

Deflating Quadratic Matrix Polynomials

Christopher J. Munro

October 2007

MIMS EPrint: **2008.55**

Manchester Institute for Mathematical Sciences
School of Mathematics

The University of Manchester

Reports available from: <http://www.manchester.ac.uk/mims/eprints>

And by contacting: The MIMS Secretary
School of Mathematics
The University of Manchester
Manchester, M13 9PL, UK

ISSN 1749-9097

DEFLATING QUADRATIC MATRIX POLYNOMIALS

A DISSERTATION SUBMITTED TO THE UNIVERSITY OF MANCHESTER
FOR THE DEGREE OF MASTER OF SCIENCE
IN THE FACULTY OF ENGINEERING AND PHYSICAL SCIENCES

October 2007

Christopher J. Munro
School of Mathematics

Contents

Abstract	8
Declaration	9
Copyright Statement	10
Acknowledgements	11
1 Introduction	12
2 Solving QEPs via Linearization	18
2.1 Determining a Linearization	18
2.2 Homogeneous Notation	21
2.3 Solving the Generalized Eigenvalue Problem	22
2.4 Outline of Existing Software	23
3 Backward Error and Conditioning	25
3.1 Backward Error of Eigenpairs of Quadratics and Matrix Pencils	25
3.2 Eigenvalue Condition Numbers for Quadratics and Matrix Pencils	27
3.3 Numerical Stability of the QZ Algorithm in Solving the Quadratic Eigenvalue Problem by Linearization	28
4 Rank-Revealing Factorizations	29
4.1 Null Space, Rank and the Singular Value Decomposition	29
4.2 Rank-Revealing Orthogonal Decompositions	30
4.2.1 UTV Decomposition	31

4.2.2	Rank-Revealing QR Factorization	33
4.2.3	Eigendecomposition	34
4.3	Rank and Null Space of Structured Matrices	35
4.4	Summary of Routines Implemented in MATLAB Code	35
4.5	Computational Cost	36
4.6	Comparison of Techniques	37
5	Quadratic Eigenproblem Algorithms	39
5.1	A Modified Version of Kublanovskaya et al's Original Approach	40
5.2	Simplification of Kublanovskaya's Method	44
5.3	Householder Deflation	46
5.3.1	Deflation of Zero Eigenvalues	48
5.3.2	Deflation of Infinite Eigenvalues	51
5.3.3	Deflation of Zero and Infinite Eigenvalues	52
5.3.4	Householder Deflation Methods and Structured Linearizations	55
5.4	Computational Cost of Methods	55
6	Applications and Test Problems	60
6.1	Test Quadratics	60
6.1.1	Speaker Box	62
6.1.2	Linear Spring and Dashpot	62
6.1.3	Mobile Manipulator Problem	64
6.1.4	Shaft on Bearing Support	65
6.1.5	Railtrack Problem	68
6.1.6	Randomly Generated Quadratics with Singular Leading and Trailing Coefficient Matrices	69
6.2	Initial Scaling of the Quadratic Matrix Polynomial	70
6.3	Conditioning of Eigenvalues of Quadratic Test Problems	71
6.4	Testing the Algorithms	74
6.4.1	Householder Deflation of Zero or Infinite Eigenvalues	75

6.4.2	Deflation of Zero or Infinite Eigenvalues using Simplified Kublanovskaya Method	75
6.4.3	Householder Deflation of Zero and Infinite Eigenvalues	77
6.4.4	Deflation of Zero and Infinite Eigenvalues using Modified original Kublanovskaya Method	79
7	Conclusion	81
A	MATLAB Programs	82
	Bibliography	111

List of Tables

4.1	Subspaces defined by the SVD of the matrix A	32
4.2	Summary of UTV decompositions.	36
4.3	Cost of low-rank ULV and SVD algorithms.	37
4.4	Conditions for <code>lu1v</code> to be faster than comparative algorithms.	37
5.1	Cost of stages of simplified Kublanovskaya approach.	56
5.2	Cost of stages of modified original Kublanovskaya approach.	57
5.3	Cost of stages of Householder approach for zero eigenvalues (for deflating infinite eigenvalues only, use the reversal polynomial and deflate the corresponding zero eigenvalues).	58
5.4	Cost of stages of Householder approach for zero and infinite eigenvalues.	59
6.1	Properties of test problems.	61
6.2	Effect of scaling on norms of test problem coefficients.	61
6.3	Effect of scaling on eigenvalue condition number for quadratic and companion linearization, for test problem quadratics.	72
A.1	Summary of MATLAB programs; $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$ is a general quadratic and $L(\lambda) = \lambda X + Y$ a matrix pencil.	82

List of Figures

6.1	Spyplot of matrix coefficients of quadratic for speaker box example.	62
6.2	Finite element representation of a speaker box.	63
6.3	Spring/dashpot with Maxwell elements.	65
6.4	Three link mobile manipulator.	66
6.5	Spyplot of matrix coefficients of quadratic for shaft example.	66
6.6	Schematic of a shaft on bearing support.	67
6.7	Model of the rail.	68
6.8	FEM representation of the rail in one sleeper bay.	69
6.9	Effect of scaling on eigenvalue condition number of the quadratic for eigenvalues of the shaft problem.	73
6.10	Effect of scaling on eigenvalue condition number of the first companion linearization for eigenvalues of the shaft problem.	73
6.11	Backward error for quadratic and companion linearization for the speaker box problem where $A_i \in \mathbb{R}^{107 \times 107}$ with $\text{rank}(A_0) = 106$ (1 zero eigenvalue contributed by A_0 coefficient deflated via Householder reflectors)	76
6.12	Backward error for quadratic and companion linearization for the mobile manipulator problem where $A_i \in \mathbb{R}^{5 \times 5}$ with $\text{rank}(A_2) = 3$ (2 infinite eigenvalues contributed by A_2 coefficient deflated via simplified Kublanovskaya method	77

6.13	Backward error for quadratic and companion linearization for the shaft problem where $A_i \in \mathbb{R}^{400 \times 400}$ with $\text{rank}(A_2) = 199$ (201 infinite eigenvalues contributed by A_2 coefficient deflated), deflation by simplified Kublanovskaya method	78
6.14	Backward error for quadratic and companion linearization for random quadratic with singular coefficients generated by block outer-product, $n = 100$, $\text{rank}(A_0) = \text{rank}(A_2) = 50$, 50 zero and infinite eigenvalues deflated by Householder reflectors.	80
6.15	Backward error for quadratic and companion linearization for random quadratic with singular coefficients generated by block outer-product, $n = 100$, $\text{rank}(A_0) = \text{rank}(A_2) = 50$, 50 zero and infinite eigenvalues deflated using modified original Kublanovskaya method	80

The University of Manchester

Christopher J. Munro

Master of Science

Deflating Quadratic Matrix Polynomials

October 2007

In this thesis we consider algorithms for solving the quadratic eigenvalue problem, $(\lambda^2 A_2 + \lambda A_1 + A_0)x = 0$ when the leading or trailing coefficient matrices are singular. In a finite element discretization this corresponds to the mass or stiffness matrices being singular and reflects modes of vibration (or eigenvalues) at zero or “infinity”. We are interested in deflation procedures that enable us to utilize knowledge of the presence of these (or any) eigenvalues to reduce the overall cost in computing the remaining eigenvalues and eigenvectors of interest. We first give an introduction to the quadratic eigenvalue problem and explain how it can be solved by a process called linearization.

We present two types of algorithms, firstly a modification of an algorithm published by Kublanovskaya, Mikhailov, and Khazanov in the 1970s that has recently been translated into English. Using these ideas we present algorithms that are able to reduce the size of the problem by “deflating” infinite and zero eigenvalues that arise when the mass or stiffness matrix (or both) are singular.

Secondly we look at methods that deflate zero and infinite eigenvalues by the use of Householder reflectors; this requires a basis for the null space of the mass or stiffness matrix (or both), so we also summarize various decompositions that can be used to give this information. We consider different applications that yield a quadratic eigenvalue problem with singular leading and trailing coefficients and after testing the implementations of the algorithms on some of these problems we comment on their stability.

Declaration

No portion of the work referred to in this dissertation has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright Statement

- i. Copyright in text of this dissertation rests with the author. Copies (by any process) either in full, or of extracts, may be made **only** in accordance with instructions given by the author. Details may be obtained from the appropriate Graduate Office. This page must form part of any such copies made. Further copies (by any process) of copies made in accordance with such instructions may not be made without the permission (in writing) of the author.
- ii. The ownership of any intellectual property rights which may be described in this dissertation is vested in the University of Manchester, subject to any prior agreement to the contrary, and may not be made available for use by third parties without the written permission of the University, which will prescribe the terms and conditions of any such agreement.
- iii. Further information on the conditions under which disclosures and exploitation may take place is available from the Head of the School of Mathematics.

Acknowledgements

I am very grateful to my supervisor Françoise Tisseur for her guidance and many helpful suggestions during the preparation of this thesis. I acknowledge the use of the MATLAB toolboxes, UTV tools [17], and a draft version of NLEVP: A collection of nonlinear eigenvalue problems [23]. I thank Rüdiger Borsdorf for his comments and suggestions on reading a draft of this work and EPSRC for financial support. I would like to thank my parents for their support during my studies.

Chapter 1

Introduction

We give a brief introduction to the Quadratic Eigenvalue Problem. The *Quadratic Eigenvalue Problem* (QEP) is to find scalar eigenvalues λ , left eigenvectors y and right eigenvectors x such that

$$y^*Q(\lambda) = 0, \quad Q(\lambda)x = 0, \quad x, y \neq 0,$$

where

$$Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0, \quad \text{with } A_i \in \mathbb{C}^{n \times n}, \quad i = 0, 1, 2, \quad \text{and } A_2 \neq 0,$$

and throughout this thesis we will assume that $Q(\lambda)$ is regular, that is $\det Q(\lambda) \neq 0$.

We call $Q(\lambda)$ a quadratic matrix polynomial, quadratic, or a λ -matrix.

Note that we specify $A_2 \neq 0$. If we allow $A_2 = 0$ then we obtain linear eigenvalue problems more generally known as

- the Standard Eigenvalue Problem (SEP) $Ax = \lambda x$, or $(-\lambda I + A)x = 0$ if we take $A_0 = A$, and $A_1 = -I$.
- the Generalized Eigenvalue Problem (GEP) $Ax = \lambda Bx$ or $(-\lambda B + A)x = 0$ on choosing $A_0 = A$ and $A_1 = -B$.

QEPs arise in a variety of applications including dynamic analysis of structures discretized by the finite element method (for example the effects of damping), fluid mechanics and vibro-acoustics (examples are reducing the level of noise of cars and

aircraft). In Chapter 6 we will give further examples of problems yielding QEPs. There are fundamental differences in the quadratic case compared to the SEP and GEP. We now outline these differences and explain some properties of QEPs.

Starting with the standard eigenvalue problem $Ax = \lambda x$, there are n eigenvalues for an n -by- n matrix (possibly repeated) and they are all finite. Direct methods such as the QR algorithm for dense problems (where factorization methods can be employed) are based on applying unitary transformations to convert the matrix A to Schur form and reveal A 's eigenvalues. The matrix $U \in \mathbb{C}^{n \times n}$ is a *unitary* matrix if $U^*U = UU^* = I_n$ (I_n is the n -by- n identity matrix). Given a matrix $A \in \mathbb{C}^{n \times n}$ there exists a unitary matrix U and an upper triangular matrix T such that

$$U^*AU = T.$$

The matrix T is a Schur form of A . (When A is real with $A \in \mathbb{R}^{n \times n}$, we can work using only real arithmetic and U can be taken orthogonal ($U^TU = UU^T = I_n$) and T is then upper quasi-triangular with 1-by-1 and 2-by-2 blocks on the diagonal. An n -by- n matrix F is called upper quasi-triangular if it has the structure

$$\begin{bmatrix} F_{11} & F_{12} & \dots & F_{1n} \\ 0 & F_{22} & \dots & F_{2n} \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & 0 & F_{nn} \end{bmatrix},$$

where the diagonal blocks are either 1-by-1 or 2-by-2.)

On moving from the standard eigenvalue problem to the generalized eigenvalue problem $Ax = \lambda Bx$, we consider the *matrix pencil* $A - \lambda B$. When we let $B = I$ we obtain the SEP, however whereas the identity matrix is nonsingular in the SEP, when we look at the GEP the matrix B can be singular (if B is singular there are some infinite eigenvalues). For an n -by- n matrix pencil there are n eigenvalues that can be finite (including zero) or infinite. We consider infinite eigenvalues as zero eigenvalues of the pencil $-B + \lambda A$. In this thesis we assume all pencils are regular, this eliminates the possibility that the pencil may have an eigenvector for which any

scalar is an eigenvalue. To illustrate this, consider the pencil $A - \lambda B$ with

$$A = B = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

since $Ae_1 = Be_1 = 0$, any scalar λ satisfies $Ax = \lambda Bx = 0$ with $x = e_1$.

For the numerical solution of dense GEPs we use the QZ algorithm which is based on applying unitary transformations to convert the pencil $A - \lambda B$ to generalized Schur form, (it can be considered as an analogue of the QR algorithm used in SEPs).

We now explain the generalized Schur decomposition (based on Stewart [30]). Given a pencil $A - \lambda B$, with $A, B \in \mathbb{C}^{n \times n}$, there exists unitary matrices U and V such that

$$U^*AV = S, \text{ and } U^*BV = T$$

where S and T are upper triangular. (When A and B are real we can work in real arithmetic, then U and V can be taken orthogonal and S is then upper quasi-triangular and T is upper triangular; this is a generalized real Schur form of the pencil.)

The QZ algorithm applied to a regular pencil $X - \lambda Y$, with $X, Y \in \mathbb{R}^{n \times n}$ determines orthogonal matrices Q and Z to convert the pencil to generalized real Schur form. We have that $Q^T X Z = S$ is upper triangular and $Q^T Y Z = T$ is upper quasi-triangular.

Finally, when moving from the GEP to the QEP, we have $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$, for an n -by- n quadratic there are $2n$ eigenvalues λ , that can be finite or infinite. If there are more than n eigenvectors then they do not form a linearly independent set. With regard to the finite and infinite eigenvalues we have that when

- A_2 is singular the λ -matrix $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$ has “infinite” eigenvalues, or that the reversal polynomial $\lambda^2 Q(\lambda^{-1}) = \lambda^2 A_0 + \lambda A_1 + A_2$ has zero eigenvalues.
- A_0 is singular the λ -matrix $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$ has zero eigenvalues, or that the reversal polynomial $\lambda^2 Q(\lambda^{-1}) = \lambda^2 A_0 + \lambda A_1 + A_2$ has “infinite” eigenvalues.

Hence we interpret “infinite” eigenvalues of a quadratic $Q(\lambda)$ as zero eigenvalues of the *reversal polynomial* $\text{rev } Q(\lambda) := \lambda^2 Q(\lambda^{-1})$. If either the leading or trailing coefficient (or both) are singular, then the matrix A_1 may contribute infinite or zero eigenvalues. A major difference between the QEP compared with the SEP and GEP is that there is no analogous Schur form. Hence for dense problems it is not possible to work directly with the quadratic in the form $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$ by applying some extension of the QR and QZ type methods.

Depending on the size of the matrices involved we distinguish between

- factorization methods that work with small to medium size problems, working with the matrices directly, storing the whole matrix for example. For the GEP, the QZ algorithm is one such method.
- iterative methods that work with large (usually sparse) problems where it may not be possible to factorize the matrix, or where the matrix cannot be stored in whole, for example there may be only a function to apply the matrix A , to a vector to form Ax . For the SEP, examples are the Arnoldi and Lanczos algorithms.

In this thesis we work with small to medium size problems where factorization methods can be employed. We consider quadratic matrix polynomials where the leading (A_2) or trailing coefficient (A_0) are singular and thus contribute infinite or zero eigenvalues. We do not solve the QEP by working directly with the quadratic, we work instead with a linear matrix pencil obtained after a process called *linearization*, this consists of transforming $Q(\lambda)$ into a linear matrix pencil of twice the dimension of the quadratic.

We look at methods of computing an eigensolution, that utilizes our knowledge of the presence of infinite and zero eigenvalues. We transform the linear problem $\lambda X + Y$ to one that has no infinite or zero eigenvalues from the leading or trailing coefficient. The process of transforming the problem is called *deflation*, and we say that the transformed problem results after we have *deflated* zero and infinite eigenvalues from the original problem. The deflation procedures should reduce the overall

operation count from computing all eigenvalues of a matrix quadratic. As already mentioned, matrix polynomials with singular leading and trailing coefficients have infinite and zero eigenvalues respectively. Our aim is to derive numerically stable procedures that deflate infinite and zero eigenvalues before computing the rest of the spectrum (the *spectrum* is the set of eigenvalues). With the exception of the MATLAB function `polyeig`, there is currently no black box software for the solution of quadratic (or polynomial) eigenvalue problems; `polyeig` does not currently have a deflation procedure.

Although theoretical results show that the QZ algorithm behaves well in the presence of infinite eigenvalues [32], numerical experiments performed using the LAPACK implementation of the QZ algorithm indicate that infinite eigenvalues are not always computed by the algorithm [25]. We now outline the content of this thesis. We focus on solving the QEP by linearization, we summarize the main details of this method in Chapter 2. In Chapter 3 we outline how the accuracy of the computed eigenpairs of the QEP can be computed, and eigenvalue condition numbers for quadratics and matrix pencils. In Chapter 5 we outline two types of algorithms, the first, based on the work of Kublanovskaya et al requires a UTV factorization of the leading or trailing coefficient matrices, the second, based on applying orthogonal transformations requires a basis for the null space of the leading or trailing coefficient matrices. We outline UTV style decompositions and rank-revealing factorizations in Chapter 4. We present some applications that yield QEPs with singular leading or trailing coefficients in Chapter 6. We also consider a method of scaling the quadratic matrix polynomial and the conditioning of eigenvalues of the test problems presented, finally, we use some of those QEPs as test problems for the algorithms presented, in addition to quadratics with leading and trailing coefficients of specified rank that were randomly generated.

In this work, I_n denotes the n -by- n identity matrix, and we generally adopt the Householder convention with regard to naming variables, thus

- matrices are denoted by capital letters: A

- elements of matrices by lower case letters of the respective matrix: a_{ij}
- vectors are denoted by lower case Latin letters: a, b, c
- scalars are denoted by Greek lower case letters: α, β, γ .

We adopt the MATLAB matrix notation, thus $A(i: j, k: l)$ represents the intersection of rows i to j and columns k to l , while $A(:, k)$ denotes the k th column, the colon means to take all elements in the k th column. For simplicity we present all algorithms working in real arithmetic, with suitable modifications they extend to complex arithmetic. “ T ” denotes transpose, while in complex arithmetic “ $*$ ” denotes conjugate transpose.

Chapter 2

Solving Quadratic Eigenvalue

Problems via Linearization

We now outline a method of solving the QEP; applying the QZ algorithm to a linearization of the original matrix polynomial. In outline, the process of solving the QEP by linearization involves taking the original quadratic polynomial and converting it to a linear matrix pencil $L(\lambda) = \lambda X + Y$. The resulting GEP is then solved; we will use the QZ algorithm. Finally eigenvectors of the QEP must be recovered from those of the GEP.

2.1 Determining a Linearization

We linearize an n -by- n quadratic by determining a matrix pencil of dimension $2n$ -by- $2n$. We then apply the QZ algorithm to a pencil of twice the dimension of the quadratic. The linearization must have the same eigenvalues as the quadratic, we give the definition of a linearization below.

Definition 1 (Linearization, see [14]). *A $2n$ -by- $2n$ pencil $L(\lambda) = A - \lambda B$ is a linearization of an n -by- n quadratic $Q(\lambda)$ if*

$$E(\lambda)L(\lambda)F(\lambda) = \begin{bmatrix} Q(\lambda) & 0 \\ 0 & I_n \end{bmatrix},$$

where $E(\lambda)$ and $F(\lambda)$ are matrix polynomials with constant nonzero determinants (and are said to be unimodular).

In this thesis we will generally take the first companion linearization that for the general quadratic $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$ has the form

$$C_1(\lambda) = \lambda \begin{bmatrix} A_2 & 0 \\ 0 & I_n \end{bmatrix} + \begin{bmatrix} A_1 & A_0 \\ -I_n & 0 \end{bmatrix}. \quad (2.1)$$

The first companion linearization is a companion type linearization, another companion linearization, the second companion linearization is given below,

$$C_2(\lambda) = \lambda \begin{bmatrix} A_2 & 0 \\ 0 & I_n \end{bmatrix} + \begin{bmatrix} A_1 & -I_n \\ A_0 & 0 \end{bmatrix}. \quad (2.2)$$

The companion linearizations have the advantage that they are always linearizations of $Q(\lambda)$. However, companion linearizations do not respect structural properties of the quadratic. When the quadratic exhibits structural properties such as symmetry it is desirable to take a structure preserving linearization and use a structure preserving numerical method to then determine the eigenvalues and eigenvectors of the linearization. We note that the QZ algorithm is not a structure preserving method.

Mackey, Mackey, Mehl and Mehrmann [27] have recently shown the existence of vector spaces of potential linearizations of matrix polynomials (including quadratics). The eigenvectors of the quadratic can be directly recovered from those of the linearization, and vector spaces containing structured (symmetric) linearizations are presented. The two vector spaces of potential linearizations are for general degree matrix polynomials but we present them for quadratics to be consistent. To define the vector spaces, we need to use the *Kronecker* product of two matrices. We denote the Kronecker product by \otimes and give a definition below.

Definition 2 (Kronecker Product, see [14]). *Given $A \in \mathbb{R}^{m \times m}$ and $B \in \mathbb{R}^{n \times n}$ the*

Kronecker product of A and B is given by

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots & a_{1m}B \\ a_{21}B & a_{22}B & \cdots & a_{2m}B \\ \vdots & \vdots & & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mm}B \end{bmatrix},$$

and $A \otimes B \in \mathbb{R}^{mn \times mn}$.

The vector spaces of pencils that are potential linearizations of a quadratic Q are denoted by $\mathbb{L}_1(Q)$, $\mathbb{L}_2(Q)$ and the intersection of the two, $\mathbb{DL}(Q)$, (here $\Lambda = [\lambda, 1]^T$),

$$\mathbb{L}_1(Q) = \{L(\lambda) : L(\lambda)(\Lambda \otimes I_n) = v \otimes Q(\lambda), v \in \mathbb{C}^2\} \quad (2.3)$$

$$\mathbb{L}_2(Q) = \{L(\lambda) : (\Lambda^T \otimes I_n)L(\lambda) = w^T \otimes Q(\lambda), w \in \mathbb{C}^2\} \quad (2.4)$$

$$\mathbb{DL}(Q) = \mathbb{L}_1(Q) \cap \mathbb{L}_2(Q). \quad (2.5)$$

It can be shown that almost all of the pencils in $\mathbb{L}_1(Q)$ and $\mathbb{L}_2(Q)$ are linearizations of Q . The first companion linearization is a pencil contained in the space $\mathbb{L}_1(Q)$ (with $v = e_1$) and the second companion linearization is contained in the space $\mathbb{L}_2(Q)$ (with $w = e_1$).

For symmetric quadratics, we can choose a symmetry preserving linearization from the vector space $\mathbb{DL}(Q)$. We determine a linearization by choosing a vector $v = [v_1, v_2]$ where v_1 and v_2 are complex scalars. If the coefficient matrices of $Q(\lambda)$ are symmetric, the pencils (2.6) to (2.9) are symmetry preserving linearizations from the vector space $\mathbb{DL}(Q)$, each linearization is uniquely characterized by a vector v .

$$v = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \lambda \begin{bmatrix} 0 & A_2 \\ A_2 & A_1 \end{bmatrix} + \begin{bmatrix} -A_2 & 0 \\ 0 & A_0 \end{bmatrix}, \quad \det A_2 \neq 0, \quad (2.6)$$

$$v = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \lambda \begin{bmatrix} A_2 & 0 \\ 0 & -A_0 \end{bmatrix} + \begin{bmatrix} A_1 & A_0 \\ A_0 & 0 \end{bmatrix}, \quad \det A_0 \neq 0, \quad (2.7)$$

$$v = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \lambda \begin{bmatrix} A_2 & A_2 \\ A_2 & A_1 - A_0 \end{bmatrix} + \begin{bmatrix} A_1 - A_2 & A_0 \\ A_0 & A_0 \end{bmatrix}, \quad \det Q(-1) \neq 0, \quad (2.8)$$

$$v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, \quad \lambda \begin{bmatrix} v_1 A_2 & v_2 A_2 \\ v_2 A_2 & v_2 A_1 - v_1 A_0 \end{bmatrix} + \begin{bmatrix} v_1 A_1 - v_2 A_2 & v_1 A_0 \\ v_1 A_0 & v_2 A_0 \end{bmatrix}, \quad \det Q\left(-\frac{v_2}{v_1}\right) \neq 0, \quad (2.9)$$

The linearizations (2.6) and (2.7) are only linearizations if the leading and trailing coefficients respectively are nonsingular. If both leading and trailing coefficients are singular we can take the linearization (2.8), provided that $\lambda = -1$ is not an eigenvalue of the quadratic. Otherwise we must determine a vector $v = [v_1, v_2]^T$ such that $-v_2/v_1$ is not an eigenvalue of the quadratic, we can then take (2.9) as a linearization of the quadratic.

