Marion P. Emerson

Approved for the Major Department

David Durst 11/23/84

Approved for the Graduate Council

AN ABSTRACT OF THE THESIS OF

Hossein Saiedian for the Master of Science

in Mathematics presented in December 1983

Title: Methods of Finding the Eigenvalues of a Real Matrix

Abstract approved: Marion P. Emerson

Certain numerical analyses primarily concern themselves with problems normally found in the subjects classified as Linear Algebra and Matrix Theory. One of the problems is the determination of the spectrum (set of eigenvalues) and the eigenspaces for a square matrix. Considering the matrix equation $Ax = \lambda x$, the problem is to determine those values of λ for which the equation has a nonzero solution X. These values of λ are called eigenvalues of A. The problem of finding the eigenvalues for A is equivalent to analyzing when the square homogeneous linear system $(A - \lambda I)X = 0$ has a nonzero solution X. This can occur when the system has infinitely many solutions which is equivalent to this condition: $|A - \lambda I| = 0$ (called the characteristic equation). Therefore one way to determine the spectrum of the matrix is to find the roots of the characteristic equation. However this method is inherently unstable since very small errors

in the coefficients of characteristic equation lead to large deviations of spectrum of the matrix. Therefore, we try to use some strategies to reduce the original matrix to a specialized matrix having the same eigenvalues but with more accuracy, and less work.

METHODS OF FINDING THE EIGENVALUES OF A REAL MATRIX

A Thesis

Presented to

the Department of Mathematics

EMPORIA STATE UNIVERSITY

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

---

by

Hossein Saiedian

December 1983

ACKNOWLEDGMENTS

I would like to extend my deep appreciation to Dr. Marion Emerson and Dr. George Poole for their assistance and encouragement and to all my professors at Emporia State University, and especially to Dr. John Carlson who made this study possible.

Acknowledgments and appreciation are also expressed to my brother, Dr. Mehdi Saiedian who encouraged me throughout my study.

TABLE OF CONTENTS

	Page
LIST OF TABLES AND FIGURES	iv
Chapter	
1. INTRODUCTION	1
2. EIGENVALUES	10
LR ALGORITHMS	15
CHOLESKY DECOMPOSITION	17
HOUSEHOLDER REFLECTORS	22
GIVENS ROTATIONS	30
QR DECOMPOSITION	36
SINGULAR VALUE DECOMPOSITION	38
3. SUMMARY	45
BIBLIOGRAPHY	49
APPENDIX	51

LIST OF TABLES AND FIGURES

Table — Page

3.1 Number of Multiplications in the Reduction to Specialized Form 45

3.2 Number of Multiplications to Evaluate, for One Trial λ, the Characteristic Polynomial Given as a Determinant 46

3.3 Number of Multiplications Per Iteration in the LR Family 47

Figure

1. 11

2. 23

3. 24

4. 39

Chapter 1

INTRODUCTION

Since the implementation in 1947 of electronic digital computers in obtaining numerical approximations to Poisson's partial differential equation, Numerical Linear Algebra has come to be recognized as a fairly well defined and active discipline of study. In fact, it would be difficult to separate the history and development of digital computers and Numerical Linear Algebra over the last thirty-five years. Computer capabilities advanced during the 1950's and 1960's to the extent that it permitted numerical analysts to consider new and more complicated problems whose solutions could be numerically approximated, or at least investigated. It now seems that with the advent of each new generation of computers, mathematical physicists feel compelled to offer a new set of problems whose level of difficulty exceeds that of the problems solved by the previous computer generation.

Certain numerical analysts primarily concern themselves with problems normally found in the subjects classified as Linear Algebra and Matrix Theory. We will refer to these topics as Numerical Linear Algebraist (NLA). There are basically two problems with which the NLA's concern themselves; the solution of linear systems of equations and the

determination of the spectrum and eigenspaces for a square matrix, both of which are related.

It goes without saying that the NLA should maintain some degree of knowledge of the state of the art in computer technology and computer capabilities since his research and applications should be influenced by such knowledge.

ANALYZING POSSIBLE SOLUTIONS TO THE LINEAR ALGEBRAIC SYSTEM $Ax = b$. There are many physical problems which when modeled by an appropriate mathematical expression leads one to consider the solution to a system of m linear equations in n unknowns which shall be represented by the matrix equation

$$Ax = b \tag{1.1}$$

A shall be called the coefficient matrix and b shall be called the constant vector.

Such linear systems arise in the physical sciences, social and behavioral sciences, economics, electrical engineering, structural engineering, operations research, computerized image reconstruction, cryptology, and many others.

If the information or data in the problem which gives rise to the system $Ax = b$ is excessive, the system may have more equations than unknowns. That is, $m>n$ and in this case we say that $Ax = b$ is an overdetermined system. If the information or data which leads to $Ax = b$ is scarce or insufficient, m will be less than n, in which case we call $Ax = b$

an underdetermined system. If the constant vector b is o, we shall call the system a homogeneous system.

In this section we shall identify the types of solutions that may occur for a given system, depending on whether it is homogeneous, overdetermined, underdetermined, square (m = n), or nonhomogeneous.

In order to determine the possible types of solutions that a given system may have, we shall assume that the reader can reduce the augmented matrix

$$B = [A\ b] \tag{1.2}$$

to Reduced Echelon Form (R.E.F.).

Given the system Ax = b, there are only three possible types of solutions: A Unique Solution, Infinitely Many Solutions, or No Solution. If either of the first cases occur, the system is called consistent. In the latter case the system is called inconsistent.

By considering the R.E.F. for each of the six possible systems Ax = b that may occur, one can easily determine the possible solution sets. These are described in the table on page 4.

Although obtaining the Reduced Echelon Form for the augmented matrix B allows one to determine the solution set for Ax = b, our study of Numerical Linear Algebra will reveal more sophisticated and economical methods for determining solution sets for Linear Systems.

System Ax = b	Possible Solution Types
Nonhomogeneous Square	A Unique Solution (consistent) Infinitely Many Solutions (consistent) No Solution (inconsistent)
Nonhomogeneous Overdetermined	A Unique Solution (consistent) Infinitely Many Solutions (consistent) No Solution (inconsistent)
Nonhomogeneous Underdetermined	Infinitely Many Solutions (consistent) No Solution (inconsistent)
Homogeneous Square	A Unique Solution (consistent) Infinitely Many Solutions (consistent)
Homogeneous Overdetermined	A Unique Solution (consistent) Infinitely Many Solutions (consistent)
Homogeneous Underdetermined	Infinitely Many Solutions (consistent)

DETERMINING EIGENVALUES AND EIGENVECTORS FOR THE SQUARE MATRIX A. The second basic problem which concerns the NLA is the algebraic Eigenvalue-Eigenvector Problem. This problem is very much related to the first basic problem, that of solving the system Ax = b.

The practical importance of the algebraic eigenvalue-eigenvector problem is revealed by considering a system of n first order linear differential equations with constant coefficients to be solved simultaneously:

$$u_1'(t) = a_{11}u_1(t) + a_{12}u_2(t) + \cdots a_{1n}u_n(t) \qquad (1.3)$$

$$u_2'(t) = a_{21}u_1(t) + a_{22}u_2(t) + \cdots a_{2n}u_n(t)$$

$$\vdots$$

$$u_n'(t) = a_{n1}u_1(t) + a_{n2}u_2(t) + \cdots a_{nn}u_n(t)$$

In standard form, the system (1.3) may be written as

$$\frac{du}{dt} = Au \tag{1.4}$$

where $u = (u_1(t), u_2(t), \ldots, u_n(t))^T$ and $\frac{du}{dt} = (u_1'(t), u_2'(t), \ldots, u_n'(t))^T$. By substitution, it is clear that $u = xe^{\lambda t}$ is a solution only if $Ax = \lambda x$. Conversely, if λ and $x \neq 0$ satisfy $Ax = \lambda x$, then $u = xe^{\lambda t}$ is a solution of (1.4). This is the algebraic eigenvalue-eigenvector problem: Determine λ and $x \neq 0$ such that $Ax = \lambda x$.

Suppose A is a real square matrix of order n. Consider the matrix equation

$$Ax = \lambda x \tag{1.5}$$

There are two problems we wish to consider in regard to the equation (1.5).

First, determine those values λ for which the equation (1.5) has a nonzero solution x. These values of λ are called eigenvalues of A. Second, for each λ for which equation (1.5) has a nonzero solution, find all the vectors which satisfy equation (1.5) for a given λ. Each nonzero vector x satisfying equation (1.5) for a given eigenvalue λ is called an eigenvector of A corresponding to λ.

The first problem of finding eigenvalues for A is equivalent to analyzing when the square homogeneous linear system

$$(A-\lambda I)x = 0 \tag{1.6}$$

has a nonzero solution x. This can occur only when the system has infinitely many solutions which is equivalent to each of the following conditions:

R.E.F. of $(A-\lambda I)$ has at least one zero row (1.7)

rank $(A-\lambda I) < n$ (1.8)

$|A-\lambda I| = 0$ (called the characteristic equation) (1.9)

$|\lambda I-A| = 0$ (1.10)

Example 1: Consider the matrix

$$A = \begin{bmatrix} 8 & 2 & -2 \\ 3 & 3 & -1 \\ 24 & 8 & -6 \end{bmatrix}$$

Using equation (1.10) above, the characteristic equation of A is

$$|\lambda I - A| = \begin{bmatrix} \lambda-8 & -2 & 2 \\ -3 & \lambda-3 & 1 \\ -24 & -8 & \lambda+6 \end{bmatrix} = 0$$

or $$\lambda^3 - 5\lambda^2 + 8\lambda - 4 = 0$$

So the eigenvalues of A are 1 and 2 and $\sigma(A) = \{1,2\}$ is called the spectrum of A.

The second problem of finding the eigenvectors for A corresponding to the eigenvalue λ_0 is equivalent to determining the solution set for the homogeneous system

$$(\lambda_0 I - A)x = 0 \qquad (1.11)$$

The solution set for (1.11) is the null space N of $(\lambda_0 I - A)$. A reformulation of the Eigenvector Problem is this: for each eigenvalue λ_0 of A, find a basis $B(\lambda_0)$ for the null space $N(\lambda_0)$ for $(\lambda_0 I - A)$. $N(\lambda_0)$ is called the Eigenspace of A corresponding to λ_0 and each element of $B(\lambda_0)$ is an eigenvector of A corresponding to λ_0.

