AN ABSTRACT OF THE THESIS OF

MICHAEL GARY GUNN (Name of student) for the MASTER OF SCIENCE (Degree)

in Mathematics (Major) presented on May 5, 1971 (Date)

Title: A STUDY OF THE EFFECT OF THE CHOICE OF PIVOT ELEMENTS ON ROUND-OFF ERRORS IN GAUSS ELIMINATION

Abstract approved: Redacted for privacy

Dr. Harry Goheen

It is established folklore in numerical analysis, for solution of a square system of linear equations by Gauss elimination, that the standard method of choice of pivot elements is to search the entire suppressed matrix for the element largest in absolute value. However, it was felt by the author, that use of this method would not minimize round-off errors when operating with unstable matrices. By the use of FORTRAN programming on the equation $Cx = b$, where C is the inverse of the Hilbert matrix, for $N = 5, 6$, and 7, and b is a column matrix of N rows with each element equal 1, it is shown the use of the standard method of search does not minimize round-off errors incurred, and an alternative method is developed.

A Study of the Effect of the Choice of Pivot Elements on Round-Off Errors in Gauss Elimination

by

Michael Gary Gunn

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

June 1971

APPROVED:

Redacted for privacy

Professor of Mathematics
in charge of major

Redacted for privacy

Acting Chairman of Department of Mathematics

Redacted for privacy

Dean of Graduate School

Date thesis is presented May 5, 1971

Typed by Opal Grossnicklaus for Michael Gary Gunn

ACKNOWLEDGEMENTS

I would like to thank Professor H. E. Goheen for his help and encouragement in the development of this thesis.

TABLE OF CONTENTS

Chapter		
I.	INTRODUCTION	1
II.	DESCRIPTION OF THE METHODS OF CHOICE	4
III.	DESCRIPTION OF THE PROGRAM	9
IV.	CONCLUSIONS	22
	BIBLIOGRAPHY	26
	APPENDICES	
	APPENDIX 1 Flow Charts	27
	APPENDIX 2 Output for Solutions	32

A STUDY OF THE EFFECT OF THE CHOICE OF PIVOT ELEMENTS ON ROUND-OFF ERRORS IN GAUSS ELIMINATION

I. INTRODUCTION

This paper reports on experiments with the choice of pivot elements in Gauss elimination, the basic method used to solve a square system of linear equations, in order to determine which such choice minimizes round-off errors incurred during solution of the equation $Cx = b$, C being the inverse of a Hilbert matrix $(1/I+J-1)$, $I = 1, 2, \ldots, N$, $J = 1, 2, \ldots, N$, and b a column matrix of N rows with each element 1, and shows, for the solution of the matrix equation $Cx = b$, with C of size 5x5, 6x6, and 7x7, neither of the standard methods of choice of pivot elements, using the diagonal elements and searching the entire suppressed matrix for the element largest in absolute value, minimize round-off errors. The suppressed matrix is defined as the $(N-I)x(N-I)$ submatrix remaining after each stage of elimination, i.e., after the first stage the suppressed matrix is of size $(N-1)x(N-1)$, with c'_{22} the first element of the main diagonal of the submatrix.

In order to perform the algorithm, certain elements, called pivots, are chosen, so as to execute the necessary row and column transformations. Each pivot element is used to eliminate a single variable from all the equations, except the one in which the pivot element occurs, i.e., the first pivot would eliminate x_1 in the first

elimination stage, the second pivot, x_2 in the second stage, ..., the $(N-1)^{st}$ pivot, x_{N-1} in the $(N-1)^{st}$ stage. There are (N-1) stages to the elimination process. Once an element has been used as a pivotal element in order to eliminate one variable, it is left unaltered in the future stages of the elimination process.

Because a computer is capable of carrying only a finite number of digits, the choice of pivot elements is of great importance, so as to minimize round-off errors incurred during computation. A standard procedure is to use the diagonal elements of the matrix as pivot elements. However, if such an element is near 0, an arithmetic overflow will occur during computation. Thus, this practice is certainly inadequate.

We study several alternatives, including the one advocated in standard texts, e.g. Wilkinson (5). This method searches the entire suppressed matrix for the element largest in absolute value. However, use of this method, when operating with unstable matrices (whose determinant is near 0), will result in large round-off errors incurred during elimination, for the following reason. The determinant of a matrix can be computed as the product of its pivot elements. Consider as the following example an unstable system of 20 equations, in the matrix equation form Ax = b, where the determinant of A equals 1. Suppose that each of the 19 pivot elements chosen is of magnitude 10^3, their product being of magnitude 10^{57}. In order

for the determinant to be near 1, $c'_{20,20}$ (element which occurs in 20^{th} row and 20^{th} column) must be of magnitude 10^{-57}. However, operating with a number of this magnitude will result in an arithmetic overflow during elimination. Since the standard method of choice, searching the entire suppressed matrix for the element largest in absolute value, seems to be inadequate when operating with unstable matrices, which method of choice of pivot elements will minimize round-off errors incurred during performance of the algorithm?

This project experiments, by the use of FORTRAN programming, on the inverses of the Hilbert matrices, for N = 5, 6, and 7, with various methods of choice of pivot elements, which will be described in other sections of this paper. One such method, searching the entire suppressed matrix for the non-zero element whose absolute value is nearest the arithmetic average of the absolute values of the elements of the suppressed matrix, minimizes round-off errors incurred in the solution of the 5x5, 6x6, and 7x7 cases discussed, the reason being that instead of round-off errors being increased by propagation, which occurs by choosing extremely large or small pivot elements, the errors are instead "averaged" by forcing the pivot elements to be nearest the arithmetic average of the absolute values of the elements of the suppressed matrix.

II. DESCRIPTION OF THE METHODS OF CHOICE

Method 1:

The common, and most obvious, method of choice is to use the diagonal element, if it does not equal 0, of the column whose number is that of the current stage of elimination, as a pivot element (c_{11} is used during the first elimination stage, c'_{22} during the second stage, ...). If the diagonal element equals 0, the first non-zero element (if it exists) below the main diagonal is used as a pivot element. Very little element searching is performed by this method; therefore, it is quite possible that extremely large or small elements can be chosen as pivots, causing large round-off errors to be incurred.

Method 2:

A method advocated in standard texts, e.g. Scarborough (4), searches the column (excluding those elements above the main diagonal), whose number is that of the current stage of elimination, for the element largest in absolute value. Although this technique is a standard method of search, it is shown to be inadequate in our test cases, yielding round-off errors much larger than those of method 6, when operating on the inverses of Hilbert matrices, for N = 5, 6, and 7.

Method 3:

Another common method advocated in standard texts, e.g. Wilkinson (5), searches the entire suppressed matrix for the element largest in absolute value. The results of this method were comparable to those of method 2.

Method 4:

An example of a poor method of choice is to search the column (excluding those elements above the main diagonal), whose number is that of the current stage of elimination, for the non-zero element smallest in absolute value. By choosing elements which are extremely small in absolute value, round-off errors are increased by propagation during performance of the algorithm. The reader is referred to appendix 2 for details.

Instead of choosing pivot elements extremely large in absolute value, methods 2 and 3, which increasingly propagates round-off errors, it was felt that the pivot elements should be more temperate, by choosing the pivots such that Min(ABS c'_{IJ}) < Pivot element < Max (ABS c'_{IJ}), methods 5-9, in order to "average", rather than increasingly propagate, round-off errors.

Method 5:

This method searches the column (excluding those elements above the main diagonal), whose number is that of the current stage

of elimination, for the non-zero element whose absolute value is nearest the arithmetic average of the absolute values of the elements of the matrix. This method, as expected, yielded poor results in the test cases. The results are summarized in appendix 2.

Method 6:

This method searches the entire (N-I) x (N-I) suppressed matrix (I denoting the current stage of elimination) for the non-zero element whose absolute value is nearest the arithmetic average of the absolute values of the elements of the suppressed matrix. Use of this method yielded the smallest round-off errors for the test cases.