2.2 Homogeneous Notation

To enable us to deal more easily with both finite and infinite eigenvalues we will often write the quadratic $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$, in *homogeneous form*, as

$$Q(\alpha, \beta) = \alpha^2 A_2 + \alpha \beta A_1 + \beta^2 A_0.$$

The linearization $L(\lambda) = \lambda X + Y$ can also be written in homogeneous form as

$$L(\alpha, \beta) = \alpha X + \beta Y.$$

The homogeneous form of the quadratic can be derived by substituting $\lambda = \alpha/\beta$ into $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$ yielding

$$Q(\alpha/\beta) = (\alpha/\beta)^2 A_2 + (\alpha/\beta) A_1 + A_0. \quad (2.10)$$

on multiplying (2.10) by β^2 and defining $Q(\alpha, \beta) = \beta^2 Q(\alpha/\beta)$ we obtain the homogeneous form $Q(\alpha, \beta) = \alpha^2 A_2 + \alpha\beta A_1 + \beta^2 A_0$. We associate an eigenvalue $\lambda = \alpha/\beta$ with the pair (α, β) . The representation of λ as α/β is not unique, since for $\xi \neq 0$ we have that $(\xi\alpha, \xi\beta)$ yields $\lambda = (\xi\alpha)/(\xi\beta) = \alpha/\beta = \lambda$. We can express λ uniquely using *normalized* form, by requiring that $|\alpha|^2 + |\beta|^2 = 1$. In normalized form $(0, 1)$ represents a zero eigenvalue and $(1, 0)$ an infinite eigenvalue.

2.3 Solving the GEP

Supposing we have taken the first companion linearization of a quadratic, we then have the pencil

$$C_1(\lambda) = \lambda \begin{bmatrix} A_2 & 0 \\ 0 & I_n \end{bmatrix} + \begin{bmatrix} A_1 & A_0 \\ -I_n & 0 \end{bmatrix}.$$

We apply the QZ algorithm to the problem of determining an *eigenpair* $(z; \lambda)$ of $C_1(\lambda)$; an eigenvalue λ with associated right eigenvector z such that $C_1(\lambda)z = 0$. In this thesis we will focus on determining only right eigenvectors (and we use the term eigenvector to mean a right eigenvector), however once we have determined an eigenvalue we can determine a left eigenvector w such that $w^* C_1(\lambda) = 0$, we can then call $(w, z; \lambda)$ an *eigen triple*.

Mackey, Mackey, Mehl and Mehrmann [27] show that the eigenvectors of the quadratic can be easily recovered from those of linearizations in the spaces $\mathbb{L}_1(Q)$ or $\mathbb{L}_2(Q)$. Higham, Li and Tisseur [19] present theorems using homogeneous notation, we give the corresponding theorem for a linearization in $\mathbb{L}_1(Q)$ below, where $\Lambda_{\alpha, \beta} = [\alpha, \beta]^T$.

Theorem 1 (Eigenvector recovery from \mathbb{L}_1 see Theorem 3.1 [19]). *If $L \in \mathbb{L}_1(Q)$ is a linearization of Q , then x is an eigenvector of Q with eigenvalue (α, β) if and only if $\Lambda_{\alpha, \beta} \otimes x$ is an eigenvector of L with associated eigenvalue (α, β) .*

Using Theorem 1, we can write the form of the right eigenvector of the linearization

as

$$\Lambda_{\alpha,\beta} \otimes x = \begin{bmatrix} \alpha x \\ \beta x \end{bmatrix}. \quad (2.11)$$

When we present algorithms in Chapter 5 we need to know the structure of eigenvectors associated to zero or infinite eigenvalues, (2.11) yields the form of an eigenvector associated to an infinite eigenvalue (2.12) and associated to a zero eigenvalue (2.13)

$$\Lambda_{1,0} \otimes x = \begin{bmatrix} x \\ 0 \end{bmatrix} \quad (\lambda = \infty), \quad (2.12)$$

$$\Lambda_{0,1} \otimes x = \begin{bmatrix} 0 \\ x \end{bmatrix} \quad (\lambda = 0). \quad (2.13)$$

For an infinite eigenvalue $(1, 0)$ the vector x in (2.12) is a null vector of the coefficient matrix A_2 (a vector y is a *null vector* of A if $Ay = 0$), or an eigenvector associated to a zero eigenvalue of the reversal polynomial $\text{rev} Q(\alpha) = \beta^2 A_2 + \alpha\beta A_1 + \alpha^2 A_0$. When the eigenvalue is zero $(0, 1)$, the vector x in (2.13) is a null vector of the coefficient matrix A_0 , or an eigenvector associated to a zero eigenvalue of the quadratic $Q(\alpha, \beta) = \alpha^2 A_2 + \alpha\beta A_1 + \beta^2 A_0$.

2.4 Outline of Existing Software

As mentioned in the introduction, the MATLAB function `polyeig` is the only black box software available for quadratic (or polynomial) eigenvalue problems. `polyeig` solves the quadratic eigenvalue problem by forming a first companion linearization of the given quadratic (the reversal of the first companion linearization of the reversed quadratic). It then applies the QZ algorithm to solve by the process explained in this chapter. The command `[X,e,s] = polyeig(A0, A1, A2)` computes eigenvalues and right eigenvectors of the quadratic $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$. X is a matrix of eigenvectors, e a vector of eigenvalues. By specifying the output variable s we obtain the normwise condition number $\kappa_Q(\alpha, \beta)$ (we explain this later in Chapter 3) of eigenvalues of the quadratic $Q(\lambda)$. `polyeig` uses a special formula to recover left eigenvectors from right eigenvectors, since the formula only holds when at least one

of the leading and trailing coefficients is nonsingular, it does not compute condition numbers when both A_0 and A_2 are singular. We note that `polyeig` does not implement the initial scaling of Fan, Lin and Van Dooren [10].

Chapter 3

Backward Error and Conditioning

In this chapter we present ways of measuring the quality of computed eigenpairs of pencils and quadratic matrix polynomials, and also the sensitivity of the eigen-solution. We will focus on *backward error* and the *conditioning* of eigenvalues and eigenvectors respectively.

When we have a problem to solve with initial sampled data, there is the possibility that the sampled data contains errors. The conditioning of the data measures the sensitivity of the solution of the problem to perturbations in the data. The extent to which the problem is well conditioned is an inherent property of the problem.

Given a method or algorithm for computing a solution to a problem we would like to assess the quality of the computed solution. *Backward error* is a measure of how much the problem must be perturbed for the computed solution to be an exact solution of the perturbed problem.

3.1 Backward Error of Eigenpairs of Quadratics and Matrix Pencils

We give the definition of backward error of a right and left eigenpair of a quadratic matrix polynomial (written in homogenous form, see Section 2.2) below. In this section ΔA_i denotes a perturbation to the A_i coefficient.

Definition 3 (Relative normwise backward error of an approximate right eigenpair).
The relative normwise backward error of an approximate right eigenpair $(x; \alpha, \beta)$ of a quadratic $Q(\alpha, \beta)$ is defined as

$$\eta_Q(x; \alpha, \beta) = \min \{ \epsilon : (Q(\alpha, \beta) + \Delta Q(\alpha, \beta))x = 0, \quad \|\Delta A_i\|_2 \leq \epsilon \|A_i\|_2, \quad i = 0 : 2 \} \quad (3.1)$$

where $\Delta Q(\alpha, \beta) = \alpha^2 \Delta A_2 + \alpha \beta \Delta A_1 + \beta^2 \Delta A_0$.

The equivalent definition for an approximate left eigenpair is given below.

Definition 4 (Relative normwise backward error of an approximate left eigenpair).
The relative normwise backward error of an approximate left eigenpair $(y^; \alpha, \beta)$ of a quadratic $Q(\alpha, \beta)$ is defined as*

$$\eta_Q(y^*; \alpha, \beta) = \min \{ \epsilon : y^*(Q(\alpha, \beta) + \Delta Q(\alpha, \beta)) = 0, \quad \|\Delta A_i\|_2 \leq \epsilon \|A_i\|_2, \quad i = 0 : 2 \} \quad (3.2)$$

with $\Delta Q(\alpha, \beta)$ defined as before.

We now present explicit expressions for relative backward errors of eigenpairs of quadratic matrix polynomials and linear pencils [31]. The relative normwise backward error of an approximate right eigenpair $(x; \alpha, \beta)$ of a quadratic $Q(\alpha, \beta)$ written in homogeneous form is given by

$$\eta_Q(x; \alpha, \beta) = \frac{\|Q(\alpha, \beta)x\|_2}{(|\alpha|^2 \|A_2\|_2 + |\alpha||\beta| \|A_1\|_2 + |\beta|^2 \|A_0\|_2) \|x\|_2}. \quad (3.3)$$

For an approximate left eigenpair $(y^*; \alpha, \beta)$ we have the expression

$$\eta_Q(y^*; \alpha, \beta) = \frac{\|y^*Q(\alpha, \beta)\|_2}{(|\alpha|^2 \|A_2\|_2 + |\alpha||\beta| \|A_1\|_2 + |\beta|^2 \|A_0\|_2) \|y\|_2}. \quad (3.4)$$

Equations (3.3) and (3.4) show that the backward error of a computed eigenpair is a scaled residual. As the representation (α, β) of an eigenvalue λ is not unique, we note that both (3.3) and (3.4) are independent of the scaling of (α, β) .

$\eta_L(z; \alpha, \beta)$ is defined in an analogous way to (3.3) and (3.4). Given an approximate right eigenpair $(z; \alpha, \beta)$ of the linearization $L(\alpha, \beta) = \alpha X + \beta Y$, the relative normwise backward error of the linearization is given by

$$\eta_L(z; \alpha, \beta) = \frac{\|L(\alpha, \beta)z\|_2}{(|\alpha| \|X\|_2 + |\beta| \|Y\|_2) \|z\|_2}, \quad (3.5)$$

and for an approximate left eigenpair $(w^*; \alpha, \beta)$,

$$\eta_L(w^*; \alpha, \beta) = \frac{\|w^* L(\alpha, \beta)\|_2}{(|\alpha| \|X\|_2 + |\beta| \|Y\|_2) \|w\|_2}. \quad (3.6)$$

3.2 Eigenvalue Condition Numbers for Quadratics and Matrix Pencils

We first consider condition numbers of eigenvalues of quadratic matrix polynomials. Higham, Mackey and Tisseur [20] present condition numbers for eigenvalues of matrix polynomials. A condition number $K_Q(\lambda)$ puts a bound on $|\Delta\lambda|/|\lambda|$, which measures the relative change in an eigenvalue. Unfortunately, this condition number is only defined for simple finite nonzero eigenvalues, (not infinite or zero eigenvalues). Another condition number $\kappa_Q(\alpha, \beta)$ that is defined for simple eigenvalues, finite or infinite including zero, provides a bound on the angle between an exact eigenvalue (α, β) and a perturbed eigenvalue $(\tilde{\alpha}, \tilde{\beta})$. The angle is based on viewing an eigenvalue as a line that goes through the origin in the complex plane to the point (α, β) , the eigenvalue is in normalized form (such that $|\alpha|^2 + |\beta|^2 = 1$) and lies on the unit circle. For the quadratic case, this condition number is defined as

$$\kappa_Q(\alpha, \beta) = \max_{\|\Delta A\| \leq 1} \frac{K(\alpha, \beta) \Delta A}{\|[\alpha, \beta]\|_2} \quad (3.7)$$

where $\Delta A = (\Delta A_2, \Delta A_1, \Delta A_0)$. $K(\alpha, \beta)$ is a differential operator defined by Dedieu and Tisseur [8], such that $K(\alpha, \beta) : (\mathbb{C}^{n \times n})^3 \rightarrow T_{(\alpha, \beta)}\mathbb{P}_1$. $T_{(\alpha, \beta)}\mathbb{P}_1$ is a tangent space at (α, β) to \mathbb{P}_1 , which is a projective space of lines through the origin in \mathbb{C}^2 . The condition number can be computed using the expression given below.

Theorem 2 (see Theorem 2.3 [20]). *The normwise condition number $\kappa_Q(\alpha, \beta)$ of a simple eigenvalue (α, β) of a quadratic $Q(\alpha, \beta)$ is given by*

$$\kappa_Q(\alpha, \beta) = \frac{(|\beta|^4 \|A_0\|_F^2 + |\alpha|^2 |\beta|^2 \|A_1\|_F^2 + |\alpha|^4 \|A_2\|_F^2)^{1/2} \|x\|_2 \|y\|_2}{|y^* (\bar{\beta}(2\alpha A_2 + \beta A_1) - \bar{\alpha}(2\beta A_0 + \alpha A_1)) x|}$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

We now focus on condition numbers of eigenvalues of linear matrix pencils. An expression for the normwise condition number of a simple eigenvalue of a linear matrix pencil written in homogeneous form as $L(\alpha, \beta) = \alpha X + \beta Y$ can be derived from Theorem [20] taking the linear (degree 1 polynomial) case. We obtain the expression below.

Theorem 3 (see Theorem 2.3 [20]). *The normwise condition number $\kappa_L(\alpha, \beta)$ of a simple eigenvalue (α, β) of a linear matrix pencil $L(\alpha, \beta)$ is given by*

$$\kappa_L(\alpha, \beta) = \frac{(|\beta|^2 \|Y\|_F^2 + |\alpha|^2 \|X\|_F^2)^{1/2} \|x\|_2 \|y\|_2}{|y^* (\bar{\beta}X - \bar{\alpha}Y) x|}$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

3.3 Numerical Stability of the QZ Algorithm in Solving the Quadratic Eigenvalue Problem by Linearization

As previously described, we solve the QEP $Q(\lambda)x = 0$ by linearizing to a matrix pencil and solving the resulting GEP $L(\lambda)z = 0$. Eigenvalues and eigenvectors of the pencil $L(\lambda)$ are computed using the QZ algorithm. The QZ algorithm is backward stable for the GEP so we expect eigenpairs to have a small backward error. However, the QZ algorithm is *not* backward stable for the QEP, when linearizing and solving the resulting GEP. This is because the QZ algorithm fails to take into account the structure of the linearization, and it introduces small perturbations to the linearization that do not respect the zero and identity blocks, thus perturbations of $L(\lambda)$ do not correspond to perturbations of $Q(\lambda)$; for more details see [19].

Chapter 4

Rank-Revealing Factorizations and the Null Space

We now focus on two main problems, firstly determining the numerical rank of a matrix and secondly computing a basis for the null space. An accurate basis for the null space is a key ingredient of the algorithms we will present in Chapter 5. We begin by giving some background material and definitions, needed when considering the rank-revealing decompositions we will later present.

4.1 Null Space, Rank and the Singular Value Decomposition

We start by considering the null space, giving a definition below.

Definition 5 (Null Space). *Given the matrix $A \in \mathbb{R}^{n \times n}$, such that $\text{rank}(A) = r$, the null space is the space spanned by the set of null vectors x_1, \dots, x_{n-r} such that $Ax_i = 0$ for $i = 1 : n - r$, equivalently $\text{null}(A) = \{x \in \mathbb{R}^n : Ax = 0\}$.*

The dimension of the null space, rank, and dimension of a matrix are related by the expression

$$\text{rank}(A) = n - \dim(\text{null}(A)),$$

that is more usually written as $n = \text{rank}(A) + \dim(\text{null}(A))$.

We now introduce the singular value decomposition, which can be used to motivate the concept of rank of a matrix. A definition is given below for the case of a square matrix (we present decompositions for square matrices since we will only require them for the square case).

Definition 6 (Singular value decomposition). *Let $A \in \mathbb{R}^{n \times n}$ then the singular value decomposition or SVD of A has the form*

$$A = U\Sigma V^T$$

where $U, V \in \mathbb{R}^{n \times n}$ are orthogonal, and $\Sigma \in \mathbb{R}^{n \times n} = \text{diag}(\sigma_1, \dots, \sigma_n)$. The σ_i s are called singular values and are ordered such that

$$\sigma_1 \geq \dots \geq \sigma_r > \sigma_{r+1} = \dots = \sigma_n = 0,$$

then the matrix A has rank r .

We comment that since we use finite precision arithmetic, we really compute numerical subspaces, for example the numerical null space, however we continue to use the term null space but realize we compute approximations to the actual null space. The SVD can be used to motivate a definition of the numerical rank of a matrix in finite precision arithmetic, which we give below.

Definition 7 (Numerical rank). *Given a scalar threshold τ , the numerical rank k is the largest integer such that $\sigma_k > \tau$.*

The choice of τ is important, since it can lead to over or underestimation of the rank of a matrix. The MATLAB toolbox UTV tools uses the tolerance

$$\tau = \sqrt{n} \|A\|_1 \epsilon_{\text{machine}}$$

where $\epsilon_{\text{machine}}$ is the machine precision.

4.2 Rank-Revealing Orthogonal Decompositions

Computing the SVD is one of the most accurate and stable methods of computing the rank and a basis for the null space (a null space basis is formed by taking the $(r+1)$ th

to n th columns of V). We now present alternative methods that are less computationally expensive to compute than the SVD, designed to give good approximations to rank and subspace information.

4.2.1 UTV Decomposition

Given a square matrix $A \in \mathbb{R}^{n \times n}$ a UTV decomposition takes the form

$$A = UTV^T \quad (4.1)$$

where $U, V \in \mathbb{R}^{n \times n}$ are orthogonal, $T \in \mathbb{R}^{n \times n}$ denotes a block triangular matrix, when T is block upper triangular (4.1) is called a URV decomposition with R denoting the block upper triangular matrix. Alternatively (4.1) is called a ULV decomposition when T is block lower triangular. A URV decomposition has the form

$$A = URV^T = [U_{Rk} \ U_{R0}] \begin{bmatrix} R_k & F \\ 0 & G \end{bmatrix} [V_{Rk} \ V_{R0}]^T, \quad (4.2)$$

supposing A has a large enough gap in the k and $(k+1)$ th singular values ($\sigma_{k+1} \ll \sigma_k$), the decomposition is said to be rank-revealing if the following two conditions hold

$$\sigma_{\min}(R_k) = \mathcal{O}(\sigma_k) \quad (4.3)$$

$$\|[F^T, G^T]\|_2 = \mathcal{O}(\sigma_{k+1}). \quad (4.4)$$

A ULV decomposition has the form

$$A = ULV^T = [U_{Lk} \ U_{L0}] \begin{bmatrix} L_k & 0 \\ H & E \end{bmatrix} [V_{Lk} \ V_{L0}]^T, \quad (4.5)$$

again supposing A has a large enough gap in the k and $(k+1)$ th singular values ($\sigma_{k+1} \ll \sigma_k$), the decomposition is said to be rank-revealing if the following two conditions hold

$$\sigma_{\min}(L_k) = \mathcal{O}(\sigma_k), \quad (4.6)$$

$$\|[H, E]\|_2 = \mathcal{O}(\sigma_{k+1}). \quad (4.7)$$

We note that the SVD can be considered as a UTV decomposition, if we allow T to be a diagonal matrix Σ , then we have

$$A = U\Sigma V^T = [U_k \ U_0] \begin{bmatrix} \Sigma_k & 0 \\ 0 & \Sigma_0 \end{bmatrix} [V_k \ V_0]^T, \quad (4.8)$$

with Σ_k and Σ_0 diagonal matrices. Table 4.1 shows the four fundamental subspaces defined by the SVD of the matrix A .

Table 4.1: Subspaces defined by the SVD of the matrix A .

$\text{range}(A) = \text{range}(U)$	$\text{range of } A$
$\text{range}(A^T) = \text{range}(V)$	$\text{row space of } A$
$\text{null}(A) = \text{range}(V_0)$	$\text{null space of } A$
$\text{null}(A^T) = \text{range}(U_0)$	$\text{null space of } A^T$

We will use only the ULV decomposition to approximate the null space since the range of its null space enjoys a smaller upper bound with the range of the SVD null space, as we later describe.

Methods of computing the ULV decomposition that we describe do so by “peeling off”, or deflating singular values one at a time, we recall that

$$\sigma_1 \geq \cdots \geq \sigma_r > \sigma_{r+1} = \cdots = \sigma_n = 0.$$

Algorithms are divided between those that work more efficiently for high rank ($\text{rank}(A) \approx n$) or low rank ($\text{rank}(A) \ll n$) matrices. When the matrix is of high rank the singular values are deflated in order of increasing size starting with the smallest, and conversely when the matrix is low rank, in order of decreasing size starting with the largest.

High-Rank ULV

Stewart [29], [28] has developed alternatives to the SVD for high rank matrices. This uses deflation based on the generalized LINPACK condition estimator. This is to estimate the smallest singular value and corresponding left or right singular vector. UTV tools implements this in the routine `hulv`.

An alternative high rank ULV decomposition is based on the fact that an accurate condition estimator produces an off diagonal block in (4.5) of small norm, leading to a smaller upper bound on the distance between the exact and numerical null space [11]. The routine `hulv_a` uses deflation based on singular vector estimation via inverse iteration.

Low-Rank ULV

Stewart’s high-rank algorithm starts with a triangular factorization of the form $A = UL$ with L lower triangular. The algorithm implemented in UTV tools is an analogue for the low-rank case, deflating large singular values in decreasing size. Low-rank algorithms need to estimate the largest singular value. For this it is not necessary to have the matrix in triangular form. The MATLAB toolbox UTV tools [17] contains implementations of two low-rank ULV algorithms, one that does an initial triangular factorization, and an alternative that avoids an initial triangular factorization. The algorithm that does an initial triangular factorization is termed “warm-started” and is implemented in `lulv_a` whilst the algorithm not doing an initial triangular factorization is termed “cold-started”, it is implemented in `lulv`.

4.2.2 Rank-Revealing QR Factorization

We give the following definition of a rank-revealing QR factorization (RRQR).

Definition 8 (Rank-revealing QR factorization.). *Given $A \in \mathbb{R}^{n \times n}$ a rank-revealing QR factorization takes the form*

$$AP = QR = Q \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix},$$

where $P \in \mathbb{R}^{n \times n}$ is a permutation matrix, $Q \in \mathbb{R}^{n \times n}$ is orthogonal, $R \in \mathbb{R}^{n \times n}$ and $\text{rank}(A) = k$.

Chandrasekaran and Ipsen [7] state the following (possibly equivalent) two constraints imposed when determining P ,

$$\sigma_{\min}(R_{11}) \approx \sigma_k(A) \quad (4.9)$$

$$\sigma_{\max}(R_{22}) \approx \sigma_{k+1}(A). \quad (4.10)$$

In this thesis we only require a basis for the null space and rank of the matrix, this can be computed by taking an RRQR factorization of the transposed matrix, A^T , in which case we take the last k columns of Q , that span the exact null space. The MATLAB command `[Q, R, P] = qr(A)` computes a column pivoted QR factorization of the form $AP = QR$, it implements the LAPACK [1] routine `xGEQP3`. The MATLAB toolbox UTV tools presents two RRQR routines `lrrqr` that implements Chan's [6] low rank revealing QR factorization using deflation by singular vectors via the power method, and `hrrqr` that implements the Chan-Foster [4, 13] high rank revealing QR algorithm where deflation steps are based on using the generalized LINPACK condition estimator.

In many applications other information aside from a basis for the null space is required, and an RRQR is taken of the matrix A (not the transpose). The null space is then approximated by the information from the RRQR factorization we do not then obtain the exact null space. Bounds have been determined for the accuracy of the null space computed via this approach (see for example [5]).

4.2.3 Eigendecomposition

We can determine a basis for the null space of the matrix $A \in \mathbb{R}^{n \times n}$ by taking an eigendecomposition. Computing the real Schur decomposition of A where $\text{rank}(A) = k$ we obtain

$$AQ = Q \begin{bmatrix} T & \tilde{A} \\ 0 & R \end{bmatrix},$$

where $T \in \mathbb{R}^{k \times k}$ is strictly upper triangular (with zeros on the diagonal), $R \in \mathbb{R}^{(n-k) \times (n-k)}$ is upper quasi-triangular, $Q \in \mathbb{R}^{n \times n}$ is orthogonal and $\tilde{A} \in \mathbb{R}^{k \times (n-k)}$.

This yields a basis for the null space, $\{q_1, \dots, q_k\}$, where the vectors are the first k columns of the matrix Q .

4.3 Rank and Null Space of Structured Matrices

When a matrix is structured it is desirable to take advantage of the structure to determine the rank and a basis for the null space without employing a rank-revealing factorization. For example, consider the structured matrix K , where

$$K = \begin{bmatrix} \tilde{K} & 0 \\ 0 & 0 \end{bmatrix},$$

with $K \in \mathbb{R}^{n \times n}$, $\tilde{K} \in \mathbb{R}^{k \times k}$ and $k < n$. The matrix K has the same structure as the stiffness matrix in the mobile manipulator example in Chapter 6. The railtrack problem stiffness matrix has similar structural properties.

The matrix \tilde{K} has full rank, and a basis for the null space of K can be shown to be of the form $\{e_{k+1}, \dots, e_n\}$, where e_j is the j th column of the n -by- n identity matrix. A basis for the null space of the stiffness matrix of the railtrack problem has a similar form.

4.4 Summary of Routines Implemented in MATLAB Code

Our MATLAB codes make available the routines listed in Table 4.2 for computing a basis for the null space of a matrix, the algorithms used to compute the decompositions are also summarized. We note that the routines implemented in the UTV tools MATLAB toolbox can optionally return an a posteriori upper bound on the numerical null space angle to measure the accuracy of the null space computed. If the user has knowledge of either the rank of the matrix whose null space is sought, or knows a lower limit for the rank, then this can be specified as an input argument when using a routine from the UTV tools toolbox. The routine will then deflate

to a given rank, this can be used to ensure the rank is not underestimated. The default rank tolerance in the UTV tools routines is $\sqrt{n} \|A\|_1 \epsilon_{\text{machine}}$ where $\epsilon_{\text{machine}}$ is the machine precision.