Example 2: Consider the matrix A in the earlier example where eigenvalues are $\lambda_1 = a$ and $\lambda_2 = 2$. To find a basis B(1) for N(1) we will determine the R.E.F. for the augmented matrix $[(\lambda_1 I - A) \quad 0]$ corresponding to the system $(\lambda_1 I - A)$ $x = 0$:

$$[\lambda_1 I - A \; -] = \begin{bmatrix} -7 & -2 & 2 & 0 \\ -3 & -2 & 1 & 0 \\ -24 & -8 & 7 & 0 \end{bmatrix} \qquad (1.12)$$

Gaussian Elimination reduces the matrix in (1.12) to

$$\begin{bmatrix} 1 & 0 & -1/4 & 0 \\ 0 & 1 & -1/8 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

whose general solution is

$$\begin{cases} x = 1/4z \\ y = 1/8z \\ z \text{ arbitrary} \end{cases}$$

That is,

$$N(\lambda_1) = \{(2z, z, 8z) \mid z \text{ is real}\}$$

and a basis for $N(\lambda_1)$ is $B(\lambda_1) = \{(2, 1, 8)\}$ (by choosing $z = 1$) $e = (2, 1, 8)$ is an eigenvector A corresponding to $\lambda_1 = 1$. The eigenspace $N(\lambda_1)$ is one dimensional and every eigenvector corresponding to $\lambda_1 = 1$ is a nonzero multiple of e.

To find a basis B(2) for N(2) we shall determine the R.E.F. for

$$[\lambda_2 I - A \; 0] = \begin{bmatrix} -6 & -2 & 2 & 0 \\ -3 & -1 & 1 & 0 \\ -24 & -8 & 8 & 0 \end{bmatrix} \qquad (1.13)$$

Gaussian Elimination reduces the matrix in (1.13) to

$$\begin{bmatrix} 1 & 1/3 & -1/3 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

whose solution is

$$\begin{cases} x = -1/3y + 1/3z \\ y \text{ arbitrary} \\ z \text{ arbitrary} \end{cases}$$

From this we can determine a basis for $N(\lambda_2 = 2)$ by choosing $y = 3$, $z = 0$ in one case and $y = 0$, $z = 3$ in the second. So $B(\lambda_2 = 2) = (-1, 3, 0), (1, 0, 3)$ and $e_1 = (-1, 3, 0)$, $e_2 = (1, 0, 3)$ are eigenvectors of A corresponding to $\lambda_2 = 2$.

The eigenspace $N(\lambda_2)$ is two dimensional and every eigenvector corresponding to $\lambda_2 = 2$ is a nonzero linear combination of e_1 and e_2.

In this chapter we have suggested that one way to determine the spectrum $\sigma(A)$ of A is to find the roots of the characteristic equation $C(\lambda) = |\lambda I - A| = 0$. In fact, earlier methods for determining $\sigma(a)$ were based on approximating roots to $C(\lambda) = 0$. But such methods are inherently unstable since very small errors in the coefficients of $C(\lambda)$ lead to large deviations in $\sigma(A)$, even though $\sigma(A)$ is not particularly sensitive to small changes in the entries of A. It then appears that the most practical and effective methods for determining $\sigma(A)$ must be more closely related to A than to the more distant problem of finding roots to $C(\lambda) = 0$.

In the next chapter we will discuss the rotational techniques suitable for finding all the eigenvalues for reasonable matrices.

A FORTRAN program is presented in the Appendix. This program, which uses seven other FORTRAN program segments (subroutines), applies some of the method and algorithms presented in Chapter 2 to compute the eigenvalues of a square real matrix.

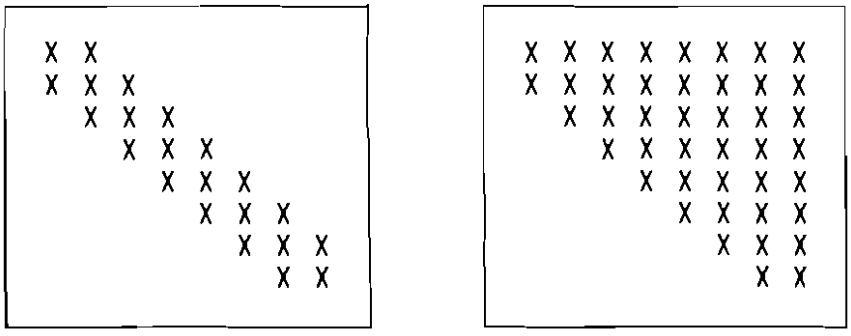
Chapter 2

EIGENVALUES

In this chapter we shall deal with methods that can produce all the eigenvalues without overmuch bother about whether they are large or small or somewhere in between. Furthermore, we shall have large matrices in mind so that we are talking, typically, about 50 to 100 eigenvalues. Since finding all 100 eigenvalues of a 100 x 100 matrix is a lot of work, we are going to be continually concerned about the efficiency of our algorithm. Further, since large matrices require many arithmetic subtractions, we must worry about the stability of the algorithms lest we obtain numerical results that are, in fact, random numbers instead of eigenvalues.

The perspicacious reader may have noticed a curious avoidance of the word eigenvector, which is not accidental. The algorithms that expediently produce all 100 eigenvalues, do not simultaneously find the corresponding eigenvectors, and so this calculation follows as nearly separate and additional labor. Fortunately, the physical problems that lead to large eigenmatrices frequently do not require the vectors to be found, so that the postponing (perhaps indefinitely!) of the topic is not solely due to expedience.

THE GRAND STRATEGY. While algorithms for completely diagonalizing a general nondefective matrix exist, they require far too much computation. The practical strategies break the problem into two parts: reducing the original matrix to a specialized matrix having many zero elements but the same eigenvalues, followed by the finding of the eigenvalues for the specialized matrix. These specialized matrices are either tridiagonal or Hessenberg, which forms are illustrated in Figure 1. The desirability of this dual



A tridiagonal matrix A Hessenberg matrix

Figure 1

strategy becomes apparent if we consider the amount of work facing a man who would find all the eigenvalues of a large general matrix. The problem, by definition, is that of finding all the zeros of a high-degree polynomial--a process that is necessarily infinite, since there are no explicit formulas. Further, any serious manipulation with a general matrix, such as multiplication by another matrix, will usually require kn^3 arithmetic operations, where k depends on

the particular manipulation. In contrast, manipulations with tridiagonal matrices tend to be proportional to n. Hessenberg matrices lie between, usually requiring kn^2 operations. But we can reduce a general matrix to tridiagonal or Hessenberg form in a finite number of arithmetic steps (the number of multiplications is close to n^3).

Thus the grand strategy is to postpone the infinite algorithm until we have reduced our matrix to one of the specialized forms--being happier about endlessly iterating an algorithm that costs kn multiplications per iteration than doing the same with one costing kn^3. When we consider that computing centers are regularly asked to find the eigenvalues of 200 x 200 matrices and larger, we realize that the difference between n^2 and n^3 are startling--not to mention n versus n^3! Only a socially irresponsible man would ignore such computational savings.

The two stages of our eigenvalue strategy are nearly independent. We shall discuss them separately. The finding of eigenvalues from tridiagonal or Hessenberg forms turns out, not very surprisingly, to be the familiar problem of finding the roots of a polynomial. The difficulties are already familiar, as are the remedies. Accordingly we shall spend most of our time discussing the first stage: The reduction of a general matrix to a specialized form while carefully preserving the eigenvalues. It is here that the crucial questions about computational labor and computational stability arise. Often the methods that seem efficient are

also unstable and hence ineffective. Since there is a variety of algorithms, each having its special utility, with no general superior tactic available, we shall be emphasizing our familiar theme of suiting the tool to the job.

THE STANDARD TOOLS. All reductions of general matrices may be phrased in terms of similarity transformations. The matrix A is said to suffer a similarity transformation if it is pre- and postmultiplied by any other matrix and its inverse. Thus B and C are similarity transformations of A if

$$B = T^{-1}AT \quad \text{or} \quad C = TAT^{-1}$$

The only restrictions are the dimensionalities being compatible with the indicated multiplications, plus the existence of the inverse of T. Similarity transformations have the property of preserving eigenvalues. Starting with the eigenproblem equation

$$Ax = \lambda x$$

where x is an eigenvector of A and λ is the corresponding eigenvalue we have

$$TAx = \lambda Tx$$

and on defining

$$Tx = y \quad \text{or} \quad x = T^{-1}y$$

we get

$$TAT^{-1}y = \lambda TT^{-1}y = \lambda y$$

Thus we have the new eigenproblem

$$Cy = \lambda y$$

with the new matrix C that is similar to A, new eigenvectors y that are simply related to x--but still the same old λ's.

The class of similarity transformations is very broad. Within it is included a much more restrictive type, the orthogonal transformation. Here the transforming matrix T is orthogonal, which implies that its transpose is also its inverse. We have

$$T^T = T^{-1}$$

and this property is enough to preserve symmetry through the similarity transformation. We note that the transpose of C is

$$C^T = (TAT^T)^T = TA^TT^T$$

which, if A is symmetric, becomes

$$C^T = TAT^T = C$$

Thus orthogonal transformations preserve both eigenvalues and symmetry, when the latter exists. They are also extremely stable. As one might suspect, any transformation that has such admirable properties must also suffer some drawbacks. Orthogonal transformations tend to require more arithmetic than their competitors. Thus we will tend to use them for

those matrices that are poorly behaved or when other methods happen to fail.

LR ALGORITHMS

Barring unfortunate accidents of degeneracy, any matrix may be decomposed into the product of a lower unit triangulat matrix and an upper triangular matrix. Symbolically we have

where the symbols are meant to convey the presence of 1's on the principal diagonal of L and no such specialization of R. The algorithm effecting such separation is given immediately by filling our schematic with components and writing out the implied matrix multiplication. It is precisely the Gaussian triangularization by elimination of variables, familiar to every college student. For stability it is usuqlly necessary that row interchanges be performed to bring large pivot elements to the diagonal.