Method 7:

This method searches the column (excluding those elements above the main diagonal), whose number is that of the current stage of elimination, for the non-zero element whose absolute value is nearest the N^{th} root of the absolute value of the determinant of the NxN matrix. An approximation of the determinant is calculated as the product of the pivot elements used in Gauss elimination (diagonal elements are used as pivots), and the N^{th} root is calculated as the absolute value of the determinant raised to the power (1/N)--N is the number of rows and columns in the matrix. Since this method requires a double performance of the elimination, it is not advocated

unless a significant decrease in round-off errors is obtained. This was not the case in the test experiments (see appendix 2).

Method 8:

This method searches the column (excluding those elements above the main diagonal), whose number is that of the current stage of elimination, for the non-zero element whose absolute value is nearest the $(N-I)^{th}$ root of the absolute value of the determinant of the matrix. During the first stage of elimination, column 1 is searched for the non-zero element whose absolute value is nearest the N^{th} root of the absolute value of the determinant of the NxN matrix (previously calculated for use in method 7). In subsequent stages of elimination, the N^{th} root of the determinant is divided by the pivot element used in the preceding stage of elimination and the result stored in the storage location of the N^{th} root of the determinant, in order to calculate an approximation of the $(N-I)^{th}$ root of the determinant. The results of method 8 were similar to those of method 7.

Method 9:

This method searches the entire suppressed matrix for the non-zero element whose absolute value is nearest one-half of (Max (ABS c'_{IJ}) + Min (ABS c'_{IJ})), c'_{IJ} being the elements of the suppressed matrix.

Method 10:

This method alternates between method 1, the "diagonal" method, and method 5, the "current column arithmetic average method". It was thought by alternating between two methods, round-off errors would be minimized.

III. DESCRIPTION OF THE PROGRAM

This FORTRAN program consists of a main program, primarily used for input and output, and various subprograms, which perform the algorithm.

The input, using a READ statement, consists of N (the size of the matrix C in the matrix equation $Cx = b$), the inverse, C, of the NxN Hilbert matrix, and the N actual values, ACTU, of the unknown column matrix, x. Each element of the column matrix, b, calculated in the program, is chosen to be 1, because it is then easier to calculate, by hand, the actual values of x. Since the matrix equation is of the form $Cx = b$, then $x = C^{-1}b$. C is the inverse of the NxN Hilbert matrix; therefore, the inverse of C is the original NxN Hilbert matrix. The actual value of each element of x is the sum of the corresponding row elements of the NxN Hilbert matrix. Because of the CDC 3300 48 bit representation of a floating point number, of which 12 bits are the exponent, and a 36 bit mantissa, the inverse, consisting of integers, of the NxN Hilbert matrix, rather than the Hilbert matrix, is used as input, so as to avoid any input error. The Hilbert matrix contains elements, such as 1/3, 1/6, 1/7, ..., which cannot be exactly represented as floating point numbers. If used as input, these fractions would contain an input error in the 10th or 11th decimal place, thus causing, because of the extremely large

condition number of both the Hilbert matrix and its inverse, a probable output error in the 3^{rd} or 4^{th} decimal place of the final answer after performance of the algorithm.

The average of the absolute values of the elements of the NxN matrix is computed by the use of a DO loop, which totals the absolute values of the elements, then divides this sum by the number of elements.

An approximation of the absolute value of the determinant is calculated as the product of the pivot elements used during Gauss elimination. The diagonal elements are used as pivot elements, because this method of choice minimizes computer execution time involved. The N^{th} root of the absolute value of the determinant is then calculated as the absolute value of the determinant raised to the power (1/N). If the matrix is singular (determinant is 0), a flag is returned to the main program, so as to print an appropriate message and terminate the algorithm.

Because of the 36 bit mantissa representation of a floating point number, only 10 or 11 decimal place accuracy, it is necessary to define an allowable tolerance, EPSILON, equal $10^{-8} \cdot ||C||_1$ (maximum absolute column sum), so that if the absolute value of a pivot element is less than EPSILON, too large a round-off error will occur, and a message to this effect will be printed. The $||C||_1 \cdot 10^{-8}$, rather than 10^{-8}, is used as a tolerance, so as to allow for performance of the

algorithm on a matrix whose elements are of varying magnitude.

In order to perform the elimination, subroutine GAUSS is then called. The square matrix, C, and the column matrix, b, must be copied to dummy matrices, A and d, respectively, so as not to destroy their contents each time the elimination is performed. A permutation array, called KR1, is initially defined to be (1 2 3...N). If a pivot element occurs in the J^{th} column, but J does not equal I (current stage of elimination), the I^{th} and J^{th} columns of the matrix C and the I^{th} and J^{th} permutation marks of KR1 must be exchanged, in effect exchanging x(I) and x(J), elements of the unknown column matrix, x.

For the inverse of an NxN Hilbert matrix, employing the Q^{th} method of choice of pivot elements (Q=1, 2, . . . , number of methods of choice), Gauss elimination is performed (N-1) times. If, by use of the Q^{th} method, too large a round-off error results, signified by a pivot element less than EPSILON, a message to this effect is printed, and use of the Q^{th} method is terminated. If (Q+1) is less than the number of methods of choice, the $(Q+1)^{st}$ method is then employed to perform the algorithm on the original, unaltered, dummy matrix A. If (Q+1) is not less than the number of methods of choice, the algorithm terminates.

During performance of the algorithm, the index I is used to denote the current stage of elimination. By use of a computed GO TO

statement, with reference to a counter, IKL, the method of choice of pivot elements is determined. During execution of the various methods of choice, parameters ICHECK and ISWITCH are initially set equal to 0 and 1, respectively. If any row or column of the matrix, C, has no non-zero element, the matrix is singular, and a value ICHECK equal 1 is returned to the elimination subroutine. If the absolute value of a pivot element is less than EPSILON, the value of ISWITCH is changed (the new value depends on the method of choice employed), before returning the parameters, i.e., if the pivot element obtained by use of method 1 is less than EPSILON, then ISWITCH equals 2, method 2, then ISWITCH equals 3, ... After each pivot element has been chosen by the appropriate method, the values of the pivot element and both the row and column indices in which the pivot element occurs are returned as parameters.

Method 1, subroutine DIAG, uses the diagonal element (if it does not equal 0) of the column, whose number is that of the current stage of elimination, as a pivot element. A row index, K, is initially set equal to I. If A(I, I) does not equal 0, then the pivot element, PIV, equals A(I, I). However, if A(I, I) equals 0, by means of a DO loop (K=I+1, N), PIV is set equal to A(K, I), K corresponding to the row occurrence of the first non-zero element (if it exists) below the main diagonal.

Method 2, subroutine LARGE, searches the column (excluding

those elements above the main diagonal), whose number is that of the current stage of elimination, for the element largest in absolute value. A row index, MAX, is initially set equal to I, and if A(MAX, I) does not equal 0, PIV is set equal to A(MAX, I). If A(MAX, I) equals 0, PIV is set equal to the first non-zero element below the main diagonal, at which time MAX contains the row index of the occurrence of PIV. By means of a DO loop (J=I+1, N), the ABS(A(J, I)), if A(J, I) is not 0, is compared to ABS(A(MAX, I)). If ABS(A(J, I)) is less than or equal to ABS(A(MAX, I)), the value of PIV remains unchanged, and the loop continues. However, if ABS(A(J, I)) is greater than ABS(A(MAX, I)), the value of J must be stored in MAX and PIV set equal to A(J, I), before continuing the loop. After the element largest in absolute value has been determined, MAX contains the row index and I the column index in which the pivot element occurs.

Method 3, subroutine LARGSM, searches the entire suppressed matrix for the element largest in absolute value. A row index, K, and a column index, L, are initially assigned equal to I, and if A(K, L) does not equal 0, PIV is set equal to A(K, L). If A(K, L) equals 0, PIV is set equal to the first occurrence (columnwise) of a non-zero element, at which time K contains the row index and L the column index of the occurrence of the pivot element. By means of two nested DO loops and an IF statement, the entire suppressed matrix is searched for the element largest in absolute value.

