PQRD-BC algorithm, are applied to the polynomial matrix $\underline{\mathbf{A}}(z)$. Over a series of iterations, these paraunitary matrices will transform the matrix into an approximately diagonal matrix.

The algorithm begins the first iteration by calculating the PQRD of the matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times q}$ such that

$$\underline{\mathbf{U}}_1(z)\underline{\mathbf{A}}(z) = \underline{\mathbf{R}}_1(z) \tag{25}$$

where $\underline{\mathbf{R}}_1(z) \in \underline{\mathbb{C}}^{p\times q}$ is an approximately upper triangular polynomial matrix and $\underline{\mathbf{U}}_1(z) \in \underline{\mathbb{C}}^{p\times p}$ is paraunitary. Following this transformation the coefficients of the polynomial elements beneath the diagonal of $\underline{\mathbf{R}}_1(z)$ will satisfy $|[\underline{\mathbf{R}}_1(z)]_{jkt}| < \epsilon$ for $j = 2, \ldots, p$, $k = 1, \ldots, \min\{j-1, q\}$ with $j > k$ and $\forall t \in \mathbb{Z}$, where $\epsilon > 0$ is the stopping criterion of the PQRD-BC algorithm.

Next, the PQRD of $\underline{\mathbf{A}}'(z) = \underline{\widetilde{\mathbf{R}}}_1(z) \in \underline{\mathbb{C}}^{q\times p}$ is calculated such that

$$\underline{\mathbf{V}}_1(z)\underline{\mathbf{A}}'(z) = \underline{\mathbf{R}}_2(z) \tag{26}$$

where $\underline{\mathbf{R}}_2(z) \in \underline{\mathbb{C}}^{q\times p}$ is an approximately upper triangular polynomial matrix and $\underline{\mathbf{V}}_1(z) \in \underline{\mathbb{C}}^{q\times q}$ is a paraunitary polynomial matrix. Again, the coefficients of the polynomial elements beneath the diagonal of the upper triangular matrix will be less than ϵ in magnitude following this transformation. This completes the first iteration of the PSVD by PQRD algorithm, where the overall decomposition following this iteration is of the form

$$\underline{\mathbf{U}}_1(z)\underline{\mathbf{A}}(z)\underline{\widetilde{\mathbf{V}}}_1(z) = \underline{\mathbf{A}}_1(z) \tag{27}$$

where $\underline{\mathbf{A}}_1(z) = \underline{\widetilde{\mathbf{R}}}_2(z)$. This iterative process is then repeated replacing $\underline{\mathbf{A}}(z)$ with $\underline{\mathbf{A}}_1(z)$ until all coefficients of the off-diagonal polynomial elements of this matrix are deemed sufficiently small according to the stopping condition

$$|a_{jk}(t)| < \epsilon \tag{28}$$

$\forall t \in \mathbb{Z}$, $j = 1, \ldots, p$, $k = 1, \ldots, q$ such that $j \neq k$ and where $\epsilon > 0$ is the same convergence measure as used when calculating the PQRDs in (25) and (26).

Following i iterations of the algorithm, the overall decomposition performed is of the form

$$\underline{\mathbf{U}}(z)\underline{\mathbf{A}}(z)\underline{\widetilde{\mathbf{V}}}(z) = \underline{\mathbf{A}}_i(z) \tag{29}$$

where $\underline{\mathbf{U}}(z) = \underline{\mathbf{U}}_i(z)\ldots\underline{\mathbf{U}}_1(z)$ and $\underline{\widetilde{\mathbf{V}}}(z) = \underline{\widetilde{\mathbf{V}}}_1(z)\ldots\underline{\widetilde{\mathbf{V}}}_i(z)$. Both of these matrices are clearly paraunitary by construction and as a result the transformation will be norm preserving, i.e., $||\underline{\mathbf{A}}(z)||_F^2 = ||\underline{\mathbf{A}}_i(z)||_F^2$. Furthermore, the matrix $\underline{\mathbf{A}}_i(z)$ will converge to a diagonal matrix according to the stopping condition given by inequality (28), provided a sufficient number of iterations has been completed. A concise description of this algorithm is given in Table II. Note that as with the PQRD, the matrices obtained from this decomposition are not unique. For example, a diagonal paraunitary matrix with nonzero elements of the form defined in (18) can be applied from the left or right to the matrix $\underline{\mathbf{\Lambda}}(z)$ to give a resulting matrix that is still diagonal. Similarly, matrices of this form could be applied from both sides and the resulting decomposition will still be a PSVD. Nonuniqueness does not degrade end-to-end performance for the proposed application of this decomposition in Section V.