If we now reverse the order of our factor matrices and remultiply them, we create another square matrix, completing one cycle of an iteration. We have

$$A_s \longrightarrow L_s R_s \qquad \text{then} \qquad R_s L_s \longrightarrow A_{s+1} \tag{2.1}$$

We see that the iteration preserves eigenvalues, for it is a similarity transformation:

$$A_{s+1} = R_s L_s = L_s^{-1} A_s L_s$$

It will be proved below that the sequence of A_s converges to upper triangular form, hence reveals the eigenvalues on the principal diagonal. A necessary and sufficient corollary is the convergence of L_s to a unit matrix, and it is this fact that will be proved. But first we would examine the utility of the algorithm.

For a general square matrix, symmetric or unsymmetric, the separation requires $n^3/3$ multiplications, the recombination another $n^3/3$. Since the algorithm produces eigenvalues it is necessarily infinite, and we have steadily warned against engaging in infinite iterations with $O(n^3)$ multiplications per iteration. We still do. The LR algorithm is not practical for general matrices. But it has modifications that are practical with specialized matrices.

If we write out the triangular decomposition algorithm (2.1) we find that triangles of zeros in the lower left corner of A are transmitted identically to the same elements of L. Likewise, triangles of zeros in the upper right corner of A turn up unchanged in R. Further, when L and R are remultiplied in the reverse order, our zeros reappear so that an original banded matrix A, has its form reproduced in A_{s+1}. In particular, tridiagonal and Hessenberg matrices keep their tridiagonal and Hessenberg forms under the LR algorithm. Since factorization of tridiagonal matrices is $O(n)$ and that of Hessenberg $O(n^2)$, we just might hope that

the LR algorithm would compete effectively for finding eigenvalues of these specialized forms. It will turn out that it does but only after elaborating the algorith with a number of devices for suppressing roots as they are found, accelerating convergence of the iterations, and splitting the problem into two or more subproblems whenever possible--in short, by using all the tricks that we need for finding roots of a high degree polynomial.

We are now in the position of hoping to use LR on specialized matrices, since the volume of work is of the right order of magnitude. Still the nagging problem of interchanges remains. Without interchanges, we know that the Gaussian triangular decomposition is unstable. With interchanges, we seem to risk loss of our specialized forms. Trial of a few examples, however, will soon convince the sceptical that zeros below the diagonal are preserved by LR with interchanges, while zeros above the diagonal are gradually lost. Thus Hessenberg form is preserved, though tridiagonal is not. Accordingly, we might expect LR_wI to be useful for unsymmetric matrices that have already been reduced to Hessenberg form.

CHOLESKY DECOMPOSITION

For certain kinds of nonsingular symmetric matrices A, not only does an LR decomposition exist, but R can be economically constructed so that $A = LR = LL^T$. However, symmetry alone will not guarantee the existence of a decomposition of

the form LL^T, a form known as the Cholesky Decomposition. For example, a nonsingular symmetric matrix with a negative element in the (1,1) position can not have a Cholesky Decomposition.

It turns out that a Cholesky Decomposition exists for positive definite (matrix A is positive definite if for each $X \neq 0$, $X^TAX > 0$) matrices and, in particular, the nonsingular Gramian matrices (of the form BB^T) which arise in least squares problems.

Suppose A is a square matrix of order n and σ is a subset of $S = \{1,2,\ldots,n\}$. Let $B[\sigma]$ be the submatrix of A obtained by deleting row i and column i of A for each $i \notin \sigma$. $B[\sigma]$ is called a principal submatrix of A for each subset σ of S. $B[\sigma]$ is the submatrix of A determined by rows i and columns i, $i \in \sigma$.

Lemma. Each principal submatrix of a positive definite matrix is also positive definite.

Proof. Suppose $B[\sigma]$ is a principal submatrix of A and A is positive definite. Assume σ contains k elements. Let $z = (z_1, z_2, \ldots, z_k)^T$ by a nonzero k-vector and define $x = (x_1, x_2, \ldots x_n)^T$ by setting

$$x_{s_i} = z \quad \text{if} \quad i \in \sigma = (s_1, s_2, \ldots, s_k)$$

$$x_i = 0 \quad \text{if} \quad i \notin \sigma$$

Since $z \neq 0$, $x \neq 0$. It is clear that $x^TAx = z^TB[\sigma]z$ and since A is positive definite ($x^TAx>0$), so is $B[\sigma]$.

From Lemma, we note that the diagonal elements of a positive definite matrix must be positive. Furthermore, as with Gaussian Elimination, the method requires no partial pivoting to determine an LR decomposition for positive definite matrices. Consequently, if A is positive definite, A has an LR decomposition. We need to know that if we assume $R = L^T$, then the Gaussian Elimination does not break down for positive definite matrices where we do not assume L is unit triangular.

Suppose we seek a lower triangular matrix $L = [\ell ij]$ such that $A = LL^T$ and A is positive definite

$$A = LL^T = \begin{bmatrix} \ell_{11}^2 & \ell_{11}\ell_{21} & \ell_{11}\ell_{31} & \cdots \\ \ell_{21}\ell_{11} & \ell_{21}^2+\ell_{22}^2 & \ell_{21}\ell_{31}+\ell_{22}\ell_{32} & \cdots \\ \ell_{31}\ell_{11} & \ell_{21}\ell_{31}+\ell_{22}\ell_{32} & \ell_{31}^2+\ell_{32}^2+\ell^2{}_{33} & \cdots \\ \vdots & \vdots & \vdots & \end{bmatrix}$$

By considering the product matrix LL^T we first note that $L^2{}_{11} = a_{11}$. Since $a_{11}>0$, $\ell_{11} = \sqrt{a_{11}}$ is well defined. The remaining elements in the first column of L and first row of L^T (which must agree) must satisfy $\ell_{t1} = a_{t1}/\ell_{11}$. Having determined the elements in the first column of L (and, hence, the first row of L^T), we note that $a_{22} = \ell_{21}^2 + \ell_{22}^2$ so that

we must have $\ell_{22} = (a_{22} - \ell_{21}^2)^{\frac{1}{2}}$. ℓ_{22} is well defined since A being positive definite implies the principal minor $a_{11}a_{22} - a_{21}^2 > 0$ and, hence, $\ell_{11}^2 a_{22} - (\ell_{11}\ell_{21})^2 > 0$ or $a_{22} - \ell_{21}^2 > 0$. We now can define the elements in the second column of L (and second row of L^T) by setting

$$\ell_{t2} = \ell_{22}^{-1} \left[a_{t2} - \sum_{k=1}^{t-1} \ell_{t1}\ell_{2k} \right]$$

for each $t = 3,4,\ldots,n$.

It is clear that we can complete the procedure inductively, using the fact that A is positive definite. The general algorithm is described below.

Now, to show that this process can be carried out inductively, assume that each symmetric positive definite matrix A of order n - 1 or less has an LR decomposition of the form $LR = LL^T$. Suppose that A is a symmetric positive definite matrix of order n and write A as

$$A = \begin{bmatrix} A_{n-1} & b \\ b^T & a_{nn} \end{bmatrix}$$

where $A_{n-1} = W$ is the leading principle submatrix of order n-1. By Lemma, W is positive definite so that by our hypothesis, W factors into ZZ^T. Since W is nonsingular, so is Z. Therefore, $Zc = b$ is solvable for c. Now for any value of x,

$$\begin{bmatrix} Z & 0 \\ c^T & x \end{bmatrix} \cdot \begin{bmatrix} Z^T & c \\ 0 & x \end{bmatrix} = \begin{bmatrix} W & b \\ b^T & c^Tc+x^2 \end{bmatrix} = A$$

If a_{nn} is set equal to $c^Tc + x^2$ and the resulting equation is solved for x, then the desired LL^T factorization of A is given above. However, to show that x is real, we take determinants of both sides of the matrix equation given above to obtain

$$\det(LL^T) = [\det(L)]^2 = x^2[\det(Z)]^2 = \det(A)$$

Since A is positive definite, $\det(A) > 0$. Hence, x can be solved for as real and positive.

CHOLESKY DECOMPOSITION ALGORITHM. If A is a positive definite matrix of order n, then A has the Cholesky Decomposition LL^T where L can be obtained as follows:

Enter the lower triangular portion of $A = [a_{ij}]$ (diagonal elements and below) since these are the only elements required in the computation of L. Also, it is possible to overwrite A with L.

[1] $a_{11} \leftarrow \ell_{11} = \sqrt{a_{11}}$

[2] Determine the first column of L by setting

$$a_{t1} \leftarrow \ell_{t1} = a_{t1}/\ell_{11} = a_{t1}/a \text{ for } t = 2,3,\ldots,n$$

[3] Set $i = 2$

[4] Set $a_{ii} \leftarrow \ell_{ii} = \left[a_{ii} - \sum_{k=1}^{i-1} a^2_{ik} \right]^{\frac{1}{2}}$

[5] If $i < n$, go to step 6

If $i = n$, go to step 8

[6] Determine the i^{th} column of L by setting

$$a_{ti} \leftarrow \ell_{ti} = \ell_{ii}^{-1} \left[a_{ti} - \sum_{k=1}^{i-1} a_{tk} a_{ik} \right]$$

for each $t = i + 1, i + 2, \ldots, n$

[7] Set $i = i + 1$ and go to step 4.

[8] Procedure is complete and A has been overwritten with L.

Because the Cholesky Decomposition requires only the computation of L for the positive definite matrix A, it is easy to conclude that the amount of arithmetic required is approximately half of the amount required for determining the general LR decomposition by Gaussian Elimination.

HOUSEHOLDER REFLECTORS

For each vector u, the matrix

$$H = I - 2 \frac{uu^T}{u^T u} \tag{2.2}$$

is called a Householder Transformation or Householder Reflector. H is called a "reflector" because H is the non-singular linear transformation which maps vectors x into

their reflection about the subspace which is orthogonal to the space generated by u in R^n. That is, for a given vector u, H maps x into the vector Hx which is the reflection about the orthogonal complement of u, namely $\langle u \rangle^{\perp}$ (see Figure 2).