Table 4.2: Summary of UTV decompositions.

Routine	Rank	Algorithm
<code>hulv</code>	High	Stewart's high rank revealing ULV algorithm [29]. Deflation based on generalized LINPACK condition estimator.
<code>hulv_a</code>	High	Alternative rank revealing ULV, using deflation based on singular vector estimation via inverse iteration.
<code>lulv</code>	Low	Warm started with initial triangular factorization, deflation using power (default) or Lanczos method.
<code>lulv_a</code>	Low	Cold started with no initial factorization.
<code>lrrqr</code>	Low	Chan and Hansen [6] low rank RRQR. Deflation by singular value estimation using power method.
<code>hrrqr</code>	High	Chan/Foster [4, 13] high rank QR. Deflation based on generalized LINPACK condition estimator.
MATLAB's <code>qr</code>	-	Column pivoted QR factorization (LAPACK <code>xGEPQ3</code>).

4.5 Computational Cost

Taking an eigendecomposition via the QR algorithm (see [15]) of the matrix A such that the real Schur form $Q^T A Q = T$ and computing T and Q has a cost of $25n^3$ flops.

We now summarize the costs of taking ULV decompositions and the SVD, when taking an SVD we require only the matrices Σ and V . We present the cost for the low-rank ULV decomposition from [12]. In Table 4.3 $W(L)$ and $W(A)$ denote the average amount of work to compute the largest singular value and vector of an upper-triangular matrix (L) or a dense matrix not in triangular form (A).

Table 4.4 shows the conditions under which the low-rank ULV (with no initial triangular factorization) is faster than the other algorithms. p is the number of

Table 4.3: Cost of low-rank ULV and SVD algorithms.

Algorithm	Cost (where $\text{rank}(A) = r$)	Conditions (if any)
<code>lulv_a</code>	$4n^3/3 + 12n^2r + (r+1)W(L)$	
<code>lulv</code>	$8n^2r + 6nr + 4n^2r + W(r+1)W(A)$	
<code>lrrqr</code>	-	
<code>hrrqr</code>	$2n^3/3 + 4n^2r^*$	
MATLAB <code>qr</code>	$4n^2r - 4r^2n + 4r^3/3$	
R-SVD	$2n^3 + 11n^2$	$m \geq 5n/3$
Golub-Reinsch SVD	$12n^3$	$5n/3 \geq m \geq n$

* assumes two inverse iteration steps per iteration.

Table 4.4: Conditions for `lulv` to be faster than comparative algorithms.

	<code>lulv_a</code>	R-SVD	Golub-Reinsch SVD
<code>lulv</code>	$n/r \leq 2p + 4$	$n/r \geq 2p + 4$	$n/r \geq (p + 3)/2$

power or Lanczos iterations per deflation step.

4.6 Comparison of Techniques

We give brief comments on some advantages and disadvantages of the methods presented.

Taking the SVD or an eigendecomposition is one of the most stable methods, and compared with the RRQR and UTV decompositions they do not have the possibility of the counter examples that can result in incorrect rank determination (such as the Kahan matrix), however there exist matrices for which certain implementations of the QR algorithm (for example in `eig` fail to converge). The main disadvantage of these two methods is that they are more expensive, and the computational cost does not depend on the rank of the matrix.

The UTV decomposition algorithms have the advantage that they can be more efficient for high or low rank cases, however both their cost and accuracy is dependent

on the condition estimator or the use of inverse iteration or the power method. If a large number of steps of inverse iteration or Lanczos are required then the cost will be larger. However, as seen in Table 4.4, the low rank ULV decomposition can be faster under certain conditions, although the conditions cannot be checked a priori.

Chapter 5

Deflation Algorithms for Quadratic Eigenproblems with Singular Leading or Trailing Coefficients

We now outline algorithms that solve the QEP $Q(\lambda)x = 0$, where

$$Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0, \quad A_i \in \mathbb{R}^{n \times n}, \quad A_2 \neq 0$$

where at least one of A_0 or A_2 are singular. When A_0 is singular we deflate the zero eigenvalue it contributes, and work with a problem of reduced dimension. Alternatively when A_2 is singular we deflate the infinite eigenvalues it contributes, again working with a problem of smaller dimension. Section 5.1 focusses on an algorithm developed by Kublanovskaya et al [26] that deflates both zero and infinite eigenvalues, when the leading and trailing coefficient are singular. A simplification of their method, when only one coefficient is singular is contained in Section 5.2. In Sections 5.3.1 and 5.3.2 we present other algorithms that deflate zero or infinite eigenvalues when either A_0 or A_2 are singular, using Householder reflectors.

5.1 A Modified Version of Kublanovskaya et al's Original Approach

Kublanovskaya, Mikhailov, and Khazanov [26] outline an algorithm to deflate infinite or zero eigenvalues from the quadratic where at least one of the leading or trailing coefficients is singular. In this section we explain how this method works, our presentation differs in a number of details that we highlight when appropriate.

In the original paper the leading and trailing coefficient matrices are represented in a “normalized form” [9]. For this we can use one of the two-sided UTV decompositions outline in Chapter 4. In our explanation we choose a rank-revealing QR factorization as our UTV decomposition (since this is closest to the presentation in the original paper). Taking a rank-revealing QR factorization of A_i^T , $i = 0, 2$ we have $A_i^T = Q_i R_i P_i$, where $Q_i, P_i \in \mathbb{R}^{n \times n}$ are orthogonal matrices (P_i is a permutation matrix), R_i is an upper-triangular matrix such that $\text{diag}(R_i) = [\xi_1, \dots, \xi_{r_i}, 0, \dots, 0]$ where $\text{rank}(R_i) = r_i$. We give a brief outline of the method. We start by making a substitution for λ depending on whether A_0 or A_2 is singular (if both are singular we can use either of the two choices of substitution). We then obtain a quadratic in μ ; $Q(\mu)$. We determine a parameter α , so we can apply the inverse of the leading coefficient of $Q(\mu)$ and then obtain a standard eigenvalue problem. Using the structure of the SEP we deflate infinite and zero eigenvalues as a result of the leading and trailing coefficients. After determining an eigenpair of the deflated matrix we then recover an eigenpair of the quadratic.

When A_0 is singular we make the substitution

$$\lambda = (\alpha^{-1} + \mu^{-1})^{-1}, \quad \alpha \neq 0. \quad (5.1)$$

We then have that

$$Q((\alpha^{-1} + \mu^{-1})^{-1}) = (\alpha^{-1} + \mu^{-1})^{-2} A_2 + (\alpha^{-1} + \mu^{-1})^{-1} A_1 + A_0. \quad (5.2)$$

Multiplying $Q((\alpha^{-1} + \mu^{-1})^{-1})$ in (5.2) by $(\alpha^{-1} + \mu^{-1})^2$ we have that

$$\begin{aligned} (\alpha^{-1} + \mu^{-1})^2 Q((\alpha^{-1} + \mu^{-1})^{-1}) &= A_2 + (\alpha^{-1} + \mu^{-1})A_1 + (\alpha^{-1} + \mu^{-1})^2 A_0. \\ &= A_2 + (\alpha^{-1} + \mu^{-1})A_1 + (\alpha^{-2} + 2(\alpha\mu)^{-1} + \mu^{-2})A_0. \end{aligned}$$

We define $Q_0(\mu) = (\alpha\mu)^2(\alpha^{-1} + \mu^{-1})^2 Q(\lambda)$, and rearrange to yield

$$Q_0(\mu) = \mu^2(\alpha^2 A_2 + \alpha A_1 + A_0) + \mu(\alpha^2 A_1 + 2\alpha A_0) + \alpha^2 A_0.$$

We now consider the case that A_0 and A_2 are singular (this substitution also applies if only A_2 is singular) and make the substitution

$$\lambda = \alpha + \mu^{-1}, \quad \alpha \neq 0. \quad (5.3)$$

We note that when both the leading and trailing coefficients are singular either substitution (5.1) or (5.3) can be made. We then have that

$$\begin{aligned} Q(\alpha + \mu^{-1}) &= (\alpha + \mu^{-1})^2 A_2 + (\alpha + \mu^{-1})A_1 + A_0. \\ &= (\alpha^2 + 2\alpha\mu^{-1} + \mu^{-2})A_2 + (\alpha + \mu^{-1})A_1 + A_0. \end{aligned} \quad (5.4)$$

Multiplying (5.4) by μ^2 , we define $Q_2(\mu) = \mu^2 Q(\lambda)$, and rearrange to obtain

$$Q_2(\mu) = \mu^2(\alpha^2 A_2 + \alpha A_1 + A_0) + \mu(A_1 + 2\alpha A_2) + A_2. \quad (5.5)$$

We define $Q(\alpha) = \alpha^2 A_2 + \alpha A_1 + A_0$. It is now necessary to apply the inverse of $Q(\alpha)$ to a matrix, after we choose a value for α so that $Q(\alpha)$ is well-conditioned. After determining a value for α we apply $Q(\alpha)^{-1}$ to $Q_2(\mu)$, (for the substitution (5.1) we apply $Q(\alpha)^{-1}$ to $Q_0(\mu)$). We then have one of

$$\tilde{Q}_2(\mu) = Q(\alpha)^{-1} Q_2(\mu) = \mu^2 I_n + \mu Q(\alpha)^{-1} (A_1 + 2\alpha A_2) + Q(\alpha)^{-1} A_2 \quad (5.6)$$

$$\tilde{Q}_0(\mu) = Q(\alpha)^{-1} Q_0(\mu). \quad (5.7)$$

A key step is to make the substitution $A_1 + 2\alpha A_2 = \alpha^{-1} Q(\alpha)^{-1} + \alpha A_2 - \alpha^{-1} A_0$, since favorable cancelation results. Making this substitution and linearizing $Q_2(\mu)$ to a matrix pencil by forming the first companion linearization we have,

$$C_1(\mu) = \mu \begin{bmatrix} I_n & 0 \\ 0 & I_n \end{bmatrix} + \begin{bmatrix} Q(\alpha)^{-1}(1/\alpha Q(\alpha) + \alpha A_2 - 1/\alpha A_0) & Q(\alpha)^{-1} A_2 \\ -I_n & 0 \end{bmatrix} =: \mu I_{2n} + C.$$

A similarity transformation of C with the matrices

$$V^{-1} = \begin{bmatrix} \alpha Q_2^T & Q_2^T \\ \alpha Q_0^T & 0 \end{bmatrix}, \quad \text{and} \quad V = \begin{bmatrix} 0 & \alpha^{-1} Q_0 \\ Q_2 & -Q_0 \end{bmatrix} \quad (5.8)$$

gives

$$\tilde{C} = V^{-1}CV = \begin{bmatrix} \alpha Q_2^T Q(\alpha)^{-1} A_2 Q_2 & -\alpha^{-1} Q_2^T Q(\alpha)^{-1} A_0 Q_0 \\ \alpha Q_0^T Q(\alpha)^{-1} A_2 Q_2 & -\alpha^{-1} Q_0^T Q(\alpha)^{-1} A_0 Q_0 + \alpha^{-1} I_n \end{bmatrix}.$$

Using the structure of $A_2 Q_2$ and $A_0 Q_0$,

$$A_i Q_i = \begin{bmatrix} P_i^T \tilde{R}_i^T & 0 \end{bmatrix}, \quad i = 0, 2,$$

we can simplify \tilde{C} to

$$\tilde{C} = \begin{matrix} & r_2 & & k_2 & & r_0 & & k_0 \\ \begin{pmatrix} \alpha Q_2^T Q(\alpha)^{-1} P_2^T \tilde{R}_2^T & 0 & -1/\alpha Q_2^T Q(\alpha)^{-1} P_0^T \tilde{R}_0^T & 0 \\ \alpha Q_0^T Q(\alpha)^{-1} P_2^T \tilde{R}_2^T & 0 & -1/\alpha Q_0^T Q(\alpha)^{-1} P_0^T \tilde{R}_0^T + 1/\alpha I_{n,r_0} & 1/\alpha I_{n,n-r_0} \end{pmatrix}, \end{matrix} \quad (5.9)$$

where $k_2 = n - r_2$ and $k_0 = n - r_0$ are the number of infinite and zero eigenvalues to be deflated. From the structure of this matrix we conclude that \tilde{C} has $(n - r_2)$ zero eigenvalues with associated right eigenvectors $\{e_{r_2+1}, \dots, e_n\}$, and $(n - r_0)$ eigenvalues equal to α^{-1} with corresponding eigenvectors $\{e_{n-r_0+1}, \dots, e_{2n}\}$.

We carry out a similarity transformation on \tilde{C} with the matrix P where

$$P = \begin{bmatrix} 0 & 0 & I_{r_2} & 0 \\ I_{n-r_2} & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{r_0} \\ 0 & I_{n-r_0} & 0 & 0 \end{bmatrix}, \quad (5.10)$$

where I_k denotes the k -by- k identity matrix. We then have that

$$F = P^T \tilde{C} P = \begin{bmatrix} G & H \\ 0 & R \end{bmatrix} \quad (5.11)$$

where $G \in \mathbb{R}^{(2n-r_0-r_2) \times (2n-r_0-r_2)}$, $H \in \mathbb{R}^{(2n-r_0-r_2) \times (r_0+r_2)}$, and $R \in \mathbb{R}^{(r_0+r_2) \times (r_0+r_2)}$.

Further we have that $G = \text{diag}(0, \dots, 0, \alpha^{-1}, \dots, \alpha^{-1})$, where there are $(n - r_0)$ zeros

and $(n - r_2)$ (α^{-1}) s on the diagonal. So we can deflate the problem, working with the submatrix R ; computing the eigenpairs of F via those of R . Since R is of smaller dimension than F , once we compute an eigenpair $(\mu; y_2)$ of G , we need to recover a subvector y_1 so we can form the corresponding eigenvector $y = [y_1, y_2]^T$ of F .

The eigenpair $(y_2; \mu)$ of R satisfies $Ry_2 = \mu y_2$, we obtain an eigenpair $(y = [y_1, y_2]^T; \mu)$ of F such that $Fy = \mu y$ by solving the linear system

$$\begin{bmatrix} G & H \\ 0 & R \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \mu \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}. \quad (5.12)$$

Of the two resulting equations

$$Gy_1 + Hy_2 = \mu y_1 \quad (5.13)$$

$$Ry_2 = \mu y_2, \quad (5.14)$$

(5.14) is satisfied automatically, (5.13) rearranges to the form $Ax = b$ where $A = (G - \mu I_{r_0+r_2})$, $x = y_1$ and $b = -Hy_2$. Provided $\mu \neq 0$, the matrix A is nonsingular and the y_1 component can be determined. The matrix A is in fact diagonal, hence only linear scalar equations must be solved, rather than a full system.

Given the eigenpair $(y; \mu)$ that satisfies $Fy = \mu y$, we apply the matrix P , to obtain the eigenpair $(Py; \mu)$ that is an eigenpair of \tilde{C} satisfying $\tilde{C}Py = \mu Py$. Finally, we obtain an eigenpair $(z; \mu)$ of C such that $Cz = \mu z$ by applying the matrix V to Py .

To obtain an eigenpair $(x; \lambda)$ of the quadratic $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$ we first make the transformation $\lambda = \alpha + \mu^{-1}$ (or $\lambda = (\alpha^{-1} + \mu^{-1})^{-1}$), then from the eigenvector of the linearization $z \in \mathbb{R}^{2n}$ we select as an eigenvector of the quadratic, one of $\xi_1 = z(1:n)$, or $\xi_2 = z(n+1:2n)$, that yields the smallest backward error.

We now justify the deflation procedure, first we note the following argument. Given that we want to solve the standard eigenvalue problem $Ax = \lambda x$, suppose x is equivalent to the row permuted eigenvector \tilde{x} , then making the substitution $x = P\tilde{x}$, where P is a permutation matrix we have that,

$$AP\tilde{x} = \lambda P\tilde{x}$$

if we then premultiply by P^T we have that

$$\begin{aligned} P^T AP\tilde{x} &= \lambda P^T P\tilde{x} \\ &= \lambda\tilde{x} \end{aligned} \tag{5.15}$$

Thus from Equation (5.15) the eigenvalues of $P^T AP$ are the same as that of A but the eigenvectors x of the original matrix are equivalent to row permuted eigenvectors \tilde{x} of $P^T AP$.

Thus using this method we are able to deflate infinite and zero eigenvalues contributed by the leading and trailing coefficient matrices. However the method does have significant drawbacks. Firstly, we need to determine a value for the parameter α such that $Q(\alpha)$ is well conditioned. Secondly we must apply the inverse, $Q(\alpha)^{-1}$ to the leading and trailing coefficient matrices. These two steps both increase the cost and could affect the accuracy and stability of the method.

5.2 Simplification of Kublanovskaya's Method

The original method of Kublanovskaya, Mikhailov, and Khazanov [26] can be simplified when only one of the leading or trailing coefficient matrices is singular. We outline the deflation of zero eigenvalues; we can deflate infinite eigenvalues when A_2 is singular using the same approach on the reversal polynomial. Let $Q(\lambda)$ be a quadratic where A_2 is nonsingular and A_0 singular, with $\text{rank}(A_0) = r_0 < n$. As in the previous section, we can take any of the two-sided UTV decompositions of Section 4. We again use a rank-revealing QR factorization of A_0^T such that

$$A_0^T P_0^T = Q_0 R_0, \tag{5.16}$$

where $Q_0, P_0 \in \mathbb{R}^{n \times n}$ are orthogonal, $R_0 \in \mathbb{R}^{n \times n}$ is upper triangular, and P_0 is a permutation matrix. Forming the first companion linearization of $Q(\lambda)$ we have

$$C_1(\lambda) = \lambda \begin{bmatrix} A_2 & 0 \\ 0 & I_n \end{bmatrix} + \begin{bmatrix} A_1 & A_0 \\ -I_n & 0 \end{bmatrix} \tag{5.17}$$

Forming $\tilde{C}_1(\lambda) := K^T C_1(\lambda) K$, where $K = \begin{bmatrix} I_n & 0 \\ 0 & Q_0 \end{bmatrix}$ we have

$$\tilde{C}_1(\lambda) = \lambda \begin{bmatrix} A_2 & 0 \\ 0 & I_n \end{bmatrix} + \begin{bmatrix} A_1 & A_0 Q_0 \\ -Q_0^T & 0 \end{bmatrix}. \quad (5.18)$$

Since

$$A_0 Q_0 = P_0^T R_0^T = P_0^T [\tilde{R}_0^T \ 0],$$

(5.18) has the form

$$\tilde{C}_1(\lambda) = \lambda \begin{bmatrix} A_2 & 0 \\ 0 & I_n \end{bmatrix} + \begin{bmatrix} A_1 & P_0^T \tilde{R}_0^T & 0 \\ -Q_0^T & 0 & 0 \end{bmatrix}.$$

On removing the last $n - r_0$ rows and columns of $\tilde{C}_1(\lambda)$, we deflate the zero eigenvalues of the pencil.

Once we have determined an eigenpair of the (smaller) deflated problem, we must recover the eigenvector of $\tilde{C}_1(\lambda)$, we can then reapply the transformation K to yield an eigenvector of the first companion linearization, finally, via a backward error argument we obtain an eigenvector of the quadratic. We first partition (5.18) as

$$\tilde{C}_1(\lambda) = \lambda \begin{bmatrix} A_2 & 0 & 0 \\ 0 & I_{r_0} & 0 \\ 0 & 0 & I_{n-r_0} \end{bmatrix} + \begin{bmatrix} A_1 & P_0^T \tilde{R}_0^T & 0 \\ -\widehat{Q}_0^T & 0 & 0 \\ -\tilde{Q}_0^T & 0 & 0 \end{bmatrix} \quad (5.19)$$

then $-\tilde{Q}_0^T \in \mathbb{R}^{(n-r_0) \times n}$ is the last $n - r_0$ rows of $-Q_0^T$ and $-\widehat{Q}_0^T \in \mathbb{R}^{r_0 \times n}$ is the first r_0 rows of $-Q_0^T$. We define \tilde{X} and \tilde{Y} to have the structure

$$\tilde{X} = \begin{bmatrix} A_2 & 0 \\ 0 & I_{r_0} \end{bmatrix}, \quad \tilde{Y} = \begin{bmatrix} A_1 & P_0^T \tilde{R}_0^T \\ -\widehat{Q}_0^T & 0 \end{bmatrix}, \quad (5.20)$$

where $\tilde{X}, \tilde{Y} \in \mathbb{R}^{(n+r_0) \times (n+r_0)}$. We then determine eigenpairs of the pencil

$$L(\lambda) = \lambda \tilde{X} + \tilde{Y}, \quad (5.21)$$

and we now explain how eigenvectors of (5.19) can be recovered from those of the

pencil (5.21). Writing (5.19) in homogeneous form we have

$$L(\alpha, \beta) = \alpha \begin{bmatrix} A_2 & 0 & 0 \\ 0 & I_{r_0} & 0 \\ 0 & 0 & I_{n-r_0} \end{bmatrix} + \beta \begin{bmatrix} A_1 & P_0^T \tilde{R}_0^T & 0 \\ -\tilde{Q}_0^T & 0 & 0 \\ -\tilde{Q}_0^T & 0 & 0 \end{bmatrix}. \quad (5.22)$$

If $(\alpha, \beta; \tilde{z})$ is an eigenpair of $L(\alpha, \beta)$ then the eigenvector is of the form $\tilde{z} = [z_1, z_2, z_3]^T$ and satisfies

$$L(\alpha, \beta)\tilde{z} = \left(\alpha \begin{bmatrix} A_2 & 0 & 0 \\ 0 & I_{r_0} & 0 \\ 0 & 0 & I_{n-r_0} \end{bmatrix} + \beta \begin{bmatrix} A_1 & P_0^T \tilde{R}_0^T & 0 \\ -\tilde{Q}_0^T & 0 & 0 \\ -\tilde{Q}_0^T & 0 & 0 \end{bmatrix} \right) \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (5.23)$$

(5.23) yields the three equations,

$$\alpha A_2 z_1 + \beta A_1 z_1 + \beta P_0^T \tilde{R}_0^T z_2 = 0 \quad (5.24)$$

$$\alpha I_{r_0} z_2 - \beta \tilde{Q}_0^T z_1 = 0 \quad (5.25)$$

$$\alpha I_{n-r_0} z_3 - \beta \tilde{Q}_0^T z_1 = 0 \quad (5.26)$$

Since we solved the GEP in Equation (5.21), Equations (5.24) and (5.25) are automatically satisfied. The vector z_3 can be determined by Equation (5.26), or if $\alpha = 0$ we can set z_3 arbitrarily, and if $\beta = 0$ we set $z_3 = 0$.

On obtaining an eigenpair (λ, \tilde{z}) of $L(\lambda)$ we recover an eigenvector of the first companion linearization by a matrix-vector product. If (λ, \tilde{z}) solves $L(\lambda)\tilde{z} = 0$ and $L(\lambda) = K^T C_1(\lambda)K$, where $C_1(\lambda)$ is the first companion linearization then $(\lambda, z = K\tilde{z})$ solves $C_1(\lambda)z = 0$. An eigenpair of the QEP is then recovered by a backward error argument applied to z .

5.3 Householder Deflation

In this section we present algorithms that deflate zero or infinite eigenvalues from quadratics with singular leading or trailing coefficients, using Householder reflectors. The technique presented can be used to deflate a quadratic given the knowledge of any eigenpair but we focus on the deflation of infinite and zero eigenvalues. First

we focus on deflating either zero or infinite eigenvalues, then we show how the two methods can be coupled to deflate zero and infinite eigenvalues from a quadratic. We first give a definition of a Householder reflector (also known as a Householder transformation).

Definition 9 (Householder reflector [15]). *A Householder reflector $H \in \mathbb{R}^{n \times n}$ is a symmetric and orthogonal matrix of the form*

$$H = I - 2 \frac{vv^T}{v^T v},$$

where $v \in \mathbb{R}^n$ is called the Householder vector.

Householder reflectors can be used to zero components of a vector, for more details see [15]. The deflation process presented has similarities with that of a constructive proof of the generalized Schur decomposition, which we give below.

