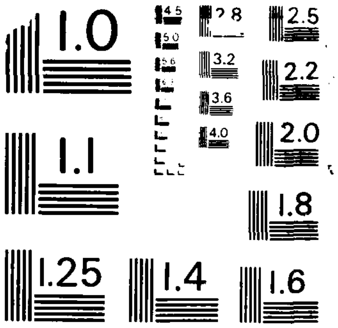
AD-A190 590 A NEW BLOCK SOLVER FOR LARGE FULL UNSYMMETRIC COMPLEX 1/1
SYSTEMS OF LINEAR ALGEBRAIC EQUATIONS(U) DAVID TAYLOR
RESEARCH CENTER BETHESDA MD E A SCHROEDER FEB 88
UNCLASSIFIED DTRC-88/003 F/G 12/4 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

4

David Taylor Research Center

Bethesda, MD 20084-5000

DTRC-88/003 February 1988

Computation, Mathematics and Logistics Department
Research and Development Report

A New Block Solver for Large, Full, Unsymmetric, Complex Systems of Linear Algebraic Equations

by

Erwin A. Schroeder

DTRC-88/003 A New Block Solver for Large, Full, Unsymmetric, Complex Systems of Linear Algebraic Equations

Approved for public release; distribution is unlimited.

MAJOR DTRC TECHNICAL COMPONENTS

CODE 011	DIRECTOR OF TECHNOLOGY, PLANS AND ASSESSMENT
12	SHIP SYSTEMS INTEGRATION DEPARTMENT
14	SHIP ELECTROMAGNETIC SIGNATURES DEPARTMENT
15	SHIP HYDROMECHANICS DEPARTMENT
16	AVIATION DEPARTMENT
17	SHIP STRUCTURES AND PROTECTION DEPARTMENT
18	COMPUTATION, MATHEMATICS & LOGISTICS DEPARTMENT
19	SHIP ACOUSTICS DEPARTMENT
27	PROPULSION AND AUXILIARY SYSTEMS DEPARTMENT
28	SHIP MATERIALS ENGINEERING DEPARTMENT

DTRC ISSUES THREE TYPES OF REPORTS:

1. **DTRC reports, a formal series,** contain information of permanent technical value. They carry a consecutive numerical identification regardless of their classification or the originating department.

2. **Departmental reports, a semiformal series,** contain information of a preliminary, temporary, or proprietary nature or of limited interest or significance They carry a departmental alphanumerical identification

3. **Technical memoranda, an informal series,** contain technical documentation of limited use and interest They are primarily working papers intended for internal use They carry an identifying number which indicates their type and the numerical code of the originating department Any distribution outside DTRC must be approved by the head of the originating department on a case-by-case basis

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

1b RESTRICTIVE MARKINGS

2a SECURITY CLASSIFICATION AUTHORITY

2b DECLASSIFICATION / DOWNGRADING SCHEDULE

3 DISTRIBUTION / AVAILABILITY OF REPORT
Approved for public release;
distribution is unlimited.

4 PERFORMING ORGANIZATION REPORT NUMBER(S)
DTRC-88/003

5 MONITORING ORGANIZATION REPORT NUMBER(S)

6a NAME OF PERFORMING ORGANIZATION
David Taylor Research Center

6b OFFICE SYMBOL *(If applicable)*
Code 1844

7a NAME OF MONITORING ORGANIZATION

6c ADDRESS *(City, State, and ZIP Code)*
Bethesda, MD 20084-5000

7b ADDRESS *(City, State, and ZIP Code)*

8a NAME OF FUNDING / SPONSORING ORGANIZATION

8b OFFICE SYMBOL *(If applicable)*

9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c ADDRESS *(City, State, and ZIP Code)*

10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO
63569N		S1255001	DN178112

11 TITLE *(Include Security Classification)*
A NEW BLOCK SOLVER FOR LARGE, FULL, UNSYMMETRIC, COMPLEX SYSTEMS OF LINEAR ALGEBRAIC EQUATIONS

12 PERSONAL AUTHOR(S)
Schroeder, Erwin A.

13a TYPE OF REPORT
Final

13b TIME COVERED
FROM ______ TO ______

14 DATE OF REPORT *(Year, Month, Day)*
1988 February

15 PAGE COUNT
29

16 SUPPLEMENTARY NOTATION

17 COSATI CODES

FIELD	GROUP	SUB-GROUP

18 SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Linear Equations Block Solvers
Equation Solvers

19 ABSTRACT *(Continue on reverse if necessary and identify by block number)*

A new block solver, OCSOLVE, for large, full, unsymmetric systems of algebraic equations with complex-valued coefficients has been developed. Although OCSOLVE was developed for use with the finite element program NASTRAN, it is designed to be easily adapted for other applications. This new solver was developed because NASTRAN's solver was not designed to solve full, unsymmetric systems efficiently; it reduced the time required to solve such a system of 500 equations with complex-valued coefficients to about 5% of the time required by the equation solver in NASTRAN. The solver is easily modified to use double precision complex arithmetic on computers on which it is available. With somewhat more effort it could be modified to solve systems of equations having real-valued coefficients.