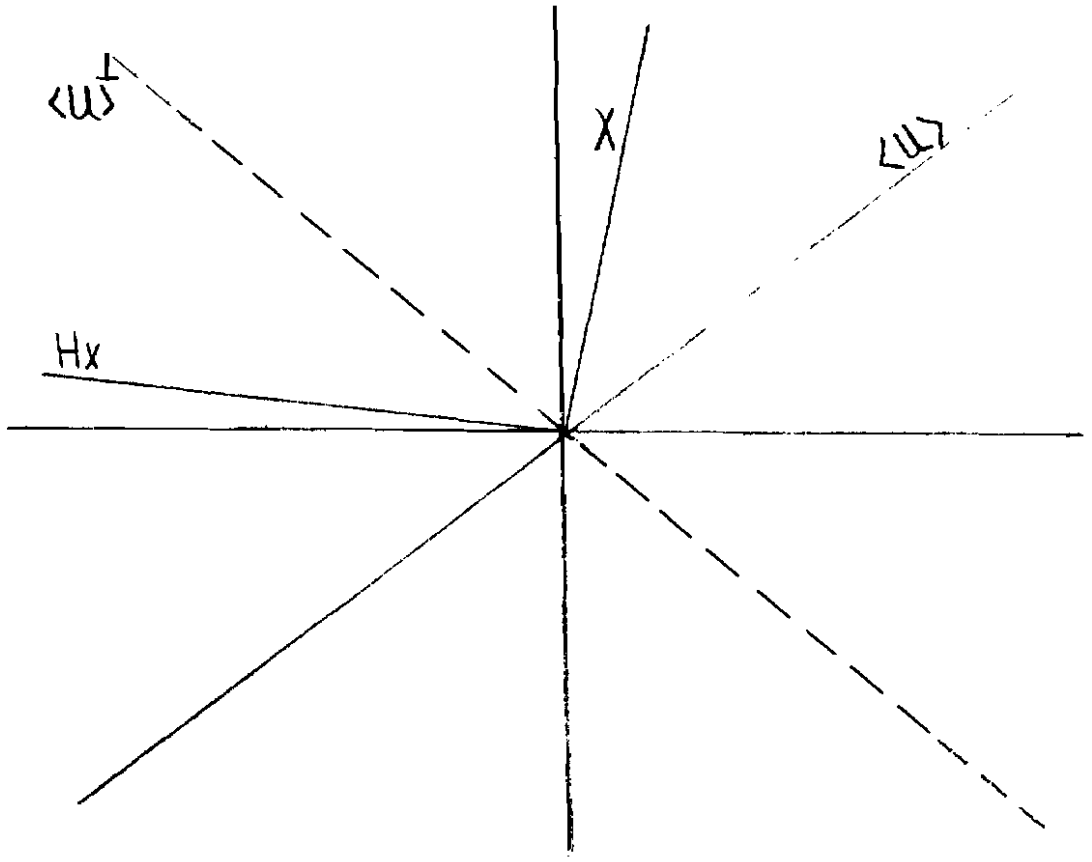


Figure 2

In particular, we shall be interested in the case where $x = (x_1, x_2, \ldots, w_n)^T$ is given and u shall be assigned the vector value

$$u = x \pm ||x|| e_1 \qquad (2.3)$$

where $e_1 = (1,0,0,\ldots,0)^T$ and the sign is determined by choosing the same sign as x_1. That is, the sign is chosen so that $||u||$ is maximum.

With u defined as in (2.3), H_u will satisfy $H_u x = \pm||x||e_1$. That is, H_u maps x into a vector all of whose components, except the first, are zero. We shall omit the subscript of u of H (see Figure 3).

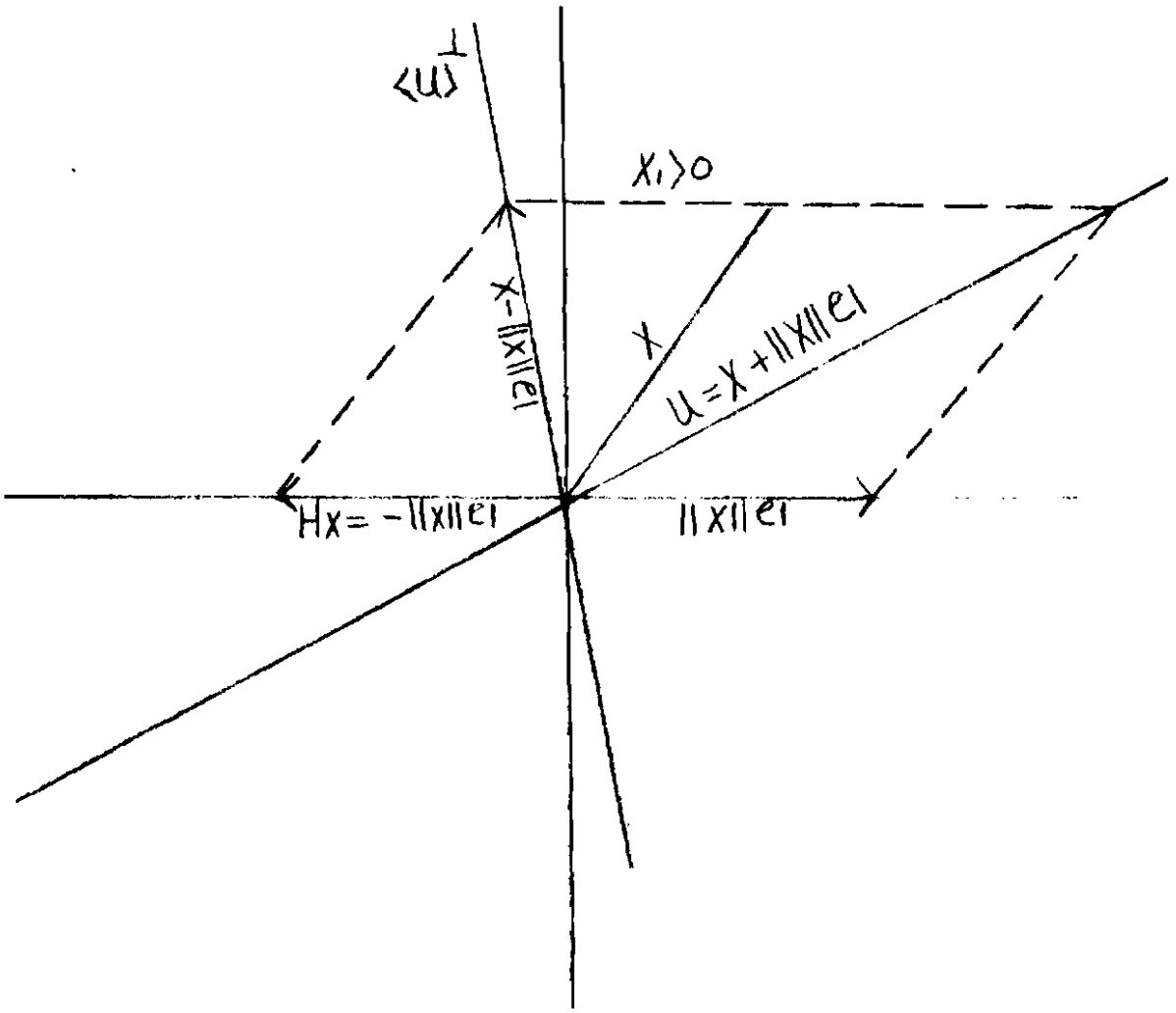


Figure 3

For each u, H is symmetric and orthogonal. For example, if $\alpha = 2/u^Tu$, $H^T = (I - \alpha uu^T)^T = I^T - \alpha(uu^T)^T = I - \alpha uu^T = H$. Furthermore, $H^TH = HH^T = (I - \alpha uu^T)(I - \alpha uu^T) = I - 2\alpha uu^T + \alpha^2 uu^TUU^T =$

$$I - \frac{4uu^T}{u^Tu} + \frac{4uu^Tuu^T}{u^Tuu^Tu} = I - \frac{4uu^T}{||u||^2} + \frac{4u(u^Tu)u^T}{||u||^4} = I.$$ That is, H is orthogonal.

The usefulness of Householder Reflectors lies in the fact that they introduce zeroes into a vector or matrix in a manner which is numerically stable.

Example 1. Consider the vector $x = (2,-3,0,6)^T$ where $||x|| = 7$. As in (2.3) define $u = (2+7, -3,0,6)^T = (9,-3,0,6)^T$. Since $u^Tu = 126$, according to (2.2)

$$H = I - \frac{2}{u^Tu}uu^T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 9/7 & -3/7 & 0 & 6/7 \\ -3/7 & 1/7 & 0 & -2/7 \\ 0 & 0 & 0 & 0 \\ 6/7 & -2/7 & 0 & 3/7 \end{bmatrix}$$

$$= \begin{bmatrix} -2/7 & 3/7 & 0 & -6/7 \\ 3/7 & 6/7 & 0 & 2/7 \\ 0 & 0 & 1 & 0 \\ 6/7 & 2/7 & 0 & 3/7 \end{bmatrix}$$

Now note that $Hx = (-7,0,0,0)^T = -||x||e_1$

Theorem. For a given vector $x = (x_1, x_2, \ldots, x_n)^T$, if u and H are defined as in (2.3) and (2.2), then $Hx = \pm ||x|| e_1$.

Proof. $Hx = (I - \frac{2}{u^T u} uu^T)\, x$

$$= x - \frac{2}{(x^T \pm ||x|| e_1^T)(x \pm ||x|| e_1)} uu^T x$$

$$= x - \frac{2}{2||x||^2 \pm 2x_1 ||x||} uu^T x$$

$$= x - \frac{1}{||x||^2 \pm x_1 ||x||} (x \pm ||x|| e_1)$$

$$[(x^T \pm ||x|| e_1^T) x]$$

$$= x - \frac{(x \pm ||x|| e_1)\,[||x||^2 \pm x_1 ||x||]}{||x||^2 \pm x_1 ||x||}$$

$$= x - (x \pm ||x|| e_1)$$

$$= \pm ||x|| e_1$$

Since H, as defined in (2.2), required a division by $u^T u = ||u||^2$, the sign in (2.3) is chosen so that $||u||$ is maximum, which maximizes numerical stability.