B. Proof of Convergence

Throughout all iterations of the PSVD by PQRD algorithm, the quantity $|a_{11}(0)|^2$ is monotonically increasing. It will increase as a consequence of driving any coefficient of any off-diagonal polynomial element in the first column or row of the matrix to zero and is never affected by any rotations applied to zero

TABLE II
SUMMARY OF THE PSVD BY PQRD ALGORITHM

Input polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times q}$.
Specify the convergence parameter ϵ, the truncation parameter μ and the maximum number of iterations of the algorithm MaxIter.
Set $\underline{\mathbf{U}}(z) \Leftarrow \mathbf{I}_p$, $\underline{\mathbf{V}}(z) \Leftarrow \mathbf{I}_q$, iter $\Leftarrow 0$ and $g \Leftarrow 1+\epsilon$.
while iter $<$ MaxIter and $g > \epsilon$ **do**
  Find indices j, k and t where $j \neq k$ such that $|a_{jk}(t)| \geq |a_{mn}(\tau)|$ holds for $m = 1,\ldots,p$, $n = 1,\ldots,q$ such that $m \neq n$ and $\forall \tau \in \mathbb{Z}$. Set $g \Leftarrow |a_{jk}(t)|$.
  if $g > \epsilon$ **then**
    iter $\Leftarrow$ iter $+ 1$.
    Calculate the PQRD of $\underline{\mathbf{A}}(z)$: $\underline{\mathbf{U}}_1(z)\underline{\mathbf{A}}(z) = \underline{\mathbf{R}}_1(z)$.
    Set $\underline{\mathbf{A}}'(z) \Leftarrow \widetilde{\underline{\mathbf{R}}}_1(z)$ and $\underline{\mathbf{U}}(z) \Leftarrow \underline{\mathbf{U}}_1(z)\underline{\mathbf{U}}(z)$.
    Calculate the PQRD of $\underline{\mathbf{A}}'(z)$: $\underline{\mathbf{V}}_1(z)\underline{\mathbf{A}}'(z) = \underline{\mathbf{R}}_2(z)$.
    Set $\underline{\mathbf{V}}(z) \Leftarrow \underline{\mathbf{V}}_1(z)\underline{\mathbf{V}}(z)$ and $\underline{\mathbf{A}}(z) = \widetilde{\underline{\mathbf{R}}}_2(z)$.
    Truncate $\underline{\mathbf{A}}(z)$, $\underline{\mathbf{U}}(z)$ and $\underline{\mathbf{V}}(z)$ according to the Appendix.
  end if
end while
Set $\underline{\mathbf{\Lambda}}(z) = \underline{\mathbf{A}}(z)$.

off-diagonal coefficients in any other column or row of the matrix. In addition, this quantity will never be affected by the application of elementary delay matrices throughout any step of the algorithm. Now, this quantity cannot continue to increase indefinitely as it is bounded above by $\|\underline{\mathbf{A}}(z)\|_F^2$. Therefore, a point must be reached whereby all off-diagonal coefficients in the first row and column of the matrix are less than ϵ in magnitude by analogy with the proof of convergence of the PQRD-BC algorithm. Note that following each PQRD, the coefficients beneath the diagonal of the matrix are guaranteed to be less than ϵ in magnitude. These coefficients could then increase in magnitude through a subsequent application of the PQRD-BC algorithm, where they are now positioned in the area above the diagonal of the matrix. However, at the next step of the algorithm, any coefficients of any polynomial elements positioned above the diagonal will be moved into columns positioned to the left of their initial position, where they will then be systematically driven to zero according to the ordering within the PQRD-BC algorithm. Due to this right to left movement of the off-diagonal elements through the matrix the algorithm will converge. Once convergence of the first column and row of the matrix has been achieved, the quantity $|a_{22}(0)|^2$ will increase monotonically until no further rotations are required in either the second row or column of the matrix. This will continue through the matrix working from left to right. As with the PQRD-BC algorithm, applying the truncation method throughout the algorithm does not invalidate this proof of convergence.

C. Calculating the PSVD Using the SBR2 Algorithm

Just as the conventional scalar matrix EVD can be used to generate the SVD of a matrix with complex scalar entries, the PEVD routine SBR2 can also be used to generate the SVD of a matrix with polynomial elements. For example, suppose we wish to calculate the PSVD of the polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times q}$ according to (24). From this matrix, the two para-Hermitian matrices $\underline{\mathbf{A}}(z)\widetilde{\underline{\mathbf{A}}}(z) \in \underline{\mathbb{C}}^{p\times p}$ and $\widetilde{\underline{\mathbf{A}}}(z)\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{q\times q}$ can be formulated. These matrices could alternatively be expressed, using the decomposition of (24), as

$$\underline{\mathbf{A}}(z)\widetilde{\underline{\mathbf{A}}}(z) = \widetilde{\underline{\mathbf{U}}}(z)\underline{\mathbf{\Lambda}}(z)\widetilde{\underline{\mathbf{\Lambda}}}(z)\underline{\mathbf{U}}(z) \tag{30}$$