(Continued on reverse side)

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT
☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS

21 ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

22a NAME OF RESPONSIBLE INDIVIDUAL
Erwin A. Schroeder

22b TELEPHONE *(Include Area Code)*
(202) 227-1645

22c OFFICE SYMBOL
Code 1844

83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

(Block 19 Continued)

Several features distinguish this linear equation solver from previous solvers. It automatically determines the dimensions of the blocks of coefficients and blocks of right-hand side vectors and avoids the need for adding extra equations by providing for blocks of more than one size. It accepts columns of the coefficient matrix and columns of the right-hand side vectors from a sequential file and returns the columns of solution vectors on a sequential file. The program OCSOLVE will solve with one call, a linear system having multiple right-hand sides. It will solve a system of linear equations if a specified minimum number of words are provided for storing the blocks; however, the more memory provided, and hence the larger the blocks, the more efficient the solution will be.

ABSTRACT

A new block solver, OCSOLVE, for large, full, unsymmetric systems of algebraic equations with complex-valued coefficients has been developed. Although OCSOLVE was developed for use with the finite element program NASTRAN, it is designed to be easily adapted for other applications. This new solver was developed because NASTRAN's solver was not designed to solve full, unsymmetric systems efficiently; it reduced the time required to solve such a system of 500 equations with complex-valued coefficients to about 5% of the time required by the equation solver in NASTRAN. The solver is easily modified to use double precision complex arithmetic on computers on which it is available. With somewhat more effort it could be modified to solve systems of equations having real-valued coefficients.

Several features distinguish this linear equation solver from previous solvers. It automatically determines the dimensions of the blocks of coefficients and blocks of right-hand side vectors and avoids the need for adding extra equations by providing for blocks of more than one size. It accepts columns of the coefficient matrix and columns of the right-hand side vectors from a sequential file and returns the columns of solution vectors on a sequential file. The program OCSOLVE will solve with one call, a linear system having multiple right-hand sides. It will solve a system of linear equations if a specified minimum number of words are provided for storing the blocks; however, the more memory provided, and hence the larger the blocks, the more efficient the solution will be.

ADMINISTRATIVE INFORMATION

This work was sponsored by the Foundation Acoustic Design Program, Task Area S1255001, Element 63569N, DTRC Work Unit 1211-801. The Naval Sea Systems Command program manager was R. Chiu (NAVSEA 55Y3). The DTRC program manager was O. Ritter, Code 1211, and the technical liaison, T. Tinley, Code 1720.1.

INTRODUCTION

The David Taylor Research Center (DTRC) has developed a capability for computing acoustic pressure fields outside submerged three-dimensional elastic structures. This capability, called NASHUA[1,2] is implemented using the finite element structural analysis program NASTRAN and includes a requirement for solving a large system of linear algebraic equations. The coefficient matrix for this system is full, complex, unsymmetric, and so large that the memory needed for storing it exceeds the memory available on the

Center's CDC Cyber 176 and Cray X-MP. Also the system must be solved for many right-hand sides. To satisfy the requirements of NASHUA, an out-of-core equation solver has been written that can solve this type of linear system. This solver is designed to solve a system of equations produced by another program (in this case NASTRAN) that writes the coefficients and right-hand side vectors on a sequential file and requires the solution vectors to be returned on a sequential file. The program is written in standard FORTRAN 77 to facilitate its use on different computers. Although the program is written for single precision complex coefficients, it is easily modified to use double precision complex arithmetic with computers for which this arithmetic is available. Also, but with somewhat more effort, the program could be modified to solve systems of equations having real-valued coefficients. When blocks of the coefficient matrix and of the right-hand side vectors are in central memory, they are stored in arrays in blank COMMON. For single precision complex arithmetic, the minimum memory required in blank COMMON for these blocks is twice the length of one column of the coefficient matrix plus 150 words. Of course, the larger the number of equations and the smaller the number of words of memory provided for storing the blocks, the greater the cost of the solution.