Householder reflectors are primarily used in the algebraic eigenvalue problem which we shall discuss later. However, they can also be used in solving the linear system $Ax = b$.

Example 2. Consider the linear system whose augmented matrix is given by

$$[a_1\ a_2\ a_3\ a_n\ b] = \begin{bmatrix} 2 & 2 & 2 & 1 & 1 \\ -3 & 1 & -1 & 2 & 0 \\ 0 & 2 & 0 & -1 & 1 \\ 6 & 1 & 0 & 3 & 0 \end{bmatrix} \qquad (2.4)$$

where ai denotes the i^{th} column of A. We can employ Householder Reflectors to find an UTF for (2.4) as follows:

The first column a_1 is the vector x given in Example 1. Therefore, if $T_1 = H$ is the Householder Reflector defined in Example 1, then

$$T_1[A\ b] = H[A\ b] = \begin{bmatrix} -7 & -1 & -1 & -2 & -2/7 \\ 0 & 2 & 0 & 3 & 3/7 \\ 0 & 2 & 0 & -1 & 1 \\ 0 & -1 & -2 & 1 & -6/7 \end{bmatrix}$$

$$= A^{(1)} = \begin{bmatrix} a_1^{(1)} & a_2^{(1)} & a_3^{(1)} & a_4^{(1)} & b^{(1)} \end{bmatrix}$$

Now define u and H with respect to all but the first component of $a_2^{(1)}$, namely $x = (2,2,-1)^T$. That is, $u = (2+3,2,-1)^T = (5,2,-1)^T$ and

$$H_1 = I - \frac{2}{u^T u} uu^T = \begin{bmatrix} -2/3 & -2/3 & 1/3 \\ 2/3 & 11/15 & 2/15 \\ 1/3 & 2/15 & 14/15 \end{bmatrix}$$

Define

$$T_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & & & \\ 0 & & H_1 & \\ 0 & & & \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -2/3 & 2/3 & 1/3 \\ 0 & -2/3 & 11/15 & 2/15 \\ 0 & 1/3 & 2/15 & 14/15 \end{bmatrix}$$

so that

$$T_2 A^{(1)} = \begin{bmatrix} -7 & -1 & -1 & -2 & -2/7 \\ 0 & -3 & 2/3 & -1 & -26/21 \\ 0 & 0 & -4/15 & -13/5 & 1/3 \\ 0 & 0 & -28/15 & 9/5 & -11/21 \end{bmatrix}$$

$$= A^{(2)}$$

A scaling of $T_2 A^{(1)}$ yields

$$\tilde{A}^{(2)} = \begin{bmatrix} 49 & 7 & 7 & 14 & 2 \\ 0 & 63 & 14 & 21 & 26 \\ 0 & 0 & 4 & 39 & -5 \\ 0 & 0 & 196 & -189 & 55 \end{bmatrix}$$

Now define u and H with respect to all but the first two components of $\tilde{a}_3^{(2)}$, namely $x = (4,196)^T$. So $u = (200.04,196)^T$ and

$$H_2 = \begin{bmatrix} -.020404 & -.99979 \\ -.99979 & .020404 \end{bmatrix}$$

consequently,

$$T_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \multicolumn{2}{c}{\multirow{2}{*}{H_2}} \\ 0 & 0 & & \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & -.202404 & -.99979 \\ 0 & 0 & -.99979 & .020404 \end{bmatrix}$$

and

$$T_c \tilde{A}^{(2)} = \begin{bmatrix} 49 & 7 & 7 & 14 & 2 \\ 0 & 63 & 14 & 21 & 26 \\ 0 & 0 & -196.04 & 188.16 & -54.887 \\ 0 & 0 & 0 & 42.848 & 6.1212 \end{bmatrix}$$

Back-substitution yields the solution

$$x = (0.0,\ .42857,\ .14286,\ -.14286)^T.$$

GIVENS ROTATIONS

Like Householder Reflections, Givens Rotations are orthogonal transformations which introduce zeroes into a vector or matrix.

The idea of computing eigenvalues of a symmetric matrix A by first reducing A to tridiagonal form was due to Wallace Givens. He used elementary rotation matrices to introduce zeroes in the off-tridiagonal positions of A. Givens was also responsible for employing and developing the use of Sturm Sequences in computing the eigenvalues of the tridiagonal matrix A. However, at the present time we shall only discuss the "introduction of Zeroes" by Givens Rotations and shall deal with the eigenvalue problem later.

Suppose $x = (x_1, x_2, \ldots, x_n)^T$ is a given vector and $1 \leq s < t \leq n$. Define the matrix $G = G(x,s,t) = [g_{ij}]$ as follows:

$$w = \sqrt{x_s^2 + x_t^2}$$

$$g_{ij} = \begin{cases} 1 & \text{if } i = j \text{ and } i \notin \{s,t\} \\ x_s/w & \text{if } i = j = s \text{ or } i = j = t \\ x_t/w & \text{if } i = s \text{ and } j = t \\ -x_t/w & \text{if } i = t \text{ and } j = s \\ 0 & \text{otherwise} \end{cases} \qquad (2.5)$$

The matrix G(x,s,t) has the following form:

$$
\begin{array}{l}
 \\
 \\
 \\
 \\
\text{row } s \\
 \\
 \\
 \\
\text{row } t \\
 \\
 \\
 \\
 \\
 \\
\end{array}
\begin{bmatrix}
1 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 & 0 \\
0 & 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 & 0 \\
\vdots & \vdots & & \vdots & & \vdots & & \vdots & \vdots \\
0 & 0 & \cdots & \frac{x_s}{w} & \cdots & \frac{x_t}{w} & \cdots & 0 & 0 \\
\vdots & \vdots & & \vdots & & \vdots & & \vdots & \vdots \\
0 & 0 & \cdots & \frac{-x_t}{w} & \cdots & \frac{x_s}{w} & \cdots & 0 & 0 \\
\vdots & \vdots & & \vdots & & \vdots & & \vdots & \vdots \\
0 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 & 0 \\
0 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 & 1
\end{bmatrix}
$$

col.s col.t

G(x,s,t) has the following properties:

1. G(x,s,t) is a rotation transformation in the s,t plane of R^n
2. G(x,s,t) is orthogonal
3. The s component of Gx is w
4. The t component of Gx is 0
5. The i component of Gx is x_i if $i \neq s,t$.

Consequently, G "zeroes" the t^{Th} component of x while replacing the s^{Th} component with w.

Example 3. Suppose we used Givens Rotations to place the following matrix in UTF:

$$\tilde{A} = \begin{bmatrix} 2 & 6 & 1 & -1 & 1 & 2 \\ 0 & -3 & 0 & 5 & 10 & -15 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 4 & 10 & -5 & 20 & 0 \\ 0 & 0 & -1 & 2 & -1 & 3 \end{bmatrix}$$

If $x = (6,-3,0,4,0)^T$ denotes the second column of A, we need to introduce a zero into the (4,2) position of A or the $t = 4$ position of x. We shall choose $s = 2$, for if we choose $s = 1$, we will destroy one of the zeroes in column one. Let $w = \sqrt{(-3)^2 + (4)^2} = 5$. Then

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & -\frac{3}{5} & 0 & \frac{4}{5} & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & -\frac{4}{5} & 0 & -\frac{3}{5} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$G\tilde{A} \quad \begin{bmatrix} 2 & 6 & 1 & -1 & 1 & 2 \\ 0 & 5 & 8 & -7 & 10 & 9 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & -6 & -1 & -20 & 12 \\ 0 & 0 & -1 & 2 & -1 & 3 \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 0 & 2 & -3 \\ 0 & 1 & -1 & 4 \\ 2 & -1 & 0 & 3 \\ -3 & 4 & 3 & -1 \end{bmatrix}$$

As we noted, one cannot use the corners of the matrix to define a Givens Rotation G which has the property that GAG^T has zeroes at the corner positions. However, we can use rows 1 and 2 (and columns 1 and 2) to annihilate the desired elements. We are interested in using a linear combination of the elements in the (1,4) and (2,4) positions to "zero" the (1,4) position:

$$w = \sqrt{(-3)^2 + (4)^2} = 5$$

$$G = \begin{bmatrix} \frac{4}{5} & \frac{3}{5} & 0 & 0 \\ -\frac{3}{5} & \frac{4}{5} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

As far as obtaining the UTF for a linear system $Ax = b$, Householder Reflectors will accomplish the same thing that Givens Rotations will accomplish, but with half the labor. So whyconsider Givens Rotations? Although we shall not have time to illustrate the fact, Givens Rotations can be employed in certain settings of either of the two basic problems ($Ax = b$ or $Ax = \lambda x$), showing a distinct advantage over the Householder Reflectors.

Furthermore, the scheme given in (2.5) is designed to introduce a zero in the t^{Th} component of a given vector x or, as demonstrated in Example 3, introduce a zero in the (t,s) position of a matrix. However, the matrix G will <u>not</u> introduce zeroes in the "corner positions" (i.e., (t,s), (s,t)) for symmetric matrices A by pre and post multiplying A by G and G^T. Unless a better scheme is found, a second Givens Rotation applied to a matrix A may destroy the zero introduced by the first Givens Rotation, particularly if one is seeking to tridiagonalize A. A slight modification of the scheme (2.5) will lead to a scheme which will annihilate symmetric elements of a symmetric matrix.

We shall illustrate the basic procedure through a simple example and trust the reader to the task of describing a general algorithm.