Theorem 4 (The Generalized Schur Decomposition). *If A and B are in $\mathbb{C}^{n \times n}$ then there exist unitary Q and Z such that $Q^*AZ = T$ and $Q^*BZ = S$ are upper triangular. If for some k , t_{kk} and s_{kk} are both zero, then $\lambda(A, B) = \mathbb{C}$ (where $\lambda(A, B)$ denotes the set of all eigenvalues of the pencil $A - \lambda B$). Otherwise*

$$\lambda(A, B) = \{t_{ii}/s_{ii} : s_{ii} \neq 0\}.$$

Proof. The proof is by construction, let v_1 be an eigenvector of the pencil $A - \lambda B$ and let V be a Householder reflector such that $Ve_1 = v_1$, and let $V = [v_1 \tilde{V}]$. Since the pencil $A - \lambda B$ is regular we have that at least one of the vectors Ax and Bx must be nonzero. We suppose Ax is nonzero and note that if $Bx \neq 0$ then it must be proportional to Ax . Let $u_1 = Ax/\|Ax\|_2$ and let U be a Householder reflector such that $Ue_1 = u_1$, and let $U = [u_1 \tilde{U}]$. We then have that

$$\begin{aligned} U^*AV &= \begin{bmatrix} u_1 & \tilde{U} \end{bmatrix}^* A \begin{bmatrix} v_1 \\ \tilde{V} \end{bmatrix} = \begin{bmatrix} u_1^*Av_1 & u_1^*A\tilde{V} \\ \tilde{U}^*Av_1 & \tilde{U}^*A\tilde{V} \end{bmatrix} \\ &= \begin{bmatrix} \alpha_{11} & s_{12}^* \\ 0 & \tilde{A} \end{bmatrix} \end{aligned}$$

and

$$\begin{aligned} U^*BV &= \begin{bmatrix} u_1 & \tilde{U} \end{bmatrix}^* B \begin{bmatrix} v_1 \\ \tilde{V} \end{bmatrix} = \begin{bmatrix} u_1^*Bv_1 & u_1^*B\tilde{V} \\ \tilde{U}^*Bv_1 & \tilde{U}^*B\tilde{V} \end{bmatrix} \\ &= \begin{bmatrix} \beta_{11} & t_{12}^* \\ 0 & \tilde{B} \end{bmatrix}. \end{aligned}$$

That $\tilde{U}^*Av_1 = 0$ follows since Av_1 is orthogonal to the column space of \tilde{U} by construction. The proof is completed by an inductive reduction of the pencil $\tilde{A} - \lambda\tilde{B}$ to triangular form. \square

5.3.1 Deflation of Zero Eigenvalues

When the trailing coefficient A_0 is singular, with $\text{rank}(A_0) = r_0 < n$, it contributes zero eigenvalues to the quadratic. We show how these can be deflated from $Q(\lambda)$ by applying Householder reflectors to the first companion linearization of the quadratic. To perform the deflation we require an orthonormal basis for the null space of A_0 ; we denote this by $\mathcal{N}_0 = \{v_1, \dots, v_{n-r_0}\}$, where the vectors v_i span the null space of A_0 . The null space basis can be computed using any of the techniques of Chapter 4. Forming the first companion linearization of the quadratic, we have

$$C_1(\lambda) = \lambda \begin{bmatrix} A_2 & 0 \\ 0 & I_n \end{bmatrix} + \begin{bmatrix} A_1 & A_0 \\ -I_n & 0 \end{bmatrix} =: \lambda X_1 + Y_1,$$

where $X_1, Y_1 \in \mathbb{R}^{2n \times 2n}$. The eigenvectors z_i of $C_1(\lambda)$ associated to the zero eigenvalue (cf. equation (2.13)) have the form

$$z_i = \begin{bmatrix} 0 \\ v_i \end{bmatrix} \in \mathbb{R}^{2n}.$$

We first deflate the eigenpair $(z_1; 0)$ from $C_1(\lambda)$. Let H_1 be a Householder reflector such that $H_1e_1 = z_1$, (where e_1 denotes the first column of the $2n$ -by- $2n$ identity matrix). Let $H_1 = [z_1 \ U_1] \in \mathbb{R}^{2n \times 2n}$. Since

$$\begin{bmatrix} A_2 & 0 \\ 0 & I_n \end{bmatrix} z_1 = \begin{bmatrix} A_2 & 0 \\ 0 & I_n \end{bmatrix} \begin{bmatrix} 0 \\ v_1 \end{bmatrix} = z_1, \quad (5.27)$$

and we have that $X_1 z_1 = z_1$. We now form $\tilde{C}_1(\lambda) = H_1^T C_1(\lambda) H_1$. (5.28) follows from $X_1 z_1 = z_1$ and $z_1^T z_1 = 1$, on forming (5.29) we note that $Y_1 z_1 = 0$.

$$H_1^T X_1 H_1 = \begin{bmatrix} z_1^T \\ U_1^T \end{bmatrix} X_1 \begin{bmatrix} z_1 & U_1 \end{bmatrix} = \begin{bmatrix} 1 & z_1^T X_1 U_1 \\ 0 & U_1^T X_1 U_1 \end{bmatrix}. \quad (5.28)$$

$$H_1^T Y_1 H_1 = \begin{bmatrix} z_1^T \\ U_1^T \end{bmatrix} Y_1 \begin{bmatrix} z_1 & U_1 \end{bmatrix} = \begin{bmatrix} 0 & z_1^T Y_1 U_1 \\ 0 & U_1^T Y_1 U_1 \end{bmatrix}. \quad (5.29)$$

Hence

$$\tilde{C}_1(\lambda) = \lambda \begin{bmatrix} 1 & z_1^T X_1 U_1 \\ 0 & U_1^T X_1 U_1 \end{bmatrix} + \begin{bmatrix} 0 & z_1^T Y_1 U_1 \\ 0 & U_1^T Y_1 U_1 \end{bmatrix} =: \lambda X_2 + Y_2,$$

where $X_2, Y_2 \in \mathbb{R}^{2n \times 2n}$. At this point we have deflated the eigenpair $(z_1; 0)$ and we now work with the pencil

$$\lambda U_1^T X_1 U_1 + U_1^T Y_1 U_1 =: \lambda \tilde{X}_2 + \tilde{Y}_2, \quad \text{where } \tilde{X}_2, \tilde{Y}_2 \in \mathbb{R}^{(2n-1) \times (2n-1)}.$$

We will now outline the next deflation to be performed, in particular the form of the next eigenvector (of $\lambda X_2 + Y_2$) associated with a zero eigenvalue. We show that

$$H_1^T z_2 = \begin{bmatrix} 0 \\ \tilde{z}_2 \end{bmatrix},$$

and then show that \tilde{z}_2 is an eigenvector of the pencil $\lambda \tilde{X}_2 + \tilde{Y}_2$. Since $z_1^T z_2 = 0$ and $H_1 e_1 = z_1$ we have $z_1^T z_2 = (e_1^T H_1^T) z_2 = 0$ viewing this as $e_1^T (H_1^T z_2) = 0$, we conclude that the first entry of $H_1^T z_2$ is zero.

We will deflate the eigenpair $(0, \tilde{z}_2)$, where $\tilde{z}_2 = [H_1^T z_2](2: 2n) \in \mathbb{R}^{2n-1}$. Let H_2 be a Householder reflector such that $H_2 e_1 = \tilde{z}_2$, and let $H_2 = [\tilde{z}_2 \ U_2] \in \mathbb{R}^{(2n-1) \times (2n-1)}$.

We have that

$$X_1 z_i = z_i \iff H_1^T X_1 H_1 (H_1^T z_i) = H_1^T z_i.$$

We then form

$$X_2 (H_1^T z_2) = \begin{bmatrix} 1 & z_1^T X_1 U_1 \\ 0 & U_1^T X_1 U_1 \end{bmatrix} \begin{bmatrix} 0 \\ \tilde{z}_2 \end{bmatrix} = \begin{bmatrix} (z_1^T X_1 U_1) \tilde{z}_2 \\ (U_1^T X_1 U_1) \tilde{z}_2 \end{bmatrix} \equiv \begin{bmatrix} 0 \\ \tilde{z}_2 \end{bmatrix}. \quad (5.30)$$

Since $(U_1^T X_1 U_1) \tilde{z}_2 = 0$ we have $X_2 \tilde{z}_2 = 0$. Considering $Y_1(H_1^T z_2)$ then $(H_1^T Y_1 H_1) H_1^T z_i = 0$

$$Y_2(H_1^T z_2) = \begin{bmatrix} 0 & z_1^T Y_1 U_1 \\ 0 & U_1^T Y_1 U_1 \end{bmatrix} \begin{bmatrix} 0 \\ \tilde{z}_2 \end{bmatrix} = \begin{bmatrix} (z_1^T Y_1 U_1) \tilde{z}_2 \\ (U_1^T Y_1 U_1) \tilde{z}_2 \end{bmatrix} \equiv \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad (5.31)$$

and we conclude that $Y_2 \tilde{z}_2 = 0$, thus in conclusion $(\lambda X_2 + Y_2) \tilde{z}_2 = 0$ and \tilde{z}_2 is an eigenvector of the pencil $\lambda X_2 + Y_2$. Proceeding as before we then deflate the eigenpair $(0, \tilde{z}_2)$.

Supposing we deflate all $(n - r_0)$ zero eigenvalues as a result of A_0 , then we obtain a matrix pencil of the form

$$L(\lambda) = \lambda X_{n-r_0} + Y_{n-r_0} = \lambda \begin{bmatrix} R_{11} & R_{12} \\ 0 & \tilde{X}_{n-r_0} \end{bmatrix} + \begin{bmatrix} T_{11} & T_{12} \\ 0 & \tilde{Y}_{n-r_0} \end{bmatrix}, \quad (5.32)$$

where $R_{11}, T_{11} \in \mathbb{R}^{(n-r_0) \times (n-r_0)}$, $R_{12}, T_{12} \in \mathbb{R}^{(n-r_0) \times (n+r_0)}$. R_{11} and T_{11} are upper triangular matrices with $\text{diag}(T_{11}) = (1, \dots, 1)$ and $\text{diag}(R_{11}) = (0, \dots, 0)$. $\tilde{X}_{n-r_0}, \tilde{Y}_{n-r_0} \in \mathbb{R}^{(n+r_0) \times (n+r_0)}$ define a pencil $\tilde{L}(\lambda) = \lambda \tilde{X}_{n-r_0} + \tilde{Y}_{n-r_0}$ that has no zero eigenvalues as a result of A_0 . After determining an eigenpair of $\tilde{L}(\lambda)$ we now show how to recover an eigenpair of (5.32). Eigenvectors associated to nonzero finite eigenvalues of the form (α, β) , $\alpha, \beta \neq 0$ can be recovered through solving a upper triangular linear system, we present the method now for the case of zero eigenvalues having been deflated (the method for infinite eigenvalues having been deflated is similar).

The deflated pencil of smaller dimension we write in homogeneous form as $\tilde{L}(\alpha, \beta) = \alpha \tilde{X}_{n-r_0} + \beta \tilde{Y}_{n-r_0}$. We now show how given an eigenpair $(\tilde{z}; \alpha, \beta)$ of $\tilde{L}(\alpha, \beta)$ such that $(\alpha \tilde{X}_{n-r_0} + \beta \tilde{Y}_{n-r_0}) \tilde{z} = 0$, an eigenvector z of $L(\alpha, \beta)$ satisfying $(\alpha X_{n-r_0} + \beta Y_{n-r_0}) z = 0$ can be recovered. We desire a solution to the system

$$\left(\alpha \begin{bmatrix} R_{11} & R_{12} \\ 0 & \tilde{X}_{n-r_0} \end{bmatrix} + \beta \begin{bmatrix} T_{11} & T_{12} \\ 0 & \tilde{Y}_{n-r_0} \end{bmatrix} \right) \begin{bmatrix} z_u \\ \tilde{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \quad (5.33)$$

Of the two equations in (5.33), the first

$$\alpha(R_{11} z_u + R_{12} \tilde{z}) + \beta(T_{11} z_u + T_{12} \tilde{z}) = 0 \quad (5.34)$$

rearranges to the form $Nx = b$, where $N = (\alpha R_{11} + \beta T_{11})$, $b = -(\alpha R_{12} + \beta T_{12}) \tilde{z}$ and $x = z_u$. The second equation, $(\alpha \tilde{X}_{n-r_0} + \beta \tilde{Y}_{n-r_0}) \tilde{z} = 0$ is automatically satisfied since

$(\tilde{z}; \alpha, \beta)$ is an eigenpair of $\tilde{L}(\alpha, \beta)$. That (5.34) has a solution follows by verifying that $\det(\alpha R_{11} + \beta T_{11}) \neq 0$. Since $\beta \neq 0$, $\text{diag}(\alpha R_{11} + \beta T_{11}) = \text{diag}(\beta T_{11}) \neq 0$ we have $\det(\beta T_{11}) \neq 0$.

5.3.2 Deflation of Infinite Eigenvalues

To deflate infinite eigenvalues from a quadratic whose leading coefficient is singular we can form the reversal polynomial, take the first companion linearization and deflate zero eigenvalues using the Householder approach of the previous section. We will later present a method that deflates zero and infinite eigenvalues from a quadratic whose leading and trailing coefficients are both singular, in this method we will need to deflate infinite eigenvalues without forming the reversal polynomial. We now outline how to deflate an infinite eigenvalue from a quadratic without forming the reversal and deflating a corresponding zero eigenvalue.

Suppose the leading coefficient A_2 is singular, with $\text{rank}(A_2) = r_2 < n$. On forming the first companion linearization of $Q(\lambda)$ we have

$$C_1(\lambda) = \lambda \begin{bmatrix} A_2 & 0 \\ 0 & I_n \end{bmatrix} + \begin{bmatrix} A_1 & A_0 \\ -I_n & 0 \end{bmatrix} =: \lambda X_1 + Y_1,$$

with $X_1, Y_1 \in \mathbb{R}^{2n \times 2n}$. Let $\mathcal{N}_2 = \{v_1, \dots, v_{n-r_2}\}$ be an orthonormal basis for the null space of A_2 . We choose Householder reflectors $H_1 = [x_1, U_1]$ and $K_1 = [z_1, V_1]$ such that

$$x_1 = \frac{Y_1 z_1}{\|Y_1 z_1\|_2}, \quad \text{and } z_1 = \begin{pmatrix} v_1 \\ 0 \end{pmatrix},$$

where $H_1, K_1 \in \mathbb{R}^{2n \times 2n}$ and $x_1, z_1 \in \mathbb{R}^{2n}$. We note that since z_1 is a null vector of X_1

$$X_1 z_1 = \begin{bmatrix} A_2 & 0 \\ 0 & I_n \end{bmatrix} \begin{bmatrix} v_1 \\ 0 \end{bmatrix} = 0,$$

and $U_1^T Y_1 z_1 = 0$ since the column space of U_1 is orthogonal to $Y_1 z_1$. We carry out the transformation $H_1^T (\lambda X_1 + Y_1) K_1$, where

$$H_1^T X_1 K_1 = \begin{bmatrix} x_1^T \\ U_1^T \end{bmatrix} X_1 \begin{bmatrix} z_1 & U_1 \end{bmatrix} = \begin{bmatrix} 0 & x_1^T X_1 V_1 \\ 0 & U_1^T X_1 V_1 \end{bmatrix} \quad (5.35)$$

and

$$H_1^T Y_1 K_1 = \begin{bmatrix} x_1^T \\ U_1^T \end{bmatrix} Y_1 \begin{bmatrix} z_1 & U_1 \end{bmatrix} = \begin{bmatrix} \xi_1 & x_1^T Y_1 V_1 \\ 0 & U_1^T Y_1 V_1 \end{bmatrix} \quad (5.36)$$

In (5.36), $\xi_1 = \|Y_1 z_1\|_2 \neq 0$. The pencil $\lambda U_1^T X_1 V_1 + U_1^T Y_1 V_1 =: \lambda \tilde{X}_2 + \tilde{Y}_2$ now has one infinite eigenvalue deflated from it (contributed by the A_2 coefficient matrix). If another infinite eigenvalue is to be deflated we work with the pencil $\lambda \tilde{X}_2 + \tilde{Y}_2$. We choose Householder reflectors $H_2 = [x_2, U_2]$ and $K_2 = [z_2, V_2]$. In particular z_2 is chosen to be a null vector of the matrix \tilde{X}_2 that takes the form $z_2 = \omega(2: 2n)^T$ where

$$\omega = K_1^T \phi, \quad \phi = \begin{pmatrix} v_2 \\ 0 \end{pmatrix} \in \mathbb{R}^{2n},$$

and v_2 is a null vector of A_2 . In fact the first entry of ω is zero since

$$K_1^T \phi = \begin{pmatrix} z_1^T \phi \\ V_1^T \phi \end{pmatrix} = \begin{pmatrix} 0 \\ V_1^T \phi \end{pmatrix}.$$

We then carry out a transformation as before. Once we have deflated all the infinite eigenvalues as a result of A_2 , we compute an eigenpair of the deflated pencil

$$\lambda X_{n-r_2} + Y_{n-r_2} \quad (5.37)$$

We then recover eigenvectors of $Q(\lambda)$ via those of (5.37) using a similar process to that used in the deflation of zero eigenvalues via Householder reflectors.

5.3.3 Deflation of Zero and Infinite Eigenvalues

We now outline an approach based on using Householder reflectors to deflate both zero and infinite eigenvalues from a quadratic matrix polynomial with singular leading and trailing coefficients. Let the quadratic $Q(\lambda)$ have singular leading and trailing matrix coefficients such that $\text{rank}(A_2) = r_2$, and $\text{rank}(A_0) = r_0$, let $k_2 = n - r_2$ and $k_0 = n - r_0$ be the number of infinite and zero eigenvalues to be deflated from the first companion linearization that we denote by $C_1(\lambda)$.

First Householder zero deflation is applied, after accumulating the Householder reflectors into the matrix H_z where z subscript denotes zero eigenvalue deflation, we

obtain the pencil $L_1(\lambda) = H_z^T C_1(\lambda) H_z$, where

$$L_1(\lambda) = \lambda X_1 + Y_1 = \lambda \begin{bmatrix} F_{11} & F_{12} \\ 0 & \tilde{X}_1 \end{bmatrix} + \begin{bmatrix} G_{11} & G_{12} \\ 0 & \tilde{Y}_1 \end{bmatrix} \quad (5.38)$$

where $F_{11}, G_{11} \in \mathbb{R}^{k_0 \times k_0}$ are both upper triangular, with $\text{diag}(F_{11}) = 1$ and $\text{diag}(G_{11}) = 0$, the pair $\tilde{X}_1, \tilde{Y}_1 \in \mathbb{R}^{(2n-k_0) \times (2n-k_0)}$ form a pencil $\lambda \tilde{X}_1 + \tilde{Y}_1$ that has no zero eigenvalues contributed from A_0 . Let $\mathcal{N}_2 = \{q_1, \dots, q_{n-r_2}\}$ be an orthonormal basis for the null space of the coefficient matrix A_2 (computed by one of the methods of Chapter 4), then let $\tilde{V} \in \mathbb{R}^{n \times k_2}$ be a matrix whose columns are the vectors of \mathcal{N}_2 . Embed \tilde{V} into a matrix of zeros to form $V \in \mathbb{R}^{2n \times k_2}$ such that

$$V = \begin{bmatrix} \tilde{V} \\ 0 \end{bmatrix}. \quad (5.39)$$

We note that the columns of \tilde{V} are the eigenvectors of the quadratic $Q(\lambda)$ corresponding to infinite eigenvalues. After applying the matrix H_z to V to form $V \leftarrow H_z V$, we then apply Householder infinite eigenvalue deflation but rather than using a basis for the null space of A_2 embedded in a zero matrix to form Householder reflectors, we use the modified basis V . We now define the following notation, let $H_z \in \mathbb{R}^{2n \times 2n}$ and $H_i, K_i \in \mathbb{R}^{2n \times 2n}$ denote the accumulated Householder reflectors for deflation of zero and infinite eigenvalues, that are embedded into an identity matrix. In addition, let $\tilde{H}_z \in \mathbb{R}^{k_0 \times k_0}$ and $\tilde{H}_i, \tilde{K}_i \in \mathbb{R}^{k_2 \times k_2}$, be the accumulated Householder reflectors that are embedded in an identity matrix to form H_z, H_i, K_i . Accumulating the Householder reflectors into matrices H_i and K_i , we obtain the pencil $L_2(\lambda) = H_i^T L_1(\lambda) K_i$ where

$$L_2(\lambda) = \lambda X_2 + Y_2 = \lambda \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \\ 0 & 0 & \tilde{X}_2 \end{bmatrix} + \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ 0 & T_{22} & T_{23} \\ 0 & 0 & \tilde{Y}_2 \end{bmatrix}. \quad (5.40)$$

$R_{11}, T_{11} \in \mathbb{R}^{k_0 \times k_0}$ with $\text{diag}(R_{11}) = 1$ and $\text{diag}(T_{11}) = 0$,

$R_{22}, T_{22} \in \mathbb{R}^{k_2 \times k_2}$ with $\text{diag}(R_{22}) = 0$ and $\text{diag}(T_{22}) \neq 0$,

$R_{12}, T_{12} \in \mathbb{R}^{k_0 \times k_2}$, $R_{13}, T_{13} \in \mathbb{R}^{k_0 \times (r_0+r_2)}$, $R_{23}, T_{23} \in \mathbb{R}^{k_2 \times (r_0+r_2)}$,

The matrices $\tilde{X}_2, \tilde{Y}_2 \in \mathbb{R}^{(r_0+r_2) \times (r_0+r_2)}$ form the pencil $\lambda \tilde{X}_2 + \tilde{Y}_2$ from which all infinite

and zero eigenvalues as a result of the A_2 and A_0 coefficient matrices have been deflated.

After determining an eigenpair $(\tilde{z}; \lambda)$ of $\lambda\tilde{X}_2 + \tilde{Y}_2$, we need to recover an eigenvector of the first companion linearization, we now give details. Overall we must do the following

1. Determine an eigenvector of $L_2(\lambda) = \lambda X_2 + Y_2$, by solving linear systems.
2. Apply the accumulated Householder reflectors from infinite eigenvalue deflation to part of the eigenvector.
3. Apply the accumulated Householder reflectors from zero eigenvalue deflation to part of the eigenvector.

We assume that λ is a finite non-zero eigenvalue such that $\lambda = \alpha/\beta$ and in homogeneous form (α, β) . Let z be an eigenvector of $L_2(\alpha, \beta) = \alpha X_2 + \beta Y_2$ that we write as

$$z = [z_1, z_2, \tilde{z}]^T,$$

with $z_1 \in \mathbb{R}^{k_0}$, $z_2 \in \mathbb{R}^{k_2}$, and $\tilde{z} \in \mathbb{R}^{r_0+r_2}$. To determine z we must solve the linear system

$$\left(\alpha \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \\ 0 & 0 & \tilde{X}_2 \end{bmatrix} + \beta \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ 0 & T_{22} & T_{23} \\ 0 & 0 & \tilde{Y}_2 \end{bmatrix} \right) \begin{bmatrix} z_1 \\ z_2 \\ \tilde{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \quad (5.41)$$

Rearranging (5.41) we obtain three linear systems,

$$(\alpha R_{11} + \beta T_{11})z_1 = -((\alpha R_{13} + \beta T_{13})\tilde{z} + (\alpha R_{12} + \beta T_{12})z_2) \quad (5.42)$$

$$(\alpha R_{22} + \beta T_{22})z_2 = -(\alpha R_{23} + \beta T_{23})\tilde{z} \quad (5.43)$$

$$(\alpha\tilde{X} + \beta\tilde{Y})\tilde{z} = 0. \quad (5.44)$$

After solving the GEP, we determine an eigenpair $(\tilde{z}; \lambda)$ of (5.44). To recover an eigenvector of the full pencil we need to determine z_1 and z_2 , to do this we first solve (5.43) for z_2 , then on substituting z_2 into (5.42) we obtain z_1 , we can then form the vector z and have an eigenpair of $L_2(\alpha, \beta)$.

We now give a brief note about reapplying the accumulated Householder reflectors back to an eigenvector of $L_2(\lambda)$ to obtain an eigenvector of the first companion linearization. We work with eigenpairs $(z; \lambda)$ of the pencil

$$L_2(\lambda) = H_i^T H_z^T C_1(\lambda) H_z K_i,$$

therefore if $(z; \lambda)$ satisfies $L_2(\lambda)z = 0$ then $(H_z K_i z; \lambda)$ satisfies $C_1(\lambda) H_z K_i z = 0$. When forming $H_z K_i z$ we first apply \tilde{K}_i to elements $(k_0 + k_2)$ onwards of z , then we apply \tilde{H}_z to elements k_0 onwards of $\tilde{K}_i z$. We then obtain our eigenpair $(H_z K_i z; \lambda)$ of the first companion linearization.

5.3.4 Householder Deflation Methods and Structured Linearizations

We note that when we extend the Householder methods to the structured (symmetric) linearizations outlined in Chapter 2 they do not preserve the symmetry of the linearization, so there is no advantage in such an approach compared with the methods using Householder reflectors we have described.

5.4 Computational Cost of Methods

We now present the cost of our algorithms in solving the QEP with a quadratic with singular leading or trailing coefficients, for simplification we assume that all the eigenvalues are real and that only right eigenvectors are computed, $W(\text{UTV})$ denotes the amount of work in computing a UTV factorization (as described in Section 4, while $W(\alpha)$ denotes the cost of determining a value of α such that $Q(\alpha)$ is well-conditioned. Tables 5.2, 5.1, 5.4, and 5.3 summarize the costs of our algorithms for various steps, we make the following observations in comparing the costs of the algorithms

- After a UTV factorization has been computed, the work for the simplified Kublanovskaya method does not depend on the rank of the matrix, whereas

for the approach based on deflating zero (or infinite) eigenvalues by the use of Householder reflectors, a lower rank increases the cost of the algorithm.

- When deflating zero (or infinite) eigenvalues with Householder reflectors, if a basis for the null space is known it can be supplied to the algorithm, in which case a UTV factorization is not required and we save on cost. However, for the simplified Kublanovskaya approach it is not possible to save on cost if a basis for the null space is known, a UTV factorization is always required.
- When we compare the costs of the two algorithms for quadratics with one singular leading or trailing coefficient and ignore the cost of the processes common to both, we find that the Householder approach requires $6n^3$ more operations than the simplified Kublanovskaya approach.
- The amount of work required to implement the modified original method of Kublanovskaya et al is influenced by the need to apply the inverse of a matrix after a suitable value for α has been determined, the best method for determining a value for α is an open problem.