These systems of equations can be solved using subroutines provided in NASTRAN. However, since it is unusual to encounter large, full, unsymmetric systems in finite element structural analyses, the procedures incorporated in NASTRAN have not been optimized to the extent that have the more commonly used equation solvers and consequently the solution is very slow. For this application, the program OCSOLVE reduced the time to solve a full, unsymmetric, complex system of order 500 to about 5% of the time required by the equation solver included in NASTRAN.

This report documents the out-of-core block solver program OCSOLVE developed by the Applied Mathematics Division at DTRC to solve large linear systems $AX = B$, where A is a full, unsymmetric matrix of complex coefficients, B is a matrix of one or more right-hand side vectors, and X is the matrix of unknown vectors. This program solves a system of linear equations by partitioning the matrix of coefficients and the matrix of right-hand side vectors into submatrices called blocks. The blocks are stored in direct access files and brought into central memory a few at a time. The program OCSOLVE will solve a system of any order with any number of right-hand side vectors. A block solver is most efficient if the blocks are as large as possible for the memory available.[3] Therefore, it is desirable that the solver automatically determine block sizes, read the data from a sequential file, store it in blocked configuration, and after solving the equations return the solution vectors in a sequential file.

Cantin[3] and Hofmeister[4] have developed block solvers for large systems of linear equations. Cantin's solver is designed for large, symmetric, banded systems; Hofmeister's solver is designed for large, unsymmetric, full systems. Each will solve a system of equations for one right-hand side vector at a time. These programs require that the matrices have already been stored in block configuration and that all the blocks are of equal size. To satisfy these requirements, the user must organize the data in block configuration and, if the number of rows of blocks does not divide the order of the system, must add equations to fill out the last row and column of blocks. Care is needed in reorganizing sequential data into block configuration to avoid excessive calls to disk storage, for although one access to disk storage is fairly quick, many calls will add significantly to the time required for solving the equations. If all blocks must be of equal size in the block elimination process, the number of equations added to the system will be no greater than the number of rows of blocks used. Thus the penalty in time to solve the equations is not too large if only a few rows of blocks are used. However, adding the extra equations requires another step in the preparation of the system for solution and requires some additional time in the solution of the system. Providing for variably sized blocks eliminates both disadvantages. Cantin and Hofmeister both use machine-dependent calls to access disk storage. Although it is not difficult to modify these calls, FORTRAN 77 now provides standard procedures for reading and writing to direct access files. Crotty[5] also developed a block solver especially for systems of equations arising in applications of the boundary integral equation method. His solver was designed for systems with large blocks of zero entries in the coefficient matrix.

The block elimination process used in the program OCSOLVE is essentially that of Cantin[3] as modified by Hofmeister.[4] The program implementing the process has been designed to make it easy to use with other programs as an intermediate routine. The following features are new to OCSOLVE:

- It solves, with one call, systems of equations with multiple right-hand sides.

- It obtains the order of the system and the number of right-hand side vectors from a sequential file and automatically determines a suitable blocking of the matrix of coefficients and the matrix of right-hand side vectors.

- It takes these matrices from the sequential file, and stores them in the block configuration required by the block solver.

- It reorganizes the solution vectors from block configuration and writes them on a sequential file after the system is solved.

- It does not require the blocks to have equal dimensions, so the order of the system of equations being solved need not be divisible by the number of rows of blocks.

The program OCSOLVE is designed to read data from sequential files with quite general format and return the solution in a sequential file with an equally general format. To do this, the program calls five very short subroutines that read a label, heading, or matrix column from the data file or write a heading or column to the solution file. Appendix A shows an example of a set of these routines that can be used with sequential files with a simple, fixed format and another set that can be used with sequential files with the NASTRAN OUTPUT2 or INPUTT2 format. If the solver is to be used with a program that uses and produces sequential files with another format, the user can easily modify these routines to accommodate that format.

To determine the size of the submatrix blocks, the program OCSOLVE uses the order of the linear system and the memory available for the blank COMMON arrays. The order of the linear system is obtained from the data file. The user determines, according to the memory available in the particular application, the number of words available for blank COMMON and supplies this number as a parameter in the FORTRAN PARAMETER statement at the beginning of the program. The user also provides in this statement the name of the file that contains the input data and the output solution vectors as well as parameters that implement single or double precision arithmetic and indicate the type of computer that is used

OUT-OF-CORE STRATEGY

In the out-of-core elimination strategy, the matrix of coefficients and the matrix of right-hand side vectors are partitioned into blocks of submatrices, producing a matrix-like array of blocks. The block elimination procedure is illustrated here for a system of four equations.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \end{Bmatrix}$$

The coefficients a_{ij} and the right-hand side terms k_i are known, and the x_i are to be determined. First the solution of the system by the usual Gaussian elimination will be described, and then the block elimination procedure will be shown to be the same procedure with matrix arithmetic replacing numeric arithmetic.

