To the Graduate Council:

I am submitting herewith a dissertation written by Mohamed Majed Sidani entitled "A Parallel Algorithm for the Non-Symmetric Eigenvalue Problem." I have examined the final copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of PhD, with a major in Mathematics.

Jack J. Dongarra, Major Professor

We have read this dissertation
and recommend its acceptance:

Efstratios Gallopoulos

Suzanne Lenhart

Steven M. Serbin

Accepted for the Council:

Associate Vice Chancellor
and Dean of The Graduate School

A Parallel Algorithm for the Non-Symmetric Eigenvalue Problem

A Dissertation
Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

Mohamad Majed Sidani
August 1991

Dedication

To my parents

Mrs. Nadida Ardroumli

and

Mr. Mahmoud Sidani.

ACKNOWLEDGMENTS

I wish to thank my advisor and teacher, Jack Dongarra, for his insightful guidance, his enthusiastic encouragement, his support and his patience throughout this project. Thank you, Jack, for two most intellectually and professionally rewarding years.

I also wish to thank my other committee members, Dr. Ohannes Karakashian for the many hours spent in his office learning math; Dr. Steve Serbin for introducing me to this field; as well as Dr. Lenhart and Dr. Esmond Ng. I also wish to acknowledge a helpful discussion I had with Dr. Pete Stewart.

I wish to thank Ms. Mary Drake for providing me with expert assistance in LaTeX. Miss Tina Garrison provided additional help.

Finally, I acknowledge the use of the computing facilities at Oak Ridge National Laboratory and Argonne National Laboratory. The project on which I report was supported in part by the National Science Foundation Science and Technology Center Cooperative Agreement No. CCR-8809615, and in part by the Applied Mathematical Sciences subprogram of the Office of Energy Research, U.S Department of Energy, under Contract DE-AC05-84OR21400.

ABSTRACT

We report on an algorithm for the solution of the non-symmetric eigenvalue problem. The algorithm is based on a divide and conquer procedure that provides initial approximations to the eigenpairs which are then refined using Newton iterations. Since the smaller subproblems can be solved independently, and since Newton iterations with different initial guesses can be started simultaneously, the algorithm -unlike the standard QR method- is ideal for parallel computers.

We also report on our investigation of deflation methods designed to obtain further eigenpairs if needed. Numerical results from implementations on a host of parallel machines (distributed and shared-memory) will be given.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

Introduction

The algebraic eigenvalue problem is one of the fundamental problems in computational mathematics. It arises in many applications and therefore represents an important area of algorithmic research. The problem has received considerable attention which has resulted in reliable methods [23, 22, 21]. However it is reasonable to expect that calculations might be accelerated through the use of parallel algorithms. A fully parallel algorithm for the symmetric eigenvalue problem was recently proposed in [7]. This algorithm is based on a divide-and-conquer procedure outlined in [4]. The latter was based on work by [12] and [1]. The fundamental principle behind this algorithm is that the partitioning by rank-one tearing interlaces the eigenvalues of the modified problem with the eigenvalues of the original problem. This approach in turn enables rapid and accurate determination, in parallel, of the eigenvalues and subsequent eigenvectors.

In this work we propose a parallel algorithm for the solution of the non-symmetric eigenvalue problem. The approach uses some of the features of the divide-and-conquer algorithm for the symmetric case mentioned earlier. In particular, the original problem is divided into two smaller and independent subproblems by a rank-one modification of the matrix. (We assume that the matrix has already been reduced to Hessenberg form and that the rank-one modification removes a subdiagonal element.) Once the eigensystems of the smaller subproblems are known, it is possible to compute those of the original matrix. In the non-symmetric case, however, the eigenvalues of the modified matrix do not interlace with the original

matrix. Indeed, the eigenvalues can scatter anywhere in the complex plane.

In our algorithm for the non-symmetric case, the eigensystem of the subproblem is used only to construct initial guesses for an iterative process that yields the desired eigensystem of the original problem. Under suitable conditions, Newton's method or a continuation method can be used to find the eigenpairs of the original problem. We report here on our application of an iterative refinement approach based on Newton's method; work on the continuation approach has been reported by [17].

CHAPTER 2

Review of the Symmetric Divide and Conquer Method

We review here the divide and conquer algorithm for the symmetric case [7]. We assume that the original symmetric matrix has been reduced to tridiagonal form and we consider the symmetric tridiagonal matrix

$$T = \begin{pmatrix} T_1 & \beta e_k e_1^T \\ \beta e_1 e_k^T & T_2 \end{pmatrix} = \begin{pmatrix} \hat{T}_1 & 0 \\ 0 & \hat{T}_2 \end{pmatrix} + \theta\beta \begin{pmatrix} e_k \\ \theta^{-1} e_1 \end{pmatrix} (e_k^T, \theta^{-1} e_1^T) \qquad (2.1)$$

where $1 \leq k \leq n$ and e_j represents the jth unit vector of appropriate length. The k^{th} diagonal element of T_1 and the first diagonal element of T_2 have been modified to give $\hat{T}_1$ and $\hat{T}_2$ respectively. The choice of $\theta = \pm 1$ is done so that severe cancellation is avoided. The next step in the algorithm consists then in solving the two smaller tridiagonal eigenvalue problems. More precisely, two eigensystems are computed

$$\hat{T}_1 = Q_1 D_1 Q_1^T, \quad \hat{T}_2 = Q_2 D_2 Q_2^T.$$

This gives

$$T = \begin{pmatrix} Q_1 D_1 Q_1^T & 0 \\ 0 & Q_2 D_2 Q_2^T \end{pmatrix} + \theta\beta \begin{pmatrix} e_k \\ \theta^{-1} e_1 \end{pmatrix} (e_k^T, \theta^{-1} e_1^T)$$

i.e.,

$$T = \begin{pmatrix} Q_1 & 0 \\ 0 & Q_2 \end{pmatrix} \left(\begin{pmatrix} D_1 & 0 \\ 0 & D_2 \end{pmatrix} + \theta\beta \begin{pmatrix} q_1 \\ \theta^{-1} q_2 \end{pmatrix} (q_1^T, \theta^{-1} q_2^T) \right) \begin{pmatrix} Q_1^T & 0 \\ 0 & Q_2^T \end{pmatrix} \qquad (2.2)$$

where $q_1 = Q_1^T e_k$ and $q_2 = Q_2^T e_1$. The problem reduces then to solving for the eigensystem of the interior matrix in 2.2 which is of the form $D + \rho z z^T$ where

$$D = \begin{pmatrix} D_1 & 0 \\ 0 & D_2 \end{pmatrix},$$

and z is a unit vector along $\begin{pmatrix} q_1 \\ \theta^{-1} q_2 \end{pmatrix}$ and ρ incorporates the constants already present and those needed in the scaling. In other terms we seek $\hat{D}$ and $\hat{Q}$ such that

$$D_b = D + \rho z z^T = \hat{Q} \hat{D} \hat{Q}^T. \tag{2.3}$$

The eigensystem of this matrix is computed via the rank one updating technique described in [1, 12]. More precisely, if all the components of z are non zero and all the diagonal elements of D are distinct, then it can be shown that none of the diagonal elements of D is an eigenvalue of D_b, and that the eigenvalues of D_b are the roots of the following secular equation [12]

$$f(\lambda) = 1 + \rho \sum_{j=1}^{n} \frac{\zeta_j^2}{\delta_j - \lambda} = 0,$$

where ζ_j is the j^{th} component of z. Assuming that ρ is positive (otherwise replace D by $-D$ and ρ by $-\rho$) and that the diagonal entries of D (the poles of f) are distinct and are such that

$$\delta_1 < \delta_2 < \cdots < \delta_n,$$

(otherwise find an appropriate permutation) then $\hat{D}$ being a rank one modification of D, its eigenvalues satisfy ([23])

$$\delta_1 < \hat{\delta_1} < \delta_2 < \hat{\delta_2} < \cdots < \delta_n < \hat{\delta_n}.$$

Because of the interlacing of the eigenvalues, the secular equation can be solved with a fast Newton-like iteration starting with a good initial guess. Now once an eigenvalue λ_j of D_b has been found, a corresponding eigenvector is

$$v_j = (D - \lambda_j)^{-1} z.$$

Finally, if some diagonal entries in D are equal, then the corresponding components of z can be annihilated using plane rotations, thereby reducing the initial problem to one of smaller order.

The algorithm lends itself easily to parallel implementation. Indeed, the matrix can be recursively subdivided into yet smaller problems, and these can be solved simultaneously and independently on different processors. The root-finding procedure permits different processes with different initial guesses to be started simultaneously on different processors. The divide and conquer approach for the symmetric eigenproblem proved to be highly efficient on parallel machines as well as on sequential machines: Indeed, it provided remarkable speedups over the symmetric QR algorithm in parallel as well as in serial mode [7, 14].

CHAPTER 3

Homotopy Methods

In light of the remarkable perfomance of the divide and conquer approach for the symmetric eigenvalue problem, it is tempting to generalize the approach to the non-symmetric eigenproblem hoping for similar benefits in performance. The way we will do this is by first reducing the non-symmetric matrix to upper-Hessenberg form, and then zeroing a subdiagonal element. This will produce a matrix whose eigensystem can be found by solving for those of two smaller upper-Hessenberg matrices (the details will be given in the next chapter); the zeroing of the subdiagonal element is a rank-one modification. However, the interlacing property in the symmetric case provided the root-finding procedure with good initial guesses. In the non-symmetric case the rank-one modification will not result in such an interlacing of the sought eigenvalues and those of the modified matrix. Indeed, as we mentioned earlier, these can scatter anywhere in the complex plane (see Fig. 1), or rather anywhere in the Gershgorin circles of the matrix which happen to contain those of the modified matrix. In the absence of guaranteed good initial guesses, homotopy methods with their global convergence properties become an interesting option.

In this chapter we introduce the class of methods known as homotopy methods and we describe the early attempts at their use in connection with the solution of the algebraic eigenproblem. We finally show how the algorithm which we introduce and study in the remainder of the thesis relates to these methods.

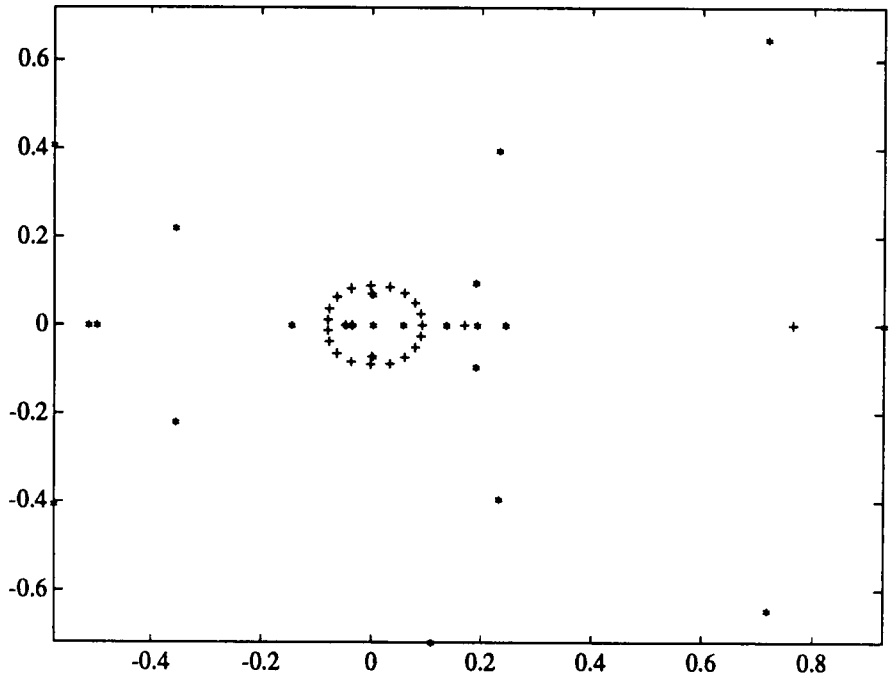

Figure 1: Distribution of the eigenvalues of the original matrix (crosses) and those of the modified matrix (stars).

3.1 Background

We list here some concepts and results from differential topology. These constitute the mathematical basis of the homotopy methods.

Given a function

$$f: \ M \rightarrow N$$

where f is a smooth map between two smooth and real manifolds M and N, let $D_x f$ denote the derivative of f at x.

Definition 3.1 *y in N is said to be a regular value of f, if, for all x such that*

$$f(x) = y \tag{3.1}$$

we have,

$$D_x f: \ TM_x \rightarrow TN_y$$

is onto. TM_x is the tangent space to M at x, TN_y is the tangent space to N at y. In particular, when M and N are open this means that the Jacobian of f at x is full rank, for all x that satisfy 3.1.

Theorem 3.2 *A connected component of a 1-dimensional smooth manifold is diffeomorphic to S^1 (the unit circle in R^2) or to some interval of real numbers.*

Proof. See Milnor, [18]. □

Theorem 3.3 (Transversality Theorem) *Let f be a smooth map between two manifolds M and N of dimensions m and n respectively. Assume $m \geq n$. If y in N is a regular value of f, then*

$$X = \{x : \ f(x) = y\},$$

is a $m - n$ dimensional smooth submanifold of M.

Proof. See Milnor, [18]. □

If $m = n+1$ (which is the case of interest to us) then X above is a 1-dimensional manifold. Therefore by Theorem 3.2 X is a disjoint union of smooth paths some of which may be closed (those diffeomorphic to S^1).

3.2 Homotopy methods

Let f be a mapping,

$$f: \ D \subset R^m \rightarrow R^n.$$

We want to find a solution in D for the problem

$$f(x) = 0. \tag{3.2}$$

Newton's method and its variants are local methods: they require "good" initial guesses. With homotopy methods, intermediate problems $P_0, \ldots, P_r$ are defined such that P_i is a modest perturbation of P_{i-1}. A known solution of P_{i-1} is then used as an initial guess for a local method to solve P_i. Local methods are likely to converge here since P_i is slight perturbation of P_{i-1}. More precisely, define

$$H: \ [0,1] \times D \subset R^{m+1} \to R^n,$$

such that

$$H(1,x) = f(x),$$

and at least one solution of $H(0,x)$ is known. Let H also be chosen in a way that a curve connecting the particular solution of $H(0,x) = 0$ to a solution of $H(1,x) = 0$ can be shown to exist. Sufficient conditions for this are for instance those that make Theorem 3.3 applicable ($m = n+1$, 0 is a regular value of H). Then the intermediate problems mentioned above correspond to intermediate values of t; namely, we choose a sequence $t_0, \ldots, t_r$, such that

$$0 = t_0 < \cdots < t_r = 1$$

and we solve

$$H(t_i, x) = 0,$$

for each i. A very (perhaps the most) common choice for H is:

$$H(t,x) = (1-t)f_0(x) + tf(x), \tag{3.3}$$

where f_0 is a smooth mapping such that a solution of:

$$f_0(x) = 0$$

is known.

