AIAA JOURNAL
Vol. 47, No. 8, August 2009

QZ-Based Algorithm for System Pole, Transmission Zero, and Residue Derivatives

Bradley T. Burchett*
Rose-Hulman Institute of Technology, Terre Haute, Indiana 47803

DOI: 10.2514/1.39759

The QZ algorithm gives a robust way of computing solutions to the generalized eigenvalue problem. The generalized eigenvalue problem is used in linear control theory to find solutions to Ricatti equations, as well as to determine system transmission zeros. In state-space linear system analysis, the system poles and transmission zeros are particularly important for determining system time and frequency response. Here, we embed calculation of the eigenvalue derivatives in the QZ algorithm such that the derivatives of system poles and transmission zeros are computed simultaneously with the poles and zeros themselves. The resulting method is further exercised in finding generalized eigenvalues and their sensitivities required for finding the derivatives of system residues. This technique should open the door to solutions of problems of interest by unconstrained gradient-based methods. Typical numerical results are presented.

Nomenclature

$\mathbf{A}$	=	numerator of the matrix pencil
$A(s)$	=	transfer function denominator polynomial
$\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, $\mathbf{d}$	=	linear system state-space description
a, b	=	various elements of the matrix pencil
$\mathbf{B}$	=	denominator of the matrix pencil
$B(s)$	=	transfer function numerator polynomial
C	=	$\cos(\theta)$ in Givens rotation
d	=	prefix indicating differentiation (e.g., $d\nu = \partial\nu/\partial\psi$)
$\mathbf{e}_1$	=	$[1 \quad 0 \quad \cdots \quad 0]^T$
$F(s)$	=	system transfer function
$\mathbf{H}$	=	Householder matrix
$\mathbf{I}$	=	identity matrix
$\bar{\mathbf{I}}$	=	skew $\mathbf{I}$ [i.e., $\mathtt{fliplr}(\mathbf{I})$]
i	=	$\sqrt{-1}$
i, j, k, kk, $i1$, $i2$	=	loop counters
$\mathbf{K}$	=	matrix of adjustable system parameters
m	=	number of system zeros
m_k	=	multiplicity of eigenvalue λ_k
n	=	matrix dimension or number of system poles
p	=	dimension of the top left block
p, q, r	=	intermediate values used in computing eigenvalues
p_i	=	system poles
$\mathbf{Q}$	=	left unitary matrix
q	=	dimension of the right bottom block
$\mathbf{R}$	=	column matrix of coefficients of $B(s)$
r_i	=	numerator of partial fraction expansion (residue)
S	=	$\sin(\theta)$ in Givens rotation
s	=	LaPlace variable or loop counter
t	=	time
$\mathbf{u}$	=	left Householder vector
$\mathbf{v}$	=	right Householder vector
$\mathbf{w}$	=	generalized eigenvector
$\mathbf{x}$	=	input/output vector to the Givens rotation
$\mathbf{y}$	=	second input/output to the Givens rotation
$\mathbf{Z}$	=	right unitary matrix
z_i	=	system zeros
η	=	coefficient of $B(s)$
θ	=	angle of the Givens rotation
λ	=	eigenvalue
μ	=	$\mathtt{sign}(a)$
ν	=	2-norm of vector or $\|a + b\|_2$
$\mathbf{\Xi}$	=	batch calculation matrix
ξ	=	a column of the $\mathbf{\Xi}$ matrix
ρ	=	$\mathtt{sign}(\mathbf{u}_1)$
τ	=	$\|a\| + \|b\|$
$\phi(.)$	=	characteristic polynomial
ψ	=	element of the $\mathbf{K}$ matrix

Subscripts

$\mathbf{A}_{i:j,k:m}$	=	block consisting of k through m columns of i through j rows of the matrix $\mathbf{A}$
$\mathbf{A}_{n,m}$	=	n and m elements of the matrix $\mathbf{A}$
$\mathbf{u}_2$	=	second element of vector $\mathbf{u}$
:	=	wild card, $\mathbf{A}_{:,m}$ is all rows, mth column of $\mathbf{A}$
(:)	=	force shape of 1-D array to be a column

Superscripts

H	=	Hermitian or complex-conjugate transpose
T	=	transpose
†	=	complex conjugate

I. Introduction

THE solution of the generalized eigenvalue problem has many applications in control theory, including solving Riccati equations [1,2], computing system transmission zeros [3], and poles. Here, the QZ algorithm is employed to provide robust solutions for the eigenvalues and transmission zeros and their derivatives. The method is further exploited to compute derivatives of the residues as well.

Considerable attention has been given to the problem of the computation of the derivatives of the eigenvalues and eigenvectors of a linear system with respect to system parameters. Murthy and Haftka [4] surveyed the different numerical methods for performing these operations and provide an extensive reference list of work performed in this area. As discussed by Murthy and Haftka, three classes of methods have emerged: adjoint, direct, and iterative. Adjoint methods generally require both left and right eigenvectors

Presented as Paper 7310 at the AIAA Guidance, Navigation, and Control Conference, Honolulu, HI, 18–22 August 2008; received 14 July 2008; revision received 22 April 2009; accepted for publication 30 April 2009.

*Associate Professor, Department of Mechanical Engineering, 5500 Wabash Avenue; burchett@rose-hulman.edu. Associate Fellow AIAA.

for derivative calculations. The derivative of a particular eigenvector is expressed as a weighted sum of the system eigenvectors. One example of an adjoint method is that by Rogers [5]. Direct methods typically involve the solution to a system of linear equations. Unlike adjoint methods, eigenvector derivatives only involve a single eigenvector and not the entire eigensystem. Garg [6] and Rudisill and Chu [7] provided examples of direct methods. More recent work by Jankovic [8] showed a method for computing eigensystem derivatives applicable to linear and nonlinear problems. Nelson [9] derived a computationally efficient direct method. Lim et al. [10] developed a method to calculate derivatives of repeated eigenvalues and their eigenvectors using the singular value decomposition. Zhang and Wei [11] presented a method using a complete modal space. Song et al. [12] invented a simplified method based upon Nelson's [9] efficient approach. Chen [13] demonstrated a method for doubly repeated eigenvalues. Wei and Zhang [14] also presented a method using the generalized inverse technique. More recently, Friswell [15], Prells and Frisell [16], and Friswell and Adhikari [17] developed methods based on Nelson's [9] method to include distinct complex eigenvalues. Zhang and Zhang [18–21] published several investigations of eigensensitivity analysis for a defective matrix. Note that none of the methods mentioned previously deal with generalized eigenvalues.

This work explores the use of a QZ-based algorithm for computing generalized eigenvalue derivatives. This method shares the stability and robustness properties of the QZ algorithm and results in a novel method for computing the sensitivities of transmission zeros. In the sequel, we will review the QZ algorithm, show how each of the subroutines are differentiated, present the overall differentiated algorithm, and present numerical examples for system poles, transmission zeros, and residues.

II. Review of the QZ Algorithm

The QZ algorithm first appeared in a seminal paper by Moler and Stewart [22]. Here, we draw on several sources [22–25], however, we are particularly indebted to Stewart [24] for his detailed presentation of the algorithm subroutines in vectorized or pseudo-MATLAB code.

The generalized eigenvalue problem is simply stated as finding nontrivial solutions λ and $\mathbf{w}$ to the equation

$$\mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w}$$

where $\mathbf{B}$ may be singular. The QZ algorithm provides a numerical solution to this problem in four steps:

1) $\mathbf{A}$ is reduced to upper Hessenberg form and $\mathbf{B}$ is reduced to upper triangular form.

2) $\mathbf{A}$ is reduced to quasi-triangular form and the triangular form of $\mathbf{B}$ is maintained.

3) The quasi-triangular matrix is reduced to triangular form and the eigenvalues are extracted.

4) The eigenvectors are obtained from the triangular matrices and transformed back into the original coordinate system [22].

Here, we focus on the differentiation of steps 1 and 2. Reduction to triangular form and computation of the eigenvectors are reserved for future efforts.

Each step of the QZ process is facilitated by orthogonal transformations. These transformations are determined by either Householder reflections or Givens rotations. In fact, various combinations of either approach may be used, such that various authors offer slightly different versions of QZ [23].

A. QZ Building Blocks

Here, we review the subroutines used to compute Householder reflections and Givens rotations and apply Givens rotations.

1. Householder Reflectors

Calculate the vector $\mathbf{u}$ such that the Householder transformation $\mathbf{H} = \mathbf{I} - \mathbf{u}\mathbf{u}^H$ renders $\mathbf{H}\mathbf{a} = \nu\mathbf{e}_1$. Note that we are using Stewart's [24] formulation in Algorithm 1 because it applies to both row and column vectors. In our previous work [26], we presented an approach that depends heavily on the transpose operator and rendering $\mathbf{u}$ with a different scaling.

Algorithm 1 Generate Householder reflector

```
1)  [u, ν] = housegen(a)
2)    u = a
3)    ν = ‖a‖_2
4)    if (ν = 0) u_1 = √2; return; end if
5)    if (u_1 ≠ 0)
6)    ρ = u_1/|u_1|
7)    else
8)    ρ = 1
9)    end if
10)   u = (ρ/ν) × u
11)   u_1 = 1 + u_1
12)   u = u/√u_1
13)   ν = −ρν
14) end housegen
```

2. Givens Rotation

The Givens rotation rotates two vectors through an angle θ about an axis mutually orthogonal to the vectors. The quantities $\cos(\theta) = C$ and $\sin(\theta) = S$ are computed directly by the following subroutine without any need to compute θ. Because of its brevity, we present the differentiated subroutine as Algorithm 2. Lines labeled with an a belong to the differentiated subroutine. A d prefix indicates the partial derivative with respect to ψ of the corresponding variable: for instance, $d\nu = \partial\nu/\partial\psi$.

The Givens rotation is applied to the vectors and their derivatives by the subroutine in Algorithm 3. Note that $d\mathbf{y}$ must be computed before $\mathbf{y}$.

3. Reduction to Hessenberg-Triangular Form

The subroutine in Algorithm 4 takes a matrix pencil of order n and reduces it to Hessenberg-triangular form by orthogonal transformations [24]. The differentiated version dHessTriForm of this subroutine is shown in Appendix A.

4. QZ Sweep

The QZ sweep transforms $\mathbf{A}$ from Hessenberg form to quasi-triangular form while preserving the triangular form of $\mathbf{B}$. This subroutine performs a QZ sweep between rows $i1$ and $i2$. One or two rows of $\mathbf{A}$ may converge to quasi-triangular form after one to several iterations of the QZ sweep. Lines 1a and 1b in Algorithm 5 calculate the initial Householder transformation, as shown by Moler and Stewart [22] (Eq. 4.1), which is shown explicitly in the next subsection.

5. Calculating the Initial Householder Transformation

The initial Householder transformation in a QZ sweep is computed using equations from Moler and Stewart [22]:

$$\mathbf{u}_1 = \left[\left(\frac{\mathbf{A}_{m,m}}{\mathbf{B}_{m,m}} - \frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}}\right)\left(\frac{\mathbf{A}_{n,n}}{\mathbf{B}_{n,n}} - \frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}}\right) - \left(\frac{\mathbf{A}_{m,n}}{\mathbf{B}_{n,n}}\right)\left(\frac{\mathbf{A}_{n,m}}{\mathbf{B}_{m,m}}\right) + \left(\frac{\mathbf{A}_{n,m}}{\mathbf{B}_{m,m}}\right)\left(\frac{\mathbf{B}_{m,n}}{\mathbf{B}_{n,n}}\right)\left(\frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}}\right)\right]\left(\frac{\mathbf{B}_{1,1}}{\mathbf{A}_{2,1}}\right) + \frac{\mathbf{A}_{1,2}}{\mathbf{B}_{2,2}} - \left(\frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}}\right)\left(\frac{\mathbf{B}_{1,2}}{\mathbf{B}_{2,2}}\right) \tag{1}$$

$$\mathbf{u}_2 = \left(\frac{\mathbf{A}_{2,2}}{\mathbf{B}_{2,2}} - \frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}}\right) - \left(\frac{\mathbf{A}_{2,1}}{\mathbf{B}_{1,1}}\right)\left(\frac{\mathbf{B}_{1,2}}{\mathbf{B}_{2,2}}\right) - \left(\frac{\mathbf{A}_{m,m}}{\mathbf{B}_{m,m}} - \frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}}\right) - \left(\frac{\mathbf{A}_{n,n}}{\mathbf{B}_{n,n}} - \frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}}\right) + \left(\frac{\mathbf{A}_{n,m}}{\mathbf{B}_{m,m}}\right)\left(\frac{\mathbf{B}_{m,n}}{\mathbf{B}_{n,n}}\right) \tag{2}$$

$$\mathbf{u}_3 = \frac{\mathbf{A}_{3,2}}{\mathbf{B}_{2,2}} \tag{3}$$

Algorithm 2 Generate Givens rotation and first derivative

1) $[a, \quad b, \quad C, \quad S, \quad da, \quad db, \quad dC, \quad dS] = d\text{rotgen}(a, \quad b, \quad da, \quad db)$
2) if $(b = 0)$
3) $C = 1; S = 0;$
   a) $dC = 0; dS = 0;$ return
4) end if
5) if $(a = 0)$
6) $C = 0; S = 1; a = b; b = 0$
   a) $dC = 0; dS = 0\ da = db; db = 0$ return
7) end if
8) $\mu = a/|a|$
   a) $d\mu = 0$
9) $\tau = |a| + |b|$
   a) $d\tau = \texttt{sign}(a) \times da + \texttt{sign}(b) \times db$
10) $\nu = \tau \times \sqrt{|a/\tau|^2 + |b/\tau|^2}$
   a) $d\nu = d\tau \sqrt{|\frac{a}{\tau}|^2 + |\frac{b}{\tau}|^2} + \tau \frac{1}{2}(|\frac{a}{\tau}|^2 + |\frac{b}{\tau}|^2)^{-1/2} \times (2a\frac{da}{\tau^2} - 2a^2\frac{d\tau}{\tau^3} + 2b\frac{db}{\tau^2} - 2b^2\frac{db}{\tau^3})$
11) $C = |a|/\nu$
   a) $dC = \texttt{sign}(a) \times da/\nu - \texttt{sign}(a) \times a \times d\nu/\nu^2$
12) $S = \mu \times b/\nu$
   a) $dS = d\mu \times b/\nu + \mu \times db/\nu - \mu \times b \times d\nu/\nu^2$
13) $a = \nu \times \mu$
   a) $da = d\nu \times \mu + \nu \times d\mu$
14) $b = 0$
   a) $db = 0$
15) end *d*rotgen

In the case that $\mathbf{B}_{2,2}$ equals zero, Kaufman [25] suggests multiplying Moler and Stewart's [22] equations by $\mathbf{B}_{2,2}$ and simplifying. This results in the following simpler set of equations:

$$\mathbf{u}_1 = \mathbf{A}_{1,2} - \mathbf{A}_{1,1}\mathbf{B}_{1,2}/\mathbf{B}_{1,1} \tag{4}$$

$$\mathbf{u}_2 = \mathbf{A}_{2,2} - \mathbf{A}_{2,1}\mathbf{B}_{1,2}/\mathbf{B}_{1,1} \tag{5}$$

$$\mathbf{u}_3 = \mathbf{A}_{3,2} \tag{6}$$

The differentiated form of these equations is reserved for Appendix A due to its length.