<u>Example 4</u>. Consider the matrix A below in which we plan to introduce zeroes in the (1,4) and (4,1) positions by a similarity involving a Givens Rotation:

Then

$$GAG^T = \begin{bmatrix} \frac{4}{5} & \frac{3}{5} & 0 & 0 \\ -\frac{3}{5} & \frac{4}{5} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 & -3 \\ 0 & 1 & -1 & 4 \\ 2 & -1 & 0 & 3 \\ -3 & 4 & 3 & -1 \end{bmatrix} G^T$$

$$= \begin{bmatrix} \frac{4}{5} & \frac{3}{5} & 1 & 0 \\ -\frac{3}{5} & \frac{4}{5} & -2 & 5 \\ 2 & -1 & 0 & 3 \\ -3 & 4 & 3 & -1 \end{bmatrix} \begin{bmatrix} \frac{4}{5} & -\frac{3}{5} & 0 & 0 \\ \frac{3}{5} & \frac{4}{5} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & -2 & 5 \\ 1 & -2 & 0 & 3 \\ 0 & 5 & 3 & -1 \end{bmatrix}$$

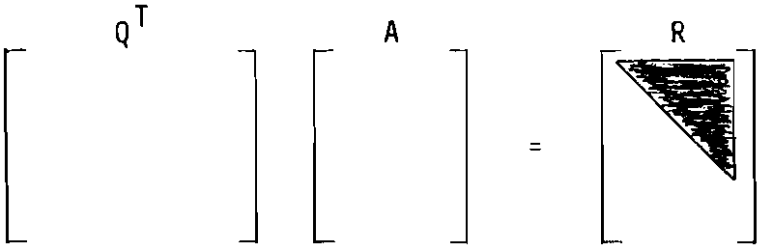
In a similar fashion, the (1,3) and (3,1) positions could be "zeroed" by a Givens Similarity Rotation by using rows (and columns) 2 and 3. Ditto for the (2,4) and (4,2) positions. Consequently, the symmetric matrix A above can be tridiagonalized.

QR DECOMPOSITION

For each matrix A there is an orthogonal matrix Q and an upper trapezoidal matrix R such that A = QR. One can use such decompositions for square matrices toconstruct an iterative sequence of matrices whose diagonal elements converged to the eigenvalues of the given matrix A. Furthermore, the QR decomposition of a matrix can be used in solving least squares problems.

There are several methods which can be used to determine a QR Decomposition for a given matrix: Gram-Schmidt Orthogonalization, Givens Rotations, and Householder Reflectors. We shall employ Householder Reflectors since they lead to a simple procedure which is easily explained.

Suppose A is an mxn matrix and t = minimum {m,n}. Let $Q^T = H_t \ldots H_2 H_1$ denote a product of Householder Reflectors such that $Q^T A = R$ where R is upper trapezoidal:



Then, since Q^T is orthogonal, A = QR. If A is square, R is square and upper triangular.

Example 5. Consider the matrix A given by

$$A = \begin{bmatrix} 2 & -\frac{14}{3} & 7 \\ -3 & 0 & 14 \\ 6 & 7 & 7 \end{bmatrix}$$

For $x = (2,-3,6)^T$, define $u = (9,-3,6)^T$ so that

$$H_1 = I - \frac{2}{u^T u} uu^T$$

$$= \begin{bmatrix} -\frac{2}{7} & \frac{3}{7} & -\frac{6}{7} \\ \frac{3}{7} & \frac{6}{7} & \frac{2}{7} \\ -\frac{6}{7} & \frac{2}{7} & \frac{3}{7} \end{bmatrix}$$

$$H_1 A = \begin{bmatrix} -7 & \frac{14}{3} & -2 \\ 0 & 0 & 17 \\ 0 & 7 & 1 \end{bmatrix}$$

For $x = (0,7)^T$, define $u^T = (7,7)$ so that

$$H_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$H_2H_1A = R = \begin{bmatrix} -7 & -\frac{14}{3} & -2 \\ 0 & -7 & -I \\ 0 & 0 & -17 \end{bmatrix}$$

Consequently

$$A = QR = (H_1H_2)R$$

$$= \begin{bmatrix} -\frac{2}{7} & \frac{6}{7} & -\frac{3}{7} \\ \frac{3}{7} & -\frac{2}{7} & -\frac{6}{7} \\ -\frac{6}{7} & -\frac{3}{7} & -\frac{2}{7} \end{bmatrix} \begin{bmatrix} -7 & \frac{14}{3} & -2 \\ 0 & -7 & -1 \\ 0 & 0 & -17 \end{bmatrix}$$

Note, although H_1 and H_2 are orthogonal and symmetric, Q need not be symmetric.

SINGULAR VALUE DECOMPOSITION

Suppose A is a real m by n matrix and $\sigma(G) = \{\sigma_1,\sigma_2,\ldots,\sigma_k\}$ is the set of nonzero non-negative real eigenvalues of the gramian matrix $G = A^TA$. If $s_i = \sqrt{\sigma_i}$ for each i, then there exist orthogonal matrices U_{mxm} and V_{nxn} such that

$$A = U \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix}_{mxn} V \tag{2.6}$$

where $D = \text{diag}\ (s_1,s_2,\ldots s_k)$.

The SVD lends nicely to the least squares problem where one is primarily concerned with the (usually) inconsistent system $Ax = b$. However, even if the system $Ax = b$ does not arise in the context of a least squares problem, but rather is a nonsquare or inconsistent system arising from some other physical problem, the SVD of A provides a least squares solution. That is, if from (2.6) we define

$$A^T = V^T \begin{bmatrix} D^{-1} & 0 \\ 0 & 0 \end{bmatrix}_{n \times m} U^T$$

then A^T is called the Moore-Penrose inverse of A and $x_0 + A^T b$ is the unique vector of smallest norm which minimizes the quantity $||Ax - b||^2$. Geometrically, x_0 is obtained by first locating the vector $z = Ay$ in the range of A where $||z-b||^2 = ||Ay - b||^2$ is smallest. Then among the vectors in the

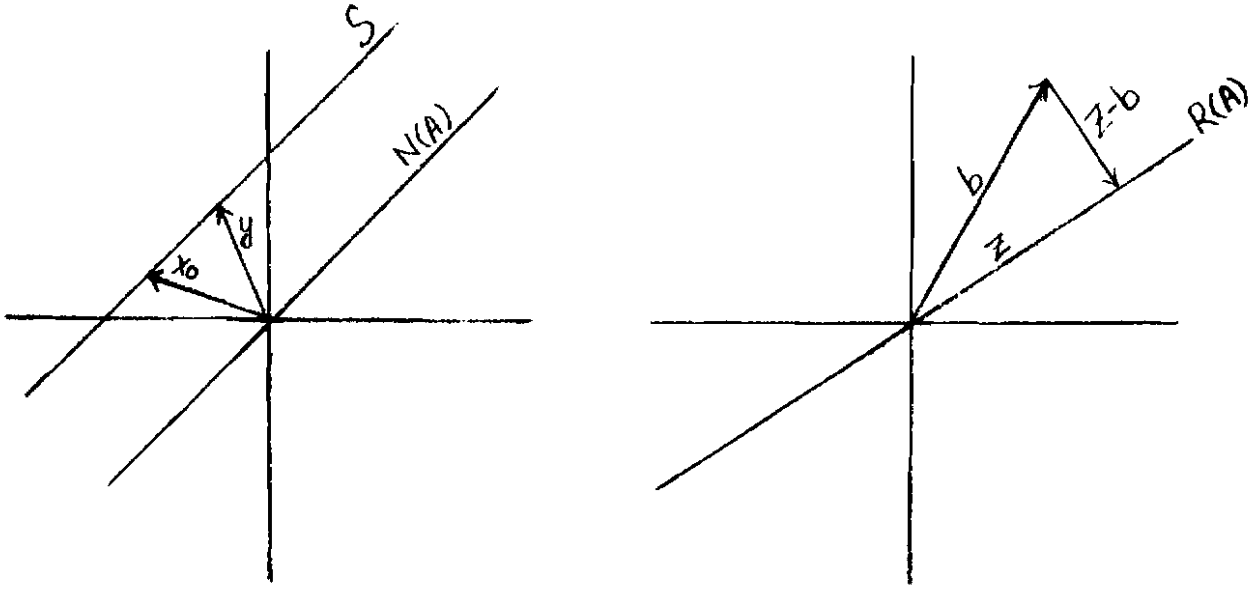


Figure 4

solution set S of the consistent system $Ax = z$ choose x_o to be of smallest length. That is, $x_o \in S = y + N(A)$ is the one vector for which $||Ax_o - b||^2$ and $||x_o||^2$ are smallest.

We shall indicate one method to find the SVD given in (2.6). However, although we shall not present the Golub-Reinsch Algorithm at this time, which employs both Givens Rotations and Householder Reflectors, one should know that it is one of the best methods which exist for finding the SVD.

Method to find SVD.

Step 1. Find the nonzero eigenvalues of $G = A^TA$, $\sigma(G) = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$ where $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \ldots \geq \sigma_k > 0$.

Step 2. Find a complete orthonormal set of eigenvectors for G, say $\{v_1, v_2, \ldots v_k, \ldots, v_n\}$.

Step 3. Find the nonzero eigenvalues of $H = AA^T$, $\sigma(H) = \sigma(G)$.

Step 4. For each $1 \leq i \leq k$ set $u_i = s_i^{-1}Av_i = (\sqrt{\sigma_i^{-1}})Av_i$. Let $\{u_{k+1}, \ldots, u_m\}$ be an orthonormal basis for $N(H)$.

Step 5. Define $V = [v_1\ v_2\ v_3\ \ldots\ v_n]^T$, $U = [u_1\ u_2\ \ldots\ u_m]$ and $D = \text{diag}\ (s_1, s_2, \ldots s_k)$ where $s_i = \sqrt{\sigma_i}$.

Example 6. Find the SVD for the matrix

$$A = \begin{bmatrix} 1 & 0 & -2 & 0 \\ 0 & 3 & 0 & 4 \end{bmatrix}$$

The eigenvalues of $G = A^TA$ and $H = AA^T = \begin{bmatrix} 5 & 0 \\ 0 & 25 \end{bmatrix}$ are $\sigma_1 = 25$ and $\sigma_2 = 5$. A complete set of eigenvectors for G can be found by finding a basis for each solution set to $(\lambda I - G)x = 0$ for each $\lambda \in \sigma(G) = \{25, 5, 0\}$.