3.3 Homotopy methods for the solution of the eigenproblem

Given a matrix A, an eigenpair (x, λ) of A can be thought of as a solution to the following polynomial system:

$$\begin{cases} Ax - \lambda x = 0 \\ L(x) = 0 \end{cases} \tag{3.4}$$

where $L(x)$ is a scalar equation. Early work on homotopy methods for the solution of the eigenproblem seems to have considered 3.4 as only an element of a more general class of polynomial systems: the class of all polynomial systems, or the class of all deficient polynomial systems. A deficient polynomial system is one with fewer roots than its total degree (the product of the degrees of the individual polynomials making up the system). The polynomial system in 3.4 is deficient since it cannot have more than n distinct roots whereas its total degree is 2^n.

Many of these methods use the homotopy in 3.3 with $f_0(x)$ being a random polynomial system having the same total degree as the system in 3.4, i.e., 2^n. Since $f_0(x)$ is chosen at random it can be expected to have 2^n distinct roots. n distinct paths are then shown to connect n of the roots of f_0 to the eigenpairs of the matrix, and the method then consists in following the n curves. Since these n roots are but a small proportion of the total number of the roots of f_0 (especially for large n), we see that the probability of starting with the wrong initial guess (one that is not connected by a path to an eigenpair) is quite high. These methods are therefore

highly inefficient and indeed were never thought of as serious eigenproblem solvers.

Recently, Chu [3] proposed a homotopy algorithm for the symmetric eigenproblem where he let $f_0(x)$ be a (simple) eigenproblem. He defined the following homotopy:

$$H: \ R \times R^n \times R \ \rightarrow R^n \times R$$

with

$$\begin{aligned} H(t,x,\lambda) &= (1-t)(Dx - \lambda x, (1-x^T x)/2) + t(Ax - \lambda x, (1-x^T x)/2) \\ &= ((1-t)D + tA)x - \lambda x, (1-x^T x)/2) \end{aligned}$$

where A is symmetric tridiagonal with non zero off-diagonal elements; D is a diagonal matrix with distinct diagonal entries. Clearly,

$$H(0,x,\lambda) = (Dx - \lambda x, (1-x^T x)/2),$$

and so the zero set for $t = 0$ consists of the eigenpairs of D which are already available. Also

$$H(1,x,\lambda) = (Ax - \lambda x, (1-x^T x)/2),$$

and so for $t = 1$ the zero set consists of the eigenpairs of A. In general, a triplet (t,x,λ) belongs to the zero set of H if and only if (x,λ) is an eigenpair of the matrix $(1-t)D + tA$. The second component of the right hand side in both cases imposes a normalizing condition on the eigenvector, namely, that its euclidean norm be one. Chu then showed that the zero set of H consisted of paths connecting eigenpairs of D with eigenpairs of A. and the paths were non intersecting. We sketch the proof: the Jacobian of H is full rank for all (t,x,λ) such that $H(t,x,\lambda) = 0$. Therefore zero is a regular value of H. Now finish up with Theorems 3.3 and 3.2. He also shows that no curves diffeomorphic to S^1 are contained in the zero set.

Chu's approach generalizes easily to the non-symmetric eigenproblem. We can use virtually the same homotopy with A being an upper-Hessenberg matrix and D being a matrix whose eigensystem is relatively cheap to compute. Then the following procedure could be used to follow the paths from the eigenpairs of D to those of A if such paths can be shown to exist: given an appropriate subdivision of the interval [0,1]

$$0 = t_0 < \cdots < t_r = 1, \tag{3.5}$$

let (x_i, λ_i) be such that

$$H(t_i, x_i, \lambda_i) = 0.$$

Use Newton's method to find a solution to

$$H(t_{i+1}, x, \lambda) = 0,$$

starting from (x_i, λ_i). This procedure can be shown to converge under certain conditions. The following theorem is due to Avila [19].

Theorem 3.4 *Assume zero is a regular value of H, and that a continuous solution $(x(t), \lambda(t))$ is known to exist; then there exists a subdivision as in 3.5 and integers $m_1, \ldots, m_r$ such that the above procedure converges with m_i Newton iterations at the i^th step.*

It can be shown that the m_i's can be chosen to be one, provided the subdivision is modified accordingly. The theorem provides only sufficient conditions for convergence that are by no means necessary. As an immediate consequence we have the following result.

Theorem 3.5 *Assume $(1-t)D + tA$ has a simple spectrum for all t in the unit interval; then H satisfies the conditions of the previous theorem.*

Proof. The Jacobian of H at a triplet (t, x, λ) is

$$D_{(t,x,\lambda)} = \begin{pmatrix} (A-D)x & D + t(A-D) - \lambda & -x \\ 0 & -x^T & 0 \end{pmatrix}.$$

We will show that the last $n+1$ columns of the Jacobian when (t, x, λ) is in the zero set are linearly independent. Assume the converse; then there exists $(y, \mu)^T$ such that

$$\begin{pmatrix} D + t(A-D) - \lambda & -x \\ -x^T & 0 \end{pmatrix} \begin{pmatrix} y \\ \mu \end{pmatrix} = 0.$$

It is easy to see that if μ is zero then y is an eigenvector of $(1-t)D + tA$ that is orthogonal to x, and that if $\mu \neq 0$ then x is an eigenvector of grade two, contradicting in both cases the hypothesis on the spectrum of $(1-t)D+tA$. Therefore the Jacobian has full rank at every (t, x, λ) in the spectrum and hence 0 is a regular value of H. $(x(t), \lambda(t))$ is also continuous as a function of t [13, 15]. □

It is in general impossible to ascertain when working in finite precision arithmetic that a given eigenvalue is multiple: roundoff errors will cause a multiple eigenvalue of the matrix to be perturbed into a cluster of very close eigenvalues. The theorem then applies in more general cases. However, nearly singular Jacobians can arise in the case of pathologically close eigenvalues.

A non-symmetric eigensolver based on this scheme has been attempted by [17]: the method lacks though a good step size estimator ("guessing" the subdivision in 3.5).

3.4 Final remarks

Our experience with the homotopy scheme above has shown that the choice of $t_1 = 1$ is sufficient for the scheme to converge if the matrix D is chosen as a rank one perturbation of A, where A is reduced to upper-Hessenberg form first. Therefore,

in the remainder of this thesis we will be presenting our study of the above scheme when t_1 is chosen to be one, i.e., when Newton iterations are applied to find the eigenpairs of A starting from the eigenpairs of D, with D obtained from the upper-Hessenberg A by zeroing a subdiagonal element. The advantage of this choice of t_1 is clear; unnecessary intermediate steps are avoided.

CHAPTER 4

The Algorithm

Given a matrix H, an eigenpair (x_0, λ_0) of H can be thought of as a solution to the following polynomial system

$$(S)\begin{cases} Hx - \lambda x = 0 \\ L(x) = 1 \end{cases}$$

where $L(x)$ is a scalar equation. Here, we set $L(x) = e_s^T x$, where e_s is the s^{th} unit vector. Let

$$F_s(x, \lambda) = \begin{pmatrix} Hx - \lambda x \\ e_s^T x - 1 \end{pmatrix}.$$

Then, finding an eigenpair of H reduces to finding a zero of F_s. In what follows, unless otherwise mentioned, H is assumed to be a real, unreduced (no zeros on the subdiagonal), upper-Hessenberg matrix of order n. This is no restriction on the type of problems we want to solve, since if H has a zero on the subdiagonal then, finding its eigenvalues reduces to finding those of the blocks on the diagonal. Indeed, if

$$H = \left(\begin{array}{c|c} H_{11} & H_{12} \\ \hline 0 & H_{22} \end{array}\right),$$

then

$$Det(H - \lambda I) = Det(H_{11} - \lambda I) Det(H_{22} - \lambda I).$$

More generally, if we have m zeros on the subdiagonal, the characteristic polynomial of H is the product of those of the diagonal blocks $H_{11},\ldots,H_{m+1,m+1}$, of H and hence we have

$$\sigma(H) = \sigma(H_{11}) \cup \cdots \cup \sigma(H_{m+1,m+1}), \tag{4.1}$$

where $\sigma(H)$ is the spectrum of H. We note also that as a consequence of our assumption that H is unreduced, the geometric multiplicity of an eigenvalue of H, i.e., the dimension of the subspace generated by the eigenvectors corresponding to λ is well determined. Indeed we have the following.

Theorem 4.1 *Let H be an upper-Hessenberg matrix with m zeros on the subdiagonal. Then the geometric multiplicity g of an eigenvalue of H is such that*

$$1 \leq g \leq m + 1.$$

Proof. Let λ be an eigenvalue of H, and let $H_{11}, \ldots, H_{m+1,m+1}$ be the diagonal blocks of H. Each of these blocks is an unreduced upper-Hessenberg matrix. By 4.1, λ is an eigenvalue of one diagonal block of H, say $H_{i,i}$. Let us denote the order of $H_{i,i}$ by n_i, and the identity of order n_i by I_i. Then by taking linear combinations of these, it is quite easy to see that the first $n_i - 1$ columns of $H_{i,i} - \lambda I_i$ are linearly independent; this follows solely from the fact that $H_{i,i}$ is unreduced. The rank of $H_{i,i} - \lambda I_i$ is therefore at least $n_i - 1$ and hence equal to $n_i - 1$. The geometric multiplicity g_i of λ as an eigenvalue of $H_{i,i}$ is equal to the dimension of the null space of $H_{i,i} - \lambda I_i$ which is one. The assertion of the theorem follows from the fact that

$$g \leq \sum g_i.$$

Indeed, let $x_1, \ldots, x_k$ be the eigenvectors of H corresponding to λ. Let us partition each of these into subvectors of lengths matching the orders of the diagonal blocks of H. Let $x_l^{(i)}, l = 1, \ldots, k, i = 1, \ldots, m+1$ be the subvector of x_k corresponding to the block H_i. Then we have

$$H_{m+1,m+1} x_l^{(m+1)} = \lambda x_l^{(m+1)},$$

for $l = 1, \ldots, k$. But $g_{m+1} \leq 1$ and so there exists at most one l, say $l = 1$ without loss of generality, for which $x_l^{(m+1)} \neq 0$. Then

$$H_{m,m} x_l^{(m)} = \lambda x_l^{(m)},$$

for $l = 2, \ldots, k$. Repeating the argument made earlier for the m^{th} block and proceeding down to the first, we see that at most $m + 1$ eigenvectors can be non zero, i.e., $k \leq m + 1$. □

We will assume for now that H has a simple spectrum. We can write H as:

$$H = \left(\begin{array}{c|c} H_{11} & H_{12} \\ \hline \alpha e_1^{(k)} e_k^{(k)T} & H_{22} \end{array} \right),$$

where H_{11} and H_{22} are upper-Hessenberg of dimensions $k \times k$ and $n - k \times n - k$, respectively; $\alpha = h_{k+1,k}$, and $e_i^{(k)}$ is the i^{th} unit vector of length k.

Let $H_0 = H - \alpha e_{k+1}^{(n)} e_k^{(n)T}$, then

$$H_0 = \left(\begin{array}{c|c} H_{11} & H_{12} \\ \hline 0 & H_{22} \end{array} \right),$$

and $\sigma(H_0) = \sigma(H_{11}) \cup \sigma(H_{22})$. The algorithm can then be described as follows: We first find the k eigenpairs of H_{11} and the $n - k$ eigenpairs of H_{22} by some method, perhaps the QR algorithm. These eigenpairs are then used to construct initial approximations to the eigenpairs of H in the following way: if λ is an eigenvalue of H_{11} and x is the corresponding eigenvector, then λ is viewed as an approximate eigenvalue of H with the corresponding approximate eigenvector taken to be $\begin{pmatrix} x \\ 0 \end{pmatrix}$, where $n - k$ zeros are appended to x. Note that $\begin{pmatrix} x \\ 0 \end{pmatrix}$ is an exact eigenvector of H_0 corresponding to λ. On the other hand, if λ is an eigenvalue of H_{22} and x is the corresponding eigenvector, then $\begin{pmatrix} 0 \\ x \end{pmatrix}$, where k zeros are prefixed to x is taken as an approximate eigenvector to H corresponding to the approximate eigenvalue λ. If

λ is not an eigenvalue of H_{11} then the last $n-k$ components of $\begin{pmatrix} 0 \\ x \end{pmatrix}$, i.e., those of x, are the trailing components of an exact eigenvector of H_0 corresponding to λ, namely: $\begin{pmatrix} (H_{11}-\lambda)^{-1}H_{12}x \\ x \end{pmatrix}$. However, if λ is an eigenvalue of H_{11}, and hence a necessarily multiple eigenvalue of H_0, then if λ has geometric multiplicity one, no eigenvector of H_0 will have the components of x as its trailing components.

Having constructed these initial guesses, we now use them as starting points for Newton's method to find the zeros of F_s, i.e., the eigenpairs of H. Hence, given an approximate zero (x,λ) of F_s the next approximation is:

$$x' = x + y; \lambda' = \lambda + \mu,$$

where (y,μ) is the solution of the system

$$\begin{pmatrix} H-\lambda I & -x \\ e_s^T & 0 \end{pmatrix} \begin{pmatrix} y \\ \mu \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix}, \tag{4.2}$$

where $r = \lambda x - Hx$.

Newton's method comes into this problem in a rather "natural" way. Indeed, suppose that (x,λ) is an approximate eigenpair of H:

$$Hx \approx \lambda x,$$

and let us find a way to compute a correction (y,μ) to this approximate eigenpair. Clearly, (y,μ) should satisfy:

$$H(x+y) = (\lambda+\mu)(x+y).$$

Rearranging the latter equation yields:

$$(H-\lambda I)y - \mu x = \lambda x - Hx + \mu y. \tag{4.3}$$

Now if we ignore the second order term μy, and if we impose a normalization condition on x, say $x_s = 1$ and assume that the desired eigenvector should satisfy the same condition, then (y, μ) is the solution of:

$$\begin{pmatrix} H - \lambda I & -x \\ e_s^T & 0 \end{pmatrix} \begin{pmatrix} y \\ \mu \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix},$$

with r as above. But, this is the same equation that we obtain when a Newton iteration is applied to the function F_s to find a correction to (x, λ), as we can see from 4.2. We note here a result from [5], where the author studied an iterative refinement technique to compute the correction (y, μ) to (x, λ) from equation 4.3 (i.e., without ignoring the second order term) and proved that in exact arithmetic, that method and Newton's method produce the same final iteration.