B. Overall QZ Process

The overall QZ process begins with the transformation to Hessenberg-triangular form, as outlined previously. If the $\mathbf{B}$ is singular, a zero will appear on the diagonal for each rank deficiency. These zeros may be ignored, in which case they will be deflated out of the top of the pencil during the first QZ sweep. However, the system derivatives will not be accurately calculated using this approach. Therefore, it becomes imperative that any zeros on the diagonal of $\mathbf{B}$ are deflated out of the bottom of the pencil, because using this approach will preserve the correct derivative information. The algorithm for deflating zeros out of the bottom, including the propagation of the derivative information, is given in Appendix A. When $\mathbf{B}$ initially has two or more zeros on the diagonal, the deflation logic becomes somewhat more complicated. A flowchart is shown in Fig. 1. The algorithm then proceeds by applying the QZ sweep to submatrices of the pencil. We follow the description provided by Golub and Van Loan [23], including differentiation.

Find the largest nonnegative q and the smallest nonnegative p such that if

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{A}_{13} \\ 0 & \mathbf{A}_{22} & \mathbf{A}_{23} \\ 0 & 0 & \mathbf{A}_{33} \end{bmatrix} \begin{matrix} p \\ n-p-q \\ q \end{matrix} \tag{7}$$
$$\begin{matrix} p & n-p-q & q \end{matrix}$$

then $\mathbf{A}_{33}$ is upper quasi-triangular and $\mathbf{A}_{22}$ is unreduced upper Hessenberg. Partition $\mathbf{B}$ is conformably

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} & \mathbf{B}_{13} \\ 0 & \mathbf{B}_{22} & \mathbf{B}_{23} \\ 0 & 0 & \mathbf{B}_{33} \end{bmatrix} \begin{matrix} p \\ n-p-q \\ q \end{matrix} \tag{8}$$
$$\begin{matrix} p & n-p-q & q \end{matrix}$$

If $q < n$, apply the qz2step to $\mathbf{A}_{22}$ and $\mathbf{B}_{22}$. The algorithm *d*qz2step is offered in Appendix A.

Algorithm 3 Apply Givens rotation and derivative

1) $[\mathbf{x}, \quad \mathbf{y}, \quad d\mathbf{x}, \quad d\mathbf{y}]$
   $= d\text{rotapp}(C, \quad S, \quad dC, \quad dS, \quad \mathbf{x}, \quad \mathbf{y}, \quad d\mathbf{x}, \quad d\mathbf{y})$
2) $\mathbf{t} = C \times \mathbf{x} + S \times \mathbf{y}$
   a) $d\mathbf{t} = dC \times \mathbf{x} + C \times d\mathbf{x} + dS \times \mathbf{y} + S \times d\mathbf{y}$
   b) $d\mathbf{y} = dC \times \mathbf{y} + C \times d\mathbf{y} - dS \times \mathbf{x} - S \times d\mathbf{x}$
3) $\mathbf{y} = C \times \mathbf{y} - S \times \mathbf{x}$
4) $\mathbf{x} = \mathbf{t}$
   a) $d\mathbf{x} = d\mathbf{t}$
5) end *d*rotapp

Algorithm 4 Reduce A and B to Hessenberg-Triangular form

1) $[\mathbf{A}, \quad \mathbf{B}, \quad \mathbf{Q}, \quad \mathbf{Z}] = \text{HessTriForm}(\mathbf{A}, \quad \mathbf{B})$
   a) Reduce $\mathbf{B}$ to triangular form.
2) $\mathbf{Q} = \mathbf{I}$
3) for $k = 1$ **to** $n - 1$
4) $[\mathbf{u}, \quad \text{temp}] = \text{housegen}(\mathbf{B}_{k:n,k})$
5) $\mathbf{v}^T = \mathbf{u}^T \times \mathbf{B}_{k:n,k:n}$
6) $\mathbf{B}_{k:n,k:n} = \mathbf{B}_{k:n,k:n} - \mathbf{u} \times \mathbf{v}^T$
7) $\mathbf{v}^T = \mathbf{u}^T \times \mathbf{A}_{k:n,k:n}$
8) $\mathbf{A}_{k:n,:} = \mathbf{A}_{k:n,:} - \mathbf{u} \times \mathbf{v}^T$
9) $\mathbf{v} = \mathbf{Q}_{:,k:n} \times \mathbf{u}$
10) $\mathbf{Q}_{:,k:n} = \mathbf{Q}_{:,k:n} - \mathbf{v} \times \mathbf{u}^T$
11) end for k
   a) Reduce $\mathbf{A}$ to Hessenberg form.
12) $\mathbf{Z} = \mathbf{I}$
13) for $k = 1$ to $n - 2$
14) for $j = n - 1$ to $k + 1$ by -1
15) $[\mathbf{A}_{j,k}, \quad \mathbf{A}_{j+1,k}, \quad C, \quad S] = \text{rotgen}(\mathbf{A}_{j,k}, \quad \mathbf{A}_{j+1,k})$
16) $[\mathbf{A}_{j,k+1:n}, \quad \mathbf{A}_{j+1,k+1:n}]$
   $= \text{rotapp}(C, \quad S, \quad \mathbf{A}_{j,k+1:n}, \quad \mathbf{A}_{j+1,k+1:n})$
17) $[\mathbf{B}_{j,j:n}, \quad \mathbf{B}_{j+1,j:n}] = \text{rotapp}(C, \quad S, \quad \mathbf{B}_{j,j:n}, \quad \mathbf{B}_{j+1,j:n})$
18) $[\mathbf{Q}_{:,j}, \quad \mathbf{Q}_{:,j+1}] = \text{rotapp}(C, \quad S, \quad \mathbf{Q}_{:,j}, \quad \mathbf{Q}_{:,j+1})$
19) $[\mathbf{B}_{j+1,j+1}, \quad \mathbf{B}_{j+1,j}, \quad C, \quad S] = \text{rotgen}(\mathbf{B}_{j+1,j+1}, \quad \mathbf{B}_{j+1,j})$
20) $[\mathbf{B}_{1:j,j+1}, \quad \mathbf{B}_{1:j,j}] = \text{rotapp}(C, \quad S, \quad \mathbf{B}_{1:j,j+1}, \quad \mathbf{B}_{1:j,j})$
21) $[\mathbf{A}_{:,j+1}, \quad \mathbf{A}_{:,j}] = \text{rotapp}(C, \quad S, \quad \mathbf{A}_{:,j+1}, \quad \mathbf{A}_{:,j})$
22) $[\mathbf{Z}_{:,j+1}, \quad \mathbf{Z}_{:,j}] = \text{rotapp}(C, \quad S, \quad \mathbf{Z}_{:,j+1}, \quad \mathbf{Z}_{:,j})$
23) end for j
24) end for k
25) end HessTriForm

Algorithm 5 QZ sweep between rows $i1$ and $i2$

1) $[\mathbf{A}, \mathbf{B}, \mathbf{Q}, \mathbf{Z}] = \text{qz2step}(\mathbf{A}, \mathbf{B}, \mathbf{u}, i1, i2, \mathbf{Q}, \mathbf{Z})$
   a) $\mathbf{u} = \text{Moler41}(\mathbf{A}, \mathbf{B})$
   b) $[\mathbf{u}, \nu] = \text{housegen}(\mathbf{u})$
2) for $k = i1$ to $i2 - 2$
3) if $(k \neq i1)$
4) $[\mathbf{u}, \nu] = \text{housegen}(\mathbf{A}_{k:k+2,k-1})$
5) $\mathbf{A}_{k,k-1} = \nu$; $\mathbf{A}_{k+1:k+2,k-1} = 0$
6) end if
7) $\mathbf{v}^T = \mathbf{u}^T \times \mathbf{A}_{k:k+2,k:n}$; $\mathbf{A}_{k:k+2,k:n} = \mathbf{A}_{k:k+2,k:n} - \mathbf{u}\mathbf{v}^T$
8) $\mathbf{v}^T = \mathbf{u}^T \times \mathbf{B}_{k:k+2,k:n}$; $\mathbf{B}_{k:k+2,k:n} = \mathbf{B}_{k:k+2,k:n} - \mathbf{u}\mathbf{v}^T$
9) $\mathbf{v} = \mathbf{Q}_{:,k:k+2} \times \mathbf{u}$; $\mathbf{Q}_{:,k:k+2} = \mathbf{Q}_{:,k:k+2} - \mathbf{v}\mathbf{u}^T$
10) $[\mathbf{u}^T, \nu] = \text{housegen}(\mathbf{B}_{k+2,k:k+2} \times \mathbf{I})$
11) $\mathbf{B}_{k+2,k+2} = \nu$; $\mathbf{B}_{k+2,k:k+1} = 0$
12) $\mathbf{u}^T = \mathbf{u}^T \times \mathbf{I}$
13) $\mathbf{v} = \mathbf{B}_{1:k+1,k:k+2} \times \mathbf{u}$; $\mathbf{B}_{1:k+1,k:k+2} = \mathbf{B}_{1:k+1,k:k+2} - \mathbf{v}\mathbf{u}^T$
14) $kk = \min(k+3, i2)$
15) $\mathbf{v} = \mathbf{A}_{1:kk,k:k+2} \times \mathbf{u}$; $\mathbf{A}_{1:kk,k:k+2} = \mathbf{A}_{1:kk,k:k+2} - \mathbf{v}\mathbf{u}^T$
16) $\mathbf{v} = \mathbf{Z}_{:,k:k+2} \times \mathbf{u}$; $\mathbf{Z}_{:,k:k+2} = \mathbf{Z}_{:,k:k+2} - \mathbf{v}\mathbf{u}^T$
17) $[\mathbf{B}_{k+1,k+1}, \mathbf{B}_{k+1,k}, C, S] = \text{rotgen}(\mathbf{B}_{k+1,k+1}, \mathbf{B}_{k+1,k})$
18) $[\mathbf{B}_{1:k,k+1}, \mathbf{B}_{1:k,k}] = \text{rotapp}(C, S, \mathbf{B}_{1:k,k+1}, \mathbf{B}_{1:k,k})$
19) $[\mathbf{A}_{1:kk,k+1}, \mathbf{A}_{1:kk,k}] = \text{rotapp}(C, S, \mathbf{A}_{1:kk,k+1}, \mathbf{A}_{1:kk,k})$
20) $[\mathbf{Z}_{:,k+1}, \mathbf{Z}_{:,k}] = \text{rotapp}(C, S, \mathbf{Z}_{:,k+1}, \mathbf{Z}_{:,k})$
21) end for k
22) $[\mathbf{A}_{i2-1,i2-2}, \mathbf{A}_{i2,i2-2}, C, S] = \text{rotgen}(\mathbf{A}_{i2-1,i2-2}, \mathbf{A}_{i2,i2-2})$
23) $[\mathbf{A}_{i2-1,i2-1:i2}, \mathbf{A}_{i2,i2-1:i2}] = \text{rotapp}(C, S, \mathbf{A}_{i2-1,i2-1:i2}, \mathbf{A}_{i2,i2-1:i2})$
24) $[\mathbf{B}_{i2-1,k2-1:i2}, \mathbf{B}_{i2,i2-1:i2}] = \text{rotapp}(C, S, \mathbf{B}_{i2-1,k2-1:i2}, \mathbf{B}_{i2,i2-1:i2})$
25) $[\mathbf{Q}_{:,i2-1}, \mathbf{Q}_{:,i2}] = \text{rotapp}(C, S, \mathbf{Q}_{:,i2-1}, \mathbf{Q}_{:,i2})$
26) $[\mathbf{B}_{i2,i2}, \mathbf{B}_{i2,i2-1}, C, S] = \text{rotgen}(\mathbf{B}_{i2,i2}, \mathbf{B}_{i2,i2-1})$
27) $[\mathbf{B}_{1:i2-1,i2}, \mathbf{B}_{1:i2-1,i2-1}] = \text{rotapp}(C, S, \mathbf{B}_{1:i2-1,i2}, \mathbf{B}_{1:i2-1,i2-1})$
28) $[\mathbf{A}_{1:i2,i2}, \mathbf{A}_{1:i2,i2-1}] = \text{rotapp}(C, S, \mathbf{A}_{1:i2,i2}, \mathbf{A}_{1:i2,i2-1})$
29) rotapp$(C, S, \mathbf{Z}_{:,i2}, \mathbf{Z}_{:,i2-1})$
30) end qz2step

$$\begin{aligned}\frac{\partial \mathbf{A}}{\partial \psi} &= \mathtt{diag}(0_p, d\mathbf{Q}, 0_q)^T \times \mathbf{A} \times \mathtt{diag}(\mathbf{I}_p, \mathbf{Z}, \mathbf{I}_q)^T \\ &+ \mathtt{diag}(\mathbf{I}_p, \mathbf{Q}, \mathbf{I}_q)^T \times d\mathbf{A} \times \mathtt{diag}(\mathbf{I}_p, \mathbf{Z}, \mathbf{I}_q) \\ &+ \mathtt{diag}(\mathbf{I}_p, \mathbf{Q}, \mathbf{I}_q)^T \times \mathbf{A} \times \mathtt{diag}(0_p, d\mathbf{Z}, 0_q) \end{aligned} \tag{9}$$