and

$$\widetilde{\underline{\mathbf{A}}}(z)\underline{\mathbf{A}}(z) = \widetilde{\underline{\mathbf{V}}}(z)\widetilde{\underline{\mathbf{\Lambda}}}(z)\underline{\mathbf{\Lambda}}(z)\underline{\mathbf{V}}(z) \tag{31}$$

where the matrices $\underline{\mathbf{\Lambda}}(z)\widetilde{\underline{\mathbf{\Lambda}}}(z) \in \underline{\mathbb{C}}^{p\times p}$ and $\widetilde{\underline{\mathbf{\Lambda}}}(z)\underline{\mathbf{\Lambda}}(z) \in \underline{\mathbb{C}}^{q\times q}$ are both diagonal. As (30) and (31) constitute, respectively, the PEVD of the matrices $\underline{\mathbf{A}}(z)\widetilde{\underline{\mathbf{A}}}(z)$ and $\widetilde{\underline{\mathbf{A}}}(z)\underline{\mathbf{A}}(z)$, the paraunitary matrices $\underline{\mathbf{U}}(z)$ and $\underline{\mathbf{V}}(z)$ can be calculated by applying the SBR2 algorithm to $\underline{\mathbf{A}}(z)\widetilde{\underline{\mathbf{A}}}(z)$ and $\widetilde{\underline{\mathbf{A}}}(z)\underline{\mathbf{A}}(z)$ in turn [1]. As with the PQRD-BC algorithm, the SBR2 algorithm operates as an iterative process and so will generate approximately diagonal matrices subject to a suitable stopping condition. For example, it can be set to drive all coefficients associated with the off-diagonal polynomial elements of the matrices $\underline{\mathbf{\Lambda}}(z)\widetilde{\underline{\mathbf{\Lambda}}}(z)$ and $\widetilde{\underline{\mathbf{\Lambda}}}(z)\underline{\mathbf{\Lambda}}(z)$ to be less in magnitude than a specified value $\delta > 0$ and so, as given in [1], a good approximation is achievable. An approximation of the diagonal matrix $\underline{\mathbf{\Lambda}}(z)$ is then calculated, using $\underline{\mathbf{U}}(z)$ and $\underline{\mathbf{V}}(z)$, according to (24).

However, there is a fundamental problem when using this method—it is impossible to have any direct control over the size of the coefficients associated with the off-diagonal polynomial elements of the matrix $\underline{\mathbf{\Lambda}}(z)$ and so the level of the approximation cannot be specified in advance. For example, if the magnitude of the off-diagonal coefficients of the polynomial elements of $\underline{\mathbf{\Lambda}}(z)\widetilde{\underline{\mathbf{\Lambda}}}(z)$ are driven less than $\delta > 0$, then these coefficients must satisfy

$$\left|\left[\underline{\mathbf{\Lambda}}(z)\widetilde{\underline{\mathbf{\Lambda}}}(z)\right]_{jkt}\right| = \left|\sum_{l=1}^{q}\sum_{t_1}\sum_{t_2-t_1=t}\lambda_{jl}(t_1)\lambda_{kl}^{*}(t_2)\right| < \delta \tag{32}$$

for $j,k = 1,\ldots,p$ where $j \neq k$ and $\forall t \in \mathbb{Z}$. However, this does not apply to the magnitude of the coefficients associated with the off-diagonal polynomial elements of $\underline{\mathbf{\Lambda}}(z)$, i.e., it is difficult to place an upper bound on the value of $|\lambda_{jk}(t)|$ for $j = 1,\ldots,p$, $k = 1,\ldots,q$ where $j \neq k$ and $\forall t \in \mathbb{Z}$. Therefore, the value of δ cannot be chosen to ensure that the polynomial matrix $\underline{\mathbf{\Lambda}}(z)$ is suitably diagonal following the decomposition. Clearly, for the potential application of the decomposition to MIMO communications, where a strictly diagonal matrix is required for channel equalization, this could affect the error rate performance. Furthermore, to generate a diagonal matrix whose coefficients of the off-diagonal polynomial elements are approximately of the same magnitude as those obtained using the PSVD-PQRD algorithm, an empirical process of trial and error must be undertaken to find suitable values for μ and δ, which cannot be determined in advance. The PSVD by PQRD algorithm proposed in Section IV-A does not suffer from this problem, as a consequence of operating directly upon the polynomial matrix $\underline{\mathbf{A}}(z)$. The proposed method for formulating a PSVD is now demonstrated by means of a simple example.