Method 4, subroutine SMALL, searches the column (excluding those elements above the main diagonal), whose number is that of the current stage of elimination, for the non-zero element smallest in absolute value. A row index, MIN, is initially set equal to I. If A(MIN, I) does not equal 0, then PIV equals A(MIN, I); otherwise, PIV is set equal to the first occurrence of a non-zero element below the main diagonal, at which time MIN contains the row index of the occurrence of the pivot element. By means of a DO loop (J=MIN+1, N), ABS(A(J, I)), if A(J, I) does not equal 0, is compared to ABS(A(MIN,I)). If ABS(A(J, I)) is greater than or equal to ABS(A(MIN, I)), the value of PIV remains unchanged, and the loop continues. However, if ABS(A(J, I)) is less than ABS(A(MIN, I)), the value of J must be stored in MIN and PIV set equal to A(J, I), before continuing the loop. After the non-zero element smallest in absolute value has been determined, MIN contains the row index and I the column index of the occurrence of the pivot element.

Method 5, subroutine MEAN, searches the column (excluding those elements above the main diagonal), whose number is that of the current stage of elimination, for the non-zero element whose absolute value is nearest the arithmetic average of the absolute values of the elements of the matrix. Two row indices, L and KT, are initially assigned equal to I. If A(KT, I) equals 0, then both L and KT are set equal to the row index of the occurrence of the first non-zero

element below the main diagonal (if it exists); otherwise, both L and KT are left unaltered. DIFF1 is then set equal to ABS(ABS(A(L, I))-XMEAN), XMEAN being the arithmetic average of the elements of the matrix. By use of a DO loop (J=L+1, N) and several IF statements, DIFF2 is calculated as ABS(ABS(A(J, I))-XMEAN), A(J, I) not equal 0. If A(J, I) equals 0, DIFF2 is not calculated; J is incremented, and the loop continues. Otherwise, DIFF1 is then compared to DIFF2. If DIFF1 is less than or equal to DIFF2, DIFF1 remains unchanged, and the loop continues. However, if DIFF1 is greater than DIFF2, signifying ABS(A(J, I)) is nearer XMEAN than ABS(A(L, I)), the value of DIFF2 must be stored in DIFF1 and KT set equal to J, before continuing the loop. After execution of the search, PIV equals A(KT, I), signifying the absolute value of the element, which occurs in row KT and column I, is nearest the arithmetic average of the absolute values of the elements of the matrix.

Method 6, subroutine MEANER, searches the entire suppressed matrix for the non-zero element whose absolute value is nearest the arithmetic average of the absolute values of the elements of the suppressed matrix. A flag, IDIG, is set equal to 1, signifying the entire suppressed matrix is searched, instead of only searching the current column. Two row indices, K and KP, and two column indices, L and LP, are initially set equal to I. By means of two nested

DO loops, the first occurrence of a non-zero element of A is determined (search conducted columnwise), at which time K and KP denote the row index and L and LP the column index of this occurrence. DIFF1 is then set equal to ABS(ABS(A(KP, LP))-CMEAN); CMEAN is the arithmetic average, calculated after each stage of elimination, of the absolute values of the elements of the (N-I)x(N-I) suppressed matrix. By means of two DO loops (J1=KP, N J2=LP, N), DIFF2 is calculated as ABS(ABS(A(J2, J1))-CMEAN), A(J2, J1) not equal to 0. If A(J2, J1) equals 0, DIFF2 is not calculated; the proper index is incremented, and the loop continues. Otherwise, DIFF1 is compared to DIFF2. If DIFF1 is less than or equal to DIFF2, DIFF1 remains unchanged, and the loop continues. However, if DIFF1 is greater than DIFF2, signifying ABS(A(J2, J1)) is nearer CMEAN than ABS(A(KP, LP)), the value of DIFF2 must be stored in DIFF1, K set equal to J2 and L set equal to J1 before continuing the loops. After execution of the search, PIV equals A(K, L), signifying the absolute value of the element occurring in row K and column L is nearest the arithmetic average of the absolute values of the suppressed matrix.

Method 7, subroutine CLOSE, searches the column (excluding those elements above the main diagonal), whose number is that of the current stage of elimination, for the non-zero element whose absolute value is nearest the N^{th} root of the absolute value of the

determinant of the NxN matrix. Two row indices, L and KT, are initially assigned equal to I. If A(KT, I) equals 0, then both KT and L are set equal to the row index of the occurrence of the first non-zero element below the main diagonal; otherwise, both KT and L remain unaltered. DIFF1 is then set equal to ABS(ABS(A(KT, I))-SROOT), SROOT being the N^{th} root of the absolute value of the determinant of the NxN matrix. By use of a DO loop (J=KT+1, N), DIFF2 is calculated as ABS(ABS(A(J, I))-SROOT), A(J, I) not equal 0. If A(J, I) equals 0, DIFF2 is not calculated; J is incremented, and the loop continues. Otherwise, DIFF1 is compared to DIFF2. If DIFF1 is less than or equal to DIFF2, DIFF1 remains unchanged, and the loop continues. However, if DIFF1 is greater than DIFF2, signifying ABS(A(J, I)) is nearer SROOT than ABS(A(KT, I)), the value of DIFF2 must be stored in DIFF1 and L set equal to J, before continuing the loop. After execution of the search, PIV equals A(L, I), signifying the absolute value of the element, which occurs in row L and column I, is nearest the N^{th} root of the absolute value of the determinant of the NxN matrix.

Method 8, subroutine CLOSER, searches the column (excluding those elements above the main diagonal), whose number is that of the current stage of elimination, for the non-zero element whose absolute value is nearest the $(N-I)^{th}$ root of the absolute value of the determinant of the matrix. This method calls CLOSE (method 7);

however, instead of the N^{th} root of the absolute value of the determinant used as a parameter, the $(N-I)^{th}$ root of the absolute value of the determinant is used. During the first stage of elimination, column 1 is searched for the non-zero element whose absolute value is nearest the N^{th} root of the absolute value of the determinant. In subsequent stages of elimination, the current column is searched for the non-zero element whose absolute value is nearest the $(N-I)^{th}$ root, calculated as the N^{th} root of the absolute value of the determinant, divided by the pivot element used in the preceding stage.

Method 9 searches the entire suppressed matrix for the non-zero element whose absolute value is nearest one-half of $(ABS(Max\ c'_{IJ}) + ABS(Min\ c'_{IJ}))$. A subroutine, LARGSM, is used to determine the largest and smallest elements in absolute value, and a variable, SPRED, is then set equal to one-half the sum of these elements. Subroutine MEANER, used in method 6, is called, using SPRED, instead of CMEAN, as a parameter, in order to determine the non-zero element whose absolute value is nearest one-half of $(ABS(Max\ c'_{IJ}) + ABS(Min\ c'_{IJ}))$.

Method 10 alternates between the use of method 1, subroutine DIAG, and method 5, subroutine MEAN. Employing the characteristic of truncation when setting a floating point number equal to a fixed point number, determines if a counter, ICTR, is even or odd. If ICTR is odd, method 1 is used; if even, method 5 is used.

After a pivot element, determined by the Q^{th} method of choice, has been returned as a parameter to the elimination subroutine, it must be compared to EPSILON, the allowable tolerance. If the pivot element is less than EPSILON, a flag is returned to the main program, signifying too large a round-off error will occur. A message to this effect is printed, and use of the Q^{th} method is terminated. If Q equals the number of methods of choice, the program terminates; otherwise, the $(Q+1)^{st}$ method of choice is employed to perform the algorithm on the original, unaltered, dummy matrix, A.

If the pivot element is greater than or equal to EPSILON, the parameter, K, denoting the row index of the occurrence of the pivot element, must be compared to I, the current stage of elimination. If K equals I, the row elements of A are left unaltered; however, if K does not equal I, signifying the pivot element does not occur in the I^{th} row, the I^{th} and K^{th} rows of A and the I^{th} and K^{th} elements of D are exchanged, performed by use of a DO loop and a temporary storage location.

The parameter, L, denoting the column index of the occurrence of the pivot element, must be compared to I. If L equals I, the column elements remain unaltered; however, if L does not equal I, signifying the pivot element does not occur in the I^{th} column, the I^{th} and the L^{th} columns of A are exchanged. The I^{th} and L^{th} permutation marks of the permutation array, KR1, must also be exchanged,

which in effect exchanges the I^{th} and L^{th} elements of the unknown column matrix, x.

Employing the selected Q^{th} method of choice, the elimination is performed (N-1) times, and the matrix A is reduced to upper triangular form.