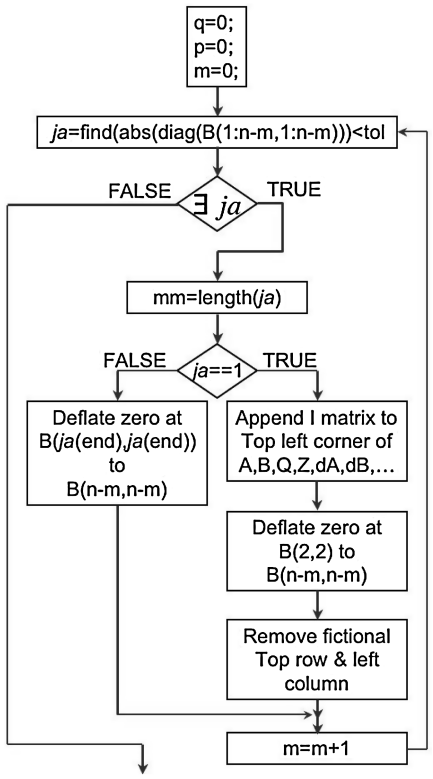

Fig. 1 Logic flowchart for deflation in multiple-input/multiple-output systems.

$$\begin{aligned}\frac{\partial \mathbf{B}}{\partial \psi} &= \mathtt{diag}(0_p, d\mathbf{Q}, 0_q)^T \times \mathbf{B} \times \mathtt{diag}(\mathbf{I}_p, \mathbf{Z}, \mathbf{I}_q)^T \\ &+ \mathtt{diag}(\mathbf{I}_p, \mathbf{Q}, \mathbf{I}_q)^T \times d\mathbf{B} \times \mathtt{diag}(\mathbf{I}_p, \mathbf{Z}, \mathbf{I}_q) \\ &+ \mathtt{diag}(\mathbf{I}_p, \mathbf{Q}, \mathbf{I}_q)^T \times \mathbf{B} \times \mathtt{diag}(0_p, d\mathbf{Z}, 0_q) \end{aligned} \tag{10}$$

$$\mathbf{A} = \mathtt{diag}(\mathbf{I}_p, \mathbf{Q}, \mathbf{I}_q)^T \times \mathbf{A} \times \mathtt{diag}(\mathbf{I}_p, \mathbf{Z}, \mathbf{I}_q) \tag{11}$$

$$\mathbf{B} = \mathtt{diag}(\mathbf{I}_p, \mathbf{Q}, \mathbf{I}_q)^T \times \mathbf{B} \times \mathtt{diag}(\mathbf{I}_p, \mathbf{Z}, \mathbf{I}_q) \tag{12}$$

After each QZ sweep, test the upper left and lower right corners of $\mathbf{A}_{22}$ for new columns in triangular form. If none appear, test the $p+1{:}p+2$, and $p+1{:}p+2$ blocks of the pencil for complex generalized eigenvalues. In this case, two columns can be concatenated to $[\mathbf{A}_{11}, \mathbf{B}_{11}]$ and truncated from $[\mathbf{A}_{22}, \mathbf{B}_{22}]$. Increment p and q appropriately and repeat from Eq. (7). Interestingly, the $d\mathbf{A}$ and $d\mathbf{B}$ matrices automatically have the same sparsity pattern as $\mathbf{A}$ and $\mathbf{B}$, except when computing sensitivities of transmission zeros.

C. Extracting the Eigenvalues

Up to this point, we have used only real arithmetic. The original QZ paper [22] proposed further reducing the real quasi-triangular $\mathbf{A}$ to complex triangular form. Here, we seek to merely extract the eigenvalues and their derivatives from the generalized Schur-form matrices. This is done by decomposing the generalized Schur matrices into 1×1 and 2×2 subproblems. The eigenvalues of the 1×1 blocks are real and found simply by $\lambda_i = a_{i,i}/b_{i,i}$. Their derivatives are likewise given by

$$\frac{\partial \lambda}{\partial \psi} = \frac{1}{b_{i,i}} \frac{\partial a_{i,i}}{\partial \psi} - \frac{\partial b_{i,i}}{\partial \psi} \frac{a_{i,i}}{b_{i,i}^2} \tag{13}$$

Eigenvalues of the 2×2 subproblems are given by Eq. (5.1) from Moler and Stewart [22]. Eigenvalue derivatives are found from the generalized Schur matrices and their derivatives by simply differentiating Eq (5.1):

$$\mu = a_{1,1}/b_{1,1} \tag{14}$$

$$\frac{\partial \mu}{\partial \psi} = \frac{1}{b_{1,1}}\frac{\partial a_{1,1}}{\partial \psi} - \frac{\partial b_{1,1}}{\partial \psi}\frac{a_{1,1}}{b_{1,1}^2} \tag{15}$$

$$a_{1,2}^{\dagger} = a_{1,2} - \mu b_{1,2} \tag{16}$$

$$\frac{\partial a_{1,2}^{\dagger}}{\partial \psi} = \frac{\partial a_{1,2}}{\partial \psi} - \frac{\partial \mu}{\partial r} b_{1,2} - \mu \frac{\partial b_{1,2}}{\partial \psi} \tag{17}$$

$$a_{2,2}^{\dagger} = a_{2,2} - \mu b_{2,2} \tag{18}$$

$$\frac{\partial a_{2,2}^{\dagger}}{\partial \psi} = \frac{\partial a_{2,2}}{\partial \psi} - \frac{\partial \mu}{\partial \psi} b_{2,2} - \mu \frac{\partial b_{2,2}}{\partial \psi} \tag{19}$$

$$p = \frac{1}{2}\left(\frac{a_{2,2}^{\dagger}}{b_{2,2}} - \frac{b_{1,2}a_{2,1}}{b_{1,1}b_{2,2}}\right) \tag{20}$$

$$\frac{\partial p}{\partial \psi} = \frac{1}{2b_{2,2}}\left(\frac{\partial a_{2,2}^{\dagger}}{\partial \psi} - \frac{a_{2,2}^{\dagger}}{b_{2,2}}\frac{\partial b_{2,2}}{\partial \psi} - \frac{\partial b_{1,2}}{\partial \psi}\frac{a_{2,1}}{b_{1,1}} - \frac{\partial a_{2,1}}{\partial \psi}\frac{b_{1,2}}{b_{1,1}} + \frac{b_{1,2}a_{2,1}}{b_{1,1}^2}\frac{\partial b_{1,1}}{\partial \psi} + \frac{b_{1,2}a_{2,1}}{b_{1,1}b_{2,2}}\frac{\partial b_{2,2}}{\partial \psi}\right) \tag{21}$$

$$q = \frac{a_{2,1}a_{1,2}^{\dagger}}{b_{1,1}b_{2,2}} \tag{22}$$

$$\frac{\partial q}{\partial \psi} = \frac{1}{b_{1,1}b_{2,2}}\left(\frac{\partial a_{2,1}}{\partial \psi}a_{1,2}^{\dagger} + a_{2,1}\frac{\partial a_{1,2}^{\dagger}}{\partial \psi} - \frac{a_{2,1}a_{1,2}^{\dagger}}{b_{1,1}}\frac{\partial b_{1,1}}{\partial \psi} - \frac{a_{2,1}a_{1,2}^{\dagger}}{b_{2,2}}\frac{\partial b_{2,2}}{\partial \psi}\right) \tag{23}$$

$$r = p^2 + q \tag{24}$$

$$\frac{\partial r}{\partial \psi} = 2p\frac{\partial p}{\partial \psi} + \frac{\partial q}{\partial \psi} \tag{25}$$

$$\lambda = \mu + p + \mathtt{sign}(p)\sqrt{r} \tag{26}$$

$$\frac{\partial \lambda}{\partial \psi} = \frac{\partial \mu}{\partial \psi} + \frac{\partial p}{\partial \psi} + \mathtt{sign}(p)\frac{1}{2}(r)^{-1/2}\frac{\partial r}{\partial \psi} \tag{27}$$

III. Batch Calculation of the Residues

The partial fraction expansion is a method of reducing a ratio of high-order polynomials to a sum of fractions, each having a first-order denominator and, in general, a complex-valued numerator. The complex numerators are commonly known as the system residues. Historically, computation of the residues has been facilitated by the cover-up method (see Oppenheim and Schafer [27] for the discrete time case or Ogata [28] for the continuous-time case).

The partial fraction expansion of a system transfer function

$$F(s) = \frac{B(s)}{A(s)}$$

$$\frac{B(s)}{A(s)} = \frac{K(s+z_1)(s+z_2)\cdots(s+z_m)}{(s+p_1)(s+p_2)\cdots(s+p_n)}, \quad \text{for } m < n$$

can be written as [28]

$$\frac{B(s)}{A(s)} = \frac{r_1}{(s+p_1)} + \frac{r_2}{(s+p_2)} + \cdots + \frac{r_n}{(s+p_n)} \tag{28}$$

A. Systems with Distinct Poles

Adding the right-hand side of Eq. (28) by finding a common denominator then equating powers of s in the numerator's right- and left-hand sides of Eq. (28) results in a set of n linear equations. This set of equations can be written in a matrix form $\mathbf{\Xi r} = \mathbf{R}$, where $\mathbf{R}$ is the array of coefficients of the original numerator polynomial in its convolved form, $\mathbf{R} = [\,\eta_1 \quad \eta_2 \quad \cdots \quad \eta_n\,]^T$:

$$B(s) = \eta_1 s^{n-1} + \eta_2 s^{n-2} + \cdots + \eta_n$$

where $\mathbf{r}$ is a column of the unknown residues r_i, and the matrix $\mathbf{\Xi}$ has a distinct pattern in terms of row i and column j:

$$\mathbf{\Xi}_{i,j} = \sum_{\substack{k,l= \\ k,l\neq j}} \binom{n-1}{i-1} (-p_k)(-p_l)\cdots \tag{29}$$

Here, the combinatoric notation

$$\binom{n-1}{i-1}$$

denotes all combinations of $i-1$ poles out of the set of $n-1$ poles when pole j is excluded. If the product of all combinations of zero poles is interpreted to be equal to unity, then Eq. (29) is an accurate way to represent the matrix defined as $\mathbf{\Xi}$. Using the `combnk` function from the statistics toolbox, Eq. (29) can be written in MATLAB code as

```
Xi(i,j) = sum(prod(combnk(-ps([1:j-1,j+1:n]),
  i-1),2))
```

Equation (29) can be unpacked for the distinct poles case as shown in Appendix B. Numerical examples are shown in the sequel.

B. Systems with Repeated Poles

In the case in which pole 1 is repeated m_k times, the correct form of the partial fraction expansion is

$$\frac{B(s)}{A(s)} = \frac{r_1}{(s+p_1)^{m_k}} + \frac{r_2}{(s+p_1)^{m_k-1}} + \cdots + \frac{r_{m_k}}{(s+p_1)} + \cdots + \frac{r_{m_k+1}}{(s+p_2)} + \cdots \frac{r_n}{(s+p_n)} \tag{30}$$

The historic method to deal with pole repetitions is to cover up the corresponding terms and then take derivatives of the resulting quotient of polynomials. This method is not difficult to implement when applying the well-known quotient rule and power rule of differentiation.

By repeating the algebraic process outlined in Sec. III.A, we discover that the resulting system of equations $\mathbf{\Xi r} = \mathbf{R}$ has the same form as Eq. (29), except that for the terms involving repeated poles, the corresponding column of the $\mathbf{\Xi}$ matrix can be built from the bottom up using Eq. (29), pretending that the system is lacking $m_k - j + 1$ occurrences of the repeated pole. For convenience, we will affiliate the first m_k columns of the $\mathbf{\Xi}$ matrix with the pole of multiplicity m_k. Thus, j is the matrix column number as defined previously. The top $m_k - j$ of said column will then be filled in with zeros.

The leftmost portion of the $\mathbf{\Xi}$ matrix (corresponding to a pole with multiplicity m_k) is unpacked in Appendix B as well. The lower trapezoidal pattern that results ensures that the $\mathbf{\Xi}$ matrix is well conditioned in the repeated pole case as well.

C. Derivatives of the Residues

Given the batch solution of the residues $\mathbf{r} = \mathbf{\Xi}^{-1}\mathbf{R}$, the first derivatives of the residues follow from the application of the product rule:

$$\frac{\partial \mathbf{r}}{\partial \psi} = \frac{\partial \Xi^{-1}}{\partial \psi}\mathbf{R} + \Xi^{-1}\frac{\partial \mathbf{R}}{\partial \psi}$$

where ψ is an adjustable parameter in the overall (closed-loop) system. It is also relatively easy to show that

$$\frac{\partial \Xi^{-1}}{\partial \psi} = -\Xi^{-1}\frac{\partial \Xi}{\partial \psi}\Xi^{-1}$$

Then substituting, we obtain

$$\frac{\partial \mathbf{r}}{\partial \mathbf{K}} = -\Xi^{-1}\frac{\partial \Xi}{\partial \psi}\Xi^{-1}\mathbf{R} + \Xi^{-1}\frac{\partial \mathbf{R}}{\partial \psi} \tag{31}$$

Equation (31) will be used to compute the derivatives of the residues such that $\partial\Xi/\partial\psi$ need not be inverted. This is certainly good news, because the top row of Eq. (29) is invariant, and thus the top row of $\partial\Xi/\partial\psi$ will be zeros, implying that $\partial\Xi/\partial\psi$ will in no case have full row rank. Knowing the individual eigenvalue sensitivities, the quantity $\partial\Xi/\partial\psi$ is obtained from element-by-element application of product rule to Eq. (29). The result can be written as

$$\frac{\partial \Xi_{i,j}}{\partial \psi} = \sum_{\substack{m=1 \\ m\neq j}}^{n}\left(-\frac{\partial p_m}{\partial \mathbf{K}}\right) \prod_{s=\binom{n-2}{i-2}_{s\neq j}} (-p_s) \tag{32}$$

Equation (32) is unpacked in Appendix B.