D. Worked Example

The PSVD by PQRD algorithm was applied to the polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{3\times 3}$ from Section III-E with the stopping

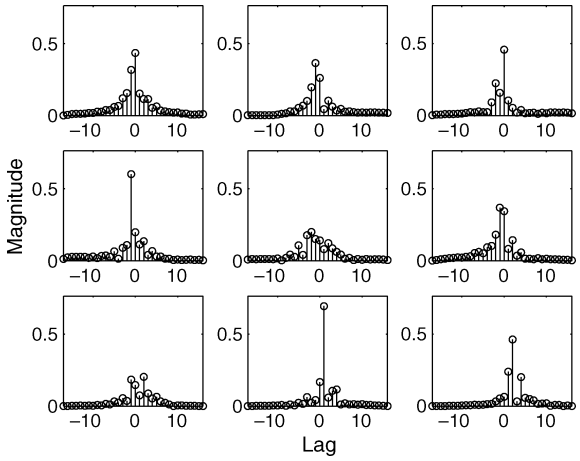

Fig. 5. Polynomial elements of the paraunitary matrix $\underline{\mathbf{U}}(z)$, obtained when the PSVD-PQRD algorithm was applied to the polynomial matrix $\underline{\mathbf{A}}(z)$.

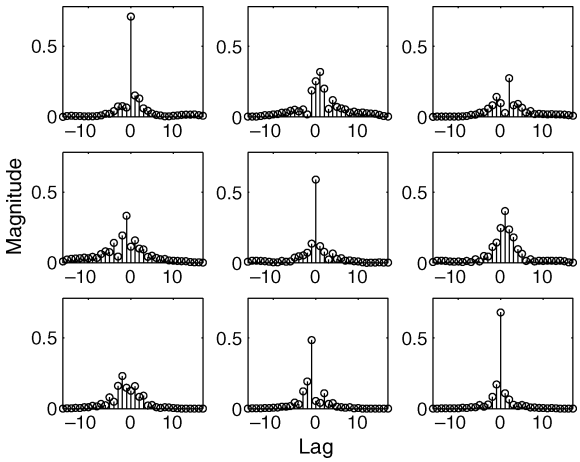

Fig. 6. Polynomial elements of the paraunitary matrix $\underline{\mathbf{V}}(z)$, obtained when the PSVD-PQRD algorithm was applied to the polynomial matrix $\underline{\mathbf{A}}(z)$.

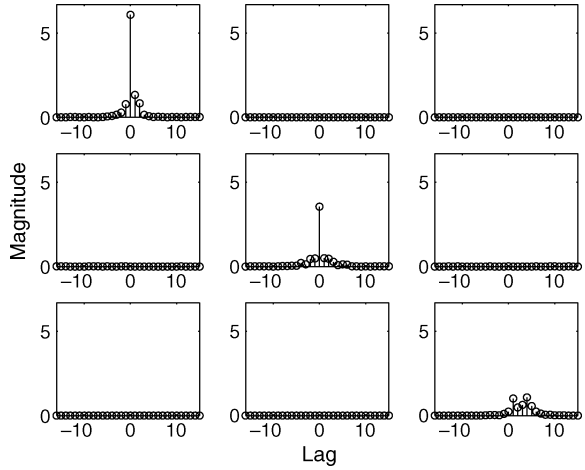

Fig. 7. Polynomial elements of the diagonal matrix $\underline{\mathbf{\Lambda}}(z)$, obtained when the PSVD-PQRD algorithm was applied to the polynomial matrix $\underline{\mathbf{A}}(z)$.

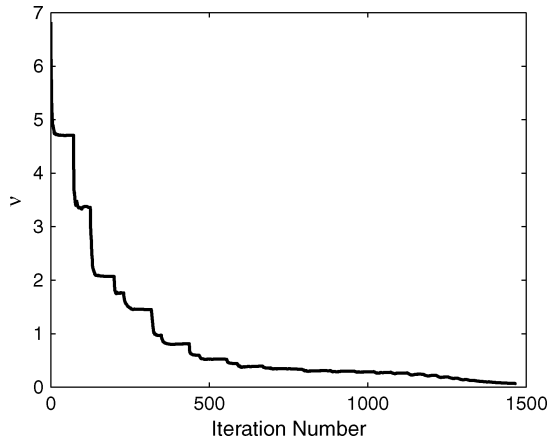

Fig. 8. F-norm of the off-diagonal polynomial elements over the series of EPGRs of the PSVD by PQRD algorithm.

criterion set as $\epsilon = 10^{-2}$ and applying the polynomial matrix truncation method with μ set equal to 10^{-6}. This algorithm required a total of 1466 EPGRs over 15 iterations to converge to a point where the coefficient with maximum magnitude situated in an off-diagonal element of the transformed polynomial matrix was found to be 9.97×10^{-3}. The paraunitary polynomial matrices $\underline{\mathbf{U}}(z)$ (of order 33) and $\underline{\mathbf{V}}(z)$ (of order 33) can be seen in Figs. 5 and 6, respectively. The approximately diagonal matrix $\underline{\mathbf{\Lambda}}(z)$ (of order 31) can be seen in Fig. 7.