The elements of the unknown column matrix, x, can now be calculated by a process called back substitution. The last equation determines x_N, which is then substituted in the next-last equation to determine x_{N-1}, and so on.

If no column exchanges were performed during execution of the algorithm, denoted by IDIG equal 0, the values of the column matrix, x, are left unaltered and returned to the main program. However, if column exchanges were performed, signified by IDIG equal 1, the permuted elements of x, in the computer's memory, are not in the correct order x(1) . . . x(N). By simultaneous arrangement of both the permutation array, KR1, which signifies what column exchanges were performed, and the column matrix, x, in the form KR1(1) ... KR1(N) and x(1) ... x(N), respectively, the values of x will be returned to the main program in the correct order, i.e., if the initial permutation array was 1 2 3 ... N, and the second and third columns of A were exchanged, the array is then 1 3 2 ... N, signifying the second and third columns of A were exchanged.

The relative error for each computed value of x is then

calculated in the form ERROR(J)=ABS((ACTU(J)-X(J))/ACTU(J))(J=1, N). Computing the relative error, rather than calculating the error as ABS(ACTU(J)-X(J)), gives a better indication of the actual computational error involved. As an example, consider a computed value of x equal to 10^{-6} and an actual value equal to 10^{-5}. The ABS(ACTU-X) equals $9x10^{-5}$, a very small number. However, the relative error equals 9.1, which is an indication a large error was incurred during computation.

The actual and computed values of x and the relative error for each computed value are then printed in table form (refer to appendix 2). If IKL, the counter corresponding to the method of choice employed, does not equal the number of methods of choice, the $(IKL+1)^{st}$ method is used to perform the algorithm on the original, unaltered, dummy matrix, A. However, if IKL equals the number of choices employed, the program terminates.

IV. CONCLUSIONS

Experiments with the choice of pivot elements in Gauss elimination were performed, in order to determine which such choice minimizes round-off errors incurred during solution of the equation $Cx = b$, C being the inverse of a Hilbert matrix, for N = 5, 6, and 7, and b a column matrix of N rows with each element equal 1. Hilbert matrices of size less than 5 were not employed, because it was felt they were not unstable enough to provide conclusive results.

The results for the solution of the matrix equation for the inverse of the 5x5 Hilbert matrix conclusively show that the common method of choice of pivot elements, using the diagonal elements, and both standard methods of search, searching the column (excluding those elements above the main diagonal), whose number is that of the current stage of elimination, for the element largest in absolute value, and searching the entire suppressed matrix for the element largest in absolute value, do not minimize round-off errors.

Employment of method 6, in the case N = 5, searching the entire (N-I)x(N-I) suppressed matrix for the non-zero element whose absolute value is nearest the arithmetic average of the absolute values of the elements of the suppressed matrix, yields relative errors which are of magnitude 10^3 less than the errors obtained by

use of the diagonal method and both of the largest in absolute value methods. With the exception of alternating between the diagonal method and the current column arithmetic average method, which yielded round-off errors comparable of those of method 6, the errors obtained by use of the other methods were similar to those of the diagonal and largest in absolute value methods.

The smallest relative error obtained by the arithmetic average method is 2.5×10^{-11}, and the largest is 6.5×10^{-11}, while the smallest error of the diagonal method is 1.9×10^{-8}, and the largest is 3×10^{-8}. The current column largest in absolute value method yields a smallest relative error of 1.7×10^{-8} and a largest error of 2.6×10^{-8}, and the smallest and largest relative errors obtained by searching the entire suppressed matrix for the element largest in absolute value are 6×10^{-9} and 9×10^{-8}, respectively--slightly better results than those obtained by use of the current column largest in absolute value method.

Of the methods used in the solution of the 5x5 case, searching the column (excluding those elements above the main diagonal), whose number is that of the current stage of elimination, for the non-zero element smallest in absolute value, yields the largest relative errors. This, of course, was expected. The reader is referred to appendix 2 for details of the results.

Because several of the methods employed in the solution of

the 5x5 case yielded large relative errors, only four methods, alternating between the diagonal and current column arithmetic average methods, the current column largest in absolute value, the entire matrix largest in absolute value, and the entire matrix arithmetic average, were employed in the solution of both the 6x6 and 7x7 cases.

Use of method 6, in both the cases N = 6, 7, searching the entire suppressed matrix for the non-zero element whose absolute value is nearest the arithmetic average of the absolute values of the elements of the suppressed matrix, yields relative errors which are of magnitude 10^2 less than the errors obtained by employment of the other methods.

For the case N = 6, the smallest relative error obtained by the arithmetic average method is 8.3×10^{-9}, and the largest is 5.4×10^{-8}, while the smallest error obtained by the "alternating" method is 2.3×10^{-7}, and the largest is 4.7×10^{-6}. The current column largest in absolute value method yields a smallest relative error of 1.5×10^{-7} and a largest error of 2.6×10^{-6}, and the smallest and largest relative errors obtained by searching the entire suppressed matrix for the element largest in absolute value are 1×10^{-7} and 3.1×10^{-6}, respectively (refer to appendix 2).

As with the cases N =5, 6, the results of the 7x7 test case show, that of the methods of choice employed, use of method 6 minimizes

relative errors incurred. The reader is referred to appendix 2 for details.

Since by use of all the methods of choice employed in the solution of the 7x7 case, element $a'_{7,7}$, obtained after performance of the algorithm, is of magnitude less than EPSILON, it appears that none of the methods of choice employed will be able to solve the case N = 8, because certain pivot elements would be less than EPSILON, thus causing too large a round-off error.

Viewing the results of the study of the effect of the choice of pivot elements on round-off errors in Gauss elimination, it appears, that of the methods of choice used in the 5x5, 6x6, and 7x7 cases discussed, searching the entire (N-I) x (N-I) suppressed matrix for the non-zero element whose absolute value is nearest the arithmetic average of the elements of the suppressed matrix, yields the smallest relative errors. Although employment of this method requires more CPU time than the diagonal method or either of the largest in absolute value methods, it seems that one would be willing to sacrifice several microseconds of CPU time for much smaller relative errors incurred during performance of the algorithm.