When this system is solved by Gaussian elimination,[6] elementary row operations are used to transform the system to an equivalent system in which all entries in the coefficient matrix below the main diagonal are zero. For this example the equations are

$$\begin{aligned} b_{11}x_1 + b_{12}x_2 + b_{13}x_3 + b_{14}x_4 &= h_1 \\ b_{22}x_2 + b_{23}x_3 + b_{24}x_4 &= h_2 \\ b_{33}x_3 + b_{34}x_4 &= h_3 \\ b_{44}x_4 &= h_4. \end{aligned}$$

The last equation can easily be solved for $x_4 = b_{44}^{-1} h_4$. With x_4 known, the second last equation can be solved for $x_3 = b_{33}^{-1} (h_3 - b_{34}x_4)$. Similarly, each of the remaining two equations is solved for the remaining unknown values x_2 and x_1, respectively. The elementary row operations use addition and multiplication of the numbers a_{ij} and k_i.

For the block elimination procedure, the coefficient matrix, the unknown vector, and the right-hand side vector are each partitioned to form two matrix equations with two vector unknowns. The partition produces a total of eight blocks of coefficients, unknowns, and right-hand side values

$$\left[\begin{array}{cc|cc} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ \hline a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{array}\right] \left(\begin{array}{c} x_1 \\ x_2 \\ \hline x_3 \\ x_4 \end{array}\right) = \left(\begin{array}{c} k_1 \\ k_2 \\ \hline k_3 \\ k_4 \end{array}\right)$$

Elementary row operations, now using multiplication and addition of matrices, transform the system into the equivalent system

$$\left[\begin{array}{cc|cc} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ \hline 0 & 0 & c_{33} & c_{34} \\ 0 & 0 & c_{43} & c_{44} \end{array}\right] \left(\begin{array}{c} x_1 \\ x_2 \\ \hline x_3 \\ x_4 \end{array}\right) = \left(\begin{array}{c} h_1 \\ h_2 \\ \hline h_3 \\ h_4 \end{array}\right)$$

The lower two rows form a linear system of order two that can be solved by the usual Gaussian elimination for x_3 and x_4. Substituting these values into the upper two rows produces another system of order two that can be solved for x_1 and x_2. In this example of a block solution, only systems of order two are solved and the procedure can be organized so that at most four blocks are required in memory at one time. When not in memory, the blocks are stored in direct access files. Even for this very small example, an in-core solution would require that all eight blocks be stored in memory at one time. The blocks are assigned to records in column order from the array of blocks. The matrix entries in each block are also stored in column order in the block's record in the direct access file.

The size of the blocks is determined by the order of the linear system, the number of right-hand side vectors, and the memory available. If the order of the matrix of coefficients is divisible by the number of blocks, the coefficient blocks will all be square and have equal dimensions, otherwise all coefficient blocks except those in the last row and column will be square and have equal dimensions, and the dimensions of the blocks in the last row and column will be decreased so that the total number of rows and columns in the block configuration equals that of the original matrix of coefficients. In general the blocks of right-hand side vectors will be rectangular. The number of rows in a right-hand side block will be the same as the number of rows in the coefficient blocks in the same block row. The number of columns in each right-hand side block is determined by the size of the problem and the memory available for storing blocks.

The block dimensions are computed using the following variable definitions:

NBRC is the number of rows and columns in a coefficient block.

NRHB is the number of columns in a right-hand side vector block.

NRHV is the number of right-hand side vectors.

LBC is the number of complex numbers available in blank COMMON.

At times there will be in memory three coefficient blocks of dimension NBRC x NBRC, one right-hand side block of dimension NBRC x NRHB, one column of working storage, and one column of pivots. The block of right-hand side vectors and the two columns, each of length NBRC, constitute the fourth block. The first estimate for NBRC follows from the possibility of four blocks in memory, and the condition that the fourth block will be no larger than the first three.

$$4 \times NBRC^2 = LBC$$

where LBC is the number of complex numbers that can be contained in blank COMMON. The number of columns in a right-hand side block is determined by the conditions that (1) NRHB is at most NRHV, and (2) the fourth block in the preceding estimate includes one column for pivot indices and one column for working storage.

$$NRHB = \min(NBRC\text{-}2, NRHV).$$

The program reads columns of the coefficient and right-hand-side matrices into the fourth block while storing the system in block configuration, therefore, this block must be at least large enough to contain one column of length NMRC. Thus, the number of columns in the fourth block is

$$NBK4 = \max(NRHV + 2, \frac{NMRC + NBRC - 1}{NBRC}).$$

Then the equation

$$3(NBRC)^2 + BBK4 x NBRC = LBC$$

is solved to determine the number of rows and columns in a coefficient block.