So far we have not attempted to answer the question of which subdiagonal entry to introduce the zero. We will use first order perturbation theory in order to shed some light on this issue. Let $E = -\alpha e_{k+1} e_k^T$, where as above: $\alpha = h_{k+1,k}$ (note that $H + E = H_0$, introduced above). Then classical results from function theory ([16], v. 2, pp. 119-134) allow us to state that in a small neighborhood of zero, we have:

$$(H + \epsilon E)x(\epsilon) = \lambda(\epsilon)x(\epsilon),$$

for all ϵ in that neighborhood and for differentiable $(x(\epsilon), \lambda(\epsilon))$ corresponding to a simple eigenpair (x, λ) of H. Clearly: $x(0) = x$ and $\lambda(0) = \lambda$. Let y^H be the left eigenvector of H corresponding to λ. Then differentiating both sides we have (with primes indicating derivatives):

$$Hx'(\epsilon) + Ex(\epsilon) + \epsilon Ex'(\epsilon) = \lambda'(\epsilon)x(\epsilon) + \lambda(\epsilon)x'(\epsilon).$$

Multiplying by y^H and setting $\epsilon = 0$ we get

$$y^H Ex = \lambda'(0) y^H x,$$

and therefore

$$|\lambda'(0)| = \frac{|y^H E x|}{|y^H x|} = \frac{|\alpha||y_{k+1}||x_k|}{|y^H x|}. \tag{4.4}$$

The quantities that vary with k in this expression for the rate of change of λ, are the factors in the numerator. However, $|\alpha|, |y_{k+1}|$ and $|x_k|$ are not really independent of one another: indeed, if $\alpha = 0$, then at least one of y_{k+1} or x_k is zero. Hence for α small, we can expect one of y_{k+1} and x_k to be correspondingly small, since the components of the eigenvectors vary continuously. Therefore, we have found it sufficient in practice to look for the smallest subdiagonal entry in a pre-specified range, and accept it as the subdiagonal entry (in that range) with respect to which the eigenvalues of the matrix are least sensitive, and set it equal to zero.

An outline of the algorithm follows:

Algorithm 4.2 *Given an upper-Hessenberg matrix H, the following algorithm computes the eigensystems of two submatrices of H and uses them as initial guesses for starting Newton iterations for determining the eigensystem of H.*

Determine subdiagonal element α where $H = \left(\begin{array}{c|c} H_{11} & H_{12} \\ \hline 0 \;\; \alpha & H_{22} \end{array}\right)$, should be split;

Determine initial guesses from eigensystems of the 2 diagonal blocks H_{11} and H_{22};

For each initial guess (λ_i, x_i) iterate until convergence:

$$\begin{pmatrix} H - \lambda_i I & -x_i \\ e_s^T & 0 \end{pmatrix} \begin{pmatrix} y \\ \mu \end{pmatrix} = \begin{pmatrix} r_i \\ 0 \end{pmatrix}; \lambda_i \leftarrow \lambda_i + \mu \;\; ; \;\; x_i \leftarrow x_i + y;$$

end

Check for duplicates and deflate if necessary;

More will be said about the last step, deflation, in chapter 5. The algorithm just described is inherently parallel. Indeed, the eigensystems of H_{11} and H_{22} can

be computed in parallel. After which Newton's method can be started with different initial guesses on different processors. Furthermore, the dividing process can be applied recursively to obtain yet smaller subproblems: i.e., H_{11} and H_{22} can be divided each into subproblems. This will permit the use of more processors. However, care must be exercised if the algorithm is to be kept efficient. More will be said about this in chapter 8 (work estimate). In practice, the original matrix will be dense and we will need to reduce it to upper-Hessenberg form as a first step; this can be done in a stable fashion through a sequence of orthogonal similarity transformations.

A study of some sufficient conditions guaranteeing the convergence of our method will be touched upon in chapter 6. Now assuming that the algorithm converges, it could happen that the same eigenpair of H is obtained more than once: i.e., starting from two (or more) distinct initial approximations Newton's method converges to the same eigenpair of H. We have investigated methods to obtain further eigenpairs of H should this happen (see chap. 5).

We end this chapter with some implementational details. Our algorithm will accept (x, λ) as an eigenpair of H when

$$\frac{\|Hx - \lambda x\|}{\|x\|\|H\|} < f(n)\epsilon,$$

where $f(n)$ is a modest function of n. Under these conditions ([13]), (x, λ) is an exact eigenpair of a matrix obtained from H by a slight perturbation: Indeed,

$$(H + \frac{1}{x^H x} r x^H)x = \lambda x,$$

where $r = \lambda x - Hx$.

Significant savings can be realized in practice by virtue of the following result.

Theorem 4.3 *Let (x, λ) and $(\bar{x}, \bar{\lambda})$ be two complex conjugate initial approximations. Then starting from these, and assuming it converges, Newton's method will do so to*

two complex conjugate zeros of F_s, or the same real zero.

Proof. To prove this it suffices to look at one iteration and show that when the current approximations are complex conjugate, Newton's method yields two complex conjugate corrections, or the same real correction. The system we have to solve starting from (x, λ) is 4.2. For $(\bar{x}, \bar{\lambda})$ this becomes:

$$\begin{pmatrix} H - \bar{\lambda} I & -\bar{x} \\ e_s^T & 0 \end{pmatrix} \begin{pmatrix} y' \\ \mu' \end{pmatrix} = \begin{pmatrix} \bar{r} \\ 0 \end{pmatrix}.$$

But this is the same linear system obtained when the conjugate of both sides in 4.2 is taken, knowing that the matrix is real and hence $\bar{H} = H$. Therefore: $y' = \bar{y}$ and $\mu' = \bar{\mu}$. □

When starting from a real initial guess, only *real* corrections are computed. Therefore the imaginary part of the eigenvalue should be perturbed if convergence to a complex eigenpair is to be made possible. In practice we have done this when real arithmetic does not lead to convergence after a pre-specified number of corrections. To illustrate the situation when perturbing the imaginary part of the eigenvalue is necessary we mention the very simple case of a 2×2 real matrix whose eigenvalues are complex.

CHAPTER 5

Deflation

When Newton's method is applied to solve the eigensystem of a matrix H, it is possible that starting from distinct initial guesses we converge to the same eigenpair of H. In fact a naive implementation of the algorithm in chapter 4 may result in many eigenvalues being found multiple times, and some other eigenvalues not being found at all. To avoid this unwanted situation, we included a deflation step in our algorithm which is designed to obtain further zeros using Newton's method. In this chapter we propose various methods for doing this. A common drawback that these methods have is that they tend to serialize the computation. It has been our experience that this happens infrequently, less than 5% of the time in our tests.

All the methods of deflation we propose but one (method of 5.2), have their theoretical basis in the following theorem.

Theorem 5.1 *Let A be a matrix (no assumption on its structure). Let n be the order of A, and (x, λ) be an (exact) eigenpair of A. Then for any non-singular matrix Q satisfying*

$$Q^{-1}x = e_i,$$

the i^{th} column of $Q^{-1}AQ$ is equal to λe_i (note that $\sigma(Q^{-1}AQ) = \sigma(A)$). In particular, if $i = 1$ or $i = n$, then

$$Q^{-1}AQ = [\lambda e_1, B_1] \text{ or } Q^{-1}AQ = [B_2, \lambda e_n],$$

respectively; where B_1 and B_2 are $n \times (n-1)$ matrices.

Proof. Indeed, the i^{th} column of $Q^{-1}AQ$ is $Q^{-1}AQe_i$ and

$$Q^{-1}AQe_i = Q^{-1}AQ(Q^{-1}x) = Q^{-1}Ax = \lambda Q^{-1}x = \lambda e_i.$$

□

Given an $n \times n$ matrix H (we will assume for simplicity that it has no multiple eigenvalues) and a known eigenpair (x, λ) of H, we will seek in general a matrix H' such that

$$\sigma(H') = \sigma(H) - \{\lambda\},$$

in order to solve the problem we set to ourselves at the beginning of the chapter. In our algorithm, we first construct a matrix $\tilde{H}$ of the form $Q^{-1}HQ$, i.e., similar to the original matrix H, and such that

$$Q^{-1}x = e_1 \ \ or \ \ Q^{-1}x = e_n.$$

H' is then the leading principal submatrix of $\tilde{H}$ of order $(n-1)$, or the lower principal submatrix of $\tilde{H}$ of the same order. One major factor that influenced our choice of Q was the need that the structure of H be inherited by H'.

5.1 Deflation ysing elementary transformations

We now describe one possible deflating similarity transformation. Since we are assuming H to be upper-Hessenberg with no zeros on the subdiagonal, then $x_n \neq 0$ and the elementary transformation

$$M = [e_1, \ldots, e_{n-1}, x]$$

i.e.,

$$M = \begin{pmatrix} 1 & & & x_1 \\ & \ddots & & \vdots \\ & & 1 & x_{n-1} \\ 0 & & & x_n \end{pmatrix}. \tag{5.1}$$

is non singular. The inverse of this matrix is

$$M^{-1} = \begin{pmatrix} 1 & & & -\frac{x_1}{x_n} \\ & \ddots & & \vdots \\ & & 1 & -\frac{x_{n-1}}{x_n} \\ 0 & & & \frac{1}{x_n} \end{pmatrix}. \tag{5.2}$$

It is easy to see that

$$M^{-1}x = e_n.$$

Now we let

$$\tilde{H} = M^{-1}HM.$$

Then we know that the last column of $\tilde{H}$ is λe_n. Furthermore the leading principal submatrix of order $n-1$ of $\tilde{H}$, which we will call H', is upper Hessenberg, has the property that

$$\sigma(H') = \sigma(H) - \{\lambda\}$$

and differs from the leading $n-1 \times n-1$ principal submatrix of H in the last column only. In fact, if we let h_{n-1} be the last column of the leading principal submatrix of H of order $n-1$, then it is straightforward to verify that the last column of H' is $h_{n-1} - h_{n,n-1}x'$, where

$$x' = \begin{pmatrix} \frac{x_1}{x_n} \\ \vdots \\ \frac{x_{n-1}}{x_n} \end{pmatrix}.$$

The strategy we have just described is given in [23] and applies regardless of whether the eigenpair (x, λ) is real or not. However, when (x, λ) is real we get a real matrix H'. In the case when (x, λ) is not real, the last column of H' alone is not real since, as we remarked earlier, the other columns are those of a leading principal submatrix of H. Hence the leading principal submatrix of H' of order $n-2$ is real. Also in this case the complex conjugate of (x, λ), $(\bar{x}, \bar{\lambda})$, is an eigenpair of H, and

therefore $\bar{\lambda}$ is an eigenvalue of $\widetilde{H}$; a corresponding eigenvector is

$$M^{-1}\bar{x} = \begin{pmatrix} \bar{x}_1 - x_1\frac{\bar{x}_n}{x_n} \\ \vdots \\ \bar{x}_{n-1} - x_{n-1}\frac{\bar{x}_n}{x_n} \\ \frac{\bar{x}_n}{x_n} \end{pmatrix}.$$

Since $\sigma(H') = \sigma(H) - \{\lambda\}$, $\bar{\lambda}$ is an eigenvalue of H' and a corresponding eigenvector is the vector $\tilde{x}$ of length $n-1$, whose components are the first $n-1$ components of a scalar multiple of $M^{-1}\bar{x}$:

$$\tilde{x} = \begin{pmatrix} (\bar{x}_1 x_n - x_1 \bar{x}_n)/i \\ \vdots \\ (\bar{x}_{n-1} x_n - x_{n-1}\bar{x}_n)/i \end{pmatrix},$$

where $i^2 = -1$. This is due to the special structure of $\widetilde{H}$. Now recall that H' has no zeros on the subdiagonal, and so we know that the last component of $\tilde{x}$ is nonzero. Note that $\tilde{x}$ is real. We can carry out a deflation that produces a matrix H'' of order $n-2$ with the property that

$$\sigma(H'') = \sigma(H') - \{\bar{\lambda}\},$$

in the same way we obtained H' from H using the elementary transformation of order $n-1$

$$M' = \begin{pmatrix} 1 & & \tilde{x}_1 \\ & \ddots & \vdots \\ & & \tilde{x}_{n-2} \\ 0 & & \tilde{x}_{n-1} \end{pmatrix}.$$

By a previous remark about this deflation strategy, we know that H'' differs from the leading principal submatrix of H' of order $n-2$ (which is real) in the last column only. The last column of H'' is $h'_{n-2} - h_{n-1,n-2}x''$, where h'_{n-2} is the last column of the leading principal submatrix of H' and x'' is the vector whose components are the first $n-2$ components of $\tilde{x}$. Since all of the quantities involved are real, H'' is real.

We remark here, as can be readily realized, that H' (or H'') is quite cheap to obtain in practice once an eigenpair of H is available. It requires $O(n)$ operations consisting of a vector normalization, a scalar-vector multiplication and a vector-vector addition. However the conditioning of the matrix M might raise concern. Indeed:

$$cond_\infty(M) = \|M\|_\infty \|M^{(-1)}\|_\infty \approx max(|x_i|) max\left(\frac{|x_i|}{|x_n|}\right)$$

which can be large if $x_n \ll x_i$. Having noted this, it is clear that the ill-conditioning of M can be easily detected, and therefore one of the more stable (and costlier) methods which we introduce next and in the following chapter can be used.

It is possible to prevent the ill-conditioning of M from bearing on the algorithm by avoiding a similarity transformation. More precisely, the eigenvalue problem we want to solve can be thought of as a generalized eigenvalue problem,

$$Hx = \lambda Bx,$$

with $B = I$. We want to find $\sigma(H) = \sigma(H, I)$. Now we know that given any non-singular M and M',

$$\sigma(H, I) = \sigma(M'HM, M'M).$$

Given a particular eigenpair (x, λ) we would like to choose M and M' in a way that solves the problem we set to ourselves at the beginning of this chapter, namely, we want to reduce the problem to one where λ is no longer in the spectrum. A closer look at the similarity transformation introduced above, reveals that its deflating property is due to the fact that $M^{-1}x = e_n$. But then this is not the only matrix that can be used to accomplish this. In fact the following matrix M' can be chosen to do the task:

$$M' = DM^{(-1)},$$

where D is the diagonal matrix with the following entries

$$\begin{cases} d_i = 1, \ if \ \frac{|x_i|}{|x_n|} \leq 1 \\ d_i = -\frac{|x_n|}{|x_i|}, \ if \ \frac{|x_i|}{|x_n|} > 1 \end{cases}$$

i.e., D is chosen so that all the entries in M' are less than or equal to one. Then we have

$$M'HM = \left(\begin{array}{c|c} H' & 0 \\ \hline \alpha & \\ \hline 0 & \gamma \end{array}\right), \quad M'M = \left(\begin{array}{c|c} D' & 0 \\ \hline 0 & \delta \end{array}\right), \tag{5.3}$$

with

$$\lambda = \gamma/\delta.$$

Also,

$$\sigma(H', D') = \sigma(H, I) - \{\lambda\}$$

and therefore by this transformation λ has been "removed" from the spectrum. Working on the solution of a generalized eigenvalue problem from this point on will not in general cause any dramatic increase in the cost of the algorithm mainly because D' is diagonal. A Newton step with this problem involves the following computation

$$\begin{pmatrix} H' - \lambda_i D' & -D'x_i \\ e_s^T & 0 \end{pmatrix} \begin{pmatrix} y \\ \mu \end{pmatrix} = \begin{pmatrix} r_i \\ 0 \end{pmatrix}, \quad \lambda_i \leftarrow \lambda_i + \mu \ ; \ x_i \leftarrow x_i + y \tag{5.4}$$

where

$$r_i = \lambda_i D' x_i - H x_i.$$

Clearly the previous computation involves an $O(n)$ increase in the cost of one step: this comes from the multiplications by D'. The details on how equation 5.4 is derived are given in chapter 10. If λ is complex, then after deflating λ and its conjugate $\bar{\lambda}$ the resulting matrices H' and D' are complex in general. This is the major drawback of this method.