The term $\partial\mathbf{R}/\partial\psi$ is found from the system state-space description and applicable derivatives. Using the familiar formula for the transfer function numerator,

$$B(s) = \phi(\mathbf{a} - \mathbf{bc}) + (\mathbf{d} - 1)A(s) \tag{33}$$

where $\phi(.)$ is the characteristic polynomial [i.e., $A(s) = \phi(\mathbf{a})$]. The roots of $A(s) = \phi(\mathbf{a})$ and $\phi(\mathbf{a} - \mathbf{bc})$ and respective derivatives of these roots can be found using the QZ algorithm as described previously. That is, the roots of $A(s)$ and their derivatives are found as the generalized eigenvalues of the matrix pencil $[\,\mathbf{a}, \quad \mathbf{I}\,]$, and the pole derivatives are found in a combined algorithm in which the pencil derivative is

$$\left[\frac{\partial \mathbf{a}}{\partial \psi}, 0\right]$$

The roots of $\phi(\mathbf{a} - \mathbf{bc})$ are found as generalized eigenvalues of the matrix pencil $[\,\mathbf{a} - \mathbf{bc}, \quad \mathbf{I}\,]$, and appropriate derivatives are found in the combined algorithm using the pencil derivative:

$$\left[\frac{\partial \mathbf{a}}{\partial \psi} - \frac{\partial \mathbf{b}}{\partial \psi}\mathbf{c} - \mathbf{b}\frac{\partial \mathbf{c}}{\partial \psi}, 0\right]$$

Knowing the roots of a characteristic polynomial and respective derivatives, an expression for the derivative of such polynomial is found as follows. Given the characteristic polynomial in factored form,

$$A(s) = (s - p_1)(s - p_2)\cdots(s - p_n) \tag{34}$$

convolving, we obtain

$$A(s) = s^n + \sum_{i=1}^{n}(-p_i)s^{n-1} + \sum \prod_{i,j=\binom{n}{2}} (-p_i)(-p_j)s^{n-2}$$

$$+ \sum \prod_{i,j,k=\binom{n}{3}} (-p_i)(-p_j)(-p_k)s^{n-3} + \cdots \tag{35}$$

When differentiated *with respect to the roots*, we first realize

$$\frac{\partial A(s)}{\partial \psi} = \sum_j \frac{\partial A(s)}{\partial p_j}\frac{\partial p_j}{\partial \psi}$$

Then

$$\frac{\partial A(s)}{\partial \psi} = \sum_j^n\left(-\frac{\partial p_j}{\partial \psi}\right)s^{n-1} + \sum_j^n\left(-\frac{\partial p_j}{\partial \psi}\right)\left[\prod_{\substack{i=1 \\ i\neq j}}(-p_i)\right]s^{n-2}$$

$$+ \sum_j^n\left(-\frac{\partial p_j}{\partial \psi}\right)\left[\prod_{i=\binom{n-1}{2}_{i\neq j}}(-p_i)\right]s^{n-3} + \cdots$$

$$+ \sum_j^n\left(-\frac{\partial p_j}{\partial \psi}\right)\left[\prod_{i=\binom{n-1}{n-1}}(-p_i)\right]s^0 \tag{36}$$

or, in general,

$$\frac{\partial A(s)}{\partial \psi} = \sum_j^n\left(-\frac{\partial p_j}{\partial \psi}\right)\left[\sum_{m=0}^{n-1}\prod_{i=\binom{n-1}{m}_{i\neq j}}(-p_i)s^{n-1-m}\right] \tag{37}$$

The first sum of Eq. (37) can be evaluated in MATLAB for loop in which the terms to the right of the first summation become the statement

```
sum(prod(combnk(-ps([1:j-1,j+1:n]),m),2))
  × (-dps(j))
```

Finally, the coefficients of $\partial\mathbf{R}/\partial\psi$ can be found by from the derivative of Eq. (33). Assuming $\mathbf{d} = 0$,

$$\frac{\partial \mathbf{R}}{\partial \psi} = \frac{\partial B(s)}{\partial \psi} = \frac{\partial}{\partial \psi}(\phi(\mathbf{a} - \mathbf{bc})) - \frac{\partial A(s)}{\partial \psi} \tag{38}$$

where the coefficients in the first term are found by appropriate substitutions in Eq. (37).

IV. Numerical Examples

A. Derivatives of the Poles

The eigenvalue derivatives were computed using the differentiated QZ algorithm for the matrix:

$$\mathbf{A} = \begin{bmatrix} -2 & 1 & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 1 & -1 \\ 0 & 1 & -2 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 1 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 \end{bmatrix}$$

where the matrix derivative was taken to be

$$\frac{\partial \mathbf{A}}{\partial \psi} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

The pencil is completed in this case by choosing $\mathbf{B} = \mathbf{I}$ and $\partial\mathbf{B}/\partial\psi = 0$.

The resulting derivatives are shown in Table 1, in which our method is labeled as derivative and the columns labeled as FD were

found by finite differencing of the MATLAB eig routine with a perturbation of 0.0001. The results match very well. Differences are mainly attributed to approximation in finite differencing because they are the same order of magnitude as the perturbation used in finite differencing. The method presented here is deemed more accurate because all quantities required for derivative calculation are evaluated at the point of interest, in which finite differencing uses a second evaluation perturbed from the point of interest by a small amount.

The method works well for a randomly chosen pencil and randomly chosen derivatives as well. An example is shown in Appendix C.

B. Derivatives of the Transmission Zeros

A rudimentary method of calculating transmission zeros for the linear system

$$\dot{\mathbf{x}} = \mathbf{a}\mathbf{x} + \mathbf{b}\mathbf{u} \qquad \mathbf{y} = \mathbf{c}\mathbf{x} + \mathbf{d}\mathbf{u}$$

is to form the composite pencil, in which

$$\mathbf{A} = \begin{bmatrix} \mathbf{a} & \mathbf{b} \\ \mathbf{c} & \mathbf{d} \end{bmatrix}$$

and $\mathbf{B}$ is conformably partitioned as

$$\mathbf{B} = \begin{bmatrix} \mathbf{I} & 0 \\ 0 & 0 \end{bmatrix}$$

The generalized eigenvalues of $(\mathbf{A}, \mathbf{B})$ are the transmission zeros of the linear system. First we demonstrate the calculation of transmission zero derivatives on a system from Emami-Naeini and Van Dooren [3]:

$$\mathbf{A} = \begin{bmatrix} \mathbf{a} & \mathbf{b} \\ \mathbf{c} & \mathbf{d} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} -2 & 1 & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 1 & -1 \\ 0 & 1 & -2 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 1 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \\ \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} & [0] \end{bmatrix} \tag{39}$$

We assume that the number zero is invariant wherever it appears in the system description. Thus, we make the pencil derivatives

$$\frac{\partial \mathbf{A}}{\partial \psi} = \begin{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \\ \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} & [0] \end{bmatrix} \tag{40}$$

and

$$\frac{\partial \mathbf{B}}{\partial \psi} = 0$$

The results for transmission zeros and their derivatives are given in Table 2. The finite difference results were obtained by finite differencing the MATLAB tzero algorithm with a perturbation of 10^{-9}.

A second example from Emami-Naeini and Van Dooren [3] demonstrates that the method works for square multivariable systems:

$$\mathbf{A} = \begin{bmatrix} \mathbf{a} & \mathbf{b} \\ \mathbf{c} & \mathbf{d} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 \end{bmatrix} & \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} \end{bmatrix}$$

Once again, the pencil derivatives are taken to be unity for nonzero elements of the numerator and zero for all elements of the denominator; that is,

$$\frac{\partial \mathbf{A}}{\partial \psi} = \begin{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} & \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} \end{bmatrix}$$

and

$$\frac{\partial \mathbf{B}}{\partial \psi} = 0$$

The results for transmission zeros and their derivatives are given in Table 3. The finite difference results were obtained by finite differencing the MATLAB tzero algorithm with a perturbation of 10^{-9}. The similarity between zero location and derivative in three of the four cases is merely coincidental, stemming from the choice of pencil derivative structure.

Results for randomly chosen $\mathbf{A}$ and $\partial\mathbf{A}/\partial\psi$ are shown in Appendix C to demonstrate that the method yields good results regardless of the sparsity pattern of the pencil derivatives. Note that when calculating transmission zeros, elements along the diagonal of the $\mathbf{A}$ matrix affiliated with the infinite transmission zeros will become unbounded. For instance, for the first example in this section, there are two finite zeros and four infinite zeros. Thus, five values along the diagonal of the reduced $\mathbf{A}$ matrix will be unbounded, which is the correct result. When computing the transmission zero sensitivities, the sparsity pattern of $\partial\mathbf{A}/\partial\psi$ does not match that of $\mathbf{A}$,

Table 1 Pole derivative results

Real(pole)	Imag(pole)	Real(derivative)	Real(FD)	Imag(derivative)	Imag(FD)
−3.38389	0	−0.617431	−0.617461	0	0
−2.19994	0	0.666242	0.666232	0	0
−0.624778-0.624778	∓1.34337	0.755544	0.755531	∓0.160447	∓0.160355
−0.0833092	±0.487702	1.22005	1.22008	±0.046311	±0.0461307

Table 2 Transmission zero results

Zero	Actual	Derivative	Finite difference
−0.9999999999999992	−1	0.5000000000000018	0.4999993752363707
−0.9999999999999999	−1	0.5000000000000002	0.5000000413701855

Table 3 Transmission zero results

Zero	tzero(.)	Derivative	Finite difference
		Real Zeros	
0.9999999999999999	0.9999999999999986	3.000000000000000	3.000000803332625
−0.6823278038420128	−0.6823278038280189	−0.6823278035700907	−0.6823281939460912
		Complex Pair	
0.3411639019239535	0.3411639019140095	0.3411639017903962	0.3411645410622555
$\pm 1.161541400001745i$	$\pm 1.161541399997253i$	$\pm 1.161541399880581i$	$\pm 1.161541307581615i$

due to the very large values on the diagonal of **A** perturbing certain offdiagonal terms of $\partial\mathbf{A}/\partial\psi$. This is not an indication of instability in the algorithm, however, because computer results are still reliable, as shown in the respective tables.

C. Batch Calculation of Residues

The first example is a system generated using the `rand` function in MATLAB. We have tested random systems up to 25th order without observing any problems with conditioning of the Ξ matrix. We present a fourth-order example here for brevity. Given the frequency-domain function

$$\frac{0.762s^3 + 0.457s^2 + 0.019s + 0.821}{s^4 + 0.243s^3 + 0.639s^2 + 0.512s + 0.938}$$

the partial fraction expansion is

$$\frac{r_1}{(s - 0.515 - 0.911i)} + \frac{r_2}{(s - 0.515 + 0.911i)} + \frac{r_3}{(s + 0.637 - 0.672i)} + \frac{r_4}{(s + 0.637 + 0.672i)}$$

and the corresponding batch equation is

$$\begin{bmatrix} \xi_1 & \xi_1^\dagger & \xi_2 & \xi_2^\dagger \end{bmatrix} \begin{Bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{Bmatrix} = \begin{Bmatrix} 0.762 \\ 0.457 \\ 0.019 \\ 0.821 \end{Bmatrix} \tag{41}$$

where

$$\xi_1 = [\,1 \quad 0.758 + 0.911i \quad 0.2 + 1.16i \quad -0.441 + 0.78i\,]^T$$
$$\xi_2 = [\,1 \quad -0.394 + 0.672i \quad -0.439 - 0.692i \quad 0.697 + 0.735i\,]^T$$

and the † indicates complex conjugate. The residues are $r_1 = 0.107 - 0.063i$, $r_2 = r_1^\dagger$, $r_3 = 0.274 - 0.296i$, and $r_4 = r_3^\dagger$.

The second example is a system with repeated poles at $s = -1 + i$ and $-1 - i$. Given the frequency-domain function

$$\frac{1}{s^4 + 4s^3 + 8s^2 + 8s + 4} \tag{42}$$

the partial fraction expansion is

$$= \frac{r_1}{(s + 1 + i)^2} + \frac{r_2}{(s + 1 - i)^2} + \frac{r_3}{(s + 1 + i)} + \frac{r_4}{(s + 1 - i)}$$

and the corresponding batch equation is

$$\begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 3 - i & 3 + i \\ 2 - 2i & 2 + 2i & -4 - 2i & -4 + 2i \\ -2i & 2i & 2 - 2i & 2 + 2i \end{bmatrix} \begin{Bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{Bmatrix} \tag{43}$$

The solution of which is $r_1 = -0.25$, $r_2 = -0.25$, $r_3 = 0.25i$, and $r_4 = -0.25i$, which is the same result obtained by established methods.

D. Sensitivities of the Residues

Next, we demonstrate batch calculation of the residue sensitivities for a case with distinct poles. The frequency-domain function used is

$$\frac{0.762s^3 + 0.457s^2 + 0.019s + 0.821}{s^4 + 0.243s^3 + 0.639s^2 + 0.512s + 0.938}$$

Results for the batch calculation matrix Ξ and residues are shown as the first example in Sec. IV.C. This example is intended to focus on finding the residue derivatives, and so we consider the pole derivatives to be known and arbitrarily choose the following values:

$$\begin{Bmatrix} \partial p_{1,2}/\partial\psi \\ \partial p_{3,4}/\partial\psi \end{Bmatrix} = \begin{Bmatrix} .0765 \mp 0.139i \\ .657 \mp .108i \end{Bmatrix}$$

We also consider the derivative of the numerator coefficients to be known and equal to zero, except for the s^0 term, which has a derivative of unity. The correct form of the $\partial\Xi/\partial\psi$ matrix is

$$\frac{\partial\Xi}{\partial\psi} = \begin{bmatrix} \xi_1 & \xi_1^\dagger & \xi_2 & \xi_2^\dagger \end{bmatrix}$$

where

$$\xi_1 = \begin{bmatrix} 0 & -1.3905 - 0.139i & -0.40214 - 1.3738i \\ & 0.44043 - 1.0133i & \end{bmatrix}^T$$
$$\xi_2 = \begin{bmatrix} 0 & -0.81000 - 0.10800i & 0.40533 + 0.0085589i \\ & -0.83052 - 0.23537i & \end{bmatrix}^T$$

The resulting residue derivatives are shown in Table 4. The finite difference results stem from a second batch calculation with system poles perturbed by $\partial\mathbf{p}/\partial\psi(10^{-4})$ and the s^0 numerator coefficient perturbed by (10^{-4}).