BIBLIOGRAPHY

1. Hartree, D. R. Numerical Analysis. London, Oxford University Press, 1958. p. 166-171.

2. Issacson, Eugene and Keller, Herbert Bishop. Analysis of Numerical Methods. New York, Wiley and Sons, 1966. p. 50-51.

3. Kunz, Kaiser S. Numerical Analysis. New York, McGraw-Hill, 1957. p. 220-221.

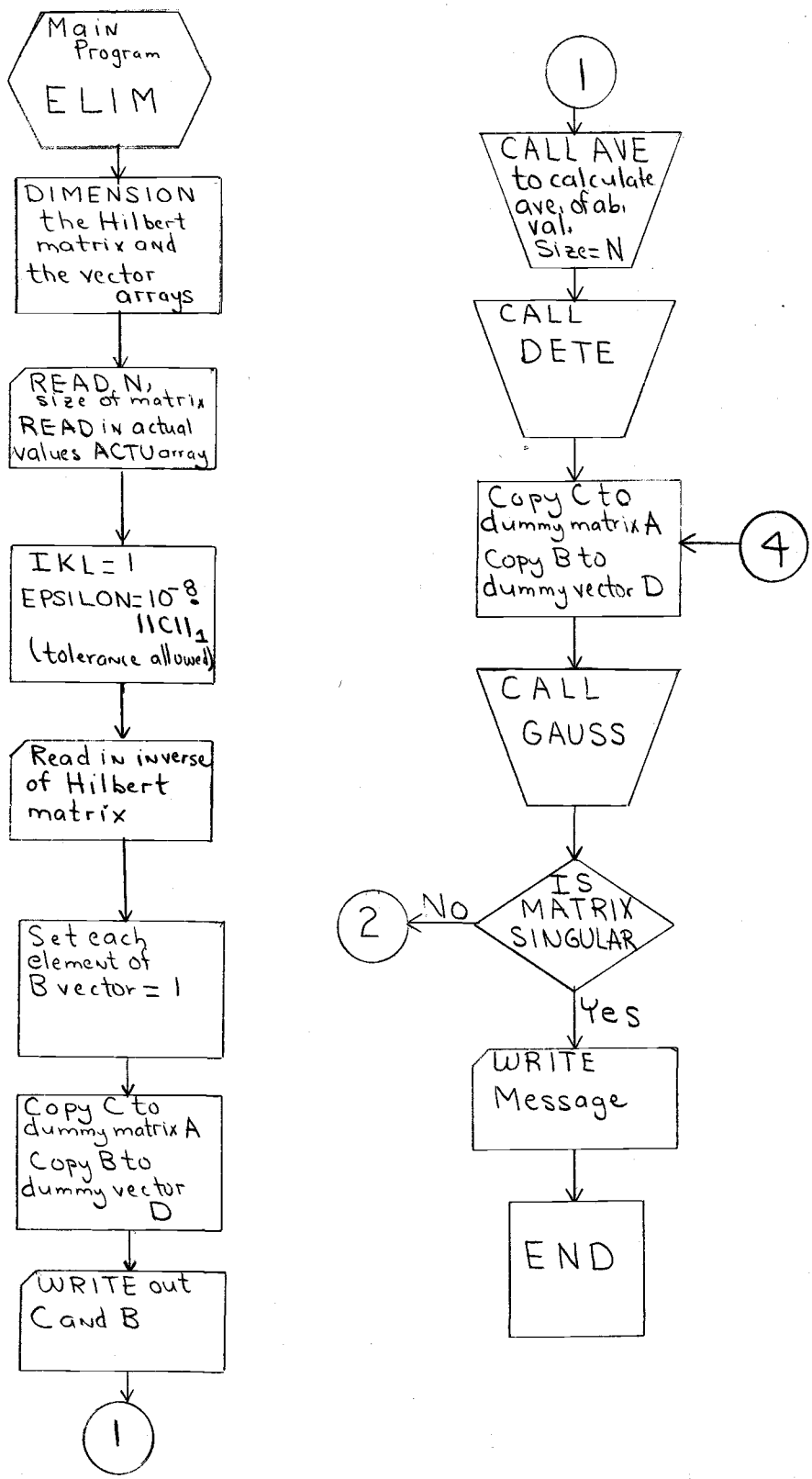
4. Scarborough, James B. Numerical Mathematical Analysis. Baltimore, The Johns Hopkins Press, 1930. p. 272-277.

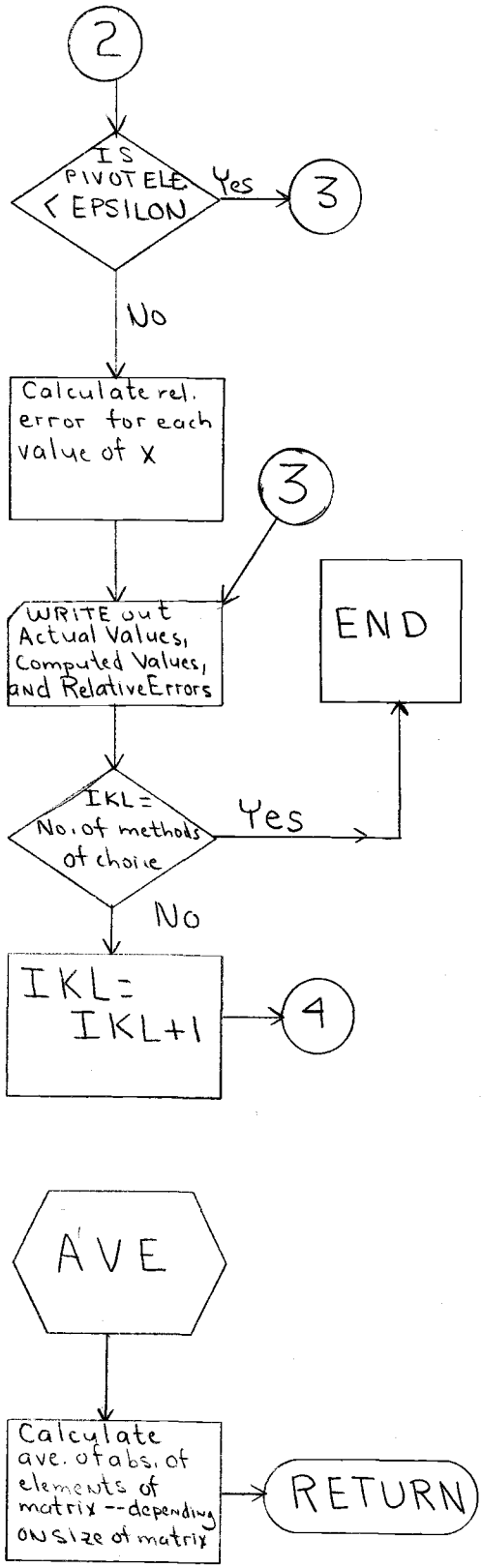
5. Wilkinson, J. H. Rounding Errors in Algebraic Processes. Englewood Cliffs, N. J., Prentice-Hall, 1963. p. 91-99.