The particular case when H is tridiagonal

It is possible to reduce the original non-symmetric matrix to tridiagonal form [9, 10]. We assume in this section that this has been done and make a corresponding change in our notation and denote the matrix by T. A tridiagonal matrix is but a particular upper-Hessenberg matrix and hence everything that has been developed in the upper-Hessenberg case applies with no modification whatsoever to the tridiagonal case. We include here some remarks about the deflation step in the tridiagonal case.

Let T be tridiagonal with no zeros on the subdiagonal, and no zeros on the super diagonal and (x, λ) an eigenpair of T. Since T has no zeros on the subdiagonal, $x_n \neq 0$; we will assume that $x_n = 1$. Then a similarity transformation with the matrix M in 5.1 results as in the upper-Hessenberg case in a matrix ([23])

$$\tilde{T} = M^{-1}TM,$$

whose leading principal submatrix of order $n-1$, T' satisfies $\sigma(T') = \sigma(T) - \{\lambda\}$. T' has the following form:

$$T' = \begin{pmatrix} * & * & & * & 0 \\ * & * & & & \\ & & \ddots & & \\ & & \ddots & \vdots & \\ & & \ddots & & \\ & & & * & \\ 0 & & & * & * \end{pmatrix}.$$

The tridiagonal structure of T allows for another elementary transformation to be used in the deflation. Since T has no zeros on the super-diagonal, $x_1 \neq 0$. Therefore we can scale so that $x_1 = 1$. Now we take, $M = [x, e_2, \ldots, e_n]$, i.e.,

$$M = \begin{pmatrix} 1 & & 0 \\ x_2 & & \\ \vdots & \ddots & \\ x_n & & 1 \end{pmatrix}. \tag{5.5}$$

With $\tilde{T}$ as above we see that in this case the first column of $\tilde{T}$ is λe_1. Therefore if we let T' be the lower $n-1 \times n-1$ principal submatrix of $\tilde{T}$ then again: $\sigma(T') = \sigma(T) - \{\lambda\}$. Moreover, T' differs from the lower $n-1 \times n-1$ principal submatrix of T in the first column only; indeed, T' in this case has the following form:

$$T' = \begin{pmatrix} * & * & & & 0 \\ * & * & & & \\ & & \ddots & & \\ \vdots & \vdots & \ddots & & \\ & & \ddots & & \\ & & & & * \\ 0 & * & & * & * \end{pmatrix}.$$

In practice, when T is tridiagonal, we choose M according to 5.1 if $|x_n| \geq |x_1|$ and according to 5.5 otherwise. Note here that the choice of M according to 5.5 is not attractive in the case of an upper-Hessenberg matrix H, since then the matrix H' (as above) is a full matrix.

5.2 Deflation with help from the left eigenvector

The difficulties associated with the method introduced in the previous section necessitated that we look for alternatives. These will be the subjects of this and the following section.

We introduce here two methods of deflation having a feature in common that they use the left eigenvector corresponding to the eigenvalue to be deflated.

5.2.1 Projection onto complementary spaces

The method we introduce now is different in spirit from the one in the previous section, in that no attempt is made at modifying the matrix.

Assume that (x, λ) is an eigenpair of H that was obtained with the algorithm

we described earlier and that λ is simple:

$$Hx = \lambda x.$$

No assumption is made about the remaining eigenvalues of H.

Let (x, X) be such that:

$$(x, X)^{-1} H (x, X) = \begin{pmatrix} \lambda & 0 \\ 0 & J \end{pmatrix}$$

where the right hand side is the Jordan canonical form of H. Now set:

$$(x, X)^{-1} = \begin{pmatrix} y^H \\ Y \end{pmatrix}.$$

Then it is clear that y^H is a left eigenvector of H corresponding to λ and furthermore that:

$$y^H X = 0.$$

This property can be used to modify Newton's method to avoid converging to the eigenpair (x, λ) a second time. Indeed, given λ we can compute the left eigenvector y^H corresponding to it and use it to confine the current eigenvector to the space X. Therefore we can expect to converge to an eigenvector linearly independent of x and hence corresponding to a different eigenpair (since (x, λ) was assumed to be simple). When (x, λ) has been obtained once, our algorithm for avoiding it then consists of the following major steps:

1) Compute the left eigenvector y^H corresponding to λ; scale so that $\|y\|_2 = 1$.

2) Given the current eigenpair (z, μ) compute the Newton correction described in chapter 2.

3) let z' be the approximate eigenvector obtained after adding the correction from 2) to z, choose the next eigenvector z'' as:

$$z'' = (I - yy^H)z'.$$

In step 3) we are just projecting z' onto the space X. The need might arise to do more than one projection if convergence to more than one known eigenvalue is to be avoided.

It is obvious why we need the known eigenpair to be simple for the algorithm just outlined to work. If λ is multiple then left eigenvectors are no longer necessarily orthogonal to X. In fact, the algorithm will be adversely affected if the eigenvalue λ is ill-conditioned, i.e., if $y^H x$ is very small. Indeed, in this case, if the current eigenvector $z = \alpha x + Xv$, where v is a vector of length $n-1$:

$$(I - yy^H)z = z - (y^H z)y = \alpha x + Xv - (\alpha y^H x)y \approx z,$$

showing that z is hardly modified by the projection and therefore suggesting that the algorithm will not necessarily prevent a second convergence to (x, λ).

The algorithm generalizes to the case when λ is multiple in the following way: let V be a right invariant subspace corresponding to λ. Let (V, V_c) be such that:

$$(V, V_c)^{-1} H (V, V_c) = \begin{pmatrix} J_\lambda & 0 \\ 0 & J \end{pmatrix},$$

where the right hand side is again the Jordan canonical form of H. J_λ is the Jordan block corresponding to λ; since H is assumed to be unreduced, there can be only one such block. If we set

$$(V, V_c)^{-1} = \begin{pmatrix} U^H \\ W \end{pmatrix},$$

then clearly U^H is a left invariant subspace corresponding to λ and furthermore:

$$U^H V_c = 0.$$

This last property will allow U^H to be used in much the same way the left eigenvector was used earlier. However, the practical usefulness of this method is restricted to

the case when the eigenvalue λ is simple: indeed, the problem of determining the invariant subspace associated with a multiple eigenvalue λ is an extremely difficult one and can be prohibitively expensive.

This method in its simplest form (using the left eigenvector) adds $O(n^2)$ work to the cost of finding one eigenpair distinct from (x, λ): this is the cost of computing the left eigenvector corresponding to λ; the cost of a single projection is $O(n)$.

5.2.2 Using the left eigenvector for deflation with elementary transformations

When the left eigenvector y corresponding to the eigenvalue λ has been computed (as a first step in the method of the previous section for example) it is possible to use it to construct an elementary transformation M as in 5.1 that produces a matrix H' with

$$\sigma(H') = \sigma(H) - \{\lambda\}.$$

The elementary matrix M is defined by rows in the following way

$$M = \begin{pmatrix} y^H \\ e_2^T \\ \vdots \\ e_n^T \end{pmatrix},$$

i.e.,

$$M = \begin{pmatrix} y_1 & \cdots & y_n \\ & 1 & \\ & \ddots & \\ 0 & & 1 \end{pmatrix}.$$

The inverse of M, $M^{(-1)}$ is then

$$M^{(-1)} = \begin{pmatrix} \frac{1}{y_1} & -\frac{y_2}{y_1} & \cdots & -\frac{y_n}{y_1} \\ & 1 & & \\ & & \ddots & \\ 0 & & & 1 \end{pmatrix}.$$

It can be verified that

$$M^{(-1)}HM = \left(\begin{array}{c|c} \lambda & 0 \\ \hline 0 & H' \end{array}\right),$$

where H' is $n-1 \times n-1$ and $\sigma(H') = \sigma(H) - \{\lambda\}$. H' differs from the lower $n-1 \times n-1$ principal submatrix of H in the first row only. An analysis similar to the one we did in 5.1 would reveal that the deflation of a complex eigenvalue and its conjugate would result in a real matrix with the remaining eigenvalues of H. We note lastly that this method is interesting only when $max(|y_i|/|y_n|)$ is moderate.

5.3 Other deflation approaches

We present now two very stable methods for deflating the matrix in order to "remove" a known eigenvalue λ. These methods might unfortunately result in the destruction of the upper-Hessenberg structure of the matrix, and if the eigenvalue in question is complex, will yield a complex matrix (even if the pair of conjugates is deflated).

5.3.1 Deflation with stabilized elementary transformations

The method we described in 5.1 can be refined in a way that contributes to its stability. This is achieved by interchanging two components of x and the corresponding columns and rows in H so that the last component of x is large enough. More precisely: Let x_s be the largest component of x (in absolute value) and let P_{ns} be the matrix obtained from the identity matrix by permuting the n^{th} and s^{th} columns. Then $(P_{ns}x, \lambda)$ is an eigenpair of the matrix $P_{ns}HP_{ns}$, since:

$$(P_{ns}HP_{ns})(P_{ns}x) = \lambda P_{ns}x.$$

Now scale the vector $P_{ns}x$ so that the last component is 1 and call that vector $\tilde{x}$. Then the elementary transformation

$$M = \begin{pmatrix} 1 & & \tilde{x}_1 \\ & \ddots & \vdots \\ 0 & & \tilde{x}_{n-1} \\ & & 1 \end{pmatrix}$$

can be used to deflate the matrix $P_{ns}HP_{ns}$ as before (Theorem 5.1).

Having thus deflated the matrix $P_{ns}HP_{ns}$, the leading principal submatrix of order $n-1$ of

$$M^{-1}P_{ns}HP_{ns}M, \tag{5.6}$$

call it H', has all the eigenvalues of H except λ (if λ is simple). However, H' is not upper-Hessenberg in general and therefore will be reduced back to Hessenberg form before applying Newton's iterations: this is meant to save on the cost of factorizing the Jacobian when solving the linear systems arising at each step of Newton's iteration. Note that it is only the trailing diagonal submatrix of order $n-s+1 \times n-s+1$ of H' that needs to be reduced and that if $s = n-1$ or $s = n$, then H' is upper-Hessenberg. s needs not be chosen so that x_s is the largest component of x (in absolute value). However, since the size of the matrix to be reduced to upper-Hessenberg form increases when s approaches 1, it is more advantageous to choose the largest s for which the ratios x_i/x_s are moderate. We wish therefore to define a threshold t for the size of these ratios on the basis of which s will be determined.

Let $\tilde{H}$ be the computed form of the matrix in 5.6. In ([23], chap. 9), Wilkinson established that if $\|x\|_\infty \leq 1$, then the eigenvalues of $\tilde{H}$ are the exact eigenvalues of a matrix $\tilde{H}'$ satisfying

$$\|H - \tilde{H}'\|_\infty \leq \|r\|_\infty + (n-1)\epsilon,$$

where r is the residual $\lambda x - Hx$ and ϵ is the machine epsilon. With no assumptions on the infinity norm of x, this inequality becomes:

$$\|H - \tilde{H}'\|_\infty \leq \|r\|_\infty + (n-1)\|x\|_\infty \epsilon. \tag{5.7}$$

Since in our algorithm, our computed eigenpairs have residuals on the order of $n\|H\|_\infty \epsilon$, we propose $\|H\|_\infty$ as a value for t.

In addition to destroying the structure of the matrix, the method of deflation in this chapter suffers from the fact that in the case when the eigenvalue to be deflated is non-real, the resulting matrix H' is complex and therefore will considerably increase the cost of finding subsequent eigenpairs if a Newton process is restarted from a real initial guess.

5.3.2 Deflation with orthogonal transformations

Another stable method for obtaining an upper-Hessenberg matrix H' with the property that it has all the eigenvalues of H except for λ is by way of plane rotations which we describe next [23]. We assume for now that the eigenpair (x, λ) is *exact*. The derivation of this method could proceed along the lines set by Theorem 5.1 ([2]). However we follow another path here.

The strategy consists of $n-1$ major steps, at each of which a new zero is introduced in the last column of $H - \lambda I$ starting from the bottom. The configuration at the beginning of the r^{th} step looks like

$$\tilde{H} = \begin{pmatrix} * & \cdots & * & * \\ * & & & \vdots \\ & \ddots & \vdots & * \\ & & * * & 0 \end{pmatrix}$$

with zeros in the last $r-1$ components of the last column. The r^{th} step then consists in a post-multiplication by G_r, where G_r is the (possibly complex) rotation in the

plane $(n-r, n)$ designed to annihilate the $(n-r, n)$ element:

$$G_r = \begin{pmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & & c_i & & -s_i \\ & & & & \ddots & \\ & & & s_i & & 1 \\ & & & & & & c_i \end{pmatrix},$$

where $\|c_i\|^2 + \|s_i\|^2 = 1$. This post-multiplication will affect columns $n-r$ and n only and therefore will not disturb zeros previously introduced in the last column. At the $(n-1)$ step, G_{n-1} which is constructed to zero the $(2, n)$ element, will also zero the $(1, n)$ element. Indeed, assume that after the $(2, n)$ element, has been zeroed we have some value α in the $(1, n)$ position; then we have:

$$(H - \lambda I)G_1 \ldots G_{n-1} = \begin{pmatrix} * & \cdots & * & \alpha \\ * & & & \\ & \ddots & \vdots & 0 \\ & & * & * \end{pmatrix}.$$

Now if we develop the determinant of this matrix by the last column we get:

$$\det[(H - \lambda I)G_1 \ldots G_{n-1}] = \alpha b_1 \ldots b_{n-1},$$

where b_i is the i^{th} subdiagonal element of

$$(H - \lambda I)G_1 \ldots G_{n-1}.$$

$\text{Det}[(H - \lambda I)G_1 \ldots G_{n-1}] = \det(H - \lambda I) = 0$, since $\det(G_i) = 1$ for all i. But $b_i \neq 0$ for all i, since we have assumed that H had no zeros on the subdiagonal and since post-multiplication by a G_i cannot but increase the modulus of a subdiagonal element in $H - \lambda I$. Thus we must have $\alpha = 0$ and therefore at the end of the $n-1$ steps just described the last column is zero. Let us set $\mathcal{G} = G_1 \ldots G_{n-1}$ to simplify the notation. Then:

$$\mathcal{G}^{-1} = \mathcal{G}^H = G_{n-1}^H \ldots G_1^H,$$

and it is straightforward to verify that the zeros of the last column of $(H-\lambda I)\mathcal{G}$ will be preserved when it is pre-multiplied by $\mathcal{G}^{-1}$, because the successive pre-multiplications by $G_i^T, i = 1, \ldots, n-1$, will preserve those zeros. Therefore the last column of

$$\tilde{H} = \mathcal{G}^H(H - \lambda I)\mathcal{G} + \lambda I$$

is equal to λe_n. The eigenvector of $\tilde{H}$ corresponding to λ is e_n. Note that $\tilde{H}$ is not upper-Hessenberg in this case though. Indeed, non zero elements will be introduced in the last row of $\tilde{H}$; in fact ,

$$\tilde{H} = \begin{pmatrix} * & \cdots & * & \\ * & & & 0 \\ & \ddots & \vdots & \\ & & * * & \\ * & \cdots & * & \lambda \end{pmatrix}.$$

This is not disturbing since if H' is the leading principal submatrix of $\tilde{H}$ of order $n-1$, then H' is upper -Hessenberg and $\sigma(H') = \sigma(H) - \{\lambda\}$.