Next, we investigate derivatives of the residues of the system shown in Eqs. (39) and (40). In transfer function form, the system is

$$\frac{B(s)}{A(s)} = \frac{4s^2 + 8s + 4}{s^6 + 7s^5 + 18s^4 + 26s^3 + 24s^2 + 8s + 4}$$

There are two system transmission zeros at $s = -1$. The partial fraction expansion is

Table 4 Residue derivative results, distinct poles

Real(residue)	Imag(residue)	Real(derivative)	Real(FD)	Imag(derivative)	Imag(FD)
0.1067	$\mp 0.0625i$	−0.17761	−0.17764	$\mp 0.28942i$	$\mp 0.28944i$
0.2743	$\mp 0.2958i$	0.17761	0.17764	$\mp 0.52449i$	$\mp 0.52450i$

$$\frac{B(s)}{A(s)} = \frac{r_1}{s+3.384} + \frac{r_2}{s+2.200} + \frac{r_3}{s+0.625+1.343i} + \frac{r_3^\dagger}{s+0.625-1.343i} + \frac{r_4}{s+0.0833-0.488i} + \frac{r_4^\dagger}{s+0.0833+0.488i}$$

The residues are $r_1 = -0.183$, $r_2 = 0.241$, $r_3 = -0.130–0.197i$, and $r_4 = 0.102–0.299i$. The system pole derivatives are shown in Table 1. For system transmission zero derivatives, see Table 2. Applying Eq. (13), we find

$$\frac{\partial \mathbf{R}}{\partial \psi} = [20 \quad 36 \quad 16]^T \tag{44}$$

which matches exactly with finite differencing. Applying Eq. (6), the residue derivatives that occur in complex-conjugate pairs are

$$\begin{Bmatrix} \partial r_1/\partial\psi \\ \partial r_2/\partial\psi \\ \partial r_3/\partial\psi \\ \partial r_4/\partial\psi \end{Bmatrix} = \begin{Bmatrix} -0.5328 \\ 0.7136 \\ -0.4067 - 0.9127i \\ 0.3164 - 1.5115i \end{Bmatrix}$$

and match finite differencing results to at least four decimal places.

As of this writing, residue sensitivities cannot be demonstrated in the case of repeated system poles. This stems from the fact that repeated poles will have distinct derivatives. In constructing the portion of the Ξ matrix corresponding to repeated poles, a column is constructed ignoring all occurrences of the repeated poles. Subsequent columns ignore one less occurrence, as the matrix is constructed from left to right, until only one occurrence is ignored. When attempting to reintroduce occurrences in the differentiated version, it is unclear in which order the pole derivatives should be included.

V. Conclusions

This paper has developed and demonstrated a novel approach for computing the first derivatives of system poles, transmission zeros, and residues by differentiating a known and trusted algorithm. Through differentiation of the QZ algorithm, reliable sensitivities of system poles and zeros become available. These results are then recombined to form an algorithm for derivatives of the system residues. Numerical results match finite differencing approximations. These algorithms may open the door to new methods for brute force optimization and direct design by pole and zero placement.

Appendix A: Longer Differential Subroutines

Algorithms A1 and A2 show the more lengthy subroutines with derivatives in place. In each subroutine call, a d prefix indicates the partial derivative with respect to ψ of the corresponding variable, for instance, $d\nu = \partial\nu/\partial\psi$:

Algorithm A2 is the differentiated subroutine for reduction to Hessenberg-triangular form:

Here are the derivatives of Moler and Stewart [22] (Eq. 4.1):

$$\begin{aligned}\frac{\partial \mathbf{u}_1}{\partial\psi} &= \left(\frac{\mathbf{B}_{1,1}}{\mathbf{A}_{2,1}}\right)\left[\left(\frac{1}{\mathbf{B}_{m,m}}\frac{\partial \mathbf{A}_{m,m}}{\partial\psi} - \frac{\partial \mathbf{B}_{m,m}}{\partial\psi}\frac{\mathbf{A}_{m,m}}{\mathbf{B}_{m,m}^2} - \frac{1}{\mathbf{B}_{1,1}}\frac{\partial \mathbf{A}_{1,1}}{\partial\psi} + \frac{\partial \mathbf{B}_{1,1}}{\partial\psi}\frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}^2}\right) \times \left(\frac{\mathbf{A}_{n,n}}{\mathbf{B}_{n,n}} - \frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}}\right) + \left(\frac{\mathbf{A}_{m,m}}{\mathbf{B}_{m,m}} - \frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}}\right) \times \left(\frac{1}{\mathbf{B}_{n,n}}\frac{\partial \mathbf{A}_{n,n}}{\partial\psi} - \frac{\partial \mathbf{B}_{n,n}}{\partial\psi}\frac{\mathbf{A}_{n,n}}{\mathbf{B}_{n,n}^2} - \frac{1}{\mathbf{B}_{1,1}}\frac{\partial \mathbf{A}_{1,1}}{\partial\psi} + \frac{\partial \mathbf{B}_{1,1}}{\partial\psi}\frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}^2}\right) \\ &\quad - \frac{\partial \mathbf{A}_{m,n}}{\partial\psi}\frac{\mathbf{A}_{n,m}}{\mathbf{B}_{n,n}\mathbf{B}_{m,m}} + \frac{\partial \mathbf{B}_{n,n}}{\partial\psi}\frac{\mathbf{A}_{m,n}\mathbf{A}_{n,m}}{\mathbf{B}_{m,m}\mathbf{B}_{n,n}^2} - \frac{\partial \mathbf{A}_{n,m}}{\partial\psi}\frac{\mathbf{A}_{m,n}}{\mathbf{B}_{n,n}\mathbf{B}_{m,m}} + \frac{\partial \mathbf{B}_{m,m}}{\partial\psi}\frac{\mathbf{A}_{m,n}\mathbf{A}_{n,m}}{\mathbf{B}_{n,n}\mathbf{B}_{m,m}^2} + \frac{\partial \mathbf{A}_{n,m}}{\partial\psi}\frac{\mathbf{B}_{m,n}\mathbf{A}_{1,1}}{\mathbf{B}_{m,m}\mathbf{B}_{n,n}\mathbf{B}_{1,1}} \\ &\quad + \frac{\partial \mathbf{B}_{m,n}}{\partial\psi}\frac{\mathbf{A}_{n,m}\mathbf{A}_{1,1}}{\mathbf{B}_{m,m}\mathbf{B}_{n,n}\mathbf{B}_{1,1}} + \frac{\partial \mathbf{A}_{1,1}}{\partial\psi}\frac{\mathbf{A}_{n,m}\mathbf{B}_{m,n}}{\mathbf{B}_{m,m}\mathbf{B}_{n,n}\mathbf{B}_{1,1}} - \frac{\partial \mathbf{B}_{m,m}}{\partial\psi}\frac{\mathbf{A}_{n,m}\mathbf{B}_{m,n}\mathbf{A}_{1,1}}{\mathbf{B}_{n,n}\mathbf{B}_{1,1}\mathbf{B}_{m,m}^2} \cdots - \frac{\partial \mathbf{B}_{n,n}}{\partial\psi}\frac{\mathbf{A}_{n,m}\mathbf{A}_{1,1}}{\mathbf{B}_{m,m}\mathbf{B}_{1,1}\mathbf{B}_{n,n}^2} \\ &\quad - \frac{\partial \mathbf{B}_{1,1}}{\partial\psi}\frac{\mathbf{A}_{n,m}\mathbf{B}_{m,n}\mathbf{A}_{1,1}}{\mathbf{B}_{m,m}\mathbf{B}_{n,n}\mathbf{B}_{1,1}^2}\Bigg] + \left(\frac{1}{\mathbf{A}_{2,1}}\frac{\partial \mathbf{B}_{1,1}}{\partial\psi} - \frac{\partial \mathbf{A}_{2,1}}{\partial\psi}\frac{\mathbf{B}_{1,1}}{\mathbf{A}_{2,1}^2}\right) \\ &\quad \times \left[\left(\frac{\mathbf{A}_{m,m}}{\mathbf{B}_{m,m}} - \frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}}\right)\left(\frac{\mathbf{A}_{n,n}}{\mathbf{B}_{n,n}} - \frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}}\right) - \frac{\mathbf{A}_{m,n}\mathbf{A}_{n,m}}{\mathbf{B}_{n,n}\mathbf{B}_{m,m}} + \frac{\mathbf{A}_{n,m}\mathbf{B}_{m,n}\mathbf{A}_{1,1}}{\mathbf{B}_{m,m}\mathbf{B}_{n,n}\mathbf{B}_{1,1}}\right] + \frac{1}{\mathbf{B}_{2,2}}\frac{\partial \mathbf{A}_{1,2}}{\partial\psi} - \frac{\partial \mathbf{B}_{2,2}}{\partial\psi}\frac{\mathbf{A}_{1,2}}{\mathbf{B}_{2,2}^2} - \frac{\partial \mathbf{A}_{1,1}}{\partial\psi}\frac{\mathbf{B}_{1,2}}{\mathbf{B}_{1,1}\mathbf{B}_{2,2}} \\ &\quad + \frac{\partial \mathbf{B}_{1,1}}{\partial\psi}\frac{\mathbf{A}_{1,1}\mathbf{B}_{1,2}}{\mathbf{B}_{2,2}\mathbf{B}_{1,1}^2} - \frac{\partial \mathbf{B}_{1,2}}{\partial\psi}\frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}\mathbf{B}_{2,2}} + \frac{\partial \mathbf{B}_{2,2}}{\partial\psi}\frac{\mathbf{A}_{1,1}\mathbf{B}_{1,2}}{\mathbf{B}_{1,1}\mathbf{B}_{2,2}^2}\end{aligned} \tag{A3}$$

$$\begin{aligned}\frac{\partial \mathbf{u}_2}{\partial\psi} &= \frac{1}{\mathbf{B}_{2,2}}\frac{\partial \mathbf{A}_{2,2}}{\partial\psi} - \frac{\mathbf{A}_{2,2}}{\mathbf{B}_{2,2}^2}\frac{\partial \mathbf{B}_{2,2}}{\partial\psi} - \frac{1}{\mathbf{B}_{1,1}}\frac{\partial \mathbf{A}_{1,1}}{\partial\psi} + \frac{\mathbf{A}_{1,1}}{bfB_{1,1}^2}\frac{\partial \mathbf{B}_{1,1}}{\partial\psi} - \frac{\partial \mathbf{A}_{2,1}}{\partial\psi}\frac{\mathbf{B}_{1,2}}{\mathbf{B}_{1,1}\mathbf{B}_{2,2}} - \frac{\partial \mathbf{B}_{1,2}}{\partial\psi}\frac{\mathbf{A}_{2,1}}{\mathbf{B}_{1,1}\mathbf{B}_{2,2}} + \frac{\mathbf{A}_{2,1}\mathbf{B}_{1,2}}{\mathbf{B}_{2,2}\mathbf{B}_{1,1}^2}\frac{\partial \mathbf{B}_{1,1}}{\partial\psi} \\ &\quad + \frac{\partial \mathbf{B}_{2,2}}{\partial\psi}\frac{\mathbf{A}_{2,1}\mathbf{B}_{1,2}}{\mathbf{B}_{1,1}\mathbf{B}_{2,2}^2} - \left(\frac{1}{\mathbf{B}_{m,m}}\frac{\partial \mathbf{A}_{m,m}}{\partial\psi} - \frac{\partial \mathbf{B}_{m,m}}{\partial\psi}\frac{\mathbf{A}_{m,m}}{\mathbf{B}_{m,m}^2} - \frac{1}{\mathbf{B}_{1,1}}\frac{\partial \mathbf{A}_{1,1}}{\partial\psi} + \frac{\partial \mathbf{B}_{1,1}}{\partial\psi}\frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}^2}\right) - \left(\frac{1}{\mathbf{B}_{n,n}}\frac{\partial \mathbf{A}_{n,n}}{\partial\psi} - \frac{\partial \mathbf{B}_{n,n}}{\partial\psi}\frac{\mathbf{A}_{n,n}}{\mathbf{B}_{n,n}^2} - \frac{1}{\mathbf{B}_{1,1}}\frac{\partial \mathbf{A}_{1,1}}{\partial\psi} + \frac{\partial \mathbf{B}_{1,1}}{\partial\psi}\frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}^2}\right) + \frac{\partial \mathbf{A}_{n,m}}{\partial\psi}\frac{\mathbf{B}_{m,n}}{\mathbf{B}_{m,m}\mathbf{B}_{n,n}} - \frac{\partial \mathbf{B}_{m,m}}{\partial\psi}\frac{\mathbf{A}_{n,m}\mathbf{B}_{m,n}}{\mathbf{B}_{n,n}\mathbf{B}_{m,m}^2} \\ &\quad + \frac{\partial \mathbf{B}_{m,n}}{\partial\psi}\frac{\mathbf{A}_{n,m}}{\mathbf{B}_{m,m}\mathbf{B}_{n,n}} - \frac{\partial \mathbf{B}_{n,n}}{\partial\psi}\frac{\mathbf{A}_{n,m}\mathbf{B}_{m,n}}{\mathbf{B}_{m,m}\mathbf{B}_{n,n}^2}\end{aligned} \tag{A4}$$

$$\frac{\partial \mathbf{u}_3}{\partial\psi} = \frac{1}{\mathbf{B}_{2,2}}\frac{\partial \mathbf{A}_{3,2}}{\partial\psi} - \frac{\mathbf{A}_{3,2}}{\mathbf{B}_{2,2}^2}\frac{\partial \mathbf{B}_{2,2}}{\partial\psi} \tag{A5}$$