The relative error for the decomposition performed is defined as

$$E_{\text{rel}} = \left\| \underline{\mathbf{A}}(z) - \underline{\widetilde{\mathbf{U}}}(z)\underline{\mathbf{\Lambda}}'(z)\underline{\mathbf{V}}(z) \right\|_F / \left\| \underline{\mathbf{A}}(z) \right\|_F \tag{33}$$

where $\underline{\mathbf{\Lambda}}'(z)$ denotes the approximately diagonal polynomial matrix $\underline{\mathbf{\Lambda}}(z)$ with all coefficients of the off-diagonal polynomial elements set equal to zero. This measure was found to be 0.0469, confirming the algorithm has performed a good approximate PSVD of the polynomial matrix $\underline{\mathbf{A}}(z)$ despite truncating the polynomial matrices and only forming an approximate decomposition. Convergence of the algorithm is illustrated in Fig. 8, which shows the F-norm of the off-diagonal elements of the matrix at each stage of the algorithm, i.e., each time a single coefficient has been driven to zero by applying an EPGR and paraunitary inverse delay matrix. The final value of this measure was found to be 0.0688, accounting for 0.91% of the F-norm of the entire matrix. Of course this could have been driven smaller by using a smaller value for the stopping criterion ϵ.

V. Potential Application of the Decompositions to MIMO Channel Equalization

It is assumed that a set of signals $\mathbf{x}(t) \in \mathbb{C}^{q\times 1}$ for $t \in \{0, \ldots, T_{\text{Max}} - 1\}$ are emitted from q independent sources through a convolutive channel, to be received at an array of p sensors, where it is assumed that $p \geq q$. In communication systems, the source signals are generally drawn from a finite constellation, such as in binary or quaternary phase shift keying (BPSK/QPSK). The mixing model for the set of convolutively mixed signals, $\mathbf{y}(t) \in \mathbb{C}^{p\times 1}$ where $t \in \{0, \ldots, T_{Max} - 1\}$, can be expressed as

$$\mathbf{y}(t) = \sum_{k=0}^{N} \mathbf{A}(k)\mathbf{x}(t-k) + \mathbf{n}(t) \tag{34}$$

where $\underline{\mathbf{A}}(z) = \sum_{k=0}^{N} \mathbf{A}(k)z^{-k}$ denotes the polynomial channel matrix where each of the coefficient matrices $\mathbf{A}(k) \in \mathbb{C}^{p\times q}$ for $k \in \{0,\ldots,N\}$, $\mathbf{n}(t) \in \mathbb{C}^{p\times 1}$ denotes an additive zero-mean circular complex Gaussian noise process. The mixing model shown in (34) can then alternatively be written in the form

$$\underline{\mathbf{y}}(z) = \underline{\mathbf{A}}(z)\underline{\mathbf{x}}(z) + \underline{\mathbf{n}}(z) \tag{35}$$

where $\underline{\mathbf{x}}(z)$, $\underline{\mathbf{y}}(z)$, and $\underline{\mathbf{n}}(z)$ denote algebraic power series of the form $\underline{\mathbf{x}}(z) = \sum_{t=-\infty}^{\infty} \mathbf{x}(t)z^{-t}$. It is now shown how either the PSVD or the PQRD can be used to decompose the polynomial channel matrix $\underline{\mathbf{A}}(z)$ allowing the MIMO channel equalization problem to be reformulated as a set of single-input–single-output (SISO) channel equalization problems.

A. The QRD of a Polynomial Matrix

The PQRD of the channel matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times q}$ can be calculated such that $\underline{\mathbf{Q}}(z)\underline{\mathbf{A}}(z) = \underline{\mathbf{R}}(z)$ where $\underline{\mathbf{Q}}(z) \in \underline{\mathbb{C}}^{p\times p}$ is paraunitary and $\underline{\mathbf{R}}(z) \in \underline{\mathbb{C}}^{p\times q}$ is upper triangular. The convolutive mixing model expressed in (35) can be rewritten as

$$\underline{\mathbf{y}}'(z) = \underline{\mathbf{R}}(z)\underline{\mathbf{x}}(z) + \underline{\mathbf{n}}'(z) \tag{36}$$

where $\underline{\mathbf{y}}'(z) = \underline{\mathbf{Q}}(z)\underline{\mathbf{y}}(z)$ and $\underline{\mathbf{n}}'(z) = \underline{\mathbf{Q}}(z)\underline{\mathbf{n}}(z)$. Note that as $\underline{\mathbf{Q}}(z)$ is paraunitary, $\underline{\mathbf{n}}'(z)$ is also a Gaussian noise process with identical spectral properties. Now provided the channel matrix is of full column rank, the MIMO channel equalization problem can be transformed into a set of q single channel equalization problems using back substitution. Starting with the qth element of $\underline{\mathbf{y}}'(z)$, this can be expressed as

$$\underline{y}'_q(z) = \underline{r}_{qq}(z)\underline{x}_q(z) + \underline{n}'_q(z) \tag{37}$$