For $\lambda = 25$, the augmented matrix for $(25I - G)x = 0$ is

$$\begin{bmatrix} 24 & 0 & 2 & 0 & 0 \\ 0 & 16 & 0 & -12 & 0 \\ 2 & 0 & 21 & 0 & 0 \\ 0 & -12 & 0 & 9 & 0 \end{bmatrix}$$

and the corresponding REF is

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -\frac{3}{4} & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

so the solution set $N(25I - G)$ has dimension one and $v_1' = (0, \frac{3}{4}, 0, 1)^T$ is a basis element. $v_1 = v_1' / ||v_1'|| = (0, \frac{3}{5}, 0, \frac{4}{5})^T$.

For $\lambda = 5$, the augmented matrix for $(5I - G)x = 0$ is

$$\begin{bmatrix} 4 & 0 & 2 & 0 & 0 \\ 0 & -4 & 0 & -12 & 0 \\ 2 & 0 & 1 & 0 & 0 \\ 0 & -12 & 0 & -11 & 0 \end{bmatrix}$$

and the corresponding REF is

$$\begin{bmatrix} 1 & 0 & \frac{1}{2} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

So the solution set $N(5I - G)$ has dimension one and $v_2' = (-\frac{1}{2}, 0, 1, 0)^T$ is a basis element. $v_2 = v_2' / ||v_2'|| = (^{-1}/\sqrt{5}, 0, ^{2}/\sqrt{5}, 0)^T$.

For $\lambda = 0$, the augmented matrix for $(0I - G)x = 0 = Gx$ is

$$\begin{bmatrix} 1 & 0 & -2 & 0 & 0 \\ 0 & 9 & 0 & 12 & 0 \\ -2 & 0 & 4 & 0 & 0 \\ 0 & 12 & 0 & 16 & 0 \end{bmatrix}$$

and the corresponding REF is

$$\begin{bmatrix} 1 & 0 & -2 & 0 & 0 \\ 0 & 1 & 0 & \frac{4}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

So the solution set N(G) has dimension two and can be described as all vectors of the form $n = (2z, -\frac{4}{3}w, z, w)^T$. Choose z, w so that $\{v_3', v_4'\}$ is an orthogonal basis, say $v_3' = (2, 0, 1, 0)^T$ and $v_4' = (0, -4, 0, 3)^T$. Define $v_3 = (2/\sqrt{5}, 0, 1/\sqrt{5}, 0)^T$ and $v_4 = (0, -\frac{4}{5}, 0, \frac{3}{5})$.

An orthonormal set of eigenvectors for $H = AA^T = \begin{bmatrix} 5 & 0 \\ 0 & 25 \end{bmatrix}$ is given by $u_1 = \frac{1}{5}A(0, \frac{3}{5}, 0, \frac{4}{5})^T = (0, 1)^T$ and $u_2 =$

$\frac{1}{\sqrt{5}} A\ (-1/\sqrt{5}, 0, 2/\sqrt{5}, 0)^T = (-1, 0)^T$. If $V = [v_1\ v_2\ v_3\ v_4]^T =$

$$\begin{bmatrix} 0 & \frac{3}{5} & 0 & \frac{4}{5} \\ -1/\sqrt{5} & 0 & 2/\sqrt{5} & 0 \\ 2/\sqrt{5} & 0 & 1/\sqrt{5} & 0 \\ 0 & -\frac{4}{5} & 0 & \frac{3}{5} \end{bmatrix} \quad \text{and } U = [u_1\ u_2] = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix},$$

then $$A = U \begin{bmatrix} 5 & 0 & 0 & 0 \\ 0 & \sqrt{5} & 0 & 0 \end{bmatrix} V$$

Furthermore, $$A^T = V^T \begin{bmatrix} \frac{1}{5} & 0 \\ 0 & \frac{1}{5} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} U^T = \begin{bmatrix} \frac{1}{5} & 0 \\ 0 & \frac{3}{25} \\ -\frac{2}{5} & 0 \\ 0 & \frac{4}{25} \end{bmatrix}.$$

Chapter 3

SUMMARY

In an attempt to distinguish the general strategies for eigenvalues, as well as the peculiar competences of our several algorithms, we here tabulate their principal properties, placing special emphasis on the amount of work that each requires. Generally our measure is the number of multiplications, although we comment additionally if the square-root is noticeable.

The finite part of our strategy, reduction to a specialized form, is covered in Table 3.1.

Table 3.1

Number of Multiplications in the Reduction to Specialized Form

	Symmetric		Nonsymmetric
	General to tridiagonal	Banded to tridiagonal	General to Hessenberg
Givens	$\frac{4}{3}n^3$ $+n^2/2$ square roots	$4n^2(m - 1/m)$	$\frac{10}{3}n^3$
Householder	$\frac{2}{3}n^3$ $+n$ square roots	-	$\frac{5}{3}n^3$

Turning to the infinite strategy that takes a trial value and then evaluates the characteristic polynomial to

see if it is, indeed, zero there, we find that each evaluation takes the number of multiplications given in Table 3.2. The advantage of tridiagonal form is immediate and overwhelming, but Hessenberg form is not to be disparaged if tridiagonal is unavailable.

Table 3.2

Number of Multiplications to Evaluate, for One Trial λ, the Characteristic Polynomial Given as a Determinant

General determinant	$n^3/6$
Hessenberg determinant	$n^2/2$
Tridiagonal determinant	$2n$

If we contemplate using Newton's method to select our next trial λ, we need not only the value of the characteristic polynomial but also that of its derivative. Since each of these takes about 2n multiplications from tridiagonal form and since we might expect perhaps five cycles from Newton to liberate one root, we estimate 20n multiplications per root. Assuming we want all n roots, this leads to a total labor for a general symmetric matrix of

$$\frac{2}{3}n^2 + 20n^2.$$

Turning to the alternative infinite strategy of the LR family, we find that the efficiency of the several algorithms is governed largely by their ability to preserve specialized patterns of zeros. Thus LR with interchanges requires fewer multiplications than QR, but messes up banded symmetric

matrices and is thus not suitable for them. Table 3.3 gives the details, showing work load and the properties that are preserved. The last column gives the statistics on general matrices--emphasizing the unsuitability of this algorithmic family for such use.

Table 3.3

Number of Multiplications Per Iteration in the LR Family

	Symmetric		Nonsymmetric	
	Banded	Tridiagonal	Hessenberg	General
QR	BS $3nm^2$	TD $5n$	H $4n^2$	$2n^3$
Cholesky	BS nm^2	TD $2(n-1)$	-	-

Simple comparisons of arithmetic labor per iteration are useful for avoiding gross misapplications of our algorithms, but they should not be pushed too far. Laboratory experience shows that eigenvalues of Hessenberg matrices are usually produced to some standard accuracy with fewer QR iterations than with LR. Thus the QR labor is not four times as great, in spite of Table 3.3.

The problem of finding eigenvectors arises far less often than that of finding eigenvalues--and this is perhaps fortunate. For the eigenvalue methods that succeed best with large matrices do not directly produce the vectors, and

thus we must expend considerable additional computational energy if we would have our vectors too. As usual, the formal statement of the problem is simple enough. We have found one (or all) of the eigenvalues, λ_i, of the system

$$AX_i = \lambda_i X_i$$

and we must now solve the linear homogeneous system of algebraic equations

$$(A - \lambda_i I)X_i = 0$$

for the corresponding vector X_i. Because of the homogeneity we are free to pick the value of one component of X_i and to reserve one equation from the system, whereafter we have a well-posed problem in linear equation solving.

BIBLIOGRAPHY

BIBLIOGRAPHY

Acton, Forman S. Numerical Methods that Work. New York: Harper & Row Publishers, 1970.

Dahlquist, G., and A. Bjorck. Numerical Methods. Englewood, N.J.: Prentice Hall, 1974.

Householder, A. The Theory of Matrices in Numberical Analysis. n.p.: Blaisdell, 1964.

Kolman, Bernard. Introductory Linear Algebra with Applications. New York: Macmillan Publishing Co., Inc., 1980.