If λ is a multiple eigenvalue of H of (algebraic) multiplicity m then λ is an eigenvalue of H' of multiplicity $m-1$.

So far we have assumed that the eigenpair (x, λ) is exact. In practice, (x, λ) will only be approximate, in the sense that $Hx - \lambda x = r$, is of the order of the machine ϵ. In this case round-off errors will in general prevent G_{n-1} from annihilating the $(1, n)$ entry. In fact, the accuracy of the computed eigenvalue λ will come into play. If λ corresponds to an ill-conditioned eigenvalue of H, then it is possible that λ be a rather poor approximation of the exact eigenvalue. As a consequence the $(1, n)$ entry might not be negligible at all and examples do exist where this is indeed the case [23]. A way around this difficulty is to construct the plane rotations $G_1, \ldots, G_{n-1}$ in a way to reduce the vector x to e_n and then apply the corresponding similarity transformation to H. Inequality 5.7 from the previous chapter holds when this is

done (with obvious modification in the definition of H'). However, this will in general result in introducing non-zero entries below the subdiagonal of H and therefore H needs to be reduced to upper-Hessenberg form again. We refer the reader to [2] for an example of such an algorithm; the generalization of that algorithm to non-real case is straightforward.

In addition to the difficulty just mentioned, the deflation with plane rotations, like the deflation with stabilized elementary transformations, suffers from the fact that the deflated matrix will be complex if the eigenvalue to be deflated is complex. Indeed, it is unfortunately not true in general that when an eigenvalue and its complex conjugate are deflated by either of these two methods, the resulting matrix is real.

Example 5.2 *When the two complex eigenvalues of the* 4×4 *upper-Hessenberg matrix,*

$$\begin{pmatrix} 0.2190 & -0.0756 & 0.6787 & -0.6391 \\ -0.9615 & 0.9032 & -0.4571 & 0.8804 \\ 0 & -0.3822 & 0.4526 & -0.0641 \\ 0 & 0 & -0.1069 & -0.0252 \end{pmatrix}$$

are deflated using plane rotations as just described we obtain

$$\begin{pmatrix} 1.1593 - 0.0000i & 0.0000 + 1.0129i & 0.0000 + 0.0000i & 0.0000 + 0.0000i \\ 0.0000 - 0.3053i & 0.1740 - 0.0000i & 0.0000 + 0.0000i & 0.0000 + 0.0000i \\ -0.1534 - 0.3540i & 0.5717 + 0.0112i & 0.1082 - 0.4681i & 0.0000 + 0.0000i \\ -0.4640 + 0.1988i & 0.2187 - 0.3465i & 0.3964 + 0.0611i & 0.1082 + 0.4681i \end{pmatrix}.$$

After the deflation, we will be working with the upper 2×2 *block of* H' *which clearly is complex.*

5.4 Remarks on identifying duplicate eigenvalues

We have already remarked that the computed eigenpairs from our algorithm are the exact eigenpairs of a nearby matrix. Under those conditions [13]

$$|\lambda - \lambda_e| \leq \|E\|/|s(\lambda_e)|,$$

where E is the error in the matrix, and $|s(\lambda_e)|$ is the condition number of the exact eigenvalue λ_e of the original matrix H. When λ_e is ill-conditioned we can expect unpredictably large errors (compared to the tolerated size of the residual) in the computed approximations to λ_e and therefore in the duplicates if any and identifying these duplicates becomes a rather daunting task: in particular, they will not be detected if their difference is compared to *tol* introduced earlier. Conversely, it can happen that under certain conditions the tolerated size of the residual be much larger than the distance between certain exact eigenvalues and therefore between certain computed eigenvalues. In this case, it could happen that distinct computed eigenvalues will be declared as duplicates if their difference is compared to *tol*.

Example 5.3 *We illustrate this last case with the following matrix*

$$H = \begin{pmatrix} 2 \cdot 10^{-7} & 0 & 0 \\ 2 & 10^{-7} & 0 \\ 0 & 2 & 10^{10} \end{pmatrix}.$$

The tolerated size of the residuals for this matrix as chosen in our algorithm is $tol \approx \|H\|\epsilon > 10^{-6}$. *Therefore if we decide to declare as duplicates those eigenvalues whose difference is less than tol then the other two distinct eigenvalues of H will be declared as duplicates.*

The problems we have just raised do not have easy solutions [11], and indeed more research is needed here.

5.5 Further mathematical properties

We include here some results that shed more light on the deflation process. More precisely, in what follows we let H be an unreduced upper-Hessenberg matrix of order n, and we suppose that we know its eigenvalues $\lambda_1, \ldots, \lambda_n$ (not necessarily distinct) and the correponding eigenvectors. Deflating the eigenvalues successively results in the writing of the original matrix H in some canonical form. Our intention here is to exhibit these canonical forms.

We first consider what happens when the successive deflations are done using plane rotations. Recalling our notation in chapter 5.3.2, we let $\mathcal{G}^{(1)}$ be the product of plane rotations described in that chapter that gives

$$\mathcal{G}^{(1)^{-1}} H \mathcal{G}^{(1)} = \left(\begin{array}{c|c} H^{(1)} & 0 \\ \hline * \cdots * & \lambda_1 \end{array} \right).$$

Then $H^{(1)}$ is upper-Hessenberg and $\sigma(H^{(1)}) = \sigma(H) - \{\lambda_1\}$. Next we deflate λ_2 from H_1 and so on, thus producing a sequence of matrices $H^{(k)}, k = 1, \ldots, n-1$, such that

$$\sigma(H^{(k)}) = \sigma(H) - \{\lambda_1, \cdots, \lambda_k\}. \tag{5.8}$$

More precisely, let

$$U^{(1)} = \mathcal{G}^{(1)},$$

then if

$$U^{(k-1)^{-1}} \ldots U^{(1)^{-1}} H U^1 \ldots U^{(k-1)} = \left(\begin{array}{c|ccc} H^{(k-1)} & 0 & & \\ \hline & \lambda_{k-1} & & \\ * & & \ddots & \\ & * & & \lambda_1 \end{array} \right)$$

we construct $\mathcal{G}^{(k)}$ such that

$$\mathcal{G}^{(k)^{-1}} H^{(k-1)} \mathcal{G}^{(k)} = \left(\begin{array}{c|c} H^{(k)} & 0 \\ \hline * \cdots * & \lambda_k \end{array} \right)$$

where $H^{(k)}$ is upper-Hessenberg, and we set

$$U^{(k)} = \left(\begin{array}{c|c} \mathcal{G}^{(k)} & 0 \\ \hline 0 & I_{k-1} \end{array} \right),$$

where I_{k-1} is the identity matrix of order $k-1$.

Theorem 5.4 *There exists a permutation matrix P such that:*

$$P^T U^{(n-1)^{-1}} \dots U^{(1)^{-1}} H U^{(1)} \dots U^{(n-1)} P = S$$

and the matrix S is the Schur form of H.

Proof. This is quite trivial since $U^{(i)}$ is obviously unitary for all i and

$$U^{(n-1)^{-1}} \dots U^{(1)^{-1}} H U^{(1)} \dots U^{(n-1)} = S'$$

is lower triangular. P is a permutation matrix such that

$$P^T S' P = S$$

is upper-triangular. □

Our algorithm provides then an alternative way to construct a Schur form for the matrix if the deflation at each step is done using plane rotations. This however requires a serial computation.

We now perform the successive deflations using elementary transformations. We let $M^{(1)}$ be the elementary transformation described in chapter 5.1 that gives

$$M^{(1)^{-1}} H M^{(1)} = \left(\begin{array}{c|c} H^{(1)} & 0 \\ \hline 0 \quad * & \lambda_1 \end{array} \right).$$

Then $H^{(1)}$ is upper-Hessenberg and $\sigma(H^{(1)}) = \sigma(H) - \{\lambda_1\}$. Next we deflate λ_2 from H_1 and so on, thus producing, as before, a sequence of matrices $H^{(k)}, k = 1, \dots, n-1$, satisfying 5.8, and a sequence of eigenpairs $(x^{(1)}, \lambda_1), \dots, (x^{(n)}, \lambda_n)$ such that for each

k, $(x^{(k)}, \lambda_k)$ (we assume that the last component of each $x^{(k)}$ is one) is an eigenpair of $H^{(k-1)}$, the matrix of order $n-k+1$ obtained after $k-1$ deflations of H. More precisely, if

$$M^{(k)^{-1}} \ldots M^{(1)^{-1}} H M^{(1)} \ldots M^{(k-1)} = \left(\begin{array}{c|c} H^{(k-1)} & 0 \\ \hline 0 \quad * & B^{(k-1)} \end{array} \right),$$

where,

$$B^{(k-1)} = \begin{pmatrix} \lambda_{k-1} & & 0 \\ * & \ddots & \\ 0 & * & \lambda_1 \end{pmatrix},$$

we construct $M'^{(k)}$ such that

$$M'^{(k)^{-1}} H^{(k-1)} M'^{(k)} = \left(\begin{array}{c|c} H^{(k)} & 0 \\ \hline 0 \quad * & \lambda_k \end{array} \right),$$

where $H^{(k)}$ is upper-Hessenberg. In order to construct $M'^{(k)}$ the eigenvector x_k of $H^{(k-1)}$ corresponding to λ_k is needed. This eigenvector can be obtained by updating the eigenvector of H corresponding to λ_k using $M^{(1)}, \ldots, M^{(k-1)}$. Finally we set

$$M^{(k)} = \left(\begin{array}{c|c} M'^{(k)} & 0 \\ \hline 0 & I_{k-1} \end{array} \right).$$

Let

$$U = M^{(1)} \ldots M^{(n-1)},$$

then U is clearly unit upper-triangular and the following theorem is true:

Theorem 5.5

$$U^{-1} H U = B = \begin{pmatrix} \lambda_n & & 0 \\ * & \ddots & \\ 0 & * & \lambda_1 \end{pmatrix}$$

Furthermore if:

$$X = [v_n, \ldots, v_1]$$

is a matrix of eigenvectors of H (with obvious indexing) then:

$$U^{-1}X = L$$

is lower triangular.

Proof. The first claim is clear. We note in passing here that the subdiagonal of B is exactly the subdiagonal of H.

As for the second claim: Since the eigenvalues of H are non-derogatory, v_1 must be a multiple of x_1 and so $M^{(1)^{-1}}v_1$ is a multiple of e_n. Thus,

$$X^{(2)} \equiv M^{(1)^{-1}}X = [M^{(1)^{-1}}v_n, \ldots, M^{(1)^{-1}}v_2, \alpha e_n]$$

for some constant α. The leading principal submatrix of order $n-1$ of $X^{(2)}$ is clearly a matrix of eigenvectors of $H^{(2)}$. Since $H^{(2)}$ is also unreduced and therefore its eigenvalues are non derogatory, we can proceed by induction. Note that no assumptions were made as to the multiplicities of the eigenvalues of H. □

CHAPTER 6

Convergence

In chapter 2, we mentioned that computing an eigenpair of H reduces to computing a zero of:

$$F_s(x,\lambda) = \begin{pmatrix} Hx - \lambda x \\ e_s^T x - 1 \end{pmatrix}.$$

The Jacobian of F_s at (x,λ) is

$$F_s'(x,\lambda) = \begin{pmatrix} D_x(Hx-\lambda x) & D_\lambda(Hx-\lambda x) \\ D_x(e_s^T x - 1) & D_\lambda(e_s^T x - 1) \end{pmatrix} = \begin{pmatrix} H-\lambda I & -x \\ e_s^T & 0 \end{pmatrix},$$

where $D_x(F)$ denotes the derivative with respect to x of the function F. In this chapter, we give sufficient conditions for the convergence of our procedure. The result is a version of the Kantorovich theorem as it applies to our case.

Theorem 6.1 (Wilkinson) *Assume (x,λ) is an exact zero of F_s. Then*

$$\begin{pmatrix} H-\lambda I & -x \\ e_s^T & 0 \end{pmatrix}$$

is singular iff λ is multiple.

Proof. Let us assume first that λ is a multiple eigenvalue of H and that x is the corresponding (exact) eigenvector, then there exists a nonzero vector y such that:

$$y^H(H-\lambda I) = 0 \text{ and } y^H x = 0;$$

y is simply a left eigenvector of H corresponding to λ. Thus, $(y^H\ 0)$ is a nonzero vector and

$$(y^H\ 0)\begin{pmatrix} H-\lambda I & -x \\ e_s^T & 0 \end{pmatrix} = 0.$$

This proves that the Jacobian is singular.

Now conversely, if $\begin{pmatrix} H-\lambda I & -x \\ e_s^T & 0 \end{pmatrix}$ is singular then there exists a nonzero vector $\begin{pmatrix} v \\ \mu \end{pmatrix}$ such that

$$\begin{pmatrix} H-\lambda I & -x \\ e_s^T & 0 \end{pmatrix} \begin{pmatrix} v \\ \mu \end{pmatrix} = 0.$$

But then $v_s = 0$ and

$$(H-\lambda I)v = \mu x.$$

If $\mu = 0$, then v is nonzero since

$$\begin{pmatrix} v \\ \mu \end{pmatrix} \neq 0,$$

and so v is an eigenvector that is linearly independent of x, since $x_s = 1$ and $v_s = 0$. Hence λ is a multiple eigenvalue in this case. If $\mu \neq 0$, then v is also nonzero; if it were then we would have

$$\mu x = (H-\lambda I)v = (H-\lambda I)0 = 0$$

and hence $x = 0$, contradicting our assumption on x. But

$$(H-\lambda I)^2 v = \mu(H-\lambda I)x = 0,$$

and thus v is a nonzero vector that has grade 2. Therefore λ is multiple in this case as well. □

Remark: Since we are assuming that H is upper-Hessenberg and unreduced, an eigenvalue can only be non-derogatory, i.e., the associated eigenspace has dimension 1.