Table 5.1: Cost of stages of simplified Kublanovskaya approach.

Process	Cost (flops)
UTV factorization of A_0 coefficient matrix	$W(\text{UTV})$
Apply transformation K to the first companion linearization (form product A_0Q_0)	n^3
QZ algorithm applied to deflated pencil	$50(n + r_0)^3$
Recover all $2n$ eigenvectors by back substitution	$(2n)^3$
Reapply Q transformation from QZ algorithm to eigenvectors	$2(2n)^3$
Reapply transformation K to eigenvectors to recover eigenvectors of the first companion linearization	$4n^3$
Total = $W(\text{UTV}) + 80n^3 + 150r_0n^2 + 150r_0^2n + r_0^3$	

Table 5.2: Cost of stages of modified original Kublanovskaya approach.

Process	Cost (flops)
UTV factorization of leading and trailing coefficient matrices	$2 \times W(UTV)$
Determine α such that $Q(\alpha)$ is well conditioned	$W(\alpha)$
Determine $Q(\alpha)^{-1}$ (GEPP)	$2n^3$
Apply the inverse of $Q(\alpha)$ to leading and trailing coefficients	$2n^3$
Carry out similarity transformation	$6n^3$
Apply permutation matrix	$(2n)^3$
Apply QR method to deflated problem SEP	$35(r_0 + r_2)^3$
Recover eigenvectors by back substitution	$(r_0 + r_2)(2n - r_0 - r_2)^2 + (r_0 + r_2)^3$
Reapply Q from QR to eigenvectors	$(r_0 + r_2)^3$
Reapply permutation matrix to eigenvectors	$(2n)^3$
Reapply transformation matrix V to eigenvectors	$6n^3$
<hr/>	
Total = $2 \times W(UTV) + 30n^3 + 4(r_0 + r_2)n^2 + 4(r_0 + r_2)^2n + 38(r_0 + r_2)^3$	

Table 5.3: Cost of stages of Householder approach for zero eigenvalues (for deflating infinite eigenvalues only, use the reversal polynomial and deflate the corresponding zero eigenvalues).

Process	Cost (flops)
UTV factorization of leading or trailing coefficient matrices for null space basis	$W(\text{UTV})$
Determine Householder reflectors	$3(2n(n - r_0) - (n - r_0)^2/2)$
Update null space basis (apply Householder reflectors)	$(4n^2(n - r_0) - 2n(n - r_0)^2 + (n - r_0)^3/3)$
Apply Householder reflectors to first companion linearization	$16(4n^2(n - r_0) - 2n(n - r_0)^2 + (n - r_0)^3/3)$
QZ algorithm applied to deflated pencil	$46(n + r_0)^3$
Recover eigenvectors by back substitution	$(2n)^3$
Apply Q from QZ to eigenvectors	$2n(n + r_0)^2$
Reapply Householder reflectors to eigenvectors	$8(2n^2(n - r_0) - n(n - r_0)^2/2)$
$\text{Total} = W(\text{UTV}) + (316/3)n^3 + n^2(9/2 + 118r_0) + n(120r_0^2 - 3r_0) + (122r_0^3/3 - 3r_0^2/2)$	

Table 5.4: Cost of stages of Householder approach for zero and infinite eigenvalues.

Process	Cost (flops)
UTV factorization of leading and trailing coefficient matrices for null space basis	$2 \times W(UTV)$
Determine Householder reflectors for infinite and zero deflation	$2(3n(n - r_0)^2 - (2n - r_0 - r_2)^3) + 3(2n(n - r_0) - (n - r_0)^2/2)$
Matrix vector product ($x = \tilde{Y}z$) before computing Householder reflector	$(n + r_0)^2(n - r_2) - (n - r_2)^2(n + r_0) + (n - r_2)^3/3$
Apply Householder reflectors to null space basis of A_0	$4(4n^2(n - r_0) - 2n(n - r_0)^2 + (n - r_0)^3/3)$
Apply Householder reflectors to null space basis of A_2	$n^3/3 + n^2(-16r_2 - 15r_0) + nr_0^2 + (-r_0 - r_2)^3 + r_0^3/3$
Apply Householder reflectors (for infinite and zero eigenvalue deflation) to first companion linearization	$8n^3/3 + 16(-r_0 - r_2)n^2 + (-r_0 - r_2)^3$
QZ algorithm applied to deflated pencil	$46(r_0 + r_2)^2$
Recover eigenvectors by back substitution	$(2n)^3$
Reapply Q from QZ to eigenvectors	$(r_0 + r_2)(2n)^2$
Reapply Householder reflectors to eigenvectors	$8(4n^2(2n - r_0 - r_2) - n(2n - r_0 - r_2)^2)$
$\text{Total} = W(UTV) + (128/3)n^3 + n^2(-69r_0 - 65r_2 - 15/2) + n(r_2 - 22(-r_0 - r_2)^2) + (-3r_0^2/2 - r_0^3 - r_2^2/3 - r_0r_2(r_0 + r_2) + 46(r_0 + r_2)^3)$	

Chapter 6

Applications and Test Problems

In many of the test problems we will present, a QEP results in part of the dynamic analysis of structures discretized by the finite element method. The equations of motion are of the form

$$M\ddot{q}(t) + C\dot{q}(t) + Kq(t) = f(t), \quad (6.1)$$

where M , C , and K are n -by- n matrices, $q(t)$ and $f(t)$ are n -dimensional vectors. M is called the mass matrix, C the damping matrix and K the stiffness matrix, $f(t)$ is a forcing term. The first step of a vibrational or dynamic analysis is the solution of the homogeneous equation, setting $f(t) = 0$ in (6.1). To do this it is necessary to obtain eigenvalues λ and eigenvectors x of the QEP

$$(\lambda^2 M + \lambda C + K)x = 0,$$

more details and examples can be found in [31].

We now present a number of test problems that result in a quadratic with infinite or zero eigenvalues. Table 6.1 lists the problems, their dimension and the rank of the leading and trailing coefficients.

6.1 Test Quadratics

We now describe applications that yield a QEP whose quadratic has singular leading or trailing coefficients, we make use of the MATLAB `spy` function when we describe

Table 6.1: Properties of test problems.

Test problem	n	$\text{rank}(A_0)$	$\text{rank}(A_2)$
Speaker box	107	106	full
Mobile manipulator	5	full	3
Railtrack problem	1005	67	67
Shaft on bearing support	400	full	199
Spring dashpot	10	full	2

Table 6.2: Effect of scaling on norms of test problem coefficients.

Test problem	$\ A_0\ _2$	$\ A_1\ _2$	$\ A_2\ _2$
Speaker box (unscaled)	9.95e+6	5.74e-2	1.00
Speaker box (scaled)	2.00	3.64e-5	2.00
Spring dashpot (unscaled)	3.75	1.13	1.04e+4
Spring dashpot (scaled)	1.99	1.13e-2	1.99
Mobile manipulator (unscaled)	1.30e+2	8.43	5.94e+1
Mobile manipulator (scaled)	1.82	1.75e-1	1.82
Shaft (unscaled)	1.81e+9	8.00e-3	2.71e-3
Shaft (scaled)	2.00	7.23e-6	2.00
Railtrack (unscaled)	2.67e+10	1.29e+11	2.67e+10
Railtrack (scaled)	3.44e-1	1.66	3.44e-1

the sparsity pattern of matrix coefficients, it puts a dot where the entry of a matrix is nonzero.

6.1.1 Speaker Box

The quadratic $Q(\lambda) = \lambda^2 M + \lambda C + K$, with $M, C, K \in \mathbb{R}^{107 \times 107}$, is from the finite element package MSC/Nastran. The matrices come from a finite element model of a speaker box that includes both structural finite elements representing the box, and finite elements representing the air contained in the box. The mass matrix M has rank 106 and contributes a zero eigenvalue and the matrix coefficients are highly structured and sparse, see Figure 6.1. Figure 6.2 on the next page shows the finite element representation of the speaker box. Further discussion of this model can be found in [18].

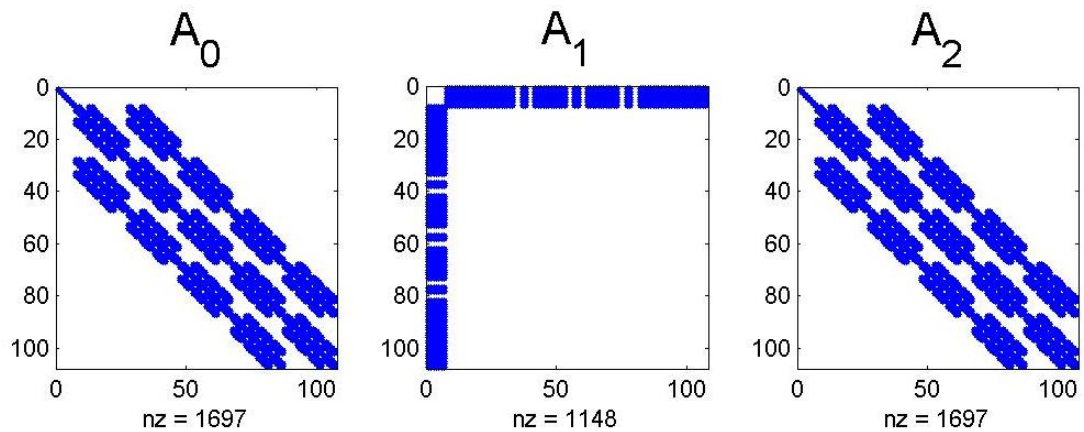


Figure 6.1: Spyplo of matrix coefficients of quadratic for speaker box example.

6.1.2 Linear Spring and Dashpot

Gotts [16] describes a QEP arising from a finite element model of a linear spring in parallel with Maxwell elements (a Maxwell element is a spring in series with a dashpot), for a diagram see Figure 6.3. The QEP is of the form $Q(\lambda)x = 0$ with

$$Q(\lambda) = \lambda^2 M + \lambda D + K, \quad M, D, K \in \mathbb{R}^{10 \times 10}$$

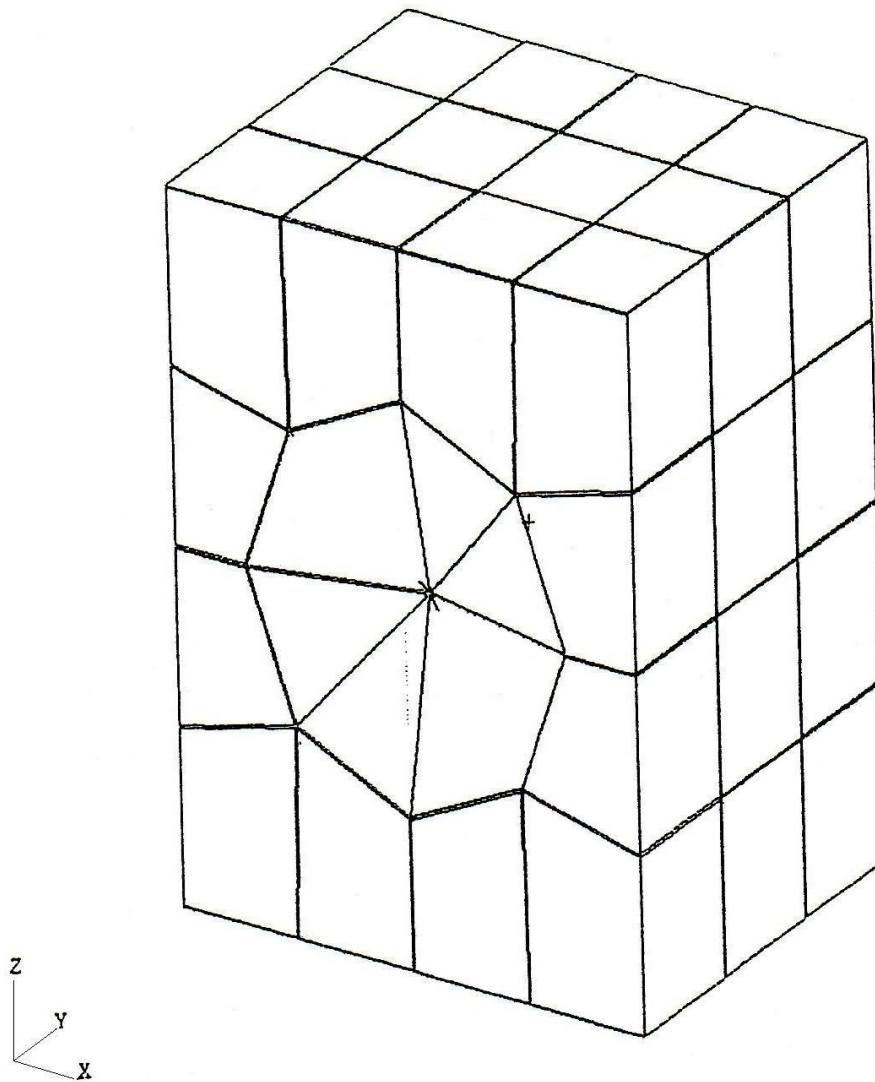


Figure 6.2: Finite element representation of a speaker box.

where the mass matrix M is rank deficient and symmetric, D the damping matrix is rank deficient and block diagonal, and K the stiffness matrix is symmetric and exhibits “arrowhead” structure. Gotts suggests that there are normally at least 4 Maxwell elements, we generated an example that reflects only the structural properties such as symmetry, 2-by-2 element mass and stiffness matrices are randomly generated to be symmetric positive definite. We give the form of the matrix for 4 Maxwell elements below

$$M = \begin{bmatrix} \rho \tilde{M}_{11} & 0 \\ 0 & 0 \end{bmatrix} \quad (6.2)$$

$$D = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & \eta_1 \tilde{K}_{11} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \eta_4 \tilde{K}_{55} \end{bmatrix} \quad (6.3)$$

$$K = \begin{bmatrix} \alpha_\rho \tilde{K}_{11} & & B & \\ & e_1 \tilde{K}_{22} & 0 & 0 \\ B^T & 0 & \ddots & 0 \\ & 0 & 0 & e_4 \tilde{K}_{55} \end{bmatrix} \quad (6.4)$$

where

$$B = \left[-\xi_1 \tilde{K}_{12}, \quad \dots, \quad -\xi_4 \tilde{K}_{15} \right].$$

\tilde{M}_{ij} and \tilde{K}_{ij} are the ij th element mass and stiffness matrices, and

$$\alpha_\rho = \sum_{k=0}^4 \xi_k.$$

η_i , $i = 1:5$, ξ_j $j = 0:5$, e_k , $k = 1:4$ and ρ (the material density) are scalar parameters.

6.1.3 Mobile Manipulator Problem

This QEP results from the modeling of a time-invariant descriptor control system of a two-dimensional three link mobile manipulator [3], [2]. Figure 6.4 shows the mobile manipulator. The quadratic is of the form

$$Q(\lambda) = \lambda^2 M + \lambda D + K, \quad M, D, K \in \mathbb{R}^{5 \times 5},$$

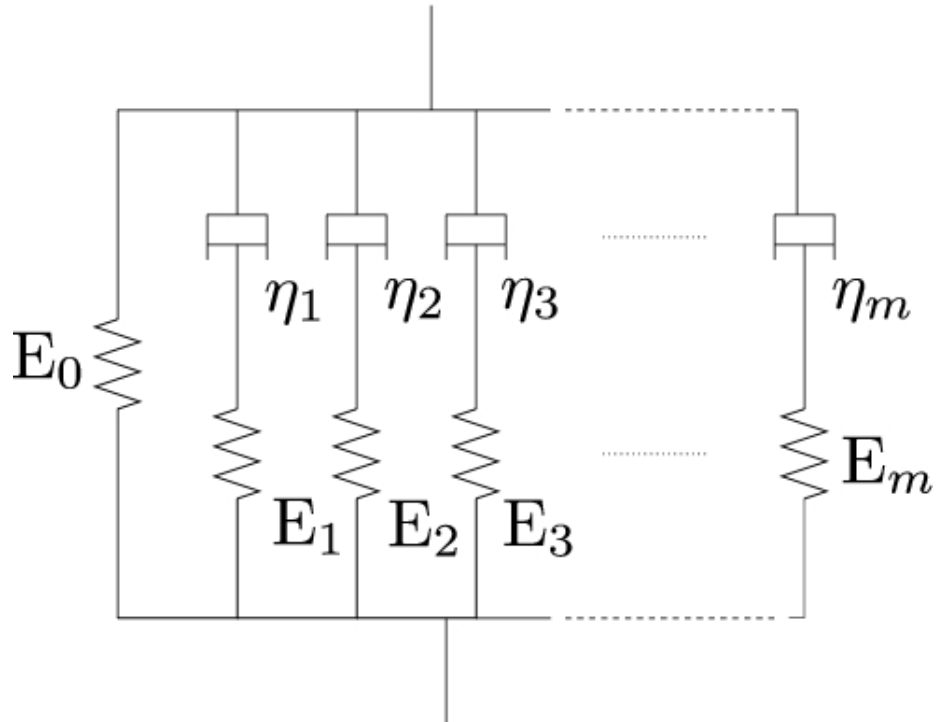


Figure 6.3: Spring/dashpot with Maxwell elements.

where,

$$M = \begin{bmatrix} M_0 & 0 \\ 0 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} D_0 & 0 \\ 0 & 0 \end{bmatrix}, \quad K = \begin{bmatrix} K_0 & -F_0 \\ F_0 & 0 \end{bmatrix}.$$

M and D are rank deficient, and in particular M contributes two infinite eigenvalues. M_0, D_0 , and $K_0 \in \mathbb{R}^{3 \times 3}$ with M_0 nonsingular, and $F_0 \in \mathbb{R}^{2 \times 3}$. The quadratic in this case is close to being nonregular [3, 22].

6.1.4 Shaft on Bearing Support

The QEP $Q(\lambda) = \lambda^2 M + \lambda C + K$, with $M, C, K \in \mathbb{R}^{400 \times 400}$, (also from the package MSC/Nastran) comes from a finite element model of a shaft on bearing supports with a damper. The rank of M is 199 and contributes a large number of infinite eigenvalues, the quadratic matrix coefficients are very sparse in this example, see Figure 6.5. Figure 6.6 on page 67 shows a schematic of a shaft on bearing support.

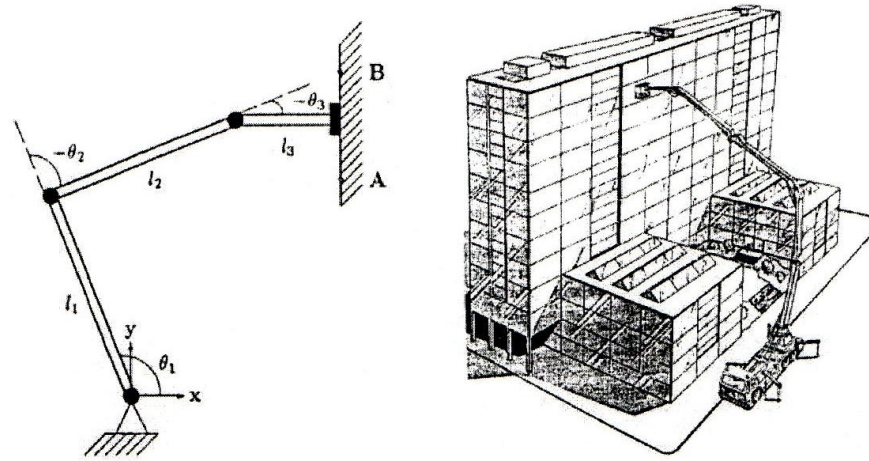


Figure 6.4: Three link mobile manipulator.

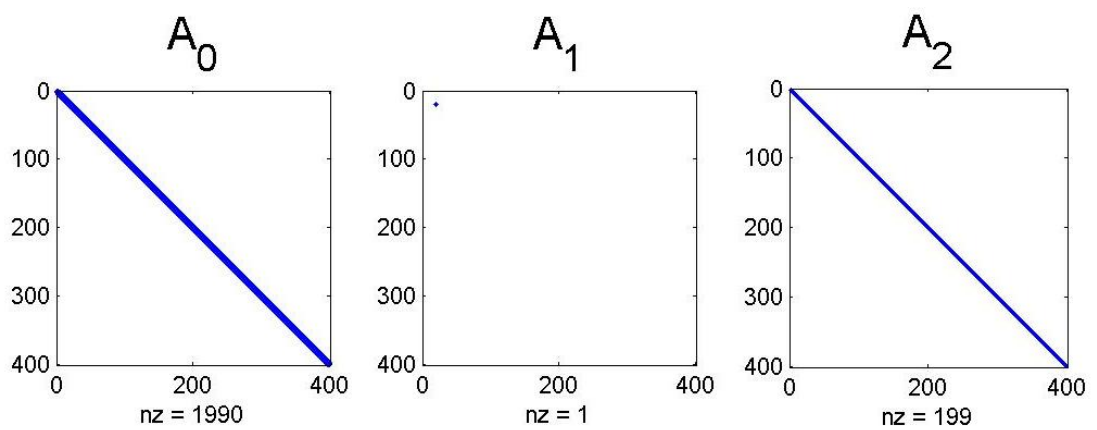


Figure 6.5: Spyplot of matrix coefficients of quadratic for shaft example.

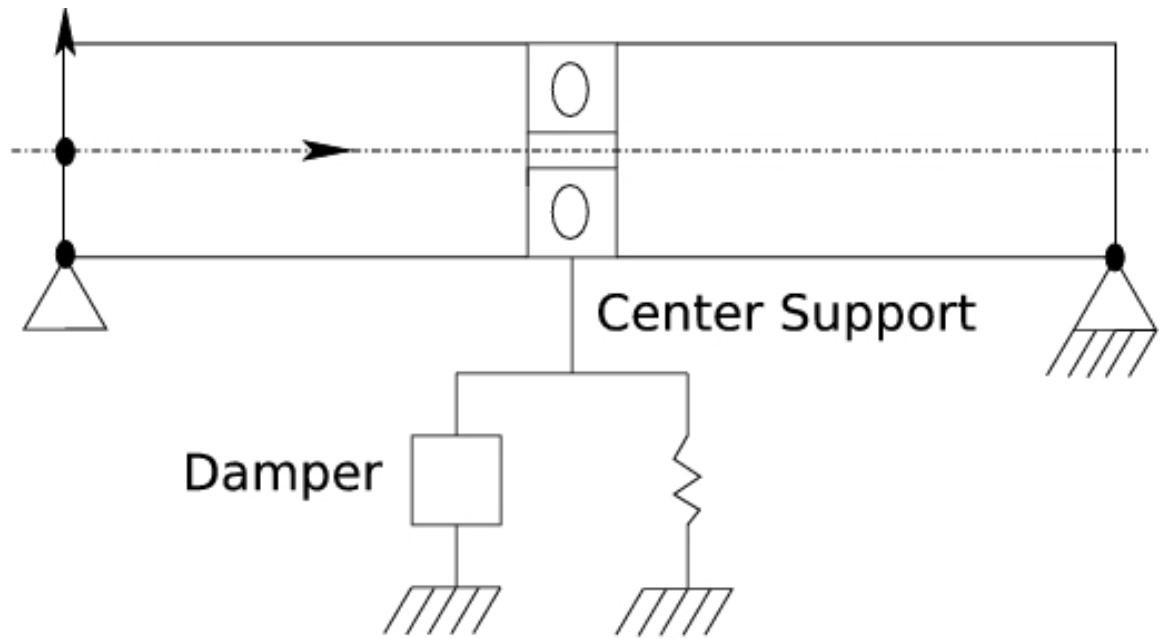


Figure 6.6: Schematic of a shaft on bearing support.

6.1.5 Railtrack Problem

Hilliges, Mehl and Mehrmann [24] consider the eigenproblem

$$\frac{1}{\kappa}(A_1^T + \kappa A_0 + \kappa^2 A_1)y = 0$$

where $A_0, A_1 \in \mathbb{C}^{1005 \times 1005}$ and $A_0 = A_0^T$ (A_0 is complex symmetric). This is a *palindromic eigenvalue problem* since starting with the original quadratic that we denote as $P(\kappa) = \kappa^2 A_1 + \kappa A_0 + A_1^T$ and transposing we have

$$P(\kappa)^T = \kappa^2 A_1^T + \kappa A_0 + A_1,$$

finally taking the reversal polynomial of $P(\kappa)^T$ we have

$$\text{rev}(P(\kappa)^T) = \kappa^2 A_1 + \kappa A_0 + A_1^T$$

which is the same as the original quadratic. The polynomial is more accurately called *T-palindromic* (T for transpose), since we took the transpose. This eigenproblem results from a vibrational analysis of rail tracks under periodic excitation using finite element analysis. The quadratic has the property that there are many infinite and zero eigenvalues contributed by the leading and trailing coefficient, since A_1 has rank 67 and is therefore highly rank-deficient. To preserve the spectral properties of the eigenproblem it is desirable to preserve the palindromic nature of the original QEP when choosing a linearization. Therefore *structure preserving* linearizations are used (see Section 2.1 on page 20 for some details of structure preserving linearizations).

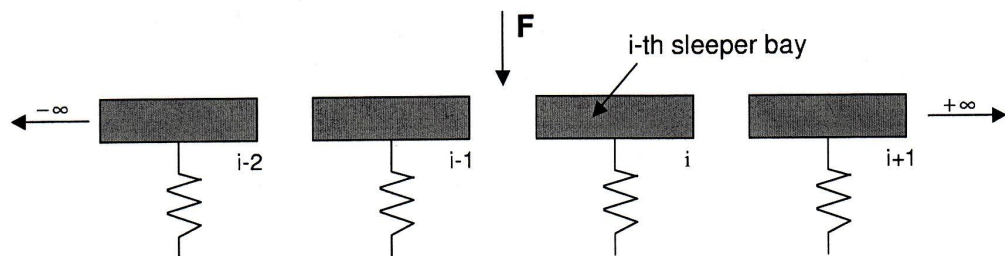


Figure 6.7: Model of the rail.