In the case that $\mathbf{B}_{2,2} = 0$, these simplify to

$$\frac{\partial \mathbf{u}_1}{\partial\psi} = \frac{\partial \mathbf{A}_{1,2}}{\partial\psi} - \frac{\partial \mathbf{A}_{1,1}}{\partial\psi}\frac{\mathbf{B}_{1,2}}{\mathbf{B}_{1,1}} - \frac{\partial \mathbf{B}_{1,2}}{\partial\psi}\frac{\mathbf{A}_{1,1}}{\mathbf{B}_{1,1}} + \frac{\partial \mathbf{B}_{1,1}}{\partial\psi}\frac{\mathbf{A}_{1,2}\mathbf{B}_{1,2}}{\mathbf{B}_{1,1}^2} \tag{A6}$$

$$\frac{\partial \mathbf{u}_2}{\partial\psi} = \frac{\partial \mathbf{A}_{2,2}}{\partial\psi} - \frac{\partial \mathbf{A}_{2,1}}{\partial\psi}\frac{\mathbf{B}_{1,2}}{\mathbf{B}_{1,1}} - \frac{\partial \mathbf{B}_{1,2}}{\partial\psi}\frac{\mathbf{A}_{2,1}}{\mathbf{B}_{1,1}} + \frac{\partial \mathbf{B}_{1,1}}{\partial\psi}\frac{\mathbf{A}_{2,1}\mathbf{B}_{1,2}}{\mathbf{B}_{1,1}^2} \tag{A7}$$

$$\frac{\partial \mathbf{u}_3}{\partial\psi} = \frac{\partial \mathbf{A}_{3,2}}{\partial\psi} \tag{A8}$$

Algorithm A1 Generate Householder reflector and first derivative

1) $[\mathbf{u}, \quad \nu, \quad d\mathbf{u}, \quad d\nu] = d\text{housegen}(\mathbf{a}, \quad d\mathbf{a})$
2) $\mathbf{u} = \mathbf{a}$
   a) $\dfrac{\partial \mathbf{u}}{\partial\psi} = \dfrac{\partial \mathbf{a}}{\partial\psi}$
3) $\nu = \|\mathbf{a}\|_2$
   a) $\dfrac{\partial\nu}{\partial\psi} = \dfrac{1}{2}\dfrac{1}{\|\mathbf{a}\|_2}(d\mathbf{a}_{(:)}^T \quad \times \mathbf{a}_{(:)} \quad + \mathbf{a}_{(:)}^T \quad \times d\mathbf{a}_{(:)})$
4) if $(\nu = 0)$
5) $\mathbf{u}_1 = \sqrt{2}$; $d\mathbf{u}_1 = 0$; return; end if
6) $\rho = \texttt{sign}(\mathbf{u}_1) + (\mathbf{u}_1 == 0)$
   a) $\mathbf{tu} = (\rho/\nu) \times \mathbf{u}$
   b) $d\mathbf{u} = (\rho/\nu) \times d\mathbf{u} - \rho \times \mathbf{u} \times d\nu/\nu^2$
7) $\mathbf{tu}_1 = 1 + \mathbf{tu}_1$
8) $\mathbf{u} = \mathbf{tu}$
9) $\mathbf{tu} = \mathbf{tu}/\sqrt{\mathbf{tu}_1}$
   a) $d\mathbf{u} = d\mathbf{u}/\sqrt{\mathbf{u}_1} - \frac{1}{2}\mathbf{u} \times (\mathbf{u}_1)^{-3/2} \times d\mathbf{u}_1$
10) $\mathbf{u} = \mathbf{tu}$
11) $\nu = -\rho\nu$
   a) $d\nu = -\rho d\nu$
12) end dhousegen

Algorithm A2 Reduce A, B, dA, and dB to Hessenberg-triangular form

1) $[\mathbf{A},\ \mathbf{B},\ \mathbf{Q},\ \mathbf{Z},\ d\mathbf{A},\ d\mathbf{B},\ d\mathbf{Q},\ d\mathbf{Z}] = d\text{HessTriForm}(\mathbf{A},\ \mathbf{B},\ d\mathbf{A},\ d\mathbf{B})$
   a) Reduce **B** to triangular form.
2) $\mathbf{Q} = \mathbf{I}$
   a) $d\mathbf{Q} = 0$
3) for $k = 1$ **to** $n-1$
4) $[\mathbf{u},\ \text{temp},\ d\mathbf{u}] = d\text{housegen}(\mathbf{B}_{k:n,k},\ d\mathbf{B}_{k:n,k})$
5) $\mathbf{v}^T = \mathbf{u}^T \times \mathbf{B}_{k:n,k:n}$
   a) $d\mathbf{v}^T = d\mathbf{u}^T \times \mathbf{B}_{k:n,k:n} + \mathbf{u}^T \times d\mathbf{B}_{k:n,k:n}$
6) $\mathbf{B}_{k:n,k:n} = \mathbf{B}_{k:n,k:n} - \mathbf{u} \times \mathbf{v}^T$
   a) $d\mathbf{B}_{k:n,k:n} = d\mathbf{B}_{k:n,k:n} - d\mathbf{u} \times \mathbf{v}^T - \mathbf{u} \times d\mathbf{v}^T$
7) $\mathbf{v}^T = \mathbf{u}^T \times \mathbf{A}_{k:n,:}$
   a) $d\mathbf{v}^T = d\mathbf{u}^T \times \mathbf{A}_{k:n,:} + \mathbf{u}^T \times d\mathbf{A}_{k:n,:}$
8) $\mathbf{A}_{k:n,:} = \mathbf{A}_{k:n,:} - \mathbf{u} \times \mathbf{v}^T$
9) $\mathbf{v} = \mathbf{Q}_{:,k:n} \times \mathbf{u}$
   a) $d\mathbf{v} = d\mathbf{Q}_{:,k:n} \times \mathbf{u} + \mathbf{Q}_{:,k:n} \times d\mathbf{u}$
10) $\mathbf{Q}_{:,k:n} = \mathbf{Q}_{:,k:n} - \mathbf{v} \times \mathbf{u}^T$
   a) $d\mathbf{Q}_{:,k:n} = d\mathbf{Q}_{:,k:n} - d\mathbf{v} \times \mathbf{u}^T - \mathbf{v} \times d\mathbf{u}^T$
11) end for k
   a) Reduce **A** to Hessenberg form.
12) $\mathbf{Z} = \mathbf{I}$
   a) $d\mathbf{Z} = 0$
13) for $k = 1$ to $n-2$
14) for $j = n-1$ to $k+1$ by -1
15) $[\mathbf{A}_{j,k},\ \mathbf{A}_{j+1,k},\ c,\ s,\ d\mathbf{A}_{j,k},\ d\mathbf{A}_{j+1,k},\ dc,\ ds] = d\text{rotgen}(\mathbf{A}_{j,k},\ \mathbf{A}_{j+1,k},\ d\mathbf{A}_{j,k},\ d\mathbf{A}_{j+1,k})$
16) $[\mathbf{A}_{j,k+1:n},\ \mathbf{A}_{j+1,k+1:n},\ d\mathbf{A}_{j,k+1:n},\ d\mathbf{A}_{j+1,k+1:n}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{A}_{j,k+1:n},\ \mathbf{A}_{j+1,k+1:n},\ d\mathbf{A}_{j,k+1:n},\ d\mathbf{A}_{j+1,k+1:n})$
17) $[\mathbf{B}_{j,j:n},\ \mathbf{B}_{j+1,j:n},\ d\mathbf{B}_{j,j:n},\ d\mathbf{B}_{j+1,j:n}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{B}_{j,j:n},\ \mathbf{B}_{j+1,j:n},\ d\mathbf{B}_{j,j:n},\ d\mathbf{B}_{j+1,j:n})$
18) $[\mathbf{Q}_{:,j},\ \mathbf{Q}_{:,j+1},\ d\mathbf{Q}_{:,j},\ d\mathbf{Q}_{:,j+1}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{Q}_{:,j},\ \mathbf{Q}_{:,j+1},\ d\mathbf{Q}_{:,j},\ d\mathbf{Q}_{:,j+1})$
19) $[\mathbf{B}_{j+1,j+1},\ \mathbf{B}_{j+1,j},\ c,\ s,\ d\mathbf{B}_{j+1,j+1},\ d\mathbf{B}_{j+1,j},\ dc,\ ds] = d\text{rotgen}(\mathbf{B}_{j+1,j+1},\ \mathbf{B}_{j+1,j},\ d\mathbf{B}_{j+1,j+1},\ d\mathbf{B}_{j+1,j})$
20) $[\mathbf{B}_{1:j,j+1},\ \mathbf{B}_{1:j,j},\ d\mathbf{B}_{1:j,j+1},\ d\mathbf{B}_{1:j,j}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{B}_{1:j,j+1},\ \mathbf{B}_{1:j,j},\ d\mathbf{B}_{1:j,j+1},\ d\mathbf{B}_{1:j,j})$
21) $[\mathbf{A}_{:,j+1},\ \mathbf{A}_{:,j},\ d\mathbf{A}_{:,j+1},\ d\mathbf{A}_{:,j}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{A}_{:,j+1},\ \mathbf{A}_{:,j},\ d\mathbf{A}_{:,j+1},\ d\mathbf{A}_{:,j})$
22) $[\mathbf{A}_{:,j+1},\ \mathbf{Z}_{:,j},\ d\mathbf{Z}_{:,j+1},\ d\mathbf{Z}_{:,j}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{A}_{:,j+1},\ \mathbf{Z}_{:,j},\ d\mathbf{Z}_{:,j+1},\ d\mathbf{Z}_{:,j})$
23) end for j
24) end for k
25) end dHessTriForm