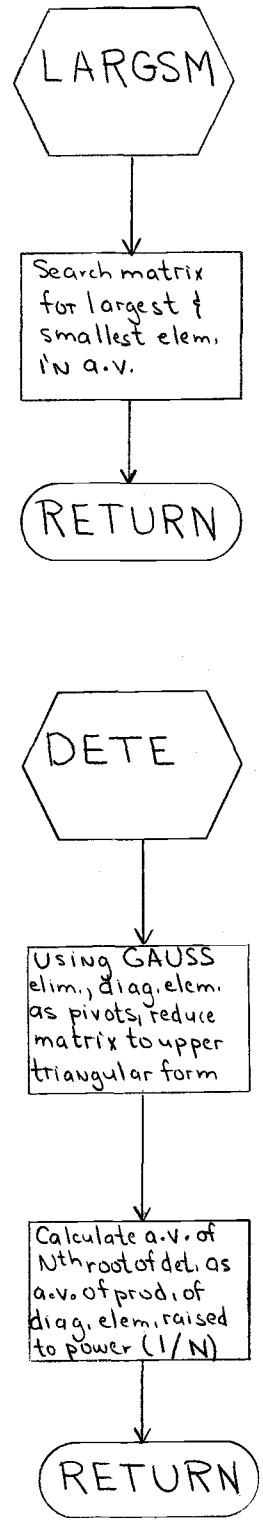
APPENDICES

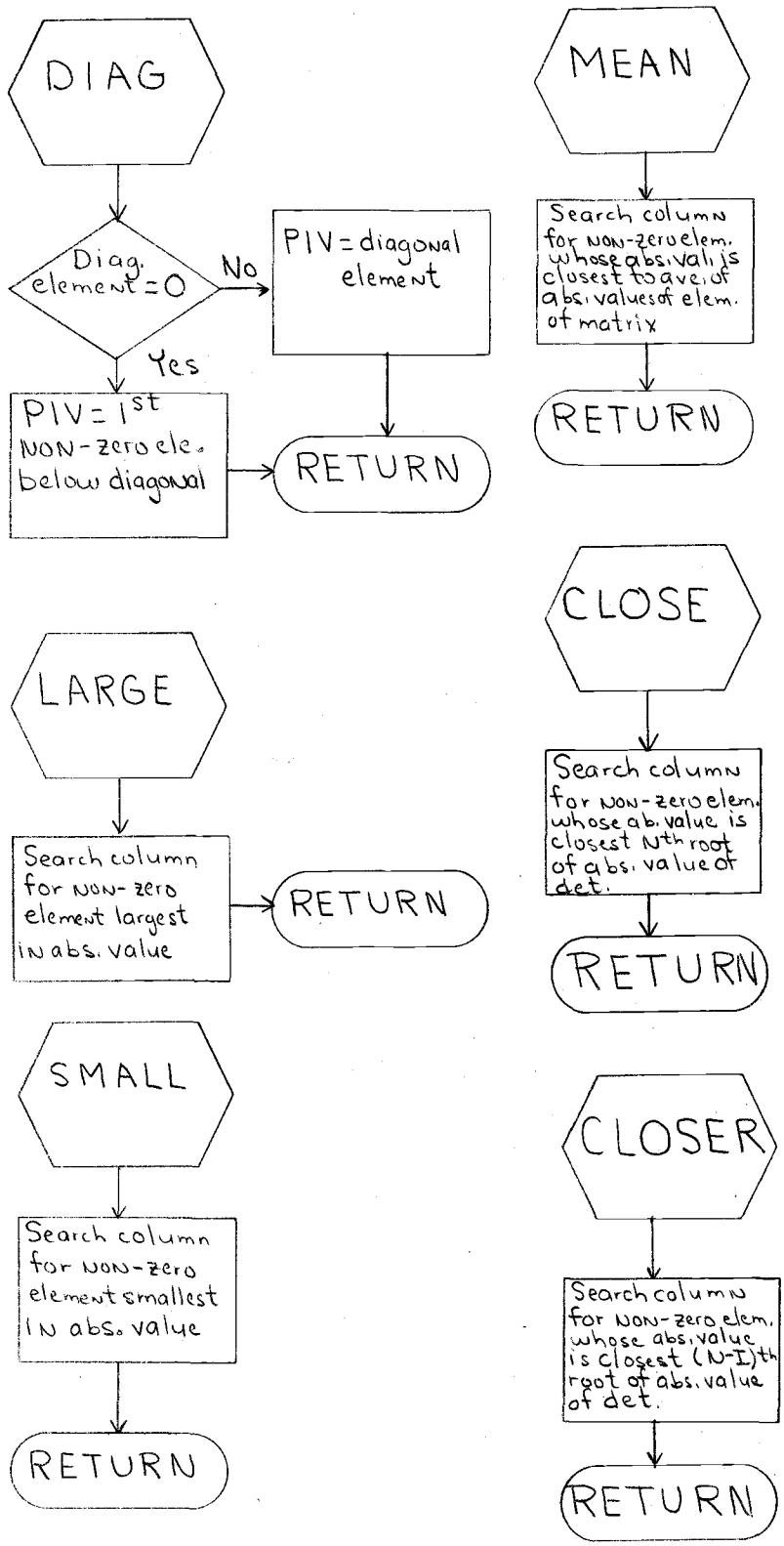
APPENDIX 1

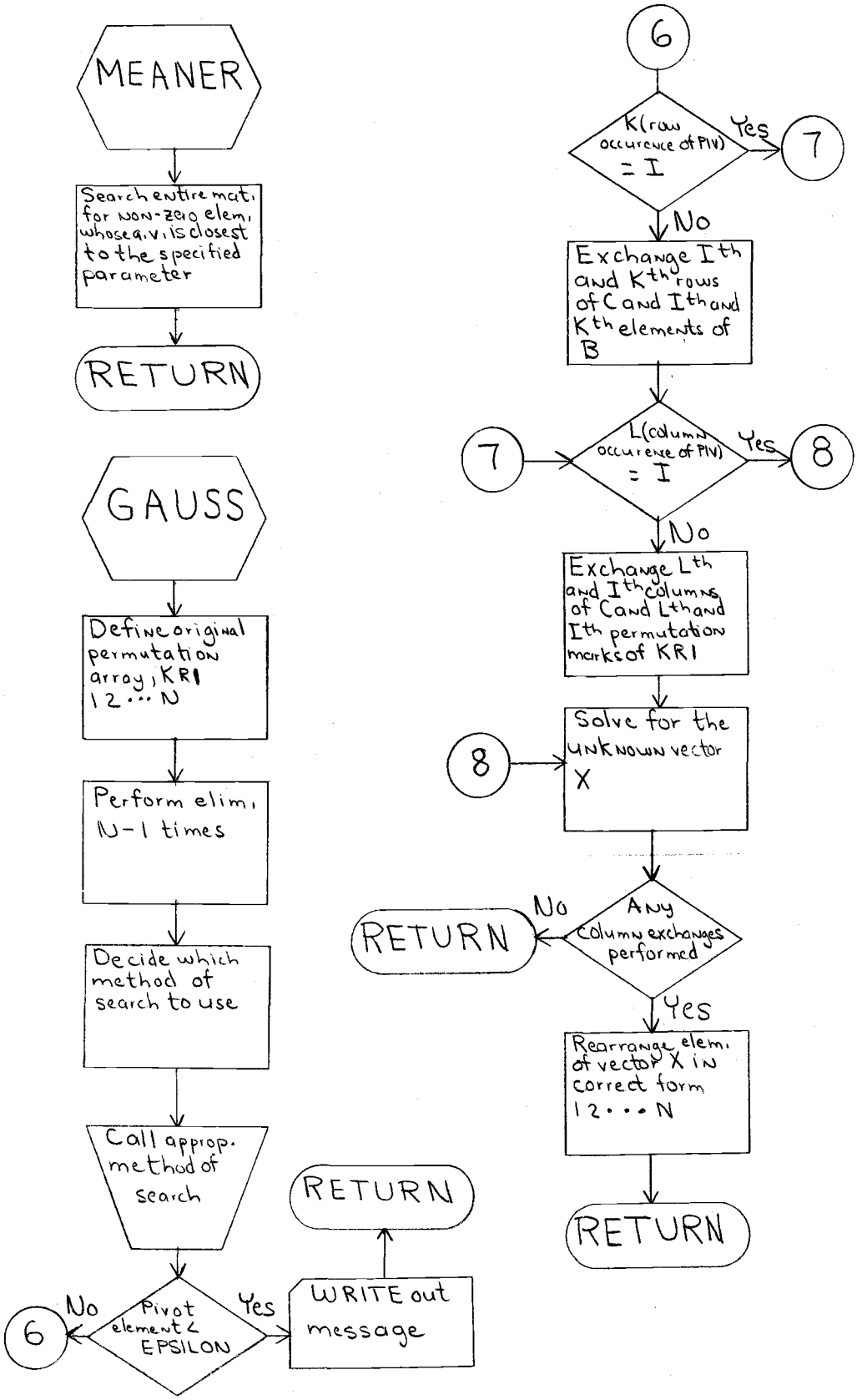
Flow Charts of the Program


Main Program
ELIM
DIMENSION the Hilbert matrix and the vector arrays
READ N, size of matrix
READ in actual values ACTU array
IKL = 1
EPSILON = 10^{-8} $\|C\|_1$
(tolerance allowed)
Read in inverse of Hilbert matrix
Set each element of B vector = 1
Copy C to dummy matrix A
Copy B to dummy vector D
WRITE out C and B
1
1
CALL AVE to calculate ave. of ab. val. Size = N
CALL DETE
Copy C to dummy matrix A
Copy B to dummy vector D
4
CALL GAUSS
IS MATRIX SINGULAR
No
2
Yes
WRITE Message
END


2
IS PIVOT ELE. < EPSILON
Yes
3
No
Calculate rel. error for each value of X
3
WRITE out Actual Values, Computed Values, and RelativeErrors
END
IKL = No. of methods of choice
Yes
No
IKL = IKL+1
4
AVE
Calculate ave. of abs. of elements of matrix -- depending on size of matrix
RETURN


LARGSM
Search matrix for largest & smallest elem. IN a.v.
RETURN
DETE
Using GAUSS elim., diag. elem. as pivots, reduce matrix to upper triangular form
Calculate a.v. of Nth root of det. as a.v. of prod. of diag. elem. raised to power (1/N)
RETURN


DIAG
Diag. element = 0
No
Yes
PIV = diagonal element
PIV = 1st non-zero ele. below diagonal
RETURN
MEAN
Search column for non-zero elem. whose abs. val. is closest to ave. of abs. values of elem. of matrix
RETURN
CLOSE
Search column for non-zero elem. whose ab. value is closest Nth root of abs. value of det.
RETURN
LARGE
Search column for non-zero element largest in abs. value
RETURN
SMALL
Search column for non-zero element smallest in abs. value
RETURN
CLOSER
Search column for non-zero elem. whose abs. value is closest (N-I)th root of abs. value of det.
RETURN


MEANER
Search entire mat. for non-zero elem. whose a.v. is closest to the specified parameter
RETURN
GAUSS
Define original permutation array, KRI 1 2 ··· N
Perform elim. N-1 times
Decide which method of search to use
Call approp. method of search
Pivot element < EPSILON
No
Yes
6
WRITE out message
RETURN
6
K(row occurence of PIV) = I
Yes
7
No
Exchange Ith and Kth rows of C and Ith and Kth elements of B
7
L(column occurence of PIV) = I
Yes
8
No
Exchange Lth and Ith columns of C and Lth and Ith permutation marks of KRI
8
Solve for the UNKNOWN vector X
ANY column exchanges performed
No
RETURN
Yes
Rearrange elem. of vector X in correct form 1 2 ··· N
RETURN