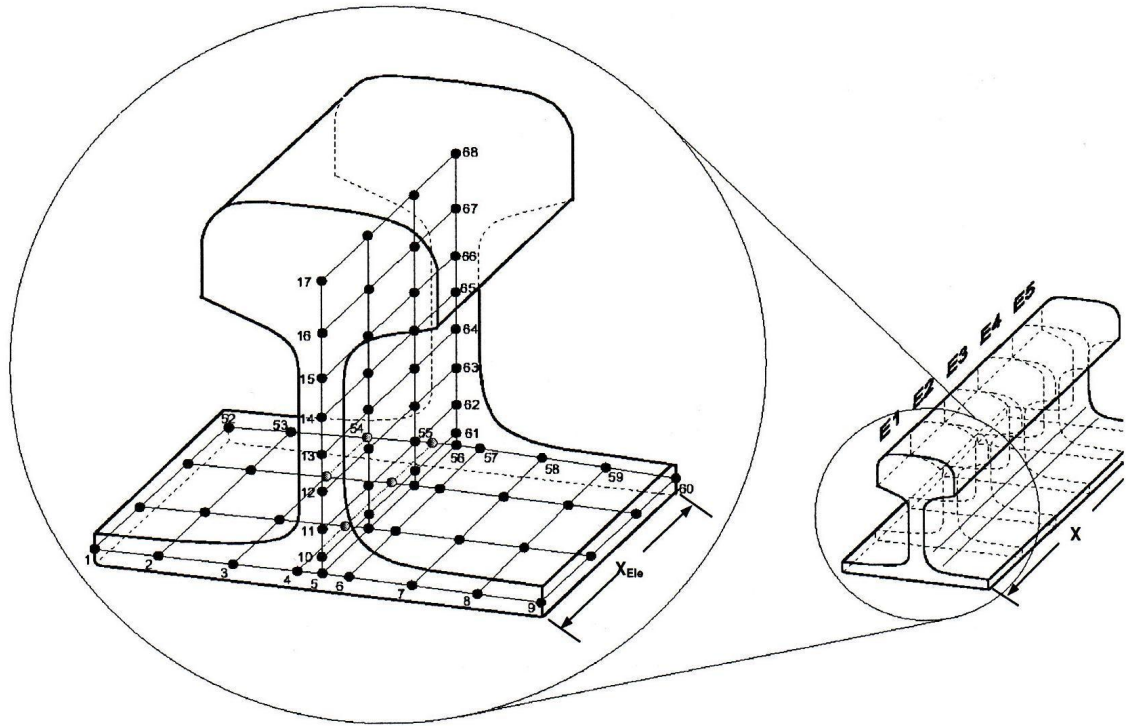


Figure 6.8: FEM representation of the rail in one sleeper bay.

6.1.6 Randomly Generated Quadratics with Singular Leading and Trailing Coefficient Matrices

In order to test the algorithms that deflate both zero and infinite eigenvalues when the leading and trailing coefficient matrices are singular, we generate our own quadratics with specified rank. The railtrack example has leading and trailing coefficients which are singular but the coefficient matrices have dimension 1007-by-1007 which is too large to use for our algorithms in a reasonable amount of time.

We use the three methods below to generate random test problems:

- Embed a $\text{randn}(r, r)$ matrix into an n -by- n zero matrix.
- Generate a random matrix \tilde{A} , compute its SVD, $\tilde{A} = U\Sigma V^T$, and let $\tilde{\Sigma}$ to be the matrix Σ with the last $n - r$ diagonal entries of Σ set to zero. Then compute A by forming $A = U\tilde{\Sigma}V^T$.
- Use a block outer product, then $A = \text{randn}(n, r) * \text{randn}(n, r)^T$.

To generate quadratics with leading and trailing coefficients that are singular with specified rank we use the following method and call the quadratics `qrnkdef1`. To mirror the structure of the matrices in the applications we have described, to generate a coefficient of rank r for a quadratic whose matrix coefficients are n -by- n we embed into an n -by- n zero matrix an r -by- r matrix with elements drawn from a normal distribution with mean zero and standard deviation one, we optionally apply column and row permutation so that the matrices are less structured.

6.2 Initial Scaling of the Quadratic Matrix Polynomial

When we obtain a quadratic from practical application, it is possible that the coefficient matrices are ‘badly scaled’. Fan, Lin and Van Dooren [10], present a method of scaling the matrix polynomial $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$, to bring the norms of the coefficients close to one. The result is the scaled matrix polynomial $\tilde{Q}(\mu) = \mu^2(A_2 \xi^2 \delta) + \mu(A_1 \xi \delta) + (A_0 \delta)$, where if

$$\gamma_2 = \|A_2\|_2, \quad \gamma_1 = \|A_1\|_2, \quad \gamma_0 = \|A_0\|_2,$$

then the choices of

$$\xi = \sqrt{\gamma_0/\gamma_2}, \quad \text{and} \quad \delta = \frac{2}{\gamma_0 + \gamma_1 \sqrt{\gamma_0/\gamma_2}},$$

result in the scaled matrix quadratic $\tilde{Q}(\mu)$ having norm close to one. The work [21] illustrates the benefit of the above scaling for a model QEP describing the motion of a beam simply supported at both ends, with damping at the midpoint. We now summarize some results from [20], regarding the effect of the scaling presented above on the conditioning of the quadratic and linearization.

Theorem 5 (see Theorem 5.1 [20]). *Let λ be a simple eigenvalue of $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$, or of the scaled quadratic $\tilde{Q}(\mu) = \mu^2 \tilde{A}_2 + \mu \tilde{A}_1 + \tilde{A}_0$ (scaled by the method of Fan, Lin and Van Dooren above). Assume that either*

- $\|A_1\|_2 \lesssim \max(\|A_0\|_2, \|A_2\|_2)$ and $\|A_2\|_2 \approx \|A_0\|_2$ then solve the unscaled QEP via linearization, taking a linearization contained in the vector space $\mathbb{DL}(Q)$
- $\|A_1\|_2 \lesssim \sqrt{\|A_0\|_2 \|A_2\|_2}$ and then scale the QEP using the Fan, Lin and Van Dooren scaling. Choose a linearization of the scaled QEP, that is contained in the vector space $\mathbb{DL}(\tilde{Q})$.

Then if A_2 is nonsingular and $|\lambda| \geq 1$, or A_0 is nonsingular and $|\lambda| \leq 1$, then it is possible to choose a linearization of the QEP so that the conditioning is near-optimal.

In applications, Theorem 5 is valid for QEPs resulting from mechanical systems with damping, that are not too heavily damped. In this thesis we focus on the first companion linearization, for the companion linearizations Section 7 of [20] gives results guaranteeing optimal conditioning, however a priori the only easily checkable condition presented is that $\|A_i\|_2 \approx 1$, $i = 0 : 2$.

6.3 Conditioning of Eigenvalues of Quadratic Test Problems

We have computed the condition numbers of eigenvalues of the quadratic and first companion linearization, with and without the scaling of Fan, Lin and Van Dooren [10]. Bounds based on the maximum and minimum values of eigenvalues condition numbers for the quadratic and linearization are summarized in Table 6.3. We note the following points with regard to the effect of the scaling on the eigenvalues condition numbers, in particular we note whether the infinite and zero eigenvalues we are deflating are well or ill-conditioned:

- **Speaker box:** Scaling the quadratic is important in this example, if we do not scale then a zero eigenvalue contributed by the A_1 coefficient is not exactly zero (10^{-8}), however when scaling is applied the zero eigenvalue is correctly returned.

Table 6.3: Effect of scaling on eigenvalue condition number for quadratic and companion linearization, for test problem quadratics.

		$\lambda = \{0, \infty\}$	λ Finite nonzero
Speaker box	Unscaled	$10^{18} \leq \kappa_Q(\alpha, \beta) \leq 10^{21}$	$10^3 \leq \kappa_Q(\alpha, \beta) \leq 10^{21}$
	Scaled	$10^{11} \leq \kappa_Q(\alpha, \beta) \leq 10^{11}$	$10^6 \leq \kappa_Q(\alpha, \beta) \leq 10^{11}$
	Unscaled	$10^{11} \leq \kappa_L(\alpha, \beta) \leq 10^{16}$	$10^6 \leq \kappa_L(\alpha, \beta) \leq 10^{16}$
	Scaled	$10^1 \leq \kappa_L(\alpha, \beta) \leq 10^1$	$10^1 \leq \kappa_L(\alpha, \beta) \leq 10^9$
Spring dashpot	Unscaled	$10^4 \leq \kappa_Q(\alpha, \beta) \leq 10^6$	$10^{-1} \leq \kappa_Q(\alpha, \beta) \leq 10^5$
	Scaled	$10^2 \leq \kappa_Q(\alpha, \beta) \leq 10^5$	$10^1 \leq \kappa_Q(\alpha, \beta) \leq 10^3$
	Unscaled	$10^4 \leq \kappa_L(\alpha, \beta) \leq 10^6$	$10^2 \leq \kappa_L(\alpha, \beta) \leq 10^5$
	Scaled	$10^3 \leq \kappa_L(\alpha, \beta) \leq 10^5$	$10^1 \leq \kappa_L(\alpha, \beta) \leq 10^4$
Mobile manipulator	Unscaled	$10^{62} \leq \kappa_Q(\alpha, \beta) \leq \infty$	$10^4 \leq \kappa_Q(\alpha, \beta) \leq 10^4$
	Scaled	$10^{61} \leq \kappa_Q(\alpha, \beta) \leq \infty$	$10^4 \leq \kappa_Q(\alpha, \beta) \leq 10^4$
	Unscaled	$10^{31} \leq \kappa_L(\alpha, \beta) \leq 10^{32}$	$10^4 \leq \kappa_L(\alpha, \beta) \leq 10^4$
	Scaled	$10^{31} \leq \kappa_L(\alpha, \beta) \leq 10^{32}$	$10^4 \leq \kappa_L(\alpha, \beta) \leq 10^4$
Shaft	Unscaled	$10^{54} \leq \kappa_Q(\alpha, \beta) \leq \infty$	$10^{-6} \leq \kappa_Q(\alpha, \beta) \leq 10^8$
	Scaled	$10^{69} \leq \kappa_Q(\alpha, \beta) \leq \infty$	$10^0 \leq \kappa_Q(\alpha, \beta) \leq 10^6$
	Unscaled	$10^8 \leq \kappa_L(\alpha, \beta) \leq 10^{33}$	$10^3 \leq \kappa_L(\alpha, \beta) \leq 10^{10}$
	Scaled	$10^{17} \leq \kappa_L(\alpha, \beta) \leq 10^{33}$	$10^1 \leq \kappa_L(\alpha, \beta) \leq 10^6$

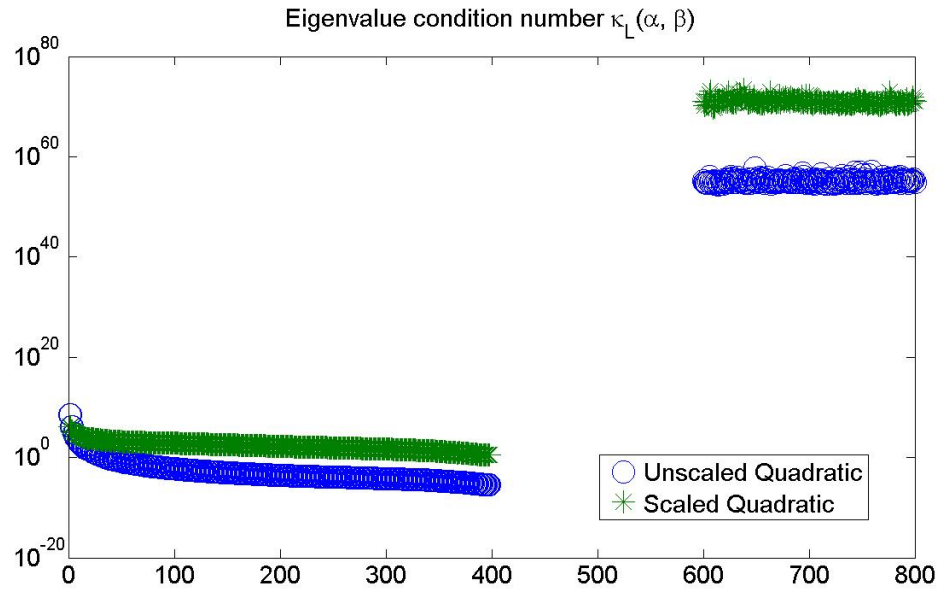


Figure 6.9: Effect of scaling on eigenvalue condition number of the quadratic for eigenvalues of the shaft problem.

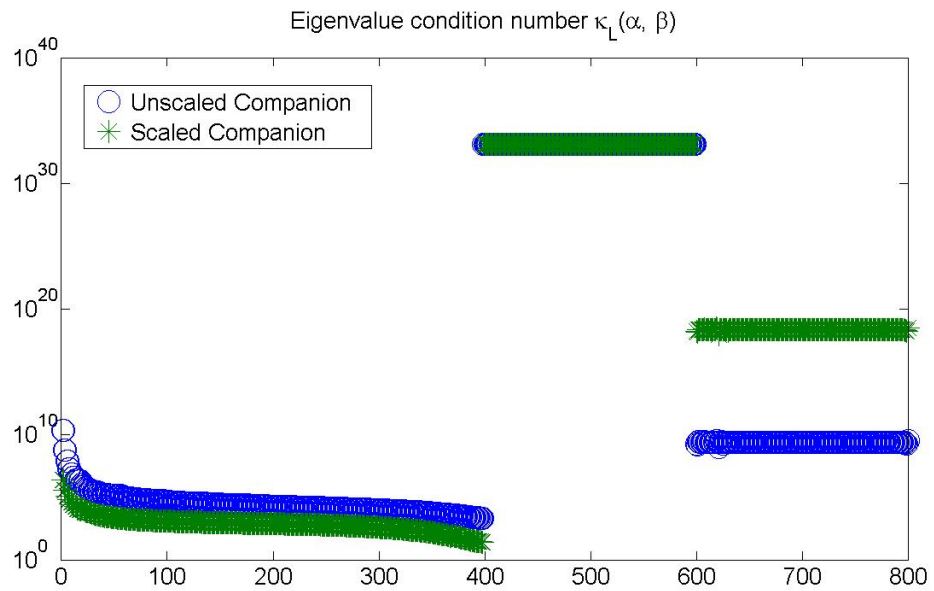


Figure 6.10: Effect of scaling on eigenvalue condition number of the first companion linearization for eigenvalues of the shaft problem.

- **Spring dashpot:** Both the infinite and finite eigenvalues are reasonably well conditioned in this case, scaling tends to reduce the condition number of eigenvalues in this example.
- **Mobile manipulator problem:** The infinite eigenvalues are particularly ill-conditioned, indicating that the infinite eigenvalues are, or are close to being defective. The finite nonzero eigenvalues (in this case one complex conjugate pair) are relatively well conditioned being of the order 10^4 .
- **Shaft:** Scaling slightly increases the condition number of finite eigenvalues for the quadratic, but decreases the condition number of finite eigenvalues for the linearization, as seen in Figure 6.9 for the quadratic, and 6.10 for the linearization, however some of the eigenvalue condition numbers are still quite large, they range from 10^1 to 10^6 when scaling is applied, compared to 10^3 to 10^{10} , so scaling is clearly desirable in this case. The infinite eigenvalues are all relatively ill-conditioned, the condition numbers are all greater than 10^8 , and for those infinite eigenvalues between 600-800 their respective condition number is increased by scaling, but the eigenvalue condition number of infinite eigenvalues between 400-600 is very large regardless of scaling the quadratic.

6.4 Testing the Algorithms

We now present results of testing the algorithms on some of the test problems previously described (we omit the railtrack example due to its large size), we also use quadratics whose coefficients have been generated to have a specified rank. We measure the backward error of eigenpairs of the quadratic and of eigenpairs of the deflated pencil. By deflated pencil we mean the pencil after a deflation procedure has been carried out. When solving the QEPs, we implement the scaling of Fan, Lin and Van Dooren [10] as described in this chapter. In our figures, the x-axis is the eigenvalue index and the eigenvalues are sorted in order of increasing magnitude (via the MATLAB `sort` function). For those quadratics with highly structured matrix

coefficients, it is possible that the backward error of eigenpairs for the quadratic is exactly zero, since we use a log axis with MATLAB's `semilogy` command such zero values are not plotted, this explains any gaps in the plots.

6.4.1 Householder Deflation of Zero or Infinite Eigenvalues

When testing the Householder deflation approach we fix the choice of null space approximation, using the SVD to give a basis for the null space, since it is the most stable method. The routine gives good (small) backward errors for eigenpairs of the quadratic on the test problems (speaker box, mobile manipulator, spring dashpot and shaft). We note that backward errors of eigenpairs with finite nonzero eigenvalues (or those zero or infinite eigenvalues that were contributed by A_1 and not deflated) are usually of a similar order as those of the linearization. However, for those eigenpairs of the quadratic with zero or infinite eigenvalues that were deflated, we can see much smaller (possibly zero) backward errors for the quadratic than we might expect. Figure 6.11 illustrates the case of the speaker box where a single eigenvalue contributed by the A_0 matrix coefficient is deflated. We can explain this by the fact that we are able to determine a basis for the null space of the singular matrix coefficient very accurately (such a basis is often spanned by unit vectors), hence we can accurately deflate infinite and zero eigenvalues.

6.4.2 Deflation of Zero or Infinite Eigenvalues using Simplified Kublanovskaya Method

We test the MATLAB implementation of the simplified Kublanovskaya method on the test problems with singular leading or trailing coefficients and note that backward errors for all eigenpairs with nondefective zero or infinite eigenvalues are small, somewhere in the region of the machine precision, the shaft problem for example, in Figure 6.13. However, when the quadratic has defective zero or infinite eigenvalues (the speaker box and mobile manipulator examples), the backward error can be large, as seen in Figure 6.12. It is known that the quadratic in the mobile manipulator is

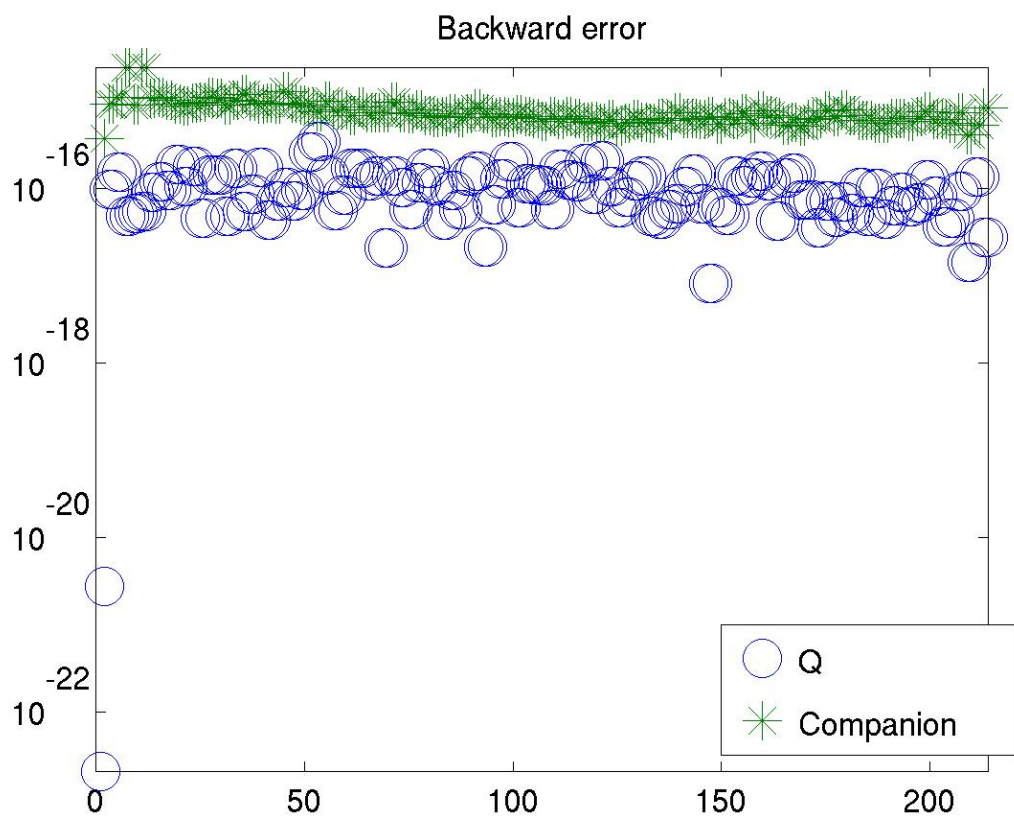


Figure 6.11: Backward error for quadratic and companion linearization for the speaker box problem where $A_i \in \mathbb{R}^{107 \times 107}$ with $\text{rank}(A_0) = 106$ (1 zero eigenvalue contributed by A_0 coefficient deflated via Householder reflectors)

close to being nonregular, which may explain our results, however the speaker box quadratic is regular, hence, further work is required to determine whether it is the method or the implementation that yields this instability.

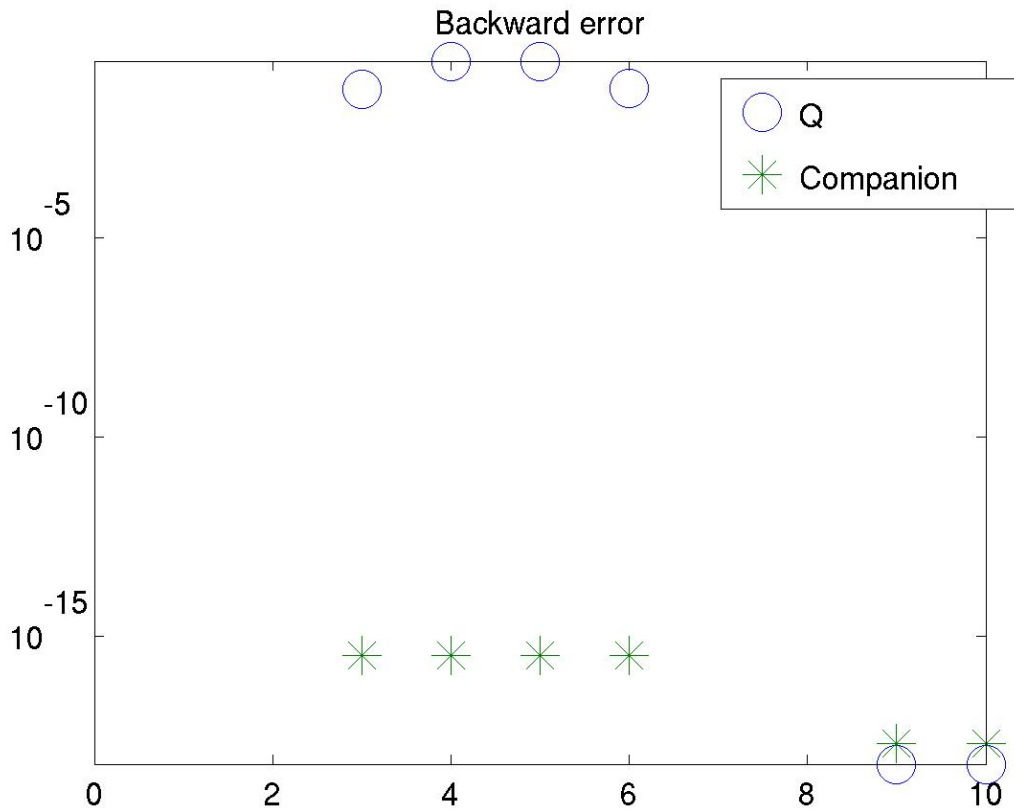


Figure 6.12: Backward error for quadratic and companion linearization for the mobile manipulator problem where $A_i \in \mathbb{R}^{5 \times 5}$ with $\text{rank}(A_2) = 3$ (2 infinite eigenvalues contributed by A_2 coefficient deflated via simplified Kublanovskaya method)

6.4.3 Householder Deflation of Zero and Infinite Eigenvalues

For the random quadratics we have generated, the backward error of quadratic eigenpairs are small, at most of the order 10^{-14} , when both leading and trailing coefficients are singular. The backward errors are relatively near to the machine precision in this case, for example Figure 6.14, using the block outer-product method to generate singular coefficients.