which is a single channel equalization problem. This can now be solved, to obtain an estimate of the qth source signal $\hat{x}_q(t)$, using a standard SISO method of equalization [4]. Furthermore, the ith single channel equalization problem can be formulated as

$$\underline{y}'_i(z) - \sum_{j=i+1}^{q} \underline{r}_{ij}(z)\underline{x}_j(z) = \underline{r}_{ii}(z)\underline{x}_i(z) + \underline{n}'_i(z) \tag{38}$$

and so, provided the set of signals is estimated according to the ordering $i = q, q-1, \ldots, 1$, this is also a SISO channel equalization problem. Each equation can then be solved to obtain an estimate of the ith transmitted signal $\hat{x}_i(t)$ using the previously estimated signals $\hat{x}_j(t)$ for $j = i+1, \ldots, q$. The PQRD has been applied to this application in [12] and [13], but remains a topic of future research with several aspects of the overall system which have yet to be fully investigated.

B. The SVD of a Polynomial Matrix

Alternatively, the PSVD of the channel matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times q}$ can be calculated such that $\underline{\mathbf{U}}(z)\underline{\mathbf{A}}(z)\underline{\tilde{\mathbf{V}}}(z) = \underline{\mathbf{\Lambda}}(z)$ where $\underline{\mathbf{U}}(z) \in \underline{\mathbb{C}}^{p\times p}$ and $\underline{\mathbf{V}}(z) \in \underline{\mathbb{C}}^{q\times q}$ are paraunitary and $\underline{\mathbf{\Lambda}}(z)$ is approximately diagonal. Before transmitting the source signals $\underline{\mathbf{s}}(z) \in \underline{\mathbb{C}}^{q\times 1}$, they are first filtered by the paraunitary matrix $\underline{\tilde{\mathbf{V}}}(z)$ such that $\underline{\mathbf{x}}(z) = \underline{\tilde{\mathbf{V}}}(z)\underline{\mathbf{s}}(z)$. The convolutive mixing model expressed in (35) can then be rewritten as

$$\underline{\mathbf{y}}'(z) = \underline{\mathbf{\Lambda}}(z)\underline{\mathbf{s}}(z) + \underline{\mathbf{n}}'(z) \tag{39}$$

where $\underline{\mathbf{y}}'(z) = \underline{\mathbf{U}}(z)\underline{\mathbf{y}}(z)$ and $\underline{\mathbf{n}}'(z) = \underline{\mathbf{U}}(z)\underline{\mathbf{n}}(z)$ is a Gaussian noise process with identical spectral properties as $\underline{\mathbf{n}}(z)$. Now provided the channel matrix is of full column rank, the MIMO channel equalization problem can be transformed into a set of q single channel equalization problems as the jth element of $\underline{\mathbf{y}}'(z)$ can be expressed as

$$\underline{y}'_j(z) = \underline{\lambda}_{jj}(z)\underline{s}_j(z) + \underline{n}'_j(z) \tag{40}$$

which is a single channel equalization problem and can be solved for $j = 1,\ldots,q$. This can now be solved to obtain an estimate of the jth source signal $\hat{s}_j(t)$, using a standard SISO method of equalization [4]. The SBR2 algorithm has previously been used for this application which is fully defined in [14]–[17], confirming that a good average BER performance is achievable when transmitting over frequency selective quasi-static channels. However, the PSVD by PQRD algorithm proposed in this paper could similarly be applied to this problem and this is the subject of future research. In particular, it will be interesting to see if using this new decomposition can improve upon the results already obtained when using the SBR2 algorithm. Another aspect of this work that is currently being investigated is that of imperfect channel state information. Previous publications have assumed that perfect knowledge of the system is available, which is clearly not realistic when the channel must first be estimated. This is currently the topic of our research and beyond the scope of this paper.

VI. Conclusion

An algorithm for calculating the QR decomposition of a polynomial matrix has been presented. It has then been shown that this algorithm can also be used to generate the SVD of a polynomial matrix and that this SVD method is clearly a more appropriate method for formulating a PSVD than the existing method for calculating this decomposition, which operates using the SBR2 algorithm. The main advantage of using the PSVD algorithm in this paper is that it operates directly upon the polynomial matrix and, as a result, allows the user more control over the quality of the decomposition obtained. Proofs of convergence have been outlined for the use of the PQRD algorithm to obtain both decompositions and some simple numerical examples were presented to illustrate both decompositions. Finally, a potential application of the decompositions to multichannel signal processing has been mentioned, demonstrating that either decomposition could potentially be used as part of a broadband MIMO communication system.