The previous result applies to the Jacobian at a zero of F_s. We wish to know more about the Jacobian at those approximations arising during Newton's iteration and before convergence is declared.

Theorem 6.2 *Assume* (x, λ) *is not an exact zero of* F_s. *Then* $\begin{pmatrix} H - \lambda I & -x \\ e_s^T & 0 \end{pmatrix}$ *is singular if and only if at least one of the following is true:*

1) λ *is an eigenvalue of* H *and the* s^{th} *component of the associated eigenvector is zero.*

2) x *belongs to the space generated by* $(c_1, \ldots, c_{s-1}, c_{s+1}, \ldots, c_n)$, *where* c_i *is the* i^{th} *column of* $H - \lambda I$ *(*λ *may or may not be an eigenvalue).*

Proof. Assume first that 1) is true and let y be an eigenvector, $y_s = 0$. Then $\begin{pmatrix} y \\ 0 \end{pmatrix}$ is clearly in the null space of the Jacobian. If 2) holds, and

$$x = (H - \lambda I)y, \quad y_s = 0$$

then $\begin{pmatrix} y \\ 1 \end{pmatrix}$ is in the null space of the Jacobian.

Conversely, if the Jacobian is singular then there exists a vector $\begin{pmatrix} y \\ \mu \end{pmatrix}$ such that,

$$\begin{pmatrix} H - \lambda I & -x \\ e_s^T & 0 \end{pmatrix} \begin{pmatrix} y \\ \mu \end{pmatrix} = 0.$$

This implies that

$$\begin{cases} (H - \lambda I)y = \mu x \\ e_s^T y = 0 \end{cases}$$

and clearly $y_s = 0$. We now consider two cases according to whether μ is zero or not. If μ is zero, then λ is an eigenvalue with corresponding eigenvector y, therefore we are in case 1). If μ is not zero then μx and hence x is in the range of $H - \lambda I$ and therefore we are in case 2) since y_s=0. Note that λ may or may not be an eigenvalue in this last case. □

Remarks: The theorem tells us that more often than not, a singular Jacobian is an indication we already have an eigenvalue and this has been our experience indeed. The singularity of the Jacobian is also an indication of an ill-conditioned

eigensystem. In fact, if we accept that the current eigenvector was moving in the "right" direction then if it satisfies condition 2) of the theorem, we can say that the eigenvalue is acting like a multiple one since the eigenvector is also in the range of $(H - \lambda I)$. In practice we have not encountered a situation where case 1) applied: this is understandable again if we accept that the eigenvector is moving in the right direction since then the eigenvalue would be acting like an eigenvalue of geometric multiplicity more than 1, which is impossible since the matrix is unreduced.

The second derivative of F_s is a constant bilinear operator with norm equal to 2. In fact,

$$F_s''(x,\lambda) = \left(\begin{array}{cc|cc} 0 & -I & -I & 0 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

Suppose now that our procedure is started with initial guess (x_0, λ_0). We now give sufficient conditions for the convergence of our procedure with the given initial guess. This is provided by the classical Kantorovich theorem [6]. Let us first introduce

$$K = \|F_s'^{-1}(x_0, \lambda_0)\|$$

and

$$c_1 = \|(x_1, \lambda_1) - (x_0, \lambda_0)\|,$$

where (x_1, λ_1) is the first iterate, i.e.,

$$\begin{pmatrix} x_1 \\ \lambda_1 \end{pmatrix} = \begin{pmatrix} x_0 \\ \lambda_0 \end{pmatrix} - F_s'^{-1}(x_0, \lambda_0) F_s(x_0, \lambda_0).$$

We call $(x_0, \lambda_0), \ldots, (x_k, \lambda_k)$ the sequence of iterates produced by the algorithm.

Theorem 6.3 *If $\beta_0 = K c_1 < 1/4$, then the sequence (x_k, λ_k) converges quadratically starting from (x_0, λ_0).*

The process can be regarded as starting from any of the iterates (x_i, λ_i) and in fact will often converge even when the conditions of the theorem are not satisfied

at (x_0, λ_0). These conditions will then be met for some (x_k, λ_k) at which stage convergence becomes quadratic.

CHAPTER 7

Defective Case

Ours being a non-stationary iteration (the iterating map is not fixed), it is not easy to analyze the behaviour of the successive approximations. We try however to address this problem in this chapter with a particular emphasis on the case when either the matrix H or the modified matrix H_0 is defective, i.e., when either one of these matrices does not have a complete set of eigenvectors. As we remarked earlier, an eigenvalue of H (with no zeros on the subdiagonal) can only have geometric multiplicity one, and therefore H is defective whenever it has a multiple eigenvalue. An eigenvalue of H_0 on the other hand, can have geometric multiplicity one or two. In what follows n is the order of H.

The connection between Newton's method and inverse iteration is well known [20]. We present a proof which motivates the subsequent analysis: we let J be the Jacobian of the map F_s at (x, λ) defined in chapter 4,

$$J = \begin{pmatrix} H - \lambda I & -x \\ e_s^T & 0 \end{pmatrix},$$

and we assume that $x_s = 1$. The order of the Jacobian is $n + 1$. Then,

$$J = J_0 + e_{n+1} v^T$$

with

$$J_0 = \begin{pmatrix} H - \lambda I & -x \\ 0 & 1 \end{pmatrix}$$

and,

$$v = e_s - e_{n+1} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ -1 \end{pmatrix}.$$

Assume J_0 it is not singular: this is true if and only if λ is not an eigenvalue. Assume also that J is not singular. The correction to (x, λ) is computed in the following manner:

$$\begin{pmatrix} y \\ \mu \end{pmatrix} = J^{-1} \begin{pmatrix} r \\ 0 \end{pmatrix},$$

where $r = \lambda x - Hx$. Using the Sherman-Morrison-Woodbury formula [13] we can write J^{-1} as,

$$J^{-1} = (J_0 + e_{n+1}v^T)^{-1} = J_0^{-1} - \frac{1}{1 + v^T J_0^{-1} e_{n+1}} J_0^{-1} e_{n+1} v^T J_0^{-1}.$$

Therefore, letting $b = \begin{pmatrix} r \\ 0 \end{pmatrix}$, we have:

$$\begin{pmatrix} y \\ \mu \end{pmatrix} = J^{-1}b = J_0^{-1}b - \frac{v^T J_0^{-1} b}{1 + v^T J_0^{-1} e_{n+1}} J_0^{-1} e_{n+1}. \tag{7.1}$$

It can be easily checked that

$$J_0^{-1}b = \begin{pmatrix} -x \\ 0 \end{pmatrix}$$

and that

$$J_0^{-1} e_{n+1} = \begin{pmatrix} \tilde{x} \\ 1 \end{pmatrix}$$

where $(H - \lambda I)\tilde{x} = x$. Now if we let (x_1, λ_1) be the next eigenpair we have from 7.1 and the subsequent equalities:

$$\begin{pmatrix} x_1 \\ \lambda_1 \end{pmatrix} = \begin{pmatrix} x \\ \lambda \end{pmatrix} + \begin{pmatrix} y \\ \mu \end{pmatrix} = \begin{pmatrix} x \\ \lambda \end{pmatrix} + \begin{pmatrix} -x \\ 0 \end{pmatrix} - \frac{v^T J_0^{-1} b}{1 + v^T J_0^{-1} e_{n+1}} \begin{pmatrix} \tilde{x} \\ 1 \end{pmatrix}.$$

Furthermore,

$$\frac{v^T J_0^{-1} b}{1 + v^T J_0^{-1} e_{n+1}} = \frac{-1}{\tilde{x}_s}$$

and so finally we see that our scheme reduces to the following: Given (x, λ) compute the next iterate (x_1, λ_1) via

$$\begin{aligned} (H - \lambda I)\tilde{x} &= x \\ x_1 &= \tfrac{1}{\tilde{x}_s}\tilde{x} \\ \lambda_1 &= \lambda + \tfrac{1}{\tilde{x}_s} \end{aligned} \tag{7.2}$$

7.1 Case when H is defective

When the algorithm is expressed as in 7.2 we can readily see some of the difficulties that arise when the matrix H is defective or almost defective, which is more likely in general due to roundoff errors. These problems are similar to the kind of problems that face the application of inverse iteration. More precisely, assume that H is almost defective with $x_1, \ldots, x_n$ as a complete set of eigenvectors. Let $\lambda_1, \ldots, \lambda_\ell$ be a cluster of eigenvalues of H and suppose that the initial approximate eigenvalue λ corresponds to one of these. The eigenvectors $x_1, \ldots, x_\ell$ corresponding to these eigenvalues are then almost linearly dependent. In general, the eigenvector corresponding to λ can be expected to converge to the space generated by $x_1, \ldots, x_\ell$. As the eigenvalue λ approaches the cluster however, continued corrections to the eigenvector cannot be expected to refine it. We refer the reader to the particularly lucid account in [20] for a justification of these claims. Solving as in inverse iteration (see INVIT [21]) is a possible way around this problem. Using extended precision arithmetic is also an obvious approach, and has been successful in practice.

When approaching a singular solution, Newton's method loses its quadratic convergence rate. We have proved earlier (Thm. 6.1) that the Jacobian is singular at multiple eigenpairs and therefore we can expect slower convergence when these are the target.

Recall the rate of change of λ that we derived in chapter 4:

$$|\lambda'(0)| = \frac{|\alpha||y_{k+1}||x_k|}{|y^H x|}.$$

We wish to caution from hastily drawing conclusions about the sensitivity of the eigenvalues of a defective matrix to our dividing process from this expression. Indeed, as an extreme case which will help illustrate our point, the eigenvalues of a defective matrix can remain virtually unchanged after a zero has been introduced in the subdiagonal. An example, is the 2×2 matrix

$$\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}.$$

After the dividing process we have

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

The starting eigenpairs are then $(\begin{pmatrix} 1 \\ 0 \end{pmatrix}, 1)$ and $(\begin{pmatrix} 0 \\ 1 \end{pmatrix}, 1)$. The second of these is of course the exact eigenpair. Now, even though the first starting eigenvector is orthogonal to the desired one, the equations arising during the first of Newton's iterations can be solved in a way to produce the desired eigenvector from the first step: zero pivots need to be replaced by small numbers on the order of the machine unit roundoff (as is done in inverse iteration, INVIT [21]).

Finally, we mention that the deflation process can contribute to the improvement of the condition of the eigenvalues. Indeed, if λ and λ' are pathologically close (and fairly distant from the rest of the eigenvalues), then by deflating λ, say, λ' will have a better condition number as an eigenvalue of the resulting matrix. Indeed, λ' is not part of a cluster anymore.

7.2 Case when H_0 is defective

It can happen in this case (if H is non-defective for example) that the initial dividing process would leave us with a number of initial approximations that is smaller than n. This will inhibit the parallelism of the algorithm in that fewer Newton processes can be started simultaneously. Furthermore, whatever eigenpairs we have can be extremely poor approximations to the desired ones. An extreme situation is illustrated by the following matrix:

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

No matter where the zero is introduced on the subdiagonal, the resulting matrix has 0 as its only eigenvalue, and we only have two initial approximations to the four distinct eigenpairs of H, namely:

$$(\begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, 0), \quad (\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}, 0),$$

if the zero is introduced in the $(3,2)$ position.

Furthermore, the Jacobian is exactly singular at each of these initial approximations. Indeed the Jacobian at $(\begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, 0)$ is

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

which is clearly of rank three since the first and the last columns are linearly dependent, and the fourth column is zero. Similarly it can be seen that the Jacobian at

$(\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}, 0)$ is also singular. A remedy to this situation is to perturb the initial guess zero by a small amount thereby making the Jacobian non singular, and indeed this has been successful in practice. The problem in this case, as in most other similar cases, results from the particular structure of the matrix. In order to obtain further eigenvalues, we need to deflate the matrix each time a new eigenpair is computed which makes the algorithm almost serial.

7.3 Known failures

Some matrices of the structure mentioned at the end of the previous section (companion-like matrices), provided us with the only cases where the algorithm failed in practice to converge to the desired eigenpairs, i.e., failed to produce eigenpairs with small residuals after a fixed number of iterations. Should these matrices be subjected to orthogonal similarity transformations however and then reduced back to upper-Hessenberg form, the dividing process will provide us with much better initial approximations and indeed, will converge for all initial approximations. We are certainly not advocating this as a viable scheme: we wanted to emphasize the fact that it is the structure of the matrix that caused the poor approximations and the failures, and not some inherent difficulty with the spectrum of these matrices.

CHAPTER 8

Work Estimates

We assume in this chapter that we are given a real dense matrix A. Then the first task in our algorithm is to reduce A to an upper-Hessenberg matrix H. This requires $\frac{14n^3}{3}$ operations (plus lower order terms) since the orthogonal matrices used in the reduction are to be accumulated [13].

The dividing process is now applied to H. Assume $n = 2^m r$, for some $m \geq 1$ where r is not necessarily relatively prime to 2. If we repeat the dividing process m times, we end up with 2^m matrices of size r, each of which is upper-Hessenberg. A submatrix of size $\frac{n}{2^i}$ obtained in the dividing process will be referred to as a matrix at the level i. The matrix H itself is at the level 0, and the r matrices referred to above are at the level m. Thus, we have $m+1$ levels in total. Let $p' = 2^m$ be the number of submatrices at the lowest level, and p the number of processors; we will assume that $p = 2^q, q \leq m$. The cost of finding the eigenpairs of a (Hessenberg) matrix at the lowest level (by the QR algorithm) is roughly $18\left(\frac{n}{p'}\right)^3$: this figure is very approximate and assumes among other things that two QR steps are needed before a real or two complex conjugate eigenvalues are identified and that the matrix has an equal number of real and complex eigenvalues [13]. Let $s_\ell = \frac{n}{2^\ell}$ be the size of one matrix at the level ℓ. Let k_ℓ be the average number of iterations needed to get one eigenpair of a matrix at the level ℓ. k_ℓ depends on the matrix and on its size of course. We will make the following simplifying assumption however: $k_\ell = k$, $\ell = 0, \cdots, m$, i.e., we assume that the average number of steps required for convergence is the same at all levels. Our experience with the algorithm suggests that this is a realistic

assumption as long as the size of the submatrices remains moderate. If the number of levels is increased to the point where we are left with small submatrices, then k_ℓ becomes significantly larger as ℓ increases.