APPENDIX 2

Output for the Solutions

of the Test Cases

SOLUTIONS FOR THE MATRIX EQUATION INVOLVING THE INVERSE OF THE 5X5 HILBERT MATRIX

USING THE DIAGONAL ELEMENTS FOR PIVOT ELEMENTS

COMPUTED SOLUTION		ACTUAL VALUE		RELATIVE ERROR
X(1) =	2.2833333760E 00	X(1) =	2.2833333333E 00	1.87369185867E-08
X(2) =	1.4500000353E 00	X(2) =	1.4500000000E 00	2.43267879408E-08
X(3) =	1.0928571724E 00	X(3) =	1.0928571429E 00	2.70037893567E-08
X(4) =	8.8452383482E-01	X(4) =	8.8452380951E-01	2.86259479105E-08
X(5) =	7.4563494279E-01	X(5) =	7.4563492063E-01	2.97230806650E-08

SEARCHING THE CURRENT COLUMN FOR THE ELEMENT LARGEST IN ABSOLUTE VALUE

COMPUTED SOLUTION		ACTUAL VALUE		RELATIVE ERROR
X(1) =	2.2833332729E 00	X(1) =	2.2833333333E 00	2.64356252714E-08
X(2) =	1.4499999692E 00	X(2) =	1.4500000000E 00	2.12558320374E-08
X(3) =	1.0928571221E 00	X(3) =	1.0928571429E 00	1.89878716093E-08
X(4) =	8.8452379379E-01	X(4) =	8.8452380951E-01	1.77842814321E-08
X(5) =	7.4563490734E-01	X(5) =	7.4563492063E-01	1.70180737623E-08

SEARCHING THE CURRENT COLUMN FOR THE NON-ZERO ELEMENT SMALLEST IN ABSOLUTE VALUE

	COMPUTED SOLUTION		ACTUAL VALUE		RELATIVE ERROR
X(1) =	2.2833333750E 00	X(1) =	2.2833333333E 00		1.87369185867E-08
X(2) =	1.4500000353E 00	X(2) =	1.4500000000E 00		2.43267879408E-08
X(3) =	1.0928571224E 00	X(3) =	1.0928571429E 00		2.70037893567E-08
X(4) =	8.8452383482E-01	X(4) =	8.8452380951E-01		2.86259479105E-08
X(5) =	7.4563494270E-01	X(5) =	7.4563492063E-01		2.97230806650E-08

SEARCHING THE CURRENT COLUMN FOR THE NON-ZERO ELEMENT WHOSE ABSOLUTE VALUE IS NEAREST THE ARITHMETIC AVERAGE OF THE ABSOLUTE VALUES OF THE ELEMENTS OF THE MATRIX

	COMPUTED SOLUTION		ACTUAL VALUE		RELATIVE ERROR
X(1) =	2.2833332720E 00	X(1) =	2.2833333333E 00		2.64356252714E-08
X(2) =	1.4499999692E 00	X(2) =	1.4500000000E 00		2.12558320374E-08
X(3) =	1.0928571221E 00	X(3) =	1.0928571429E 00		1.89878716093E-08
X(4) =	8.8452379778E-01	X(4) =	8.8452380951E-01		1.77842814321E-08
X(5) =	7.4563490794E-01	X(5) =	7.4563492063E-01		1.70180737623E-08

SEARCHING THE ENTIRE SUPPRESSED MATRIX FOR THE NON-ZERO ELEMENT WHOSE ABSOLUTE VALUE IS
NEAREST THE ARITHMETIC AVERAGE OF THE ABSOLUTE VALUES OF THE ELEMENTS OF THE SUPPRESSED MATRIX

	COMPUTED SOLUTION	ACTUAL VALUE	RELATIVE ERROR
X(1) =	2.2833333332E 00	X(1) = 2.2833333333E 00	2.54924062401E-11
X(2) =	1.4500000000E 00	X(2) = 1.4500000000E 00	2.00716072117E-11
X(3) =	1.0928571428E 00	X(3) = 1.0928571429E 00	5.32619119459E-11
X(4) =	8.8452380945E-01	X(4) = 8.8452380951E-01	6.58067768055E-11
X(5) =	7.4563492059E-01	X(5) = 7.4563492063E-01	5.85484189063E-11

SEARCHING THE CURRENT COLUMN FOR THE NON-ZERO ELEMENT WHOSE ABSOLUTE VALUE IS
NEAREST THE N-TH ROOT OF THE ABSOLUTE VALUE OF THE DETERMINANT OF THE MATRIX

	COMPUTED SOLUTION	ACTUAL VALUE	RELATIVE ERROR
X(1) =	2.2833334486E 00	X(1) = 2.2833333333E 00	5.05004567618E-08
X(2) =	1.4500000832E 00	X(2) = 1.4500000000E 00	5.74248682323E-08
X(3) =	1.0928572097E 00	X(3) = 1.0928571429E 00	6.11446749151E-08
X(4) =	8.8452386570E-01	X(4) = 8.8452380951E-01	6.35199913126E-08
X(5) =	7.4563496925E-01	X(5) = 7.4563492063E-01	6.52034225233E-08

SEARCHING THE CURRENT COLUMN FOR THE NON-ZERO ELEMENT WHOSE ABSOLUTE VALUE IS
NEAREST THE (N-I)TH ROOT OF THE ABSOLUTE VALUE OF THE DETERMINANT OF THE MATRIX

	COMPUTED SOLUTION		ACTUAL VALUE		RELATIVE ERROR
X(1) =	2.2833334486E 00	X(1) =	2.2833333333E 00		5.05004567618E-08
X(2) =	1.4500000832E 00	X(2) =	1.4500000000E 00		5.74248682323E-08
X(3) =	1.0928572097E 00	X(3) =	1.0928571429E 00		6.11446749151E-08
X(4) =	8.8452386570E-01	X(4) =	8.8452380951E-01		6.35199913126E-08
X(5) =	7.4563496925E-01	X(5) =	7.4563492063E-01		6.52034225233E-08

ALTERNATING BETWEEN THE DIAGONAL METHOD AND THE CURRENT COLUMN ARITHMETIC AVERAGE METHOD

	COMPUTED SOLUTION		ACTUAL VALUE		RELATIVE ERROR
X(1) =	2.2833333355E 00	X(1) =	2.2833333333E 00		9.68711437122E-10
X(2) =	1.4500000001E 00	X(2) =	1.4500000000E 00		8.02864288469E-11
X(3) =	1.0928571428E 00	X(3) =	1.0928571429E 00		2.66309559729E-11
X(4) =	8.8452380949E-01	X(4) =	8.8452380951E-01		3.29033884028E-11
X(5) =	7.4563492062E-01	X(5) =	7.4563492063E-01		1.95161396352E-11

SEARCHING THE ENTIRE SUPPRESSED MATRIX FOR THE NON-ZERO ELEMENT WHOSE ABSOLUTE VALUE IS NEAREST (ABS(MAX ELEMENT)+ABS(MIN ELEMENT))/2

	COMPUTED SOLUTION		ACTUAL VALUE		RELATIVE ERROR
X(1) =	2.2833331357E 00	X(1) =	2.2833333333E 00		8.65467191837E-08
X(2) =	1.4499993293E 00	X(2) =	1.4500000000E 00		1.17719976293E-07
X(3) =	1.0928569973E 00	X(3) =	1.0928571429E 00		1.33234672729E-07
X(4) =	8.8452368369E-01	X(4) =	8.8452380949E-01		1.42241348073E-07
X(5) =	7.4563481020E-01	X(5) =	7.4563492060E-01		1.48068951420E-07

SEARCHING THE ENTIRE SUPPRESSED MATRIX FOR THE ELEMENT LARGEST IN ABSOLUTE VALUE

	COMPUTED SOLUTION		ACTUAL VALUE		RELATIVE ERROR
X(1) =	2.2833333947E 00	X(1) =	2.2833333333E 00		2.68689961761E-08
X(2) =	1.4500000087E 00	X(2) =	1.4500000000E 00		6.00141055614E-09
X(3) =	1.0928571693E 00	X(3) =	1.0928571429E 00		2.41542770667E-08
X(4) =	8.8452386910E-01	X(4) =	8.8452380949E-01		6.73861394520E-08
X(5) =	7.4563498790E-01	X(5) =	7.4563492060E-01		9.02621458168E-08