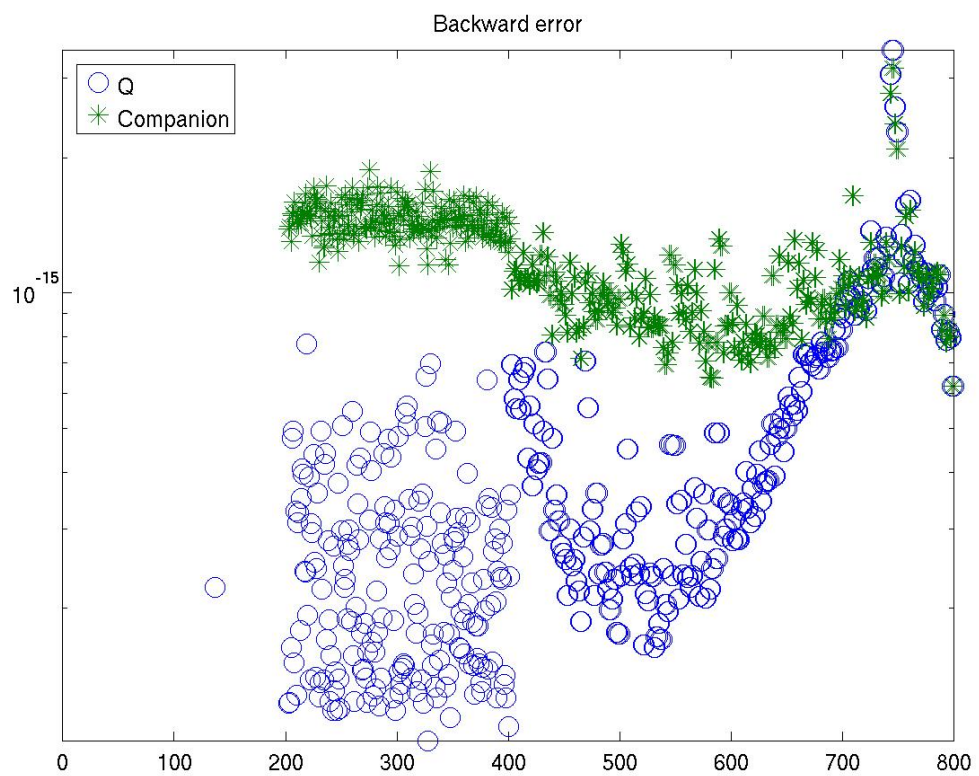


Figure 6.13: Backward error for quadratic and companion linearization for the shaft problem where $A_i \in \mathbb{R}^{400 \times 400}$ with $\text{rank}(A_2) = 199$ (201 infinite eigenvalues contributed by A_2 coefficient deflated), deflation by simplified Kublanovskaya method

6.4.4 Deflation of Zero and Infinite Eigenvalues using Modified original Kublanovskaya Method

The backward errors for quadratic eigenpairs using the simplified Kublanovskaya method are at most of the order 10^{-13} , thus, slightly larger than using Householder reflectors, this can be attributed to the need to apply the inverse of the matrix $Q(\alpha)$. Figure 6.15 shows the results for a quadratic with singular coefficients generated by forming a block outer-product.

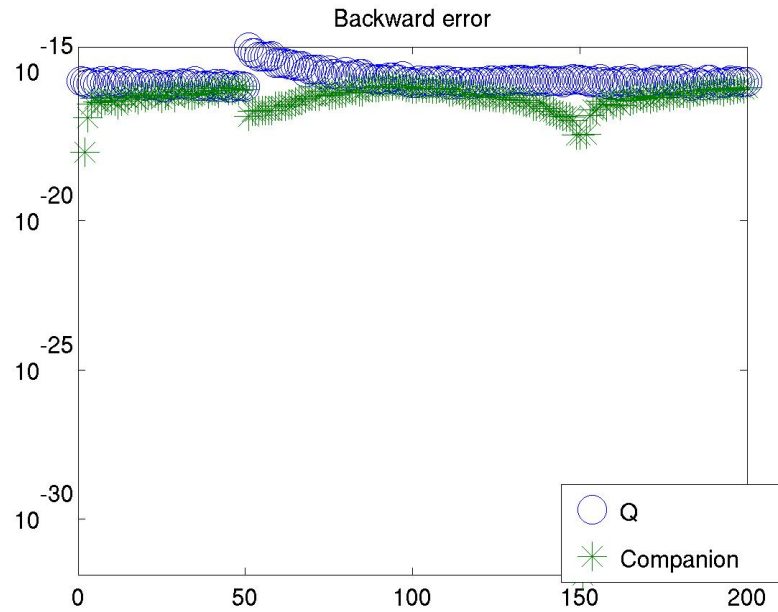


Figure 6.14: Backward error for quadratic and companion linearization for random quadratic with singular coefficients generated by block outer-product, $n = 100$, $\text{rank}(A_0) = \text{rank}(A_2) = 50$, 50 zero and infinite eigenvalues deflated by Householder reflectors.

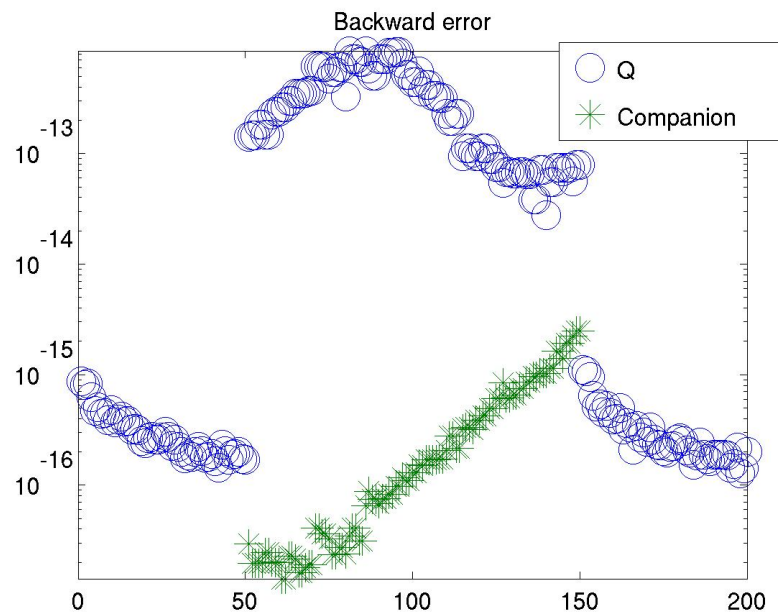


Figure 6.15: Backward error for quadratic and companion linearization for random quadratic with singular coefficients generated by block outer-product, $n = 100$, $\text{rank}(A_0) = \text{rank}(A_2) = 50$, 50 zero and infinite eigenvalues deflated using modified original Kublanovskaya method

Chapter 7

Conclusion

We have presented two main types of algorithms; those based on the ideas of Kublanovskaya et al, and those on applying Householder reflectors to the first companion linearization. For both these methods we require a UTV decomposition, which we have outlined, either to form the transformation matrices (Kublanovskaya method) or to provide a basis for the null space of singular leading and trailing coefficient (Householder reflector algorithms). We note that the algorithms differ in that the Kublanovskaya type methods try to deflate all zero and/or infinite eigenvalues at once in a global manner, whilst the Householder reflectors methods deflate one eigenvalue at a time in a local sense. From testing the MATLAB implementations on the test problems, deflating either zero or/and infinite eigenvalues using Householder reflectors is stable for the test problems, including examples such as the speaker box, with a defective zero eigenvalue. When Householder reflectors are used to deflate zero or infinite eigenvalues we saw that eigenpairs of the quadratic can be returned to high accuracy with a smaller than expected backward error. The modified original approach of Kublanovskaya et al can give slightly larger backward errors for eigenpairs of the quadratic, this is likely to arise from the need to apply the inverse of $Q(\alpha)$ to the leading and trailing coefficients. The simplified Kublanovskaya method appears stable, with the exception of defective infinite or zero eigenvalues, however, rigorous error analysis is necessary to determine the stability of the methods.

Appendix A

MATLAB Programs

We list the MATLAB programs in Table A.1.

Table A.1: Summary of MATLAB programs; $Q(\lambda) = \lambda^2 A_2 + \lambda A_1 + A_0$ is a general quadratic and $L(\lambda) = \lambda X + Y$ a matrix pencil.

General functions		Page
<code>qepnormscale.m</code>	Scale $Q(\lambda)$ by method of Fan, Lin and Van Dooren	83
<code>linberrright.m</code>	Backward error of right eigenpair for $L(\lambda)$	84
<code>qepberrright.m</code>	Backward error of right eigenpair for $Q(\lambda)$	85
<code>rec_quad_evec.m</code>	Recover eigenvectors of $Q(\lambda)$ from those of $L(\lambda)$ using a backward error argument	86
<code>nullspace.m</code>	Compute basis for null space of matrix	88
Deflation procedures		
<code>deflate1.m</code>	Deflation of zero or infinite eigenvalues (A_0 or A_2 singular but not both), using simplified Kublanovskaya method or Householder reflector approach	91
<code>deflate2.m</code>	Deflation of zero and infinite eigenvalues (A_0 and A_2 both singular), using modified version of original Kublanovskaya approach or Householder reflectors	96
<code>simpl_kubl.m</code>	Simplification of original method of Kublanovskaya et al	102
<code>house{zero, inf}.m</code>	Deflation of {zero, infinite} eigenvalues using Householder reflectors	{104, 107}

```
function [S0, S1, S2, gamma, delta] = qepnormscale(A0, A1, A2)
%QEPNORMSCALE Scale quadratic matrix polynomial.
% [S0, S1, S2, GAMMA, DELTA] = QEPNORMSCALE(A0, A1, A2)
% scales the quadratic:
%
%           lambda^2*A2 + lamda*A1 + A0.
%
% The input is A0, A1, A2 which are n-by-n matrices.
% The output is the corresponding scaled matrices, S0, S1, S2,
% also n-by-n matrices, GAMMA and DELTA are the scale factors.
%
% QEPNORMSCALE implements the method of Fan, Lin, and Van Dooren.
% The aim is to make the norms of the coefficients A0, A1, and A2
% close to one after scaling.
%
% Reference: H.-Y. Fan, W.-W. Lin and P. Van Dooren, Normwise scaling
% of second order polynomial matrices.
% SIAM J. Matrix Anal. Appl., 26(2004), pp. 252-256.
%
% 25-Jan-2008 14:55:54

gamma = sqrt(norm(A0)/norm(A2));
delta = 2 / (norm(A0) + norm(A1)*gamma );

S0 = delta * A0;
S1 = delta * gamma * A1;
S2 = delta * gamma * gamma * A2;
```

```
function berr = linberrright(Z, alpha, beta, X, Y)
%LINBERRRIGHT Backward error of right eigenpair of linearization.
%           BERR = LINBERRRIGHT(Z, ALPHA, BETA, X, Y)
%           returns BERR the backward errors, where Z is
%           a matrix of right eigenvectors of the pencil
%           ALPHA * X + BETA * Y
%           with associated eigenvalue (ALPHA, BETA) in homogeneous form.
%
% 09-Sep-2007 13:25:51

nX = norm(X);
nY = norm(Y);
n = size(X)/2;

for i =1:length(alpha)
berr(i) = norm((alpha(i)*X + beta(i)*Y)*Z(:,i))/ ...
    ((abs(alpha(i))*nX + abs(beta(i))*nY)*norm(Z(:,i)));
end
```

```

function berr = qepberrright(X, alpha, beta, A0, A1, A2, flag)
%QEPBERRRIGHT Backward error of QEP - right eigenpairs.
%   BERR = QEPBERRRIGHT(X, ALPHA, BETA, A0, A1, A2, FLAG)
%   returns backward error BERR, where X is a matrix of right
%   eigenvectors with corresponding eigenvalues (ALPHA, BETA)
%   in homogeneous form for the quadratic
%
%   ALPHA^2*A2 + ALPHA*BETA*A1 + BETA^2 * A0
%
%   FLAG is a string taking values upper or lower to select
%   (upper) X(1:n,k) or (lower) X(n+1:2*n,k)
%   as the k-th eigenvector of the QEP.
%
numevs = length(alpha);
n = size(A0,2);
switch lower(flag)
    case 'lower'
        for i =1:numevs
            num = norm((alpha(i)^2 * A2 + alpha(i) * beta(i) *A1 + ...
                beta(i)^2*A0)*X(n+1:2*n,i));
            denom = ((abs(alpha(i))^2*norm(A2) + ...
                abs(alpha(i))*abs(beta(i))*norm(A1) + ...
                abs(beta(i))^2 * norm(A0))*norm(X(n+1:2*n,i)));
            berr(i) = num/denom;
        end
    case 'upper'
        for i =1:numevs
            num = norm((alpha(i)^2 * A2 + alpha(i) * beta(i) *A1 + ...
                beta(i)^2*A0)*X(1:n,i));
            denom = ((abs(alpha(i))^2*norm(A2) + ...
                abs(alpha(i))*abs(beta(i))*norm(A1) + ...
                abs(beta(i))^2 * norm(A0))*norm(X(1:n,i)));
            berr(i) = num/denom;
        end
end
end

```

```

function [X, berrqep] = rec_quad_evec(alpha, beta, V, A0, A1, A2)
%REC_QUAD_EVEC Recover eigenvector of quadratic from that of linearization.
% [X, BERRQEP] = REC_QUAD_EVEC(ALPHA, BETA, A0, A1, A2, V), where ALPHA,
% and BETA represent the eigenvalues in homogeneous form of the quadratic
% Q(lambda), with corresponding eigenvectors that are columns of V.
% The quadratic Q(lambda) is defined by the coefficient matrices
% A0, A1, and A2 and Q(lambda) = lambda^2*A2 +lambda*A1 + A0.
%
% 25-Jan-2008 14:55:54

n = size(A0,1);
warning('off','MATLAB:divideByZero');
for i = 1:length(alpha)
    if and(isfinite(abs(alpha(i)) ./ abs(beta(i))), ...
isfinite(abs(beta(i)) ./ abs(alpha(i))))
        %Finite nonzero eigenvalue.
        %Select component of eigenvector of linearization that yields
        %the smallest backward error.
        berr_u = qepberrright(V(:,i), alpha(i), beta(i), A0, A1, A2, 'upper');
        berr_l = qepberrright(V(:,i), alpha(i), beta(i), A0, A1, A2, 'lower');

        if (abs(berr_u) < abs(berr_l))
            %Select upper component.
            Zr(1:size(A0,1),i) = V(1:size(A0,1),i);
            berrqep(i) = berr_u;
        else
            %Select lower component.
            Zr(1:size(A0,1),i) = V(size(A0,1)+1:2*size(A0,1),i);
            berrqep(i) = berr_l;
        end

        if norm(Zr(1:size(A0,1),i))==0
            Zr(1:size(A0,1),i) = V(1:n,i);
            berrqep(i) = qepberrright(V(:,i), alpha(i), beta(i), A0, A1, A2, 'upper');
        end
    elseif isinf(abs(alpha(i)) ./ abs(beta(i)))

```

```
%Infinite eigenvalue.
Zr(1:size(A0,1),i) = V(1:size(A0,1),i); %Select first n components.
berrqep(i) = qepberrright(V(:,i), alpha(i), beta(i), A0, A1, A2, 'upper');
elseif isinf(abs(beta(i)) ./ abs(alpha(i)) )
%Zero eigenvalue.
Zr(1:size(A0,1),i) = V(n+1:2*size(A0,1),i); %Select last n components.
berrqep(i) = qepberrright(V(:,i), alpha(i), beta(i), A0, A1, A2, 'lower');
end
end

X = Zr;
warning('on', 'MATLAB:divideByZero');
```

```

function [V, nr, Q, R, P] = nullspace(A, flag)
%NULLSPACE Basis for null space of a matrix.
%
% [V, NR, Q, R, P] = NULLSPACE(A, FLAG) computes a basis for the numerical null space
% of the matrix A, where FLAG is a string specifying the method used to
% compute a basis for the null space, it takes a value
%   'svd' for the singular value decomposition
%   'eig' for an eigendecomposition
%       or for ULV decompositions
%   'rrqr' for a rank-revealing QR factorization.
%   'hulv' for high rank ULV decomposition
%       (Stewart's high rank algorithm)
%   'hulv_a' for high rank ULV decomposition (using inverse iteration)
%   'lulv' for low rank ULV decomposition
%       (with initial triangular factorization)
%   'lulv_a' for low rank ULV decomposition
%       (without initial triangular factorization)
%   'lrrqr' for Chan and Hansen low rank RRQR
%   'hrrqr' for Chan/Foster high rank RRQR
% The null space basis is returned in the matrix V, and the numerical
% rank as NR. If FLAG is specified as 'rrqr' and Q, R, and P are specified
% as output arguments then Q, R, and P are the matrices that come from a
% rank-revealing QR factorization of A'.
%
% 25-Jan-2008 14:55:54

switch(lower(flag))
    case 'svd'
        %Singular value decomposition.
        [U, S, Vtilde] = svd(A);
        S = diag(S);
        tol = max(size(A))*eps(max(S));
        nr = sum(S > tol);
        V = Vtilde(:,nr+1:end);
    case 'eig'
        %Eigendecomposition.

```

```

V = zeros(length(A),0);
dimnull = 0;
n = length(A);
tol = max(size(A))*eps(normest(A));
[EV, LAMBDA] = eig(A);
for i = 1:n
    if abs(LAMBDA(i,i)) < tol
        dimnull = dimnull + 1;
        V(:, dimnull) = EV(:,i);
    end
end
nr = n - dimnull;
case 'hulv'
    %Stewart's high rank-revealing ULV algorithm.
    [nr, L, Vtilde, U, vec] = hulv(A);
    L = diag(L);
    n = length(A);
    V = Vtilde(:,nr+1:n);
case 'rrqr'
    %Rank-revealing QR factorization.
    V = zeros(length(A),0);
    [Q, R, P] = qr(A');
    dimnull = 0;
    R = diag(R);
    n = length(A);
    tol = max(size(A))*eps;
    for i = 1:n
        if abs(R(i)) < tol
            dimnull = dimnull + 1;
            V(:, dimnull) = Q(:,i);
        end
    end
    nr = n - dimnull;
case 'hulv_a'
    %Alternative high rank ULV decomposition, deflation based on
    %singular vector estimation by inverse iteration.

```

```
[nr, L, Vtilde, U, vec] = hulv_a(A);
L = diag(L);
n = length(A);
V = Vtilde(:,nr+1:n);
case 'lulv'
    %Low rank ULV with initial triangular factorization and deflation
    %using the power method.
    [nr, L, Vtilde, U, vec] = lulv(A);
    L = diag(L);
    n = length(A);
    V = Vtilde(:,nr+1:n);
case 'lrrqr'
    %Chan and Hansen low rank RRQR.
    [nr, R, Pi, Q, W, vec] = lrrqr(A');
    n = length(A);
    V = Q(:,nr+1:n);
case 'hrrqr'
    %Chan/Foster high rank RRQR.
    [nr, R, Pi, Q, W, vec] = hrrqr(A');
    n = length(A);
    V = Q(:,nr+1:n);
case 'lulv_a'
    %Alternative low rank ULV with no initial triangular factorization
    %and deflation using Householder reflectors.
    [nr, L, Vtilde, U, vec] = lulv_a(A);
    L = diag(L);
    n = length(A);
    V = Vtilde(:,nr+1:n);
end
```

```

function [X, e, etaQ, etaL] = deflate1(P0, P1, P2, algo, ns_method)
%DEFLATE1 Quadratic eigenvalue problem, with singular leading or trailing
% coefficients.
% [E, X, ETAQ, ETAL] = DEFLATE1(P0, P1, P2, ALGO)
% solves the quadratic eigenvalue problem
% (P0 + LAMBDA*P1 + LAMDBA^2*P2)*x = 0.
% P0, P1, and P2 are square matrices with A0 or A2 (but not both) singular/
% A singular P0 yields zero eigenvalues, while a singular A2 yields
% infinite eigenvalues.
% ALGO is a string that specifies the method of deflating the
% zero and infinite eigenvalues it takes one of the values
% 'householder' for deflation by use of Householder reflectors,
% or
% 'kublanovskaya' for deflation by using a simplified version of the
% algorithm of Kublanovskaya.
%
% NS_METHOD is a string specifying the method used to approximate the null
% space of P0 or P2, it takes a value
% 'svd' for the singular value decomposition
% 'eig' for an eigendecomposition
% or for ULV decompositions
% 'rrqr' for a rank-revealing QR factorization.
% 'hulv' for high rank ULV decomposition
% (Stewart's high rank algorithm)
% 'hulv_a' for high rank ULV decomposition (using inverse iteration)
% 'lulv' for low rank ULV decomposition
% (with initial triangular factorization)
% 'lulv_a' for low rank ULV decomposition
% (without initial triangular factorization)
% 'lrrqr' for Chan and Hansen low rank RRQR
% 'hrrqr' for Chan/Foster high rank RRQR
%
% On output LAMBDA is a vector of eigenvalues with associated eigenvectors
% that are columns of X.
% ETAQ and ETAL are vectors with the backward error of
% eigenpairs for the quadratic and linearization respectively. No

```

```
% backward errors are returned for deflated eigenpairs for the
% linearization.

% By default the original quadratic with coefficient matrix P0,P1,P2 is
% scaled by the method of Fan, Lin and Van Dooren.
%
% Reference: V. N. Kublanovskaya, V. B. Mikhailov, and V. B. Khazanov,
% Eigenvalue problem for an irregular  $\lambda$ -matrix,
% Journal of Mathematical Sciences, 13(1980), pp. 251260.

n = length(P0);
k = 0;

%Check singularity of A0 and A2 using condest.
A0condest = condest(P0); A2condest = condest(P2);

if(A0condest > n*1/eps)
    fprintf('Coefficient matrix P0 singular\n')
    def_evs = 'zero'; A0singular = true;
else
    fprintf('Coefficient matrix P0 nonsingular\n')
    A0singular = false;
end

if(A2condest > n*1/eps)
    fprintf('Coefficient matrix P2 singular\n')
    def_evs = 'inf'; A2singular = true;
else
    fprintf('Coefficient matrix P2 nonsingular\n')
    A2singular = false;
end

if or(and(strcmp(algo,'kublanovskaya'), A0singular == true), ...
      or( and(strcmp(algo,'householder'), A0singular == true), ...
          and(strcmp(algo,'householder'), A2singular == true) ))
    %Deflate zero eigenvalues using first companion linearization.
```

```

    %Apply Fan, Lin and Van Dooren scaling.
    [A0, A1, A2, gamma, delta] = qepnormscale(P0, P1, P2);
elseif and(strcmp(algo,'kublanovskaya'), A2singular == true)
    %Deflate infinite eigenvalues via taking linearization of reversal
    %polynomial, using simplified Kublanvoskaya method.
    def_evs = 'zero';
    [A0, A1, A2, gamma, delta] = qepnormscale(P2, P1, P0);
end

%Form first companion linearization.
X = [A2 zeros(n,n); zeros(n,n) eye(n,n)];
Y = [A1 A0 ; -eye(n,n) zeros(n,n)];

if and(strcmp(algo,'householder'), strcmp(def_evs,'zero'))
    [defX, defY, r, H, betaH, N] = housedefzero(X, Y, A0, ns_method);
    k = n - r; %Number of zero eigenvalues deflated.
elseif and(strcmp(algo,'householder'), strcmp(def_evs,'inf'))
    [defX, defY, r, H, betaH, N] = housedefinf(X, Y, A2, ns_method);
    k = n - r; %Number of infinite eigenvalues deflated.
elseif strcmp(algo,'kublanovskaya')
    [defX, defY, k, H, N] = smp1_kubl(X, Y, A0);
    %k is number of deflated zero eigenvalues.
    r = n - k;
end

%Compute eigenvectors of smaller linearization of deflated pencil.
if strcmp(algo,'householder')
    %Householder methods, deflated eigenvalues appear along
    %north-west diagonal of pencil.
    if A0singular == true
        keyboard
        %Zero eigenvalues deflated.
        for i = 1:k
            defX(i+1:end,i) = 0;
            defY(i:end,i) = 0;
        end
    end
end

```

```

elseif A2singular == true
    for i = 1:k
        defX(i+1:end,i) = 0;
        defY(i+1:end,i) = 0;
    end
end
elseif strcmp(algo,'kublanovskaya')
    %Zero elements that are theoretically zero.
    defX(n+1:end,n+1:end) = eye(n);
    defY(:,2*n-k:end)=0;
end
[Z, LAMBDA] = eig(-defY, defX);

%Put eigenvalues in homogeneous form.
alpha = diag(LAMBDA);
beta = ones(size(alpha));
i = isinf(alpha); alpha(i) = 1; beta(i) = 0;

etaL = linberrright(Z, alpha, beta, defX, defY);

warning('off','MATLAB:divideByZero');

%Reapply accumulated Householder reflectors.
if strcmp(algo, 'householder')
    for j = k:-1:1
        %Recover right eigenvectors.
        temp = Z(j:end,:)' - ...
            (Z(j:end,:)'* H(1:2*n-j+1,j))*betaH(j)*H(1:2*n-j+1,j)' ;
        Z(j:end,:)=temp';
        Z(:,j) = Z(:,j)/norm(Z(:,j));
    end
elseif strcmp(algo, 'kublanovskaya')
    Z(1:n,:) = Z(1:n,:);
    Z(n+1:2*n,:) = H'*Z(n+1:2*n,:);
end