Algorithm A3 QZ sweep with first derivative

1) $[\mathbf{A},\ \mathbf{B},\ \mathbf{Q},\ \mathbf{Z},\ d\mathbf{A},\ d\mathbf{B},\ d\mathbf{Q},\ d\mathbf{Z}] = d\text{qz2step}(\mathbf{A},\ \mathbf{B},\ \mathbf{Q},\ \mathbf{Z},\ d\mathbf{A},\ d\mathbf{B},\ d\mathbf{Q},\ d\mathbf{Z},\ i1,\ i2)$
   a) $[\mathbf{u},\ d\mathbf{u}] = d\text{moler41}(,\ d\mathbf{A},\ d\mathbf{B})$
   b) $[\mathbf{u},\ \nu,\ d\mathbf{u},\ d\nu] = d\text{housegen}(\mathbf{u},\ d\mathbf{u})$
2) for$k = i1$ to $i2-2$
3) if $(k \neq i1)$
4) $[\mathbf{u},\ \nu,\ d\mathbf{u},\ d\nu] = d\text{housegen}(\mathbf{A}_{k:k+2,k-1},\ d\mathbf{A}_{k:k+2,k-1})$
5) $\mathbf{A}_{k,k-1} = \nu$; $\mathbf{A}_{k+1:k+2,k-1} = 0$
   a) $d\mathbf{A}_{k+1:k+2,k-1} = 0$
6) end if
7) $\mathbf{v}^T = \mathbf{u}^T \times \mathbf{A}_{k:k+2,k:n}$; $d\mathbf{v}^T = d\mathbf{u}^T \times \mathbf{A}_{k:k+2,k:n} + \mathbf{u}^T d\mathbf{A}_{k:k+2,k:n}$; $\mathbf{A}_{k:k+2,k:n} = \mathbf{A}_{k:k+2,k:n} - \mathbf{u}\mathbf{v}^T$; $d\mathbf{A}_{k:k+2,k:n} = d\mathbf{A}_{k:k+2,k:n} - d\mathbf{u}\mathbf{v}^T - \mathbf{u}d\mathbf{v}^T$
8) $\mathbf{v}^T = \mathbf{u}^T \times \mathbf{B}_{k:k+2,k:n}$; $d\mathbf{v}^T = d\mathbf{u}^T \times \mathbf{B}_{k:k+2,k:n} + \mathbf{u}^T \times d\mathbf{B}_{k:k+2,k:n}$; $\mathbf{B}_{k:k+2,k:n} = \mathbf{B}_{k:k+2,k:n} - \mathbf{u}\mathbf{v}^T$; $d\mathbf{B}_{k:k+2,k:n} = d\mathbf{B}_{k:k+2,k:n} - d\mathbf{u}\mathbf{v}^T - \mathbf{u}d\mathbf{v}^T$
9) $\mathbf{v} = \mathbf{Q}_{:,k:k+2} \times \mathbf{u}$; $d\mathbf{v} = d\mathbf{Q}_{:,k:k+2} \times \mathbf{u} + \mathbf{Q} \times d\mathbf{u}$; $\mathbf{Q}_{:,k:k+2} = \mathbf{Q}_{:,k:k+2} - \mathbf{v}\mathbf{u}^T$; $d\mathbf{Q}_{:,k:k+2} = d\mathbf{Q}_{:,k:k+2} - d\mathbf{v}\mathbf{u}^T - \mathbf{v}d\mathbf{u}^T$
10) $[\mathbf{u},\ \nu,\ d\mathbf{u},\ d\nu] = d\text{housegen}(\mathbf{B}_{k+2,k:k+2} \times \overline{\mathbf{I}},\ d\mathbf{B}_{k+2,k:k+2} \times \overline{\mathbf{I}})$
11) $\mathbf{B}_{k+2,k+2} = \nu$; $d\mathbf{B}_{k+2,k+2} = d\nu$; $\mathbf{B}_{k+2,k:k+1} = 0$; $d\mathbf{B}_{k+2,k:k+1} = 0$
12) $\mathbf{u}^T = \mathbf{u}^T \times \overline{\mathbf{I}}$
   a) $d\mathbf{u}^T = d\mathbf{u}^T \times \overline{\mathbf{I}}$
13) $\mathbf{v} = \mathbf{B}_{1:k+1,k:k+2} \times \mathbf{u}$; $d\mathbf{v} = d\mathbf{B}_{1:k+1,k:k+2} \times \mathbf{u} + \mathbf{B}_{1:k+1,k:k+2} \times d\mathbf{u}$; $\mathbf{B}_{1:k+1,k:k+2} = \mathbf{B}_{1:k+1,k:k+2} - \mathbf{v}\mathbf{u}^T$; $d\mathbf{B}_{1:k+1,k:k+2} = d\mathbf{B}_{1:k+1,k:k+2} - d\mathbf{v}\mathbf{u}^T - \mathbf{v}d\mathbf{u}^T$
14) $kk = \min(k+3, i2)$
15) $\mathbf{v} = \mathbf{A}_{1:kk,k:k+2} \times \mathbf{u}$; $d\mathbf{v} = d\mathbf{A}_{1:kk,k:k+2} * \mathbf{u} + \mathbf{A}_{1:kk,k:k+2} \times d\mathbf{u}$; $\mathbf{A}_{1:kk,k:k+2} = \mathbf{A}_{1:kk,k:k+2} - \mathbf{v}\mathbf{u}^T$; $d\mathbf{A}_{1:kk,k:k+2} = d\mathbf{A}_{1:kk,k:k+2} - d\mathbf{v}\mathbf{u}^T - \mathbf{v}d\mathbf{u}^T$
16) $\mathbf{v} = \mathbf{Z}_{:,k:k+2} \times \mathbf{u}$; $d\mathbf{v} = d\mathbf{Z}_{:,k:k+2} \times \mathbf{u} + \mathbf{Z}_{:,k:k+2} \times d\mathbf{u}$ $\mathbf{Z}_{:,k:k+2} = \mathbf{Z}_{:,k:k+2} - \mathbf{v}\mathbf{u}^T$; $d\mathbf{Z}_{:,k:k+2} = d\mathbf{Z}_{:,k:k+2} - d\mathbf{v}\mathbf{u}^T - \mathbf{v}d\mathbf{u}^T$
17) $[\mathbf{B}_{k+1,k+1},\ \mathbf{B}_{k+1,k},\ c,\ s,\ d\mathbf{B}_{k+1,k+1},\ d\mathbf{B}_{k+1,k},\ dc,\ ds] = d\text{rotgen}(\mathbf{B}_{k+1,k+1},\ \mathbf{B}_{k+1,k},\ d\mathbf{B}_{k+1,k+1},\ d\mathbf{B}_{k+1,k})$
18) $[\mathbf{B}_{1:k,k+1},\ \mathbf{B}_{1:k,k},\ d\mathbf{B}_{1:k,k+1},\ d\mathbf{B}_{1:k,k}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{B}_{1:k,k+1},\ \mathbf{B}_{1:k,k},\ d\mathbf{B}_{1:k,k+1},\ d\mathbf{B}_{1:k,k})$
19) $[\mathbf{A}_{1:kk,k+1},\ \mathbf{A}_{1:kk,k},\ d\mathbf{A}_{1:kk,k+1},\ d\mathbf{A}_{1:kk,k}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{A}_{1:kk,k+1},\ \mathbf{A}_{1:kk,k},\ d\mathbf{A}_{1:kk,k+1},\ d\mathbf{A}_{1:kk,k})$
20) $[\mathbf{Z}_{:,k+1},\ \mathbf{Z}_{:,k},\ d\mathbf{Z}_{:,k+1},\ d\mathbf{Z}_{:,k}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{Z}_{:,k+1},\ \mathbf{Z}_{:,k},\ d\mathbf{Z}_{:,k+1},\ d\mathbf{Z}_{:,k})$
21) end for k
22) $[\mathbf{A}_{i2-1,i2-2},\ \mathbf{A}_{i2,i2-2},\ c,\ s,\ d\mathbf{A}_{i2-1,i2-2},\ d\mathbf{A}_{i2,i2-2},\ dc,\ ds] = d\text{rotgen}(\mathbf{A}_{i2-1,i2-2},\ \mathbf{A}_{i2,i2-2},\ d\mathbf{A}_{i2-1,i2-2},\ d\mathbf{A}_{i2,i2-2})$
23) $[\mathbf{A}_{i2-1,i2-1:i2},\ \mathbf{A}_{i2,i2-1:i2},\ d\mathbf{A}_{i2-1,i2-1:i2},\ d\mathbf{A}_{i2,i2-1:i2}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{A}_{i2-1,i2-1:i2},\ \mathbf{A}_{i2,i2-1:i2},\ d\mathbf{A}_{i2-1,i2-1:i2},\ d\mathbf{A}_{i2,i2-1:i2})$
24) $[\mathbf{B}_{i2-1,k2-1:i2},\ \mathbf{B}_{i2,i2-1:i2},\ d\mathbf{B}_{i2-1,k2-1:i2},\ d\mathbf{B}_{i2,i2-1:i2}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{B}_{i2-1,k2-1:i2},\ \mathbf{B}_{i2,i2-1:i2},\ d\mathbf{B}_{i2-1,k2-1:i2},\ d\mathbf{B}_{i2,i2-1:i2})$
25) $[\mathbf{Q}_{:,i2-1},\ \mathbf{Q}_{:,i2},\ d\mathbf{Q}_{:,i2-1},\ d\mathbf{Q}_{:,i2}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{Q}_{:,i2-1},\ \mathbf{Q}_{:,i2},\ d\mathbf{Q}_{:,i2-1},\ d\mathbf{Q}_{:,i2})$
26) $[\mathbf{B}_{i2,i2},\ \mathbf{B}_{i2,i2-1},\ c,\ s,\ d\mathbf{B}_{i2,i2},\ d\mathbf{B}_{i2,i2-1},\ dc,\ ds] = d\text{rotgen}(\mathbf{B}_{i2,i2},\ \mathbf{B}_{i2,i2-1},\ d\mathbf{B}_{i2,i2},\ d\mathbf{B}_{i2,i2-1})$
27) $[\mathbf{B}_{1:i2-1,i2},\ \mathbf{B}_{1:i2-1,i2-1},\ d\mathbf{B}_{1:i2-1,i2},\ d\mathbf{B}_{1:i2-1,i2-1}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{B}_{1:i2-1,i2},\ \mathbf{B}_{1:i2-1,i2-1},\ d\mathbf{B}_{1:i2-1,i2},\ d\mathbf{B}_{1:i2-1,i2-1})$
28) $[\mathbf{A}_{1:i2,i2},\ \mathbf{A}_{1:i2,i2-1},\ d\mathbf{A}_{1:i2,i2},\ d\mathbf{A}_{1:i2,i2-1}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{A}_{1:i2,i2},\ \mathbf{A}_{1:i2,i2-1},\ d\mathbf{A}_{1:i2,i2},\ d\mathbf{A}_{1:i2,i2-1})$
29) $[\mathbf{Z}_{:,i2},\ \mathbf{Z}_{:,i2-1},\ d\mathbf{Z}_{:,i2},\ d\mathbf{Z}_{:,i2-1}] = d\text{rotapp}(c,\ s,\ dc,\ ds,\ \mathbf{Z}_{:,i2},\ \mathbf{Z}_{:,i2-1},\ d\mathbf{Z}_{:,i2},\ d\mathbf{Z}_{:,i2-1})$
30) end $d$qz2step

Algorithm A4 Deflate zeros with first derivatives

1) $[\mathbf{A}, \mathbf{B}, \mathbf{Q}, \mathbf{Z}, d\mathbf{A}, d\mathbf{B}, d\mathbf{Q}, d\mathbf{Z}] = \text{deflate}(\mathbf{A}, \mathbf{B}, \mathbf{Q}, \mathbf{Z}, d\mathbf{A}, d\mathbf{B}, d\mathbf{Q}, d\mathbf{Z}, k, n)$
2) for $j = k$ to $n-1$
3) $[t_1, t_2, c, s, dt_1, dt_2, dc, ds] = d\text{rotgen}(\mathbf{B}_{j,j+1}, \mathbf{B}_{j+1,j+1}, d\mathbf{B}_{j,j+1}, d\mathbf{B}_{j+1,j+1})$
4) $[\mathbf{A}_{j,:}, \mathbf{A}_{j+1,:}, d\mathbf{A}_{j,:}, d\mathbf{A}_{j+1,:}] = d\text{rotapp}(c, s, dc, ds, \mathbf{A}_{j,:}, \mathbf{A}_{j+1,:}, d\mathbf{A}_{j,:}, d\mathbf{A}_{j+1,:})$
5) $[\mathbf{B}_{j,:}, \mathbf{B}_{j+1,:}, d\mathbf{B}_{j,:}, d\mathbf{B}_{j+1,:}] = d\text{rotapp}(c, s, dc, ds, \mathbf{B}_{j,:}, \mathbf{B}_{j+1,:}, d\mathbf{B}_{j,:}, d\mathbf{B}_{j+1,:})$
6) $[\mathbf{Q}_{:,j}, \mathbf{Q}_{:,j+1}, d\mathbf{Q}_{:,j}, d\mathbf{Q}_{:,j+1}] = d\text{rotapp}(c, s, dc, ds, \mathbf{Q}_{:,j}, \mathbf{Q}_{:,j+1}, d\mathbf{Q}_{:,j}, d\mathbf{Q}_{:,j+1})$
7) $[t_1, t_2, c, s, dt_1, dt_2, dc, ds] = d\text{rotgen}(\mathbf{A}_{j+1,j}, \mathbf{A}_{j+1,j-1}, d\mathbf{A}_{j+1,j}, d\mathbf{A}_{j+1,j-1})$
8) $[\mathbf{B}_{:,j}, \mathbf{B}_{:,j-1}, d\mathbf{B}_{:,j}, d\mathbf{B}_{:,j-1}] = d\text{rotapp}(c, s, dc, ds, \mathbf{B}_{:,j}, \mathbf{B}_{:,j-1}, d\mathbf{B}_{:,j}, d\mathbf{B}_{:,j-1})$
9) $[\mathbf{A}_{:,j}, \mathbf{A}_{:,j-1}, d\mathbf{A}_{:,j}, d\mathbf{A}_{:,j-1}] = d\text{rotapp}(c, s, dc, ds, \mathbf{A}_{:,j}, \mathbf{A}_{:,j-1}, d\mathbf{A}_{:,j}, d\mathbf{A}_{:,j-1})$
10) $[\mathbf{Z}_{:,j}, \mathbf{Z}_{:,j-1}, d\mathbf{Z}_{:,j}, d\mathbf{Z}_{:,j-1}] = d\text{rotapp}(c, s, dc, ds, \mathbf{Z}_{:,j}, \mathbf{Z}_{:,j-1}, d\mathbf{Z}_{:,j}, d\mathbf{Z}_{:,j-1})$
11) end for j
12) $j = n$
13) $[t_1, t_2, c, s, dt_1, dt_2, dc, ds] = d\text{rotgen}(\mathbf{A}_{j,j}, \mathbf{A}_{j,j-1}, d\mathbf{A}_{j,j}, d\mathbf{A}_{j,j-1})$
14) $[\mathbf{B}_{:,j}, \mathbf{B}_{:,j-1}, d\mathbf{B}_{:,j}, d\mathbf{B}_{:,j-1}] = d\text{rotapp}(c, s, dc, ds, \mathbf{B}_{:,j}, \mathbf{B}_{:,j-1}, d\mathbf{B}_{:,j}, d\mathbf{B}_{:,j-1})$
15) $[\mathbf{A}_{:,j}, \mathbf{A}_{:,j-1}, d\mathbf{A}_{:,j}, d\mathbf{A}_{:,j-1}] = d\text{rotapp}(c, s, dc, ds, \mathbf{A}_{:,j}, \mathbf{A}_{:,j-1}, d\mathbf{A}_{:,j}, d\mathbf{A}_{:,j-1})$
16) $[\mathbf{Z}_{:,j}, \mathbf{Z}_{:,j-1}, d\mathbf{Z}_{:,j}, d\mathbf{Z}_{:,j-1}] = d\text{rotapp}(c, s, dc, ds, \mathbf{Z}_{:,j}, \mathbf{Z}_{:,j-1}, d\mathbf{Z}_{:,j}, d\mathbf{Z}_{:,j-1})$
17) end deflate

Algorithm A3 is the full algorithm for executing a QZ sweep with derivatives:

Algorithm A4 will deflate zeros along the diagonal of $\mathbf{B}$ out of the bottom of the pencil while preserving the correct derivative information, given that $\mathbf{B}_{k,k} = 0$ and $\mathbf{B}$ is n by n.

Appendix B: Residue Batch Calculation Matrix Unpacked

Here is the expanded matrix form of the equations for finding the residues of a system with distinct poles.