SOLUTIONS OF THE MATRIX EQUATION INVOLVING THE INVERSE OF THE 6X6 HILBERT MATRIX

SEARCHING THE ENTIRE SUPPRESSED MATRIX FOR THE NON-ZERO ELEMENT WHOSE ABSOLUTE VALUE IS NEAREST THE ARITHMETIC AVERAGE OF THE ABSOLUTE VALUES OF THE ELEMENTS OF THE SUPPRESSED MATRIX

	COMPUTED SOLUTION		ACTUAL VALUE		RELATIVE ERROR
X(1) =	2.4500002231E 00	X(1) =	2.4500000000E 00		9.43201689073E-09
X(2) =	1.5928571187E 00	X(2) =	1.5928571429E 00		1.52018572713E-08
X(3) =	1.2178572462E 00	X(3) =	1.2178571429E 00		8.48124879762E-08
X(4) =	9.9563492888E-01	X(4) =	9.9563492059E-01		8.33095696871E-09
X(5) =	8.4563496048E-01	X(5) =	8.4563492059E-01		4.71850802086E-08
X(6) =	7.3654405129E-01	X(6) =	7.3654401150E-01		5.40354514169E-08

ALTERNATING BETWEEN THE DIAGONAL METHOD AND THE CURRENT COLUMN ARITHMETIC AVERAGE METHOD

	COMPUTED SOLUTION		ACTUAL VALUE		RELATIVE ERROR
X(1) =	2.4499992357E 00	X(1) =	2.4500000000E 00		3.11969304259E-07
X(2) =	1.5928567813E 00	X(2) =	1.5928571429E 00		2.27022928593E-07
X(3) =	1.2178579351E 00	X(3) =	1.2178571429E 00		6.50491947785E-07
X(4) =	9.9563957241E-01	X(4) =	9.9563492059E-01		4.67220528808E-06
X(5) =	8.4563410250E-01	X(5) =	8.4563492059E-01		9.67431810452E-07
X(6) =	7.3654376121E-01	X(6) =	7.3654401150E-01		3.39820754796E-07

SEARCHING THE CURRENT COLUMN FOR THE ELEMENT LARGEST IN ABSOLUTE VALUE

	COMPUTED SOLUTION		ACTUAL VALUE	RELATIVE ERROR
X(1) =	2.4499996217E 00	X(1) =	2.4500000000E 00	1.54404729907E-07
X(2) =	1.5928536419E 00	X(2) =	1.5928571429E 00	2.19794737804E-06
X(3) =	1.2178577406E 00	X(3) =	1.2178571429E 00	4.90784464800E-07
X(4) =	9.9563754760E-01	X(4) =	9.9563492059E-01	2.63851638191E-06
X(5) =	8.4563402128E-01	X(5) =	8.4563492059E-01	1.06345396492E-06
X(6) =	7.3654389870E-01	X(6) =	7.3654401150E-01	1.53136666885E-07

SEARCHING THE ENTIRE SUPPRESSED MATRIX FOR THE ELEMENT LARGEST IN ABSOLUTE VALUE

	COMPUTED SOLUTION		ACTUAL VALUE	RELATIVE ERROR
X(1) =	2.4499997025E 00	X(1) =	2.4500000000E 00	1.21428308132E-07
X(2) =	1.5928569821E 00	X(2) =	1.5928571429E 00	1.00949833440E-07
X(3) =	1.2178579529E 00	X(3) =	1.2178571429E 00	6.65021672426E-07
X(4) =	9.9563805929E-01	X(4) =	9.9563492059E-01	3.15246334835E-06
X(5) =	8.4563429870E-01	X(5) =	8.4563492059E-01	7.35412687017E-07
X(6) =	7.3654298760E-01	X(6) =	7.3654401150E-01	1.39012372834E-06

SOLUTIONS FOR THE MATRIX EQUATION INVOLVING THE INVERSE OF THE 7X7 HILBERT MATRIX

SEARCHING THE ENTIRE SUPPRESSED MATRIX FOR THE NON-ZERO ELEMENT WHOSE ABSOLUTE VALUE IS NEAREST THE ARITHMETIC AVERAGE OF THE ABSOLUTE VALUES OF THE ELEMENTS OF THE SUPPRESSED MATRIX

	COMPUTED SOLUTION	ACTUAL VALUE	RELATIVE ERROR
X(1) =	2.5928576286E 00	X(1) = 2.5928571429E 00	1.87316421958E-07
X(2) =	1.7178572892E 00	X(2) = 1.7178571429E 00	8.51671259804E-08
X(3) =	1.3289679542E 00	X(3) = 1.3289682539E 00	2.25499850226E-07
X(4) =	1.0956342315E 00	X(4) = 1.0956349206E 00	6.28969088674E-07
X(5) =	9.3654373279E-01	X(5) = 9.3654401149E-01	2.97581616062E-07
X(6) =	8.1987698788E-01	X(6) = 8.1987734488E-01	4.35433590785E-07
X(7) =	7.3013421730E-01	X(7) = 7.3013375510E-01	6.33031863335E-07

ALTERNATING BETWEEN THE DIAGONAL METHOD AND THE CURRENT COLUMN ARITHMETIC AVERAGE METHOD

	COMPUTED SOLUTION	ACTUAL VALUE	RELATIVE ERROR
X(1) =	2.5928197853E 00	X(1) = 2.5928571429E 00	1.44078969676E-05
X(2) =	1.7178371025E 00	X(2) = 1.7178571429E 00	1.16659309942E-05
X(3) =	1.3289853681E 00	X(3) = 1.3289682539E 00	1.28778218830E-05
X(4) =	1.0956279113E 00	X(4) = 1.0956349206E 00	6.39748531883E-06
X(5) =	9.3653986522E-01	X(5) = 9.3654401149E-01	4.42721020593E-06
X(6) =	8.1986795310E-01	X(6) = 8.1987734488E-01	1.14551177621E-05
X(7) =	7.3013662340E-01	X(7) = 7.3013375510E-01	3.92845678848E-06

SEARCHING THE CURRENT COLUMN FOR THE ELEMENT LARGEST IN ABSOLUTE VALUE

	COMPUTED SOLUTION		ACTUAL VALUE	RELATIVE ERROR
X(1)=	2.5928519765E 00	X(1)=	2.5928571429E 00	1.99254925456E-06
X(2)=	1.7178360251E 00	X(2)=	1.7178571429E 00	1.22931046240E-05
X(3)=	1.3289761324E 00	X(3)=	1.3289682539E 00	5.92830061726E-06
X(4)=	1.0956038653E 00	X(4)=	1.0956349206E 00	2.83445698122E-05
X(5)=	9.3653870211E-01	X(5)=	9.3654401149E-01	5.66912245576E-06
X(6)=	8.1986430949E-01	X(6)=	8.1987734488E-01	1.58992098222E-05
X(7)=	7.3013993569E-01	X(7)=	7.3013375510E-01	8.46501241159E-06

SEARCHING THE ENTIRE SUPPRESSED MATRIX FOR THE ELEMENT LARGEST IN ABSOLUTE VALUE

	COMPUTED SOLUTION		ACTUAL VALUE	RELATIVE ERROR
X(1)=	2.5928501534E 00	X(1)=	2.5928571429E 00	2.69568176311E-06
X(2)=	1.7178375097E 00	X(2)=	1.7178571429E 00	1.14288963293E-05
X(3)=	1.3289506713E 00	X(3)=	1.3289682539E 00	1.32302516286E-05
X(4)=	1.0956152986E 00	X(4)=	1.0956349206E 00	1.79092559498E-05
X(5)=	9.3653915359E-01	X(5)=	9.3654401149E-01	5.18704396684E-06
X(6)=	8.1986551390E-01	X(6)=	8.1987734488E-01	1.44302049477E-05
X(7)=	7.3012015319E-01	X(7)=	7.3013375510E-01	1.86293372954E-05