```

```
%Compute backward error of eigenpairs of quadratic.
clear('X')
[X, etaQ] = rec_quad_evec(alpha, beta, Z, A0, A1, A2);

%Sort eigenvalues into order of increasing magnitude.
[e, sorteigind] = sort(alpha./beta);
alpha = alpha(sorteigind); beta = beta(sorteigind); X = X(:,sorteigind);
etaQ = etaQ(sorteigind); etaL = etaL(sorteigind);
if or(and(strcmp(algo,'kublanovskaya'), A0singular == true), ...
      or(and(strcmp(algo,'householder'), A0singular == true), ...
          and(strcmp(algo,'householder'), A2singular == true) ))
    j = find(isfinite(e));
    e(j) = (alpha(j)./beta(j))*gamma; %Rescale eigenvalues.
elseif and(strcmp(algo,'kublanovskaya'), A2singular == true)
    e = gamma./e; %Take reciprocal if using reversal polyn.
end
warning('on','MATLAB:divideByZero');
```

```

function [X, e, etaQ, etaL] = deflate2(P0, P1, P2, algo, ns_method)
%DEFLATE2 Quadratic eigenvalue problem, with singular leading and trailing
%coefficients.
%[X, E, ETAQ, ETAL] = DEFLATE2(P0, P1, P2, ALGO, NS_METHOD)
% solves the quadratic eigenvalue problem
% (P0 + LAMBDA*P1 + LAMDBA^2*P2)*x = 0.
% P0, P1, and P2 are square matrices with both P0 and P2 singular. A
% singular P0 yields zero eigenvalues and a singular P2 yields infinite
% eigenvalues.
% ALGO is a string that specifies the method of deflating the
% zero and infinite eigenvalues, it takes one of the values
% 'householder' for deflation by use of Householder reflectors,
% or
% 'kublanovskaya' for deflation by using a modified version of the
% algorithm of Kublanovskaya
%
% NS_METHOD is a string specifying the method used to approximate the null
% space of P0 or P2, it takes a value
% 'svd' for the singular value decomposition
% 'eig' for an eigendecomposition
% or for ULV decompositions
% 'rrqr' for a rank-revealing QR factorization.
% 'hulv' for high rank ULV decomposition
% (Stewart's high rank algorithm)
% 'hulv_a' for high rank ULV decomposition (using inverse iteration)
% 'lulv' for low rank ULV decomposition
% (with initial triangular factorization)
% 'lulv_a' for low rank ULV decomposition
% (without initial triangular factorization)
% 'lrrqr' for Chan and Hansen low rank RRQR
% 'hrrqr' for Chan/Foster high rank RRQR
%
% By default the original quadratic with coefficient matrix P0,P1,P2 is
% scaled by the method of Fan, Lin and Van Dooren.
%
% On output E is a vector of eigenvalues with associated eigenvectors that

```

```

% are columns of X. ETAQ and ETAL are vectors with the backward error of
% eigenpairs for the quadratic and linearization respectively. No
% backward errors are returned for deflated eigenpairs for the
% linearization.
%
% Reference: V. N. Kublanovskaya, V. B. Mikhailov, and V. B. Khazanov,
% Eigenvalue problem for an irregular \lambda-matrix,
% Journal of Mathematical Sciences, 13(1980), pp. 251260.

[A0, A1, A2, gamma, delta] = qepnormscale(P0, P1, P2);
n = length(P0);
Z = zeros(2*n,2*n);

switch lower(algo)
    case('householder')
        %Form first companion linearization.
        X = [A2 zeros(n,n); zeros(n,n) eye(n,n)];
        Y = [A1 A0 ; -eye(n,n) zeros(n,n)];

        [defX, defY, r0, Hz, betaHz, Nzero] = ...
            housedefzero(X, Y, A0, ns_method);
        k0 = n - r0; %Number of deflated zero eigenvalues.

        for i = 1:k0
            defX(1+i:2*n, i) = 0;
            defY(i:2*n, i) = 0;
        end

        [defX(k0+1:2*n,k0+1:2*n),...
         defY(k0+1:2*n,k0+1:2*n), r2, Hi1, betaHi1,...
         Ntwo, Hi2, betaHi2] = ...
            housedefinf(defX(k0+1:2*n,k0+1:2*n), ...
                defY(k0+1:2*n,k0+1:2*n), A2, ns_method,k0, n, Hz, betaHz);
        k2 = n - r2; %Number of deflated infinite eigenvalues.

        for i = k0+1:k0+k2

```

```

        defY(i+1:2*n, i) = 0;
        defX(i+1:2*n, i) = 0;
    end

    [Z, LAMBDA] = eig(-defY, defX);

    %Put in homogeneous form - not including deflated e'vals.
    alpha = diag(LAMBDA);
    beta = ones(size(alpha));
    k = isinf(abs(alpha)); alpha(k) = 1; beta(k) = 0;
    etaL = linberrright(Z(:, :), alpha(:), beta(:), defX, defY);

    for i = k2:-1:1
        %Apply Infinite deflation Householder
        %reflectors to matrix of eigenvectors.
        Z(k0+i:end, :) = Z(k0+i:end, :) - ...
            Hi1(1:2*n-i+1-k0,i)*(betaHi1(i)*Hi1(1:2*n-i+1-k0,i)')...
            * Z(k0+i:end, :));
    end

    for i = k0:-1:1
        %Reapply zero deflation Householder
        %reflectors to matrix of eigenvectors.
        Z(i:end, :) = Z(i:end, :) - ...
            Hz(1:2*n-i+1,i) * (betaHz(i)...
            * Hz(1:2*n-i+1,i)')*Z(i:end, :));
    end

    case('kublanovskaya')
        if strcmp(ns_method, 'rrqr')
            [Nzero, r0, Q0, R0, P0] = nullspace(A0, ns_method);
            [Ntwo, r2, Q2, R2, P2] = nullspace(A2, ns_method);
            %Number of zero and infinite eigenvalues to deflate.
            k0 = n - r0; k2 = n - r2;
        end
        alpha = fminsearch(@(z) condest(z*z*A2 + z*A1 + A0), rand*n)

```

```

kalpha = alpha;
Qalpha = alpha^2 * A2 + alpha * A1 + A0;
C = condest(Qalpha);
fprintf('Estimate of condition number of matrix
Q(alpha) is %6.2e \n',C)

%Store product of inv(Qalpha) applied to
%leading and trailing coefficients.
invQalphaA2 = inv(Qalpha);
invQalphaA0 = invQalphaA2*A0;
invQalphaA2 = invQalphaA2*A2;

F11 = alpha * Q2' * invQalphaA2 * Q2;
F12 = -1/alpha * Q2' * invQalphaA0 * Q0;
F21 = alpha * Q0' * invQalphaA2 * Q2;
F22 = -1/alpha * Q0' * invQalphaA0 * Q0 + 1/alpha *eye(n);

P = [zeros(r2,2*n-r2-r0) eye(r2,r2) zeros(r2,r0)
      eye(n-r2,n-r2) zeros(n-r2, n+r2)
      zeros(r0, 2*n-r0) eye(r0,r0)
      zeros(n-r0,n-r2) eye(n-r0,n-r0) zeros(n-r0,r0+r2)];

F = [F11 F12; F21 F22];

%Apply permutation matrix to F.
Ftilde = P'*F*P;

G = Ftilde(1:2*n-r0-r2,1:2*n-r0-r2);
H = Ftilde(1:2*n-r0-r2, 2*n-r0-r2+1:2*n);
R = Ftilde(2*n-r0-r2+1:2*n, 2*n-r0-r2+1:2*n);

%Compute eigenvalues and vector of submatrix.
[XR LAMBDA] = eig(R);
LAMBDA = diag(LAMBDA);

for i = 1:size(XR, 2)

```

```

        Xvec(1:2*n-r0-r2,i) = (G-LAMBDA(i)*eye(size(G))) ...
            \ (-H*XR(:,i));
        Xvec(2*n-r0-r2+1:2*n,i) = XR(:,i);
        Xvec(:,i) = Xvec(:,i)/norm(Xvec(:,i));
    end

    %Compute backward error of eigenpairs of deflated pencil.
    %Exclude deflated eigenvalues.
    etaL = linberrright(Xvec, LAMBDA, ones(size(LAMBDA)), ...
        eye(size(Ftilde)), -Ftilde);
    nuevs = size(LAMBDA,1);
    Z(:,1:nuevs) = P*Xvec;

    %Apply V to eigenvectors using structure of V.
    Z = [1/kalpha*Q0*Z(n+1:2*n,:); Q2*Z(1:n,:)-Q0*Z(n+1:2*n,:)];
    LAMBDA = diag(alpha - 1./diag(LAMBDA));
    k = n - r0 + n - r2; %Number of deflated eigenvalues.

    alpha = LAMBDA;
    beta = ones(size(alpha));
    i = isinf(alpha); alpha(i) = 1; beta(i) = 0;
end

e = alpha./beta*gamma; %rescale eigenvalues.

if strcmp(algo,'kublanovskaya')
    etaL = [etaL NaN*ones(1,n-r0+n-r2)];
    %The eigenvectors associated to infinite and zero eigenvalues must
    %be formed explicitly, and eigenvalues must be added back.
    alpha = [alpha; zeros(k0,1); ones(k2,1)];
    beta = [beta; ones(k0,1); zeros(k2,1) ];
    %Add eigenvectors associated to deflated eigenvalues.
    K = kron([0,1]', Nzero);
    Z(:,nuevs+1:nuevs+k0) = K;
    K = kron([1,0]', Ntwo);
    Z(:,nuevs+k0+1:2*n) = K;

```

```
end
```

```
%Compute backward error of eigenpairs of quadratic.
```

```
[qepevecs, etaQ] = rec_quad_evec(alpha, beta, Z, A0, A1, A2);
```

```
clear('X')
```

```
X = qepevecs;
```

```
[e, sorteigind] = sort(alpha./beta);
```

```
alpha = alpha(sorteigind); beta = beta(sorteigind); X = X(:,sorteigind);
```

```
etaQ = etaQ(sorteigind); etaL = etaL(sorteigind);
```

```

function [defX, defY, k, Q, N] = simpl_kubl(X, Y, A0)
%SMPL_KUBL    Simplified Kublanovskaya deflation algorithm.
%    [defX, defY, k, Q, Nzero] = simpl_kubl(X, Y, A0)
%
% X and Y define the pencil
%
% L(lambda) = lambda * X + Y
%
% with X = [A2 zeros(n,n); zeros(n,n) eye(n,n)]
%    Y = [A1 A0 ; -eye(n,n) zeros(n,n)];
%
% that is the first companion linearization of the QEP
%
% Q(lambda) = A0 + lambda * A1 + lambda^2 A2,
%
% with A0, A1, and A2 n-by-n matrices.
% The coefficient A0 is assumed rank-deficient thus contributing zero
% eigenvalues.
% defX and defY are 2n-by-2n matrices that define the pencil
% Ltilde(lambda) = lambda * defX + defY,
% that has k zero eigenvalues deflated (in the south-east corner).
% Q is an orthogonal matrix that results from a rank-revealing QR
% factorization from A0', where rank(A0) = k.
% N is a matrix whose columns are a basis for the null space of A0.
%
% 25-Jan-2008 14:55:54

n = size(A0,1);

%Take RRQR factorization of A0'.
[Q0, R0, P] = qr(A0');
dimnull = 0;
R0 = diag(R0);
n = length(A0);
tol = max(size(A0))*eps(normest(A0));
for i = 1:n

```

```
    if abs(R0(i)) < tol
        dimnull = dimnull + 1;
    end
end
r0 = dimnull;
k = dimnull;
r0 = n - r0;

K = [eye(n,n) zeros(n,n); zeros(n,n) Q0'];

defX = X;
defY = [Y(1:n,1:n), Y(1:n,n+1:2*n)*Q0; ...
        Q0'*Y(n+1:2*n,1:n) Q0'*Y(n+1:2*n, n+1:2*n)*Q0];
Q = Q0';
N = Q0(:,find(abs(R0)<tol));
```

```

function [defX, defY, r0, H, beta, Vzero] = housedefzero(X, Y, A0, flag)
%HOUSEDEFZERO Householder deflation of zero eigenvalues in QEP from A0.
% Deflates zeros eigenvalues of the linearization
%
%  $L(\lambda) = \lambda X + Y$ 
%
% that is the first companion linearization of a QEP whose trailing
% coefficient (A0) is singular.
%
% [DEFX, DEFY, R2, K, BETAK, VZERO] = HOUSEDEFZERO(X, Y, A0, FLAG)
% where X and Y are 2n-by-2n matrices defining the linearization above,
% FLAG is a string that specifies which method is used to compute a
% basis for the null space of A0. It takes one of the values
%
% 'svd' for the singular value decomposition
% 'eig' for an eigendecomposition
% or for ULV decompositions
% 'rrqr' for a rank-revealing QR factorization.
% 'hulv' for high rank ULV decomposition
% (Stewart's high rank algorithm)
% 'hulv_a' for high rank ULV decomposition (using inverse iteration)
% 'lulv' for low rank ULV decomposition
% (with initial triangular factorization)
% 'lulv_a' for low rank ULV decomposition
% (without initial triangular factorization)
% 'lrrqr' for Chan and Hansen low rank RRQR
% 'hrrqr' for Chan/Foster high rank RRQR
%
% DEFX and DEFY are matrices forming the pencil
%
%  $L(\lambda) = \lambda X + Y$ 
%
% with zero eigenvalues contributed by A2 deflated. R0 is the rank
% of the matrix A0. K and BETAK define the Householder transformation
% that is applied to recover the right eigenvector of the original
% pencil. VZERO is a basis for the null space of A0.

```

```

%
% 25-Jan-2008 14:55:54

n = length(A0);
defX = X;
defY = Y;

[Vzero, r0] = nullspace(A0, flag);

beta = zeros(n-r0,1); %To store Householder betas.
H = zeros(2*n,n-r0); %To store Householder vectors h.
V = [zeros(n,n-r0); Vzero];

for k = 1:n-r0
    %Compute the Householder vector.
    [H(1:2*n-k+1,k), beta(k), s] = gallery('house', V(k:end,k));

    %Carry out similarity transformation on the pencil lambda*defX + defY.

    if (k>1)
        defX(1:k-1, k:end) = defX(1:k-1, k:end) - ...
(beta(k) * defX(1:k-1, k:end) * H(1:2*n-k+1,k)) * H(1:2*n-k+1,k)';
        defY(1:k-1, k:end) = defY(1:k-1, k:end) - ...
(beta(k) * defY(1:k-1, k:end) * H(1:2*n-k+1,k)) * H(1:2*n-k+1,k)';
    end

    defX(k:end, k:end) = defX(k:end, k:end) - ...
H(1:2*n-k+1,k) * (beta(k) * H(1:2*n-k+1,k) * defX(k:end, k:end));

    defY(k:end, k:end) = defY(k:end, k:end) - ...
H(1:2*n-k+1,k) * (beta(k) * H(1:2*n-k+1,k) * defY(k:end, k:end));

    defX(k:end, k:end) = defX(k:end, k:end) - ...
(beta(k) * defX(k:end, k:end) * H(1:2*n-k+1,k)) * H(1:2*n-k+1,k)';

```

```
defY(k:end, k:end) = defY(k:end, k:end) - ...  
(beta(k) * defY(k:end, k:end) * H(1:2*n-k+1,k)) * H(1:2*n-k+1,k)';  
  
%Apply Householder transformations to null space basis.  
V(k:end, k:end) = V(k:end, k:end) - ...  
H(1:2*n-k+1,k) * (beta(k) * H(1:2*n-k+1,k)'*V(k:end, k:end));  
end
```

```

function [defX, defY, r2, K, betaK, Vtwo, H, betaH] = ...
housedefinf(X, Y, A2, flag, k, n, Hz, betaHz)
%HOUSEDEFINF Householder deflation of block companion linearizations
%of quadratic polynomials with singular leading coefficient.
%
% Deflates infinite eigenvalues of the linearization
%
%  $L(\lambda) = \lambda X + Y$ 
%
% that is the first companion linearization of a QEP whose leading
% coefficient (A2) is singular.
%
% [DEFX, DEFY, R2, K, BETAK, VTWO, H, BETAH] = HOUSEDEFINFMARG(X, Y, A2,
% FLAG, K, N, HZ, BETAHZ)
% where X and Y are 2n-by-2n matrices defining the linearization above,
% flag is a string that specifies which method is used to compute a
% basis for the null space of A2. It takes one of the values
%
% 'svd' for the singular value decomposition
% 'rrqr' for a rank-revealing QR factorization
% 'eig' for an eigendecomposition
% 'ulv' for an ULV decomposition.
%
% HZ and BETAHZ define the Householder reflectors for deflating zero
% eigenvalues, (needed if zero eigenvalues have already been deflated,
% K is then the number of deflated zero eigenvalues).
% N is the dimension of the matrix coefficients that are N-by-N
% dimension.
%
% If eight input arguments are specified then HOUSEDEFINFMARG will
% deflate infinite eigenvalues from a pencil that has already had K zero
% eigenvalues deflated. If less than eight arguments (four) are specified
% then HOUSEDEFINFMARG deflates infinite eigenvalues from a pencil that
% is the first companion linearization of a quadratic matrix polynomial.
%
% DEFX and DEFY are matrices forming the pencil

```

```

%
% Ltilde(lambda) = lambda * DEFX + DEFY
%
% with infinite eigenvalues contributed by A2 deflated. R2 is the rank
% of the matrix A2. K and BETAK define the Householder transformation
% that is applied to recover the right eigenvector of the original
% pencil. Vtemp is a basis for the null space of A2.
%
% 25-Jan-2008 14:55:54

if nargin ~= 8
    n = length(A2);
end

defX = X;
defY = Y;

[Vtwo, r2] = nullspace(A2, flag);

if nargin == 8

    V = [Vtwo; zeros(n,n-r2)];
    for j = k:-1:1
        %Apply Householder reflectors from zero deflation to null space
        %basis.
        V(j:end, :) = V(j:end, :) - ...
            Hz(1:2*n-j+1,j)*(betaHz(j)*Hz(1:2*n-j+1,j))' * V(j:end, :));
    end
end

if nargin ~= 8
    V = [Vtwo; zeros(n,n-r2)];
    k = 0;
end

K = zeros(2*n-k,n-r2); %Preallocate storage.

```

```

H = zeros(2*n-k,n-r2);
betaK = zeros(n-r2,1);
betaH = zeros(n-r2,1);

for i = 1:n-r2
    z = V(k+i:end,i);
    x = defY(i:end, i:end)*z;
    x = x/norm(x); %Normalize.

    %Form right Householder reflector.
    [K(1:2*n-i+1-k,i), betaK(i), s] = gallery('house',z);

    %Form left Householder reflector.
    [H(1:2*n-i+1-k,i), betaH(i), s] = gallery('house',x);

    %Carry out the transformation H' * (lambda defX + defY) * K.
    if (i>1)
        defX(1:i-1, i:end) = defX(1:i-1, i:end) - ...
(betaK(i) * defX(1:i-1, i:end) * K(1:2*n-i+1-k,i)) * K(1:2*n-i+1-k,i)';
        defY(1:i-1, i:end) = defY(1:i-1, i:end) - ...
(betaK(i) * defY(1:i-1, i:end) * K(1:2*n-i+1-k,i)) * K(1:2*n-i+1-k,i)';
    end
    defY(i:end, i:end) = defY(i:end, i:end) - ...
H(1:2*n-i+1-k,i)*(betaH(i)*H(1:2*n-i+1-k,i)' * defY(i:end, i:end));
    defX(i:end, i:end) = defX(i:end, i:end) - ...
H(1:2*n-i+1-k,i)*(betaH(i)*H(1:2*n-i+1-k,i)' * defX(i:end, i:end));

    defX(i:end, i:end) = defX(i:end, i:end) - ...
(betaK(i) * defX(i:end, i:end) * K(1:2*n-i+1-k,i)) * K(1:2*n-i+1-k,i)';
    defY(i:end, i:end) = defY(i:end, i:end) - ...
(betaK(i) * defY(i:end, i:end) * K(1:2*n-i+1-k,i)) * K(1:2*n-i+1-k,i)';

    %Apply Householder transformations to nullspace basis.
    V(k+i:end, i:end) = V(k+i:end, i:end) - ...
K(1:2*n-i+1-k,i)*(betaK(i)*K(1:2*n-i+1-k,i)' * V(k+i:end, i:end));

```

end

Bibliography

- [1] E. ANDERSON, Z. BAI, C. H. BISCHOF, S. BLACKFORD, J. W. DEMMEL, J. J. DONGARRA, J. J. D. CROZ, A. GREENBAUM, S. J. HAMMARLING, A. MCKENNEY, AND D. C. SORENSEN, *LAPACK Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, third ed., 1999.
- [2] A. BUNSE-GERSTNER, R. BYERS, V. MEHRMANN, AND N. K. NICHOLS, *Feedback design for regularizing descriptor systems.*, Linear Algebra Appl., 299 (1999), pp. 119–151.
- [3] R. BYERS, C. HE, AND V. MEHRMANN, *Where is the nearest non-regular pencil?*, Linear Algebra Appl., 285 (1998), pp. 81–105.
- [4] T. F. CHAN, *Rank revealing QR factorizations*, Linear Algebra Appl., 88/89 (1987), pp. 67–82.
- [5] T. F. CHAN AND P. C. HANSEN, *Computing truncated singular value decomposition least squares solutions by rank revealing QR-factorizations*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 519–530.
- [6] ———, *Low-rank revealing QR factorizations*, Journal of Numerical Linear Algebra with Applications, 1 (1994), pp. 33–44.
- [7] S. CHANDRASEKARAN AND I. C. F. IPSEN, *On rank-revealing factorisations*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 592–622.
- [8] J.-P. DEDIEU AND F. TISSEUR, *Perturbation theory for homogeneous polynomial eigenvalue problems*, Linear Algebra Appl., 358 (2003), pp. 71–94.

- [9] D. K. FADDEEV, V. N. KUBLANOVSKAYA, AND V. N. FADDEEVA, *Linear algebraic systems with rectangular matrices*, in Contemporary Numerical Methods, vol. 1 of Materials Internat. Summer School on Numerical Methods [in Russian], Kiev, 1966, pp. 16–75.
- [10] H.-Y. FAN, W.-W. LIN, AND P. VAN DOOREN, *Normwise scaling of second order polynomial matrices*, SIAM J. Matrix Anal. Appl., 26 (2004), pp. 252–256.
- [11] R. D. FIERRO AND J. R. BUNCH, *Bounding the subspaces from rank revealing two-sided orthogonal decompositions*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 743–759.
- [12] R. D. FIERRO AND P. C. HANSEN, *Low-rank revealing ULV decompositions*, Numerical Algorithms, 15 (1997), pp. 37–55.
- [13] L. FOSTER, *Rank and null space calculations using matrix decomposition without column interchanges*, Linear Algebra Appl., 74 (1986), pp. 47–71.
- [14] I. GOHBERG, P. LANCASTER, AND L. RODMAN, *Matrix Polynomials*, Academic Press, New York, 1982.
- [15] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, USA, third ed., 1996.
- [16] A. GOTTS, *Report regarding model reduction, model compaction research project, University of Nottingham*. Unpublished manuscript, 2005.
- [17] P. C. HANSEN, R. D. FIERRO, AND P. S. HANSEN, *UTV tools: Matlab templates for rank-revealing UTV decompositions*, Numerical Algorithms, 20 (1999), pp. 165–194.
- [18] D. HERTING, *MSC/Nastran Advanced Dynamic Analysis User's Guide*, MSC. Software Corporation, 1997.

- [19] N. J. HIGHAM, R.-C. LI, AND F. TISSEUR, *Backward error of polynomial eigenvalue problems solved by linearization*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 1218–1241.
- [20] N. J. HIGHAM, D. S. MACKEY, AND F. TISSEUR, *The conditioning of linearizations of matrix polynomials*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 1005–1028.
- [21] N. J. HIGHAM, D. S. MACKEY, F. TISSEUR, AND S. D. GARVEY, *Scaling sensitivity and stability in the numerical solution of quadratic eigenvalue problems*, Internat. J. Numer. Methods Eng., 73 (2007), pp. 344–360.
- [22] N. J. HIGHAM AND F. TISSEUR, *More on pseudospectra for polynomial eigenvalue problems and applications in control theory*, Linear Algebra Appl., 351–352 (2002), pp. 435–453.
- [23] N. J. HIGHAM, F. TISSEUR, T. BETCKE, V. MEHRMANN, AND C. SCHRÖDER, *NLEVP: A Collection of Nonlinear Eigenvalue Problems*, University of Manchester. Unpublished manuscript, 2007.
- [24] A. HILLIGES, C. MEHL, AND V. MEHRMANN, *On the solution of palindromic eigenvalue problems*, in Proceedings of the European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS 2004), Jyväskylä, Finland, P. Neittaanmäki, T. Rossi, S. Korotov, E. Oñate, J. Périaux, and D. Knörzer, eds., 2004. <http://www.mit.jyu.fi/eccomas2004/proceedings/proceed.html>.
- [25] D. KRESSNER, *Private communication, Dusseldorf 2006*.
- [26] V. N. KUBLANOVSKAYA, V. B. MIKHAILOV, AND V. B. KHAZANOV, *Eigenvalue problem for an irregular λ -matrix*, Journal of Mathematical Sciences, 13 (1980), pp. 251–260.
- [27] D. S. MACKEY, N. MACKEY, C. MEHL, AND V. MEHRMANN, *Vector spaces*

- of linearizations of matrix polynomials*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 971–1004.
- [28] G. W. STEWART, *An updating algorithm for subspace tracking*, IEEE Trans. Signal Processing, 40 (1992), pp. 1535–1541.
- [29] ———, *Updating a rank revealing ULV decomposition*, SIAM J. Matrix Anal. Appl., 14 (1993), pp. 494–499.
- [30] ———, *Matrix Algorithms. Volume II: Eigensystems*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001.
- [31] F. TISSEUR AND K. MEERBERGEN, *The quadratic eigenvalue problem*, SIAM Rev., 43 (2001), pp. 235–286.
- [32] D. S. WATKINS, *Performance of the QZ algorithm in the presence of infinite eigenvalues*, SIAM J. Matrix Anal. Appl., 22 (2000), pp. 364–375.