The cost of computing one correction at the level ℓ is roughly $6s_\ell^2$: indeed, one Newton iteration involves the solution of a linear system that is upper-Hessenberg, but for possible non-zeros in the last row; the order of this linear system is $s_\ell + 1$. Therefore 2 multipliers at most need be computed per column and, when updating the matrix, each of these will be used in $2(s_\ell + 1 - i)$ multiplications and additions, where i is the index of the column. The factorization of the Jacobian requires then $2s_\ell^2$ operations in addition to $O(s_\ell)$ operations (including divisions and comparisons). The forward solve is $O(s_\ell)$ work and is negligible. The backsolve requires s_ℓ^2 operations and computing the residual requires another s_ℓ^2 operations. Therefore for a real current approximation one correction comes at the cost of $4s_\ell^2$. For a complex current eigenpair, this becomes $16s_\ell^2$ since the dominant operations are a roughly equal number of multiplications and additions. Since, as we indicated in chapter 2, only one of a conjugate pair of complex eigenpairs need be corrected, we can assume that $8n^2$ operations are required for correcting a complex eigenpair. Assuming again that the matrix has an equal number of real and complex eigenvalues, our estimate for the amount of work required for computing one correction at the level ℓ becomes $6s_\ell^2$ operations.

We shall use $\frac{n}{p}$ initial guesses to start $\frac{n}{p}$ Newton processes on each processor. We now have the following work estimate on one processor, assuming all processors share equally the cost of all the stages of the algorithm:

$$W_p = \frac{14n^3}{3p} + 18\left(\frac{p'}{p}\right)\left(\frac{n}{p'}\right)^3 + \sum_{\ell=0}^{m-1} 6(\frac{n}{p})ks_\ell^2 + 2\frac{n^3}{p},$$

where: $\frac{14n^3}{3p}$ is the processor's contribution to the reduction of the original matrix to upper-Hessenberg form; $18\left(\frac{p'}{p}\right)\left(\frac{n}{p'}\right)^3$ is the cost of applying the QR algrithm to $\frac{p'}{p}$ matrices of size $\frac{n}{p'}$; $\sum_{\ell=0}^{m-1} 6(\frac{n}{p})ks_l^2$ is the cost of solving k linear systems with matrices of size s_ℓ, repeating this for $\frac{n}{p}$ initial guesses and for $\ell = 0, \cdots, m-1$; $2\frac{n^3}{p}$ is the processor's contribution to the computation of the eigenvectors of the original dense matrix A once those of H have been computed. The expression for the work can be rewritten:

$$W_p = \frac{n^3}{p}\left(\frac{20}{3} + \frac{18}{p'^2} + \frac{24k}{3}(1 - (\frac{1}{4})^m)\right).$$

But $p' = 2^m$ and therefore $p'^2 = 4^m$ and hence:

$$W_p = \frac{n^3}{p}\left(\frac{20}{3} + 8k + (18 - 8k)\frac{1}{4^m}\right). \tag{8.1}$$

The cost of getting the eigenvalues and eigenvectors by the QR algorithm is $25n^3$ [13]. A reasonable value for k is 3, however there are cases when k is 2 or less. There are also cases where k is larger than 3, mostly with matrices of small order or defective matrices.

It is easy to verify that for $k \leq 2$ and for $m = 1$ (one split) the model for the cost of the algorithm predicts sequential speedup over QR. Our model does not predict sequential speedup for problems where k equals or exceeds 3. Here are some sample values of W_p, assuming that $k = 3$ and $p = p' = 2^m$. which means that the original problem is subdivided into a number of problems equal to the number of available processors.

$$p = 128 \rightarrow W_p = 0.2396n^3 \ ; \ p = 1024 \rightarrow W_p = 0.0299n^3.$$

Here we have assumed that the problem is large enough to allow the efficient use of that many processors. Note that the ratio of the work estimate from our model to the

work estimate of QR is independent of n. This is due to the simplifying assumption on k made at the beginning of this chapter. That assumption has in effect "hidden" the dependency on n of the coefficient n^3 in our model. Figures 2, 3, 4 and 5 show plots of the work estimate for various values of the parameters involved.

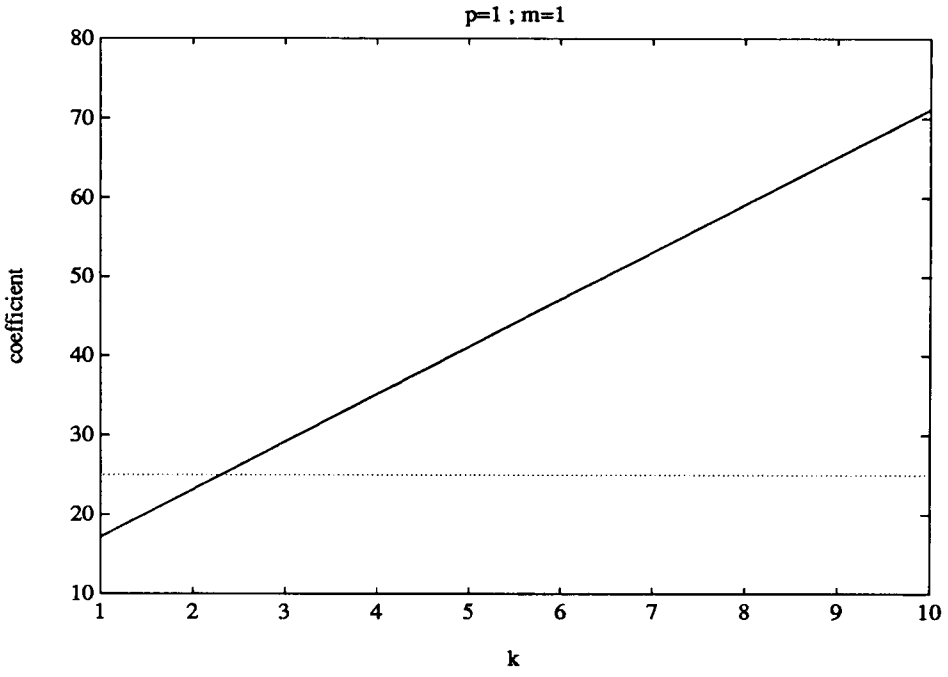

Figure 2: Variation of the Coefficient of n^3 in work estimate model in terms of number of steps, with one split and one processor.

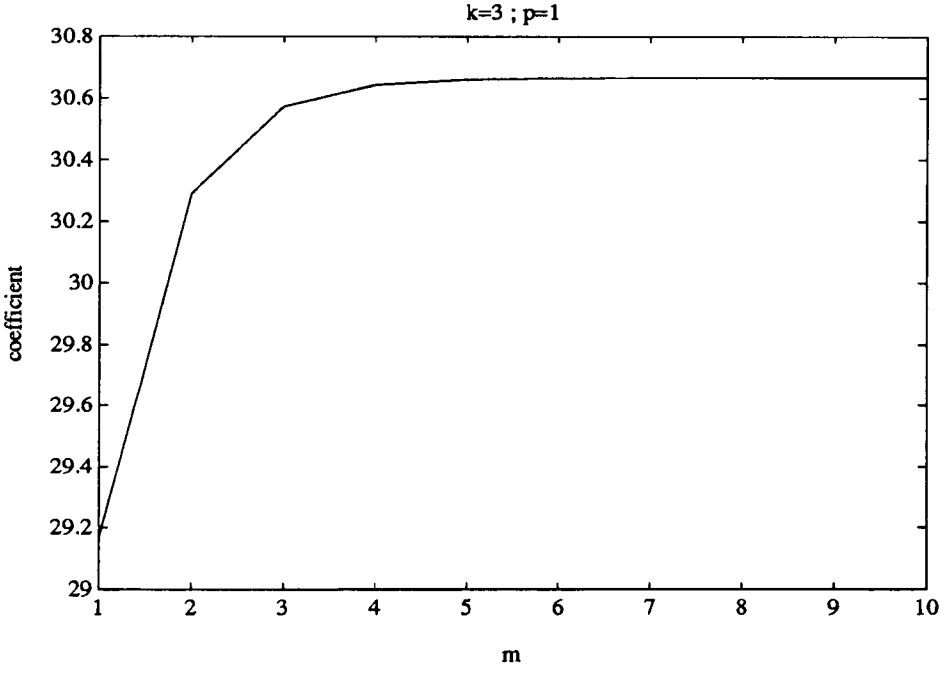

Figure 3: Variation of the Coefficient of n^3 in work estimate model in terms of the number of splits, with one processor.

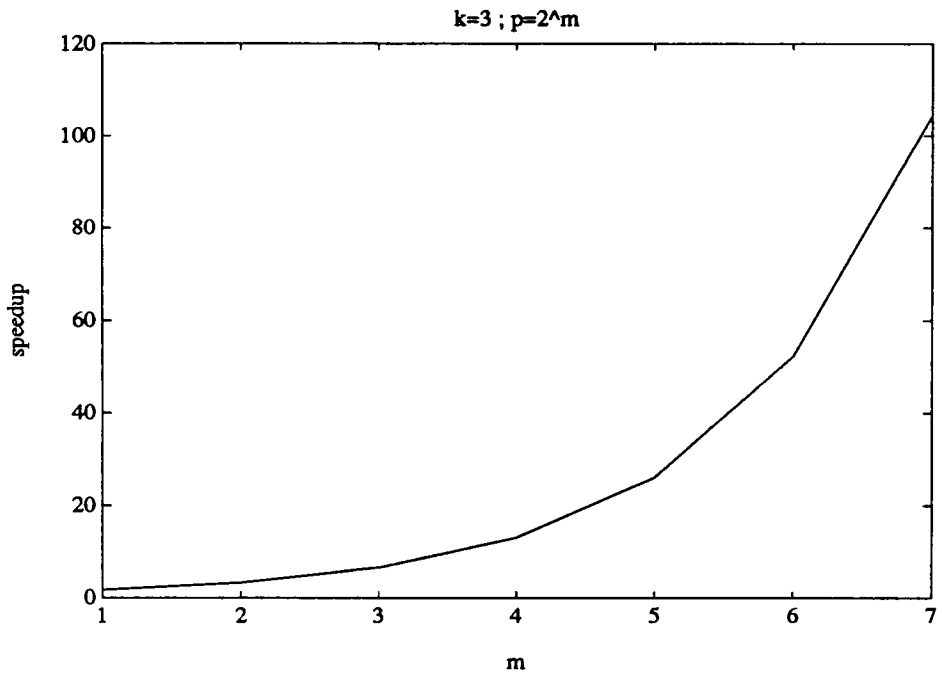

Figure 4: Predicted speedup in terms of number of splits.

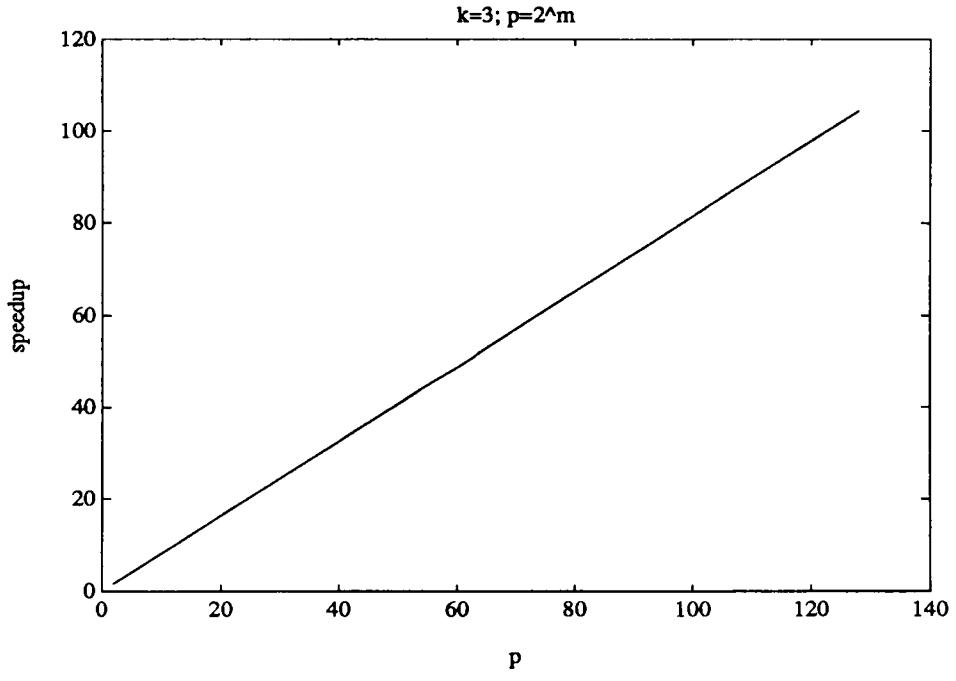

Figure 5: Predicted speedup in terms of number of processors.

CHAPTER 9

Parallel Algorithms Details and Performance

It is fairly straightforward to see from chapter 4 how to obtain a parallel algorithm. We discuss here certain details. The given, generally dense, matrix is first reduced to upper-Hessenberg form using a blocked algorithm. Next comes the partitioning phase or "divide". This phase amounts to constructing a binary tree with each node representing a partition into two subproblems. It has been our practice to partition the matrix into a number of subproblems (at the lowest level) equal to the number of processors available on the target machine. Each of these problems may be spawned independently without fear of data conflicts; the computation at this level (the lowest) consists of calls to the EISPACK routine HQR2. The tree is then traversed in reverse order with Newton's method applied at each node using the results from the sons as initial approximations. Note here that the computation at a node does not have to wait for both sons to complete in order to start: as a matter of fact, it can start as soon as one son has computed one eigenpair of a subproblem. In order to stress this point, we mention here that this is quite different from the situation in the symmetric case [7], where information from both sons is needed before computations can start at the node. However, in practice we have allowed computations to start at a node only after at least one son has completed; the need to check for duplicate eigenvalues and deflate if necessary has imposed further synchronization.

The algorithm has been implemented on computers with a shared memory and computers with distributed memory architectures.

9.1 Shared memory implementation

So far we have used SCHEDULE [8] to implement the algorithm on shared memory computers. SCHEDULE is a package of FORTRAN and C subroutines designed to aid in programming explicitly parallel algorithms for numerical calculations. An important part of this package is the provision of a mechanism for dynamically spawning processes even when such a capability is not present within the parallel language extensions provided for a given machine.

9.2 Distributed memory implementation

The current implementation on distributed memory machines requires that the matrix be stored on each processor. This obviously puts constraints on the size of problems that can be solved. With this implementation however, communication is needed only at the deflation phase. This implementation is best described through the contribution of a particular processor. Suppose that we have 4 processors at our disposal, $p_0, \ldots, p_3$ and that accordingly the matrix H has been divided into 4 subproblems: $H_0, \ldots, H_3$, that their common size is $n/4$ and that they occurr in this order on the diagonal of the matrix. We describe now the contribution of p_2:

1) Call HQR2 to solve for the eigensystem of the matrix H_2.