Appendix
Polynomial Matrix Truncation Method

This method of truncation has been previously used in [9] and is similar to that developed in [1] and [18]. A more detailed

explanation as to why the order can become unnecessarily large is also found in these papers. A suitable truncation method for a polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{p\times q}$, with coefficient matrices $\mathbf{A}(t) \in \mathbb{C}^{p\times q}$ for $t = t_1, \ldots, t_2$ can be implemented as follows: find a maximum value for T_1 and a minimum value for T_2 such that

$$\frac{\sum_{\tau=t_1}^{T_1}\sum_{l=1}^{p}\sum_{m=1}^{q}|a_{lm}(\tau)|^2}{||\underline{\mathbf{A}}(z)||_F^2} \leq \frac{\mu}{2} \tag{41}$$

and

$$\frac{\sum_{\tau=T_2}^{t_2}\sum_{l=1}^{p}\sum_{m=1}^{q}|a_{lm}(\tau)|^2}{||\underline{\mathbf{A}}(z)||_F^2} \leq \frac{\mu}{2} \tag{42}$$

where μ defines the proportion of $||\underline{\mathbf{A}}(z)||_F^2$ permitted to be truncated from the polynomial matrix $\underline{\mathbf{A}}(z)$, with one implementation of the truncation method. The coefficient matrices $\mathbf{A}(\tau)$ for $\tau = t_1, \ldots, T_1$ and $\tau = T_2, \ldots, t_2$ can subsequently be trimmed from the matrix. Clearly, when this technique is to be applied to the para-Hermitian polynomial matrices within the SBR2 algorithm, then the function can be simplified as discussed in [1]. Note that when using this truncation method to obtain either a PQRD or a PSVD, then the transformation will no longer be energy (or norm) preserving. However, this does not affect the proofs of convergence for either decomposition.

ACKNOWLEDGMENT

The authors would like to thank Dr. S. Lambotharan for his invaluable comments and suggestions to this work. They would also like to thank the reviewers for their helpful comments which have enabled them to improve the presentation of this paper.

REFERENCES

[1] J. G. McWhirter, P. D. Baxter, T. Cooper, S. Redif, and J. Foster, "An EVD algorithm for Para-Hermitian polynomial matrices," *IEEE Trans. Signal Process.*, vol. 55, no. 6, pp. 2158–2169, Jun. 2007.

[2] P. Vaidyanathan, *Multirate Systems and Filter Banks*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[3] G. H. Golub and C. F. Van Loan, *Matrix Computations (Third Edition)*. Baltimore, MD: The John Hopkins Univ. Press, 1996.

[4] J. Proakis, *Digital Communications*, 4th ed. New York: McGraw-Hill, 2001.

[5] A. J. Paulraj, R. Nabar, and D. Gore, *Introduction to Space-Time Wireless Communications*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[6] X. Gao, T. Q. Nguyen, and G. Strang, "On factorization of M-channel paraunitary filterbanks," *IEEE Trans. Signal Process.*, vol. 49, no. 7, pp. 1433–1446, Jul. 2001.

[7] J. G. McWhirter and P. D. Baxter, "A novel technique for broadband SVD," in *Proc. 12th Annu. Workshop Adaptive Sensor Array Process.*, 2004.

[8] J. A. Foster, "Algorithms and techniques for polynomial matrix decompositions," Ph.D. dissertation, School Eng., Cardiff Univ., Cardiff, U.K., 2008.

[9] J. A. Foster, J. G. McWhirter, and J. A. Chambers, "An algorithm for computing the QR decomposition of a polynomial matrix," in *Proc. 15th Int. Conf. Digital Signal Process.*, 2007, pp. 71–74.

[10] J. A. Foster, J. G. McWhirter, and J. A. Chambers, "A polynomial matrix QR decomposition with application to MIMO channel equalisation," in *Proc. 41st Asilomar Conf. Signals Syst. Comput.*, 2007, pp. 1379–1383.

[11] J. A. Foster, J. G. McWhirter, and J. A. Chambers, "A novel algorithm for calculating the QR decomposition of a polynomial matrix," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2009, pp. 3177–3180.

[12] M. Davies, S. Lambotharan, J. A. Foster, J. A. Chambers, and J. G. McWhirter, "Polynomial matrix QR decomposition and iterative decoding of frequency selective MIMO channels," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2009.

[13] M. Davies, S. Lambotharan, J. A. Foster, J. A. Chambers, and J. G. McWhirter, "A polynomial QR decomposition based turbo equalization technique for frequency selective MIMO channels," in *Proc. 69th Veh. Technol. Conf.*, 2009.

[14] M. Davies, S. Lambotharan, J. A. Chambers, and J. G. McWhirter, "Broadband MIMO beamforming for frequency selective channels using the sequential best rotation algorithm," in *Proc. 67th Veh. Technol. Conf.*, 2008, pp. 1147–1151.

[15] C. H. Ta and S. Weiss, "A design of precoding and equalisation for broadband MIMO systems," in *Proc. 41st Asilomar Conf. Signals Syst. Comput.*, 2007, pp. 1616–1620.

[16] H. Zamiri-Jafarian and M. Rajabzadeh, "A polynomial matrix SVD approach for time domain broadband beamforming in MIMO-OFDM systems," in *Proc. 67th Veh. Technol. Conf.*, Spring, 2008, pp. 802–806.