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \sum_{k=2}^{n} -p_k & \sum_{\substack{k=1 \\ k\neq 2}}^{n} -p_k & \cdots & \sum_{k=1}^{n-1} -p_k \\ \sum_{\substack{k,l=\binom{n-1}{2} \\ k,l\neq 1}} (-p_k)(-p_l) & \sum_{\substack{k,l=\binom{n-1}{2} \\ k,l\neq 2}} (-p_k)(-p_l) & \cdots & \sum_{\substack{k,l=\binom{n-1}{2} \\ k,l\neq n}} (-p_k)(-p_l) \\ \vdots & \vdots & & \vdots \\ \prod_{k=2}^{n}(-p_k) & \prod_{\substack{k=1 \\ k\neq 2}}^{n}(-p_k) & \cdots & \prod_{k=1}^{n-1}(-p_k) \end{bmatrix} \begin{Bmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{Bmatrix} = \begin{Bmatrix} \eta_1 \\ \eta_2 \\ \vdots \\ \eta n \end{Bmatrix} \tag{B1}$$

Here is the expanded portion of the $\boldsymbol{\Xi}$ matrix corresponding to a pole of multiplicity m_k:.

$$\begin{bmatrix} 0 & 0 & \cdots & 1 & \cdots \\ \vdots & \vdots & \cdots & \sum_{k=1}^{n-1}(-p_k) - (m_k - 1)p_1 & \\ 0 & 1 & \cdots & \sum_{\substack{k,l=\binom{n-1}{2} \\ k,l\neq n}} (-p_k)(-p_l) & \\ 1 & \sum_{\substack{k=1 \\ k\neq 2}}^{n} -p_k - p_1 & \cdots & \sum_{\substack{k,l,m=\binom{n-1}{3} \\ k,l\neq n}} (-p_k)(-p_l)(-p_m) & \\ \sum_{k=2}^{n} -p_k & \sum_{\substack{k,l=\binom{n-m_k+1}{2} \\ k,l\neq 2}} (-p_k)(-p_l) & \cdots & \vdots & \cdots \\ \sum_{\substack{k,l=\binom{n-1}{2} \\ k,l\neq 1}} (-p_k)(-p_l) & \sum_{\substack{k,l,m=\binom{n-m_k+1}{3} \\ k,l\neq 1}} (-p_k)(-p_l)(-p_m) & \cdots & \vdots & \\ \vdots & \vdots & & \vdots & \\ \prod_{k=2}^{n}(-p_k) & p_1 \prod_{k=2}^{n}(-p_k) & \cdots & p_1^{m_k-1} \prod_{k=2}^{n}(-p_k) & \end{bmatrix} \tag{B2}$$

Here is the expanded $\partial\boldsymbol{\Xi}$ matrix:

$$\begin{bmatrix} 0 & 0 & \cdots & 0 \\ \sum_{i=2}^{n} -\frac{\partial p_i}{\partial \psi} & \sum_{\substack{i=1\\ i\neq 2}}^{n} -\frac{\partial p_i}{\partial \psi} & \cdots & \sum_{i=1}^{n-1} -\frac{\partial p_i}{\partial \psi} \\ \vdots & \vdots & & \vdots \\ \sum_{m=2}^{n}\left(-\frac{\partial p_m}{\partial \psi}\right) \prod_{\substack{s=\binom{n-2}{i-2}\\ s\neq 1}} (-p_s) & \sum_{\substack{m=1\\ m\neq 2}}^{n}\left(-\frac{\partial p_m}{\partial \psi}\right) \prod_{\substack{s=\binom{n-2}{i-2}\\ s\neq 2}} (-p_s) & \cdots & \sum_{m=1}^{n-1}\left(-\frac{\partial p_m}{\partial \psi}\right) \prod_{\substack{s=\binom{n-2}{i-2}\\ s\neq n}} (-p_s) \\ \vdots & \vdots & & \vdots \\ \sum_{m=2}^{n}\left(-\frac{\partial p_m}{\partial \psi}\right)\prod_{\substack{s=2\\ s\neq m}}(-p_s) & \sum_{\substack{m=1\\ m\neq 2}}^{n}\left(-\frac{\partial p_m}{\partial \psi}\right)\prod_{\substack{s=1\\ s\neq 2\\ s\neq m}}^{n}(-p_s) & \cdots & \sum_{m=1}^{n-1}\left(-\frac{\partial p_m}{\partial \psi}\right)\prod_{\substack{s=1\\ s\neq m}}^{n-1}(-p_s) \end{bmatrix} \tag{B3}$$

Appendix C: Pseudorandom Examples

The following examples use a matrix pencil and derivatives that were generated in a pseudorandom fashion to demonstrate that the derivatives are calculated accurately for arbitrary pencils and pencil derivatives. To demonstrate calculation of the generalized eigenvalue derivatives, take the pencil to be

$$\mathbf{A} = \begin{bmatrix} 0.9501 & 0.8913 & 0.8214 & 0.9218 \\ 0.2311 & 0.7621 & 0.4447 & 0.7382 \\ 0.6068 & 0.4565 & 0.6154 & 0.1763 \\ 0.4860 & 0.0185 & 0.7919 & 0.4057 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0.9355 & 0.0579 & 0.1389 & 0.2722 \\ 0.9169 & 0.3529 & 0.2028 & 0.1988 \\ 0.4103 & 0.8132 & 0.1987 & 0.0153 \\ 0.8936 & 0.0099 & 0.6038 & 0.7468 \end{bmatrix}$$

$$\frac{\partial \mathbf{A}}{\partial \psi} = \begin{bmatrix} 0.4451 & 0.8462 & 0.8381 & 0.8318 \\ 0.9318 & 0.5252 & 0.0196 & 0.5028 \\ 0.4660 & 0.2026 & 0.6813 & 0.7095 \\ 0.4186 & 0.6721 & 0.3795 & 0.4289 \end{bmatrix}$$

$$\frac{\partial \mathbf{B}}{\partial \psi} = \begin{bmatrix} 0.3046 & 0.3028 & 0.3784 & 0.4966 \\ 0.1897 & 0.5417 & 0.8600 & 0.8998 \\ 0.1934 & 0.1509 & 0.8537 & 0.8216 \\ 0.6822 & 0.6979 & 0.5936 & 0.6449 \end{bmatrix}$$

Results of this test are shown in Table C1.

A second example demonstrates calculation of the transmission zeros with randomly selected $\mathbf{A}$ and $\partial\mathbf{A}/\partial\psi$:

$$\mathbf{A} = \begin{bmatrix} 0.65510 & 0.58527 & 0.89090 & 0.84072 & 0.19660 & 0.58526 \\ 0.16261 & 0.22381 & 0.95929 & 0.25428 & 0.25108 & 0.54972 \\ 0.11900 & 0.75127 & 0.54722 & 0.81428 & 0.61604 & 0.91719 \\ 0.49836 & 0.25510 & 0.13862 & 0.24352 & 0.47329 & 0.28584 \\ 0.95974 & 0.50596 & 0.14929 & 0.92926 & 0.35166 & 0.75720 \\ 0.34039 & 0.69908 & 0.25751 & 0.34998 & 0.83083 & 0.75373 \end{bmatrix}$$

Table C1 Pole derivative results and randomly generated pencil

Real(pole)	Imag(pole)	Real(derivative)	Real(FD)	Imag(derivative)	Imag(FD)
−10.4206	0	−32.8206	−32.8250	0	0
0.3762	0	−0.3834	−0.3835	0	0
0.2884	$\mp 1.3502i$	−0.4257	−0.4255	∓ 2.9924	∓ 2.9920

Table C2 Transmission zero results, randomly generated pencil

Zero	tzero(.)	Derivative	Finite difference
		Complex Pair	
0.20412	0.20412	0.69642	0.69641
$\pm 0.82248i$	$\pm 0.82248i$	$\pm 0.039508i$	$\pm 0.039464i$
		Real Zeros	
−0.37431	−0.37431	0.14779	0.14775
0.26469	0.26469	0.15942	0.15948

$$\frac{\partial \mathbf{A}}{\partial \psi} = \begin{bmatrix} 0.38045 & 0.53080 & 0.56882 & 0.16218 & 0 & 0 \\ 0.56782 & 0.77917 & 0.46939 & 0.79428 & 0 & 0 \\ 0.075854 & 0.93401 & 0.011902 & 0.31122 & 0 & 0 \\ 0.05395 & 0.12991 & 0.33712 & 0.52853 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$\mathbf{B}$ is conformably partitioned with a 4×4 identity matrix in the upper right corner and zeros elsewhere. For the transmission zero derivatives, $\partial\mathbf{B}/\partial\psi$ is set to zero. The results of this test are shown in Table C2.

References

[1] Van Dooren, P., "A Generalized Eigenvalue Approach for Solving Riccati Equations," *SIAM Journal on Scientific and Statistical Computing*, Vol. 2, No. 2, June 1981, pp. 121–135. doi:10.1137/0902010

[2] Van Dooren, P. M., "The Generalized Eigenstructure Problem in Linear System Theory," *IEEE Transactions on Automatic Control*, Vol 26, No. 1, Feb. 1981, pp. 111–129. doi:10.1109/TAC.1981.1102559

[3] Emami-Naeini, A., and Van Dooren, P., "Computation of Zeros of Linear Multivariable Systems," *Automatica*, Vol. 18, No. 4, 1982, pp. 415–430. doi:10.1016/0005-1098(82)90070-X

[4] Murthy, D. V., and Haftka, R. T., "Derivatives of Eigenvalues and Eigenvectors of a General Complex Matrix," *International Journal for Numerical Methods in Engineering*, Vol. 26, No. 2, 1988, pp. 293–311. doi:10.1002/nme.1620260202

[5] Rogers, L. C., "Derivatives of Eigenvalues and Eigenvectors," *AIAA Journal*, Vol. 8, No. 5, 1970, pp. 943–944. doi:10.2514/3.5795

[6] Garg, S., "Derivatives of Eigensolutions for a General Matrix," *AIAA Journal*, Vol. 11, No. 8, 1973, pp. 1191–1194. doi:10.2514/3.6892

[7] Rudisill, C. S., and Chu, Y., "Numerical Methods for Evaluating the Derivatives of Eigenvalues and Eigenvectors," *AIAA Journal*, Vol. 13, No. 6, 1975, pp. 834–837.
doi:10.2514/3.60449

[8] Jankovic, M. S., "Exact *n*th Derivatives of Eigenvalues and Eigenvectors," *Journal of Guidance, Control, and Dynamics*, Vol. 17, No. 1, 1994, pp. 136–144.
doi:10.2514/3.21170

[9] Nelson, R. B., "Simplified Calculation of Eigenvector Derivatives," *AIAA Journal*, Vol. 14, No. 9, 1976, pp. 1201–1205.
doi:10.2514/3.7211

[10] Lim, K. B., Juang, J., and Ghaemmaghami, P., "Eigenvector Derivatives of Repeated Eigenvalues Using Singular Value Decomposition," *Journal of Guidance, Control, and Dynamics*, Vol. 12, No. 2, 1989, pp. 282–283.
doi:10.2514/3.20405

[11] Zhang, D., and Wei, F., "Computation of Eigenvector Derivatives with Repeated Eigenvalues Using a Complete Modal Space," *AIAA Journal*, Vol. 33, No. 9, 1995, pp. 1749–1753.
doi:10.2514/3.12723

[12] Song, D., Han, W., Chen, S., and Qiu, Z., "Simplified Calculation of Eigenvector Derivatives with Repeated Eigenvalues," *AIAA Journal*, Vol. 34, No. 4, 1996, pp. 859–862.
doi:10.2514/3.13156

[13] Chen, T., "Eigenvector Derivatives for Doubly Repeated Eigenvalues," *AIAA Journal*, Vol. 34, No. 7, 1996, pp. 1531–1533.
doi:10.2514/3.13263

[14] Wei, F., and Zhang, D., "Eigenvector Derivatives with Repeated Eigenvalues Using Generalized Inverse Technique," *AIAA Journal*, Vol. 34, No. 10, 1996, pp. 2206–2209.
doi:10.2514/3.13379

[15] Friswell, M. I., "The Derivatives of Repeated Eigenvalues and Their Associated Eigenvectors," *Journal of Vibration and Acoustics*, Vol. 118, No. 3, 1996, pp. 390–397.
doi:10.1115/1.2888195

[16] Prells, U., and Friswell, M. I., "Partial Derivatives of Repeated Eigenvalues and Their Eigenvalues," *AIAA Journal*, Vol. 35, No. 8, 1997, pp. 1363–1368.
doi:10.2514/2.245

[17] Friswell, M. I., and Adhikari, S., "Derivatives of Complex Eigenvectors Using Nelson's Method," *AIAA Journal*, Vol. 38, No. 12, 2000, pp. 2355–2357.
doi:10.2514/2.907

[18] Zhang, Z.-Y., and Zhang, H.-S., "Eigensensitivity Analysis of a Defective Matrix," *AIAA Journal*, Vol. 39, No. 3, 2001, pp. 473–479.
doi:10.2514/2.1329

[19] Zhang, Z.-Y., and Zhang, H.-S., "Higher-Order Eigensensitivity Analysis of a Defective Matrix, *AIAA Journal*, Vol. 40, No. 4, 2002, pp. 751–757.
doi:10.2514/2.1708

[20] Zhang, Z.-Y., and Zhang, H.-S., "Eigensensitivity Analysis of a Defective Matrix with Zero First-Order Eigenvalue Derivatives," *AIAA Journal*, Vol. 42, No. 1, 2004, pp. 114–123.
doi:10.2514/1.1915

[21] Zhang, Z.-Y., and Zhang, H.-S., "Calculation of Eigenvalue and Eigenvector Derivatives of a Defective Matrix," *Applied Mathematics and Computation*, Vol. 176, No. 1, 2006, pp. 7–26.
doi:10.1016/j.amc.2005.09.061

[22] Moler, C. B., and Stewart, G. W., "An Algorithm for Generalized Matrix Eigenvalue Problems," *SIAM Journal on Numerical Analysis*, Vol. 10, No. 2, 1973, pp. 241–256.
doi:10.1137/0710024

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

[24] Stewart, G. W., *Matrix Algorithms, Volume II: Eigensystems*, Society for Industrial and Applied Mathematics, Philadelphia, 1998.

[25] Kaufman, L., "Some Thoughts on the QZ Algorithm for Solving the Generalized Eigenvalue Problem," *ACM Transactions on Mathematical Software*, Vol. 3, No. 1, 1977, pp. 65–75.
doi:10.1145/355719.355725

[26] Burchett, B. T., and Costello, M., "QR-Based Algorithm for Eigenvalue Derivatives," *AIAA Journal*, Vol. 40, No. 11, Nov. 2002, pp. 2319–2322.
doi:10.2514/2.1569

[27] Oppenheim, A. V., and Schafer, R. W., *Digital Signal Processing*, Prentice–Hall, Englewood Cliffs, NJ, 1975.

[28] Ogata, K., *Modern Control Engineering*, 4th ed., Prentice–Hall, Upper Saddle River, NJ, 2002.

J. Wei
Associate Editor