2) Refine the output from 1) to get 1/2 the number of (i.e., $n/4$) eigenpairs of the matrix $H_{\frac{1}{2}}$:

$$H_{\frac{1}{2}} = \left(\begin{array}{c|c} H_2 & B \\ \hline 0 \quad \alpha & H_3 \end{array} \right),$$

where $H_{\frac{1}{2}}$ is a submatrix of H.

3) Refine the output from 2) to get 1/4 of the number of eigenpairs (i.e., $n/4$)

of the matrix H.

As can be readily realized no communication between processors is needed except for checking for eigenpairs to which convergence occurred from more than 1 initial approximation: for example, the eigenpairs of $H_{\frac{1}{2}}$ are generated on p_2 and p_3, and therefore we need to check for duplicate eigenpairs (on each processor separately which requires no communications, and across both which requires communications).

We are currently developing another implementation where blocks of columns of the matrix are stored on different processors. This storage scheme has been dictated to us by the need to call HQR2 at the lowest level: indeed, the call to HQR2 requires that contiguous columns of the matrix reside on the same processor. Therefore storage schemes more advantageous for linear system solving, such as wrap mapping of columns or rows, could not be used. The communication between processors for this implementation is more intensive. Communication is indeed needed when solving the linear systems arising in Newton's iterations as well as for the deflation phase. Also because of the storage scheme, we can expect the processors to become successively idle during the factorization of the Jacobian and the back solve for the correction. However, we have implemented efficiently a scheme where the Jacobian is repartitioned by rows before the back solve takes place: the "reshuffling" of the submatrices takes place between processors that became idle after doing their part of the Gaussian elimination.

9.3 Numerical results and performance

In this chapter we present the results of the implementation of the algorithm on a number of machines. The serial version of the code is available through NETLIB

where it's called "nonsymdc".

The same algorithm has been run on the IBM RS/6000/500, the Alliant FX/8, the Intel iPSC/2 and the Intel iPSC/860. We compared our results to those of HQR2 from the EISPACK collection. We have used randomly generated matrices in these tests. In these implementations the linear solver used for the computation of the correction at each Newton step is column oriented. The storage requirement for our algorithm in a serial implementation is $4n^2+O(n)$. The following are the results from the IBM RS/6000/500 implementation. The IBM RS/6000/500 is a single processor computer with a RISC-based Architecture.

Order	HQR2	D&C	Ratio HQR2/D&C	Distinct λ
100	1.04	1.12	.93	99
200	9.31	9.18	1.01	196
300	34.1	28.1	1.2	293
400	94.0	65.3	1.4	395
500	196	136	1.4	490
1000	1741	992	1.7	1000

In an implementation on shared memory machine, the storage allocated to the Jacobian (in serial mode) is multiplied by the number of processors used; this is meant to prevent concurrent write to the same memory locations. The following are some results from the Alliant FX/8 implementation. The Alliant FX/8 is a parallel machine with 8 vector processors.

Order	no.of procs.	Levels	Ratio HQR2/D&C
100	2	1	1.7
	4	2	2.4
	8	3	4.0

The results on the Alliant were in general disappointing. The storage scheme for the Jacobian that we used on that machine seems to have inhibited the compiler performed vector optimizations. HQR2, running on a single processor *was* vector-optimized.

We finally present results from the runs on the hypercubes. The following are some results from the Intel iPSC/2 implementation. The largest size used in each case was dictated by the memory capacity of a single node.

Order	no.of procs.	Levels	Ratio HQR2/D&C
100	2	1	2.2
	4	2	3.7
	8	3	6.0
	16	4	8.2
200	2	1	2.2
	4	2	3.5
	8	3	5.2
	16	4	9.1
300	2	1	2.2
	4	2	3.2
	8	3	6.3
	16	4	9.8
	32	5	13.2
	64	6	21.3

The following are some results from the Intel iPSC/860 implementation.

Order	no.of procs.	Levels	Ratio HQR2/D&C
100	2	1	1.9
	4	2	3.3
	8	3	5.1
400	2	1	2.4
	4	2	3.2
	8	3	6.0
	16	4	8.4
600	8	3	7.5
	16	4	13.5
	32	5	23
	64	6	32

We observe here that the speedups realized by our algorithm over the QR algorithm, did not remain linear for a large number of processors. This is due to the fact that our algorithm is much less efficient on small matrices and we had to work with this kind of matrices when the number of processors became large. For example, with a matrix of order 600 and using 64 processors matrices of average size 20 had to be solved on each node at level 5.

CHAPTER 10

The Generalized Eigenvalue Problem

We show here how the ideas behind our algorithm can be used to solve the generalized eigenvalue problem.

10.1 Basic algorithm

Given an upper-Hessenberg matrix H and an upper-triangular matrix U:

$$H = \left(\begin{array}{c|c} H_{11} & H_{12} \\ \hline 0 \ \overset{\alpha}{|} & H_{22} \end{array}\right), \quad U = \left(\begin{array}{c|c} U_{11} & U_{12} \\ \hline 0 & U_{22} \end{array}\right),$$

we want to solve the generalized eigenvalue problem:

$$Hx = \lambda Ux. \tag{10.1}$$

Without loss of generality, we can assume H to be unreduced and U to be non-singular since otherwise the problem reduces to a smaller problem. Then our algorithm generalizes easily. Indeed, set

$$H_0 = H - \alpha e_{k+1} e_k^T,$$

and consider solutions of (or approximations of these)

$$H_0 x = \lambda Ux, \tag{10.2}$$

as initial approximations to the sought eigenpairs. More precisely, solving 10.2 reduces to solving

$$H_{11}x = \lambda U_{11}x \ \ and \ \ H_{22}x = \lambda U_{22}x.$$

Let's denote these solutions by $(\tilde{x}_1, \lambda_1), \ldots, (\tilde{x}_n, \lambda_n)$ (we have n solutions because of our assumption that U is non-singular). These can be used to construct initial approximations $(x_1, \lambda_1), \ldots, (x_n, \lambda_n)$ to the eigenpairs of the original generalized eigenproblem in much the same way we did it in the case $U = I$. More precisely, we take

$$(x_i, \lambda_i) = (\begin{pmatrix} \tilde{x}_i \\ 0 \end{pmatrix}, \lambda_i),$$

if $\lambda_i \in \sigma(H_{11}, U_{11})$ and

$$(x_i, \lambda_i) = (\begin{pmatrix} 0 \\ \tilde{x}_i \end{pmatrix}, \lambda_i),$$

if $\lambda_i \in \sigma(H_{22}, U_{22})$ (an appropriate number of zeros in each case).

Finally solving the generalized eigenvalue problem 10.1 above is the same as solving the problem

$$\begin{cases} Hx - \lambda Ux = 0 \\ L(x) = 1 \end{cases}$$

where $L(x)$ is a scalar equation, which we take to be a normalizing condition: $e_s^T x = 1$. Then for each initial eigenpair (x_i, λ_i), successive corrections can be computed via

$$\begin{pmatrix} H - \lambda_i U & -Ux_i \\ e_s^T & 0 \end{pmatrix} \begin{pmatrix} y \\ \mu \end{pmatrix} = \begin{pmatrix} r_i \\ 0 \end{pmatrix}, \quad \lambda_i \leftarrow \lambda_i + \mu \ ; \ x_i \leftarrow x_i + y$$

where

$$r_i = \lambda_i U x_i - H x_i.$$

A possible stopping criterion for this scheme is the condition

$$\frac{\|Hx_i - \lambda_i U x_i\|}{\|x_i\|} \leq f(n)\|H\|\|U\|\epsilon,$$

where $f(n)$ is a modest function of n. It is easy to see that starting from two complex conjugate initial guesses the algorithm will compute complex conjugate corrections, therefore allowing similar savings to those in the case $U = I$.

10.2 Deflation in the generalized case

The method of deflation presented in section 5.1 generalizes to this case in the following manner. Let $Hx = \lambda Ux$ (of course, in practice we only have an approximate eigenpair). Then it is true that if $w = Ux$ then $w_n \neq 0$: Indeed, if $w_n = 0$, then it can be shown that w is zero using the fact that H is unreduced. U being non singular this implies that x is zero which is not true.

Let

$$\begin{pmatrix} -v \\ 1 \end{pmatrix} = w/w_n,$$

and

$$u = \begin{pmatrix} v \\ 1 \end{pmatrix}.$$

We know that $\sigma(H, U)$ is the same as $\sigma(M'HM, M'UM)$ for any non-singular M and M'. For our purposes, we take

$$M = [e_1, \ldots, e_{n-1}, x]$$

and

$$M' = [e_1, \ldots, e_{n-1}, u].$$

Then it is easy to verify that we have

$$M'HM = \left(\begin{array}{c|c} H' & 0 \\ \hline 0 \quad \alpha & \gamma \end{array}\right), \quad M'UM = \left(\begin{array}{c|c} U' & 0 \\ \hline 0 & \delta \end{array}\right), \tag{10.3}$$

where H' is upper-Hessenberg, U' is upper-triangular and

$$\frac{\gamma}{\delta} = \lambda. \tag{10.4}$$

In fact in this particular case we have

$$\gamma = \lambda$$

and

$$\delta = 1.$$

Clearly:

$$\sigma(H', U') = \sigma(H, U) - \{\lambda\}.$$

Therefore, having "removed" λ from the spectrum we can get further eigenpairs. We note that, just as in the case $U = I$, H' and U' are very cheap to obtain once M' has been determined. Indeed, H' differs from the $n-1 \times n-1$ principal submatrix of H in the last column only, whereas U' is the $n-1 \times n-1$ principal submatrix of U. The computation of M' requires a matrix-vector multiply.

It is also easy to verify that deflating two consecutive complex conjugate eigenpairs results in real H' and U'.

The condition number of M' might raise concern. We have

$$cond_\infty(M') \approx max(|w_i|)^2.$$

It is therefore easy to detect an ill-conditioned M'. We propose to handle this situation in the generalized case in the following manner. Let D be a diagonal matrix with its diagonal elements d_i defined by

$$\begin{cases} d_i = 1, & if \ \frac{w_i}{w_n} \leq 1 \\ d_i = -\frac{w_n}{w_i}, & if \ \frac{w_i}{w_n} > 1 \end{cases}$$

then clearly

$$\sigma(DM'HM, DM'UM) = \sigma(H, U),$$

since D is non-singular. The multiplication by D cancels all the large entries in the last column of M'. It is easy to see that 10.3 and 10.4 are satisfied when the pre-multiplication is done with DM' instead of M'. The disadvantage of having to premultiply by D is that after the deflation of two complex conjugate eigenpairs, the resulting H' and U' might still be complex.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] J. R. Bunch, C. P. Nielsen, and D. C. Sorensen. Rank-one modification of the symmetric eigenproblem. *Numer. Math.*, 31:31–48, 1978.

[2] P. A. Businger. Numerically stable deflation of Hessenberg and symmetric tridiagonal matrices. *BIT*, 11:262–270, 1971.

[3] M.T. Chu. A simple application of the homotopy method to symmetric eigenvalue problems. *Lin. Alg. & Appl.*, 59:85–90, 1984.

[4] J. Cuppen. A divide and conquer method for the symmetric tridiagonal eigenproblem. *Numer. Math.*, 36:177–195, 1981.

[5] J. Dongarra. Improving the accuracy of computed matrix eigenvalues. Technical Report ANL-80-84, Argonne National Lab., August 1980.

[6] J. Dongarra, C. Moler, and J. Wilkinson. Improving the accuracy of computed eigenvalues and eigenvectors. *SIAM J. Numer. Anal.*, 20:23–45, 1982.

[7] J. Dongarra and D. Sorensen. A fully parallel algorithm for the symmetric eigenvalue problem. *SIAM J. Sci. Statist. Comput.*, 8:s139–s154, 1987.

[8] J. Dongarra, D. Sorensen, K. Connolly, and J. Patterson. Programming methodology and performance issues for advanced computer architectures. *Parallel Computing*, 8:41–58, 1988.

[9] G. A. Geist. Reduction of a general matrix to tridiagonal form. Technical report, Oak Ridge National Laboratory, February 1989. ORNL/TM-10991.

[10] G. A. Geist, A. Lu, and E. L. Wachspress. Stabilized Gaussian reduction of an arbitrary matrix to tridiagonal form. Technical report, Oak Ridge National Laboratory, February 1989. ORNL/TM-11089.

[11] G. Golub and J.H. Wilkinson. Ill-conditioned eigensystems and the computation of the jordan canonical form. *SIAM Rev.*, 18:578–619, 1976.

[12] G. H. Golub. Some modified matrix eigenvalue problems. *SIAM Rev.*, 15:318–334, 1973.

[13] Gene Golub and Charles Van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, MD, second edition, 1989.

[14] E.R. Jessup and Ilse C.F. Ipsen. Solving the symmetric tridiagonal eigenvalue problem on the hypercube. *SIAM J. Sci. Statist. Comput.*, 11(2):203–229, 1990.

[15] T. Kato. *A Short Introduction to Perturbation Theory for Linear Operators.* Springer Verlag, New York, 1982.

[16] Konrad Knopp. *Theory of Functions.* Dover Publications, New York, NY, 1945.

[17] T.Y. Li, Zhonggang Zeng, and Luan Cong. Solving eigenvalue problems of real nonsymmetric matrices with real homotopies. Preprint, Michigan State University, E. Lansing, MI 48824-1027, 1990.

[18] John W. Milnor. *Topology from the Differentiable Viewpoint.* The University Press of Virginia, Charlottesville, 1965.

[19] J.M. Ortega and W.C. Rheinboldt. *Iterative solution of nonlinear equations in several variables.* Academic Press, New York, NY, 1970.

[20] G. Peters and J.H. Wilkinson. Inverse iteration, ill-conditioned equations and Newton's method. *SIAM Review*, 21:339–360, 1979.

[21] B. T. Smith, J. M. Boyle, J. J. Dongarra, B. S. Garbow, Y. Ikebe, V. C. Klema, and C. B. Moler. *Matrix Eigensystem Routines – EISPACK Guide*, volume 6 of *Lecture Notes in Computer Science.* Springer-Verlag, Berlin, 1976.

[22] J. Wilkinson and C. Reinsch. *Handbook for Automatic Computation: Volume II - Linear Algebra.* Springer-Verlag, New York, 1971.

[23] J. H. Wilkinson. *The Algebraic Eigenvalue Problem.* Oxford University Press, 1965.

VITA

Mohamad Majed Sidani was born on November 30, 1961 in Beirut, Lebanon. He attended elementary and part of complementary school at the Collège de la Sagesse in Beirut, after which he moved to the Lycée Franco-Libanais of the Mission Laïque Française where he graduated in 1980. In the academic year 1980-1981 he attended the Lycée Saint Louis in Paris, France. After that he came back to Lebanon and enrolled in the Faculty of Sciences of the Lebanese University from which he graduated in 1985 with a Licence in Mathematics. In January 1988 he entered the graduate program of the department of Mathematics at the University of Tennessee in Knoxville.