[17] W. Al-Hanafy, A. P. Millar, C. H. Ta, and S. Weiss, "Broadband SVD and non-linear precoding applied to broadband MIMO channels," in *Proc. 42nd Asilomar Conf. Signals Syst. Comput.*, 2008, pp. 2053–2057.

[18] J. A. Foster, J. G. McWhirter, and J. A. Chambers, "Limiting the order of polynomial matrices within the SBR2 algorithm," in *Proc. IMA Conf. Math. Signal Process.*, 2006, pp. 93–97.

Joanne A. Foster (M'09) received the undergraduate Masters degree in mathematics (MMath) from the University of Bath, Bath, U.K., in 2004 and the Ph.D. degree from the School of Engineering, Cardiff University, Cardiff, U.K., under the supervision of Prof. J. McWhirter and Prof. J. Chambers in 2008. The subject of her Ph.D. dissertation was algorithms and applications of polynomial matrix decompositions.

Currently, she is a Postdoctoral Research Assistant at Loughborough University, Loughborough, U.K.

John G. McWhirter received the first class honors degree in mathematics and the Ph.D. degree in theoretical physics from the Queen's University of Belfast, Belfast, Ireland, in 1970 and 1973, respectively.

He joined the Royal Radar Establishment in Malvern (later to become the Royal Signals and Radar Establishment, and now part of QinetiQ Ltd.) in 1973, where he became a Senior Fellow in the Centre for Signal and Information Processing Group. In 2007, he left QinetiQ to take up his current post as Distinguished Research Professor in the Engineering Department, Cardiff University, Cardiff, U.K. He is also a Visiting Professor in Electrical Engineering at the Queen's University of Belfast. He has been carrying out research on adaptive signal processing since 1980 and was awarded the J. J. Thomson Medal by the Institution of Electrical Engineers in 1994 for his research on systolic arrays. He has published more than 140 research papers and holds numerous patents. His current research is devoted to broadband sensor arrays, convolutive blind signal separation, and polynomial matrix techniques. The signal processing group which he built up in Malvern over many years, received the EURASIP Group Technical Achievement Award for 2003.

Dr. McWhirter was elected as a Fellow of the Royal Academy of Engineering in 1996 and as a Fellow of the Royal Society in 1999. He is a Fellow of the Institute of Mathematics and its Applications (IMA) and served as President of the IMA in 2002 and 2003. He is also a Fellow of the Institute of Electrical Engineers, a Fellow of the Institute of Physics, and a member of the London Mathematical Society.

Martin R. Davies (S'08) received the B.Eng. and M.Sc. (distinction) degrees from Cardiff University, Cardiff, U.K., in 2004 and 2005, respectively, both in electronic engineering. He is awaiting the viva for his Ph.D. degree with the Advanced Signal Processing Group, School of Engineering, Loughborough University, Loughborough, U.K., under the supervision of Dr. S. Lambotharan.

He is currently based at Phase Vision Ltd., Loughborough, U.K., developing algorithms for application in the field of optical metrology.

Jonathon A. Chambers (S'83–M'90–SM'98) was born in Peterborough, U.K., in 1960. He received the B.Sc. (honors) degree in electrical engineering from the Polytechnic of Central London, London, U.K., in 1985 and the Ph.D. degree in digital signal processing from the University of London, London, U.K., in 1990.

He was at Peterhouse, Cambridge University, Cambridge, U.K., and the Imperial College London, U.K. From 1979 to 1982, he was as an Artificer Apprentice in Action, Data, and Control in the Royal Navy. He has held academic and industrial positions at Bath University, Cardiff University, Imperial College London, King's College London, and Schlumberger Cambridge Research, U.K. In July 2007, he joined the Department of Electronic and Electrical Engineering, Loughborough University, Loughborough, U.K., as a Professor of Communications and Signal Processing, and currently leads the Advanced Signal Processing Research Group and a team of researchers in the analysis, design, and evaluation of novel algorithms for digital signal processing with application in acoustics, biomedicine, and wireless communications. He is also the Director of Research in the Department. He has authored or coauthored more than 300 research outputs including two monographs and more than 100 journal articles.

Dr. Chambers is a member of the IEEE Technical Committee on Signal Processing Theory and Methods. He was the Technical Program Chair for the 2009 IEEE Workshop on Statistical Signal Processing and is the Technical Program Co-Chair for the 2011 International Conference on Acoustics, Speech, and Signal Processing, Prague, Czech Republic. He was the Chairman of the Institute for Electrical Engineers (IEE) Professional Group E5, Signal Processing. He was an Associate Editor for the IEEE SIGNAL PROCESSING LETTERS and the IEEE TRANSACTIONS ON SIGNAL PROCESSING for two terms. He was awarded the first QinetiQ Visiting Fellowship in 2007 for outstanding contributions to adaptive signal processing.