APPENDIX

```
C     Main Program:

      DOUBLE PRECISION A(10,10), T(10,10), H(10,10), P(10,10),
     -                 G(10,10), B(10,10), R(10,10), Q(10,10),
     -                 QT(10,10), AB(10,10), AT(10,10), X(10),
     -                 D, AD, TS, SIGN

C     A is the square matrix of order N which is read in.
C     The eigenvalues of A will be returned as the diagonal
C     elements of square array T of order N. The remaining
C     variables H, P, G, ..., SIGN are required by the
C     subroutines which are used. The main subroutine is
C     SUBROUTINE SYMEIG which handles all the logic involved.

C     Inputting the matrix A:

      READ (5,10) N
10    FORMAT (I3)

      DO 40 I = 1, N
         DO 30 J = 1, N
            READ (5,20) A(I,J)
20          FORMAT (1X, F10.5)
30       CONTINUE
40    CONTINUE

C     Printing the matrix A:

      DO 60 I = 1, N
         WRITE (6,50) (A(I,J), J = 1, N)
50       FORMAT (1X, '| ', 10F13.5, ' |')
60    CONTINUE

      CALL SYMIEG (A, N, T, X, H, P, Q, R, G, QT, AT, B, AB)

C     Printing the eigenvalues:

      DO 80 I = 1, N
         WRITE (6,70) I, T(I,I)
70       FORMAT (1X, 'EIGENVALUE #', I3, ' = ', F10.5)
80    CONTINUE

      CALL EXIT
      END
```

```
      SUBROUTINE SYMIEG(A, N, T, X, H, P, Q, R, G, QT, AT, B, AB)

C     A is a square array of order N (Real, Double Precision).
C     Eigenvalues of A are returned as the diagonal elements of
C     array T of order N.
C     The subroutines which are used are the following:
C               Primary Subroutines:   TRIHES (T, N, X, H, P)
C                                      QRSYMM (T, N, Q, R, G, QT, P)
C                                      MATPRD (A, B, AB, I1, I2, I3)
C
C               Secondary Subroutines: GIVENS (G, N, I, J, XC, YC)
C                                      HOUSER (X, N, L, H)
C                                      TRANSP (A, AT, I1, I2, I3)

      DOUBLE PRECISION A(N,N), T(N,N), X(N), H(N,N), P(N,N), Q(N,N),
     -                 R(N,N), G(N,N), QT(N,N), AT(N,N), AT(N,N),
     -                 B(N,N), AB(N,N), D, AD, TS, SIGMA, SIGN

      DO 10 I = 1, N
         DO 10 J = 1, N
            T(I,J) = A(I,J)
10    CONTINUE

C     Produce symmetric Tridiagonal from T

      CALL TRIHES(T, N, X, H, P)

C     Implement QR algorithm with Wilkinson shift SIGMA

      M = N - 1
      DO 20 I = 1, M
          J = N - I

C         Compare off-diagonal elements to "zero"

50        IF (DABS(T(J+1,J)) .LT. 1.D-12) GOTO 20

C         If T(J+1,J) is close to zero, goto 20 for "CONTINUE"

          D = (T(J,J) - T(J+1,J+1))/2.00

C         When D = 0 (Diagonal enteries equal) bypass to statement 90
C         to avoid division by zero.

          AD = DABS(D)
          IF (AD .LT. 1.D-12) GOTO 90
          TS = T(J+1,J)*T(J+1,J)
          SIGN = D / AD
          SIGMA = T(J+1,J+1) - SIGN*TS/(AD + DSQRT(D*D + TS))
          GOTO 60
90        SIGMA = T(J,J) - DABS(T(J+1,J))

C         Perform Shift:     T = T - SIGMA

60        DO 30 I = 1, N
```

```
            T(L,L) = T(L,L) + SIGMA
30        CONTINUE

C         Call for QR Decomposition

          CALL QRSYMM(T, N, Q, R, G, QT, P)

C         Set T = QR + SIGMA

          CALL MATPRD(R, Q, T, N, N, N)

          DO 40 L = 1, N
            T(L,L) = T(L,L) + SIGMA
40        CONTINUE

          GOTO 50

20    CONTINUE

      RETURN
      END

/INCLUDE TRIHES
/INCLUDE QRSYMM
/INCLUDE MATPRD
```

```
      SUBROUTINE TRIHES(A, N, X, H, P)

C     This subroutine puts an N by N symmetric matrix into
C     Tridiagonal, or an N by N matrix into Upper Hessenberg form.
C     The calling format is TRIHES(A, N, X, H, P).
C     This subroutine uses two other subroutines:
C                                          HOUSER(X, N, L, H)
C                                          MATPRD(A, B, AB, I1, I2, I3)
C     The Tridiagonal or Upper Hessenberg form will be returned as
C     the matrix A.

      DOUBLE PRECISION A(N,N), X(N), H(N,N), P(N,N)

      NM2 = N - 2
      DO 200 J = 1, NM2
         L = N - J

C     Next loop builds the single-dimentioned vector X which equals
C     the Jth column of A to be sent into subroutine HOUSER.

         DO 100 I = 1, N
            X(I) = A(I,J)
100      CONTINUE

C     The next call statement finds the HOUSEHOLDER of the last L
C     components of the X vector.

         CALL HOUSER (X, N, L, H)

C     The next two call statements multiply the matrix A on the left
C     and then on the right by the HOUSHOLDER found in the call
C     statement above. This zeroes out the element in the Jth column
C     below the main subdiagonal in a symmetric or nonsymmetric matrix
C     and also the elements in the Jth row to the right of the main
C     superdiagonal in a symmetric matrix.

         CALL MATPRD (H, A, P, N, N, N)
         CALL MATPRD (P, H, A, N, N, N)

200   CONTINUE

      RETURN
      END

/INCLUDE HOUSER
/INCLUDE MATPRD
```

```
      SUBROUTINE QRSYMM (T, N, Q, R, G, QT, P)

C     Subroutone QRSYMM finds Q, the transpose of GIVENS ROTATIONS,
C     and R, an upper tringular matrix, from a given tridiagonal
C     symmetric matrix T such that T = QR.
C
C     This subroutine uses subroutines GIVENS, MATPRD, TRANSP.

      DOUBLE PRECISION T(N,N), Q(N,N), QT(N,N), G(N,N), P(N,N),
     -                 R(N,N), XC, YC

C     Initializing QT, product of GIVENS ROTATIONS, as IDENTITY.

      DO 20 I = 1, N
         DO 10 J = 1, N
             QT(I,J) = 0.D0
10       CONTINUE
20    CONTINUE

      DO 30 I = 1, N
          QT(I,I) = 1.D0
30    CONTINUE

C     XC is the element to be used to zero the element YC in their
C     respective Ith and Jth position (using GIVENS ROTATIONS).

      M = N - 1
      DO 60 I = 1, M
            J = I + 1
            XC = T(I,I)
            YC = T(J,I)
               CALL GIVENS (G, N, I, J, XC, YC)
               CALL MATPRD (G, T, R, N, N, N)

C              R is stored into T for next multiplication.

               DO 40 L = 1, N
                   DO 40 K = 1, N
                      T(L,K) = R(L,K)
40             CONTINUE

               CALL MATPRD (G, QT, P, N, N, N)

C              P is stored back into QT, the product of GIVENS.

               DO 50 L = 1, N
                  DO 50 K = 1, N
                     QT(L,K) = P(L,K)
50             CONTINUE

60    CONTINUE

C     In solving R = QT*T for T, we must take the inverse of QT;
C     rather the transpose, since QT is orthogonal.
```

```
      CALL TRANSP (QT, Q, N, N)
      RETURN
      END

/INCLUDE GIVENS
/INCLUDE MATPRD
/INCLUDE TRANSP
```

```
      SUBROUTINE MATPRD(A, B, C, I1, I2, I3)

C     A and B are the two matrices to be multiplied.
C           A is I1 by I2,
C           B is I2 by I3,
C           C is the product.

      DOUBLE PRECISION A(I1, I2), B(I2, I3), C(I1, I3)

      DO 300 I = 1, I1
         DO 200 J = 1, I3
            C(I,J) = 0.
            DO 100 K = 1, I2

               C(I,J) = C(I,J) + A(I,K)*B(K,J)

100         CONTINUE
200      CONTINUE
300   CONTINUE

      RETURN
      END
```

```
      SUBROUTINE GIVENS(G, N, I, J, X, Y)

C     This subroutine returns a GIVENS ROTATION which uses the Ith
C     component X to zero out the Jth component Y.
C     G is an N by N array.
C     If Y = 0, then G is returned as Identity.
C     If X = 0, then G is returned as the permutation matrix which
C     premutes rows I and J.

      DOUBLE PRECISION G(N,N), X, Y, XYLEN

C     Set G as Identity.

      DO 10 L = 1, N
         DO 10 K = 1, N
            G(L,K) = 0.D0
10    CONTINUE

      DO 20 L = 1, N
         G(L,L) = 1.D0
20    CONTINUE

      IF (DABS(Y) .LT. 1.D-12) GOTO 100
      IF (DABS(X) .GT. 1.D-12) GOTO 80

C     Interchange rows I and J of G when X equal 0 and Y not equal 0.

      G(I,I) = 0.D0
      G(J,J) = 0.D0
      G(J,I) = 1.D0
      G(I,J) = 1.D0
      GOTO 100

C     Compute XYLEN, length of vector (X,Y).

80    XYLEN = DSQRT(X*X + Y*Y)

C     Test for J < I.

      IF (J .LT. I) GOTO 90
      G(I,I) = X /  XYLEN
      G(J,J) = G(I,I)
      G(I,J) = Y / XYLEN
      G(J,I) = -G(I,J)
      GOTO 100

90    G(I,J) = Y / XYLEN
      G(J,J) = G(I,I)
      G(J,I) = X / XYLEN
      G(I,J) = -G(J,I)

100   RETURN
      END
```

```
      SUBROUTINE HOUSER(X, N, L, H)

C     Subroutine Houser is called to compute Householder Reflector H
C     of order N.
C     L is a number which designates the last l components of the
C     single subscripted variable X of length N on which the
C     Householder Reflector H is to be determined.

      DOUBLE PRECISION X(N), H(N,N), XLEN, COFF

C     Initialize H to b Identity.

      DO 10 I = 1, N
        DO 10 J = 1, N
           H(I,J) = 0.DO
10    CONTINUE

      DO 20 I = 1, N
         H(I,I) = 1.DO
20    CONTINUE

C     Compute starting value of component index for X, namely the
C     index at which true vector begins.

      NSTART  = N - L + 1

C     Compute length of vector X from Nstart to N

      XLEN = 0.DO

      DO 30 I = NSTART, N
         XLEN = XLEN + X(I)*X(I)
30    CONTINUE

C           Test Xlen for zero. If zero, send H back as Identity

      IF (XLEN .LT. 1.D-12) GOTO 100
      XLEN = DSQRT(XLEN)

C     Reset component Nstart of X to plus or minus Xlen

      IF (X(NSTART) .GE. 0.DO) GOTO 40

      XLEN = -XLEN
40    X(NSTART) =  X(NSTART) + XLEN

C     Now compute coefficient  2 / (UT * U)

      COEF = 0.DO
      DO 50 I = NSTART, N
         COEF = COEF + X(I)*X(I)
50    CONTINUE
      COEF = 2.DO / COEF

C     Now compute final enteries of Reflector H
```

```
      DO 70 I = NSTART, N
         DO 60 = NSTART, N
            H(I,J) =  H(I,J) - COEF*X(I)*X(J)
 60      CONTINUE
 70   CONTINUE

100   RETURN
      END
```

```
      SUBROUTINE TRANSP(A, AT, IR, IC)

C     This subroutine will find the Transpose of Matrix A with the
C     number of rows equal to IR and the number of columns equal
C     to IC. The transpose will return as matrix AT.

      DOUBLE PRECISION A(IR, IC), AT(IC, IR)

      DO 20 J = 1, IC
         DO 10 K = 1, IR
            AT(J,K) = A(K,J)
10       CONTINUE
20    CONTINUE

      RETURN
      END
```