After the sizes of the blocks have been computed, the system of equations is stored in block configuration on direct access files. Then the blocked system of equations is solved by Gaussian elimination. The block elimination procedure follows the same steps as conventional Gaussian elimination for linear equations.[3,6]

ARRANGEMENT OF THE PROGRAM OCSOLVE

The program OCSOLVE consists of three principal subroutines, three secondary subroutines, and several user-provided subroutines. The principal subroutines are

- STORAB, which reads the matrix of coefficients and the matrix of right-hand side vectors and stores them in block configuration;
- BLKSLV, which performs the block elimination process; and
- UNPACK, which retrieves the solution vectors from the block configuration and writes them on a sequential file.

The secondary subroutines are

- SOLVEC, which solves a linear system of equations with complex coefficients;
- MULT, which multiplies two matrices; and
- BLKSIZ, which computes the sizes of the submatrix blocks.

The remaining user-provided subroutines read and write the data files (see Appendix A).

USER-PROVIDED PARAMETERS AND SUBROUTINES

To determine the size of the blocks, in addition to the sizes of the matrices of coefficients and right-hand side vectors, the program OCSOLVE requires information on the memory available in blank COMMON. Blank COMMON contains the arrays for the blocks and the column of pivots, and thus the size of blank COMMON determines how large the blocks may be. For several machine-dependent parameters, OCSOLVE needs the type of computer being used. The program also needs the logical file name assigned to the input and output files. The following FORTRAN statement located at the beginning of the source code of the program OCSOLVE provides this information:

- PARAMETER(MACHNR=m,LGTHBC=n,DATFIL ='name',NPREC=i)

The integer n is the length of blank COMMON in terms of real words; the length can be calculated by subtracting the number of words of memory occupied by all coding and variables from the total field length. (The variable LBC, which is the number of complex numbers in blank COMMON, is computed from LGTHBC and depends on whether single or double precision complex numbers are being used.) The integer m is a parameter, provided by the user, that indicates the type of computer used, and is given by

COMPUTER	MACHNR
CRAY	m = 1
CDC	m = 2
IBM	m = 3
VAX	m = 4
APOLLO	m = 5

The character string 'name' is the logical name of the file that contains the matrix of coefficients and the right-hand side vectors which are input to OCSOLVE, and the solution vectors which are output from OCSOLVE. The integer i is set by the user to equal either 1 or 2 when implementing single or double precision arithmetic. (When changing from single to double precision, or back, a few other changes must be made; see Appendix B.)

The user-provided subroutines read from and write to the data and solution file. The program OCSOLVE requires a minimum of data on the input data file and requires the data to be in a prescribed order, however, the file may contain more data and the specific format may be varied, as long as the requirements of this section are satisfied. The input data file, FILE 4, may begin with a label; following any label must be a heading that specifies the number of rows and columns of the matrix of right-hand side vectors. Following this first heading are the records containing the columns of the right-hand side vectors, each record containing at most one column. Next is a second heading that specifies the number of rows and columns of the matrix of coefficients (equal to the number of rows of the right-hand side vectors) followed by the columns of the matrix of coefficients, again each record containing at most one column.

The data file is read by three user-provided subroutines. To design these subroutines, one must know the format of the data file provided to OCSOLVE and that of the solution file to be returned. Appendix A gives examples of the user-provided subroutines designed for straight-forward unformatted FORTRAN files and for unformatted NASTRAN files. The subroutines must be designed to be consistent with the format of the data files and also must satisfy certain specifications to be consistent with OCSOLVE. The subroutines and their specifications are

- RDLBL, which rewinds the data file and then reads and discards any information between the beginning of the file and the heading. If there is no information before the heading, this subroutine must at least rewind FILE 4. The subroutine call is

  CALL RDLBL

  The subroutine RDLBL must leave the pointer positioned so that the subroutine RDHDG can read the heading.

- RDHDG, which reads the heading preceding each of the matrices of right-hand side vectors and coefficients. On the first call, before the matrix of right-hand side vectors, it returns NR, the number of rows of a right-hand side vector, and NC, the number of columns of right-hand side vectors. On the second call, before the matrix of coefficients, it returns NR and NC, which have the same value and are equal to the number of rows and columns of the matrix of coefficients. The subroutine call is

  CALL RDHDG(NR,NC)

  The subroutine RDHDG must leave the pointer positioned so that the subroutine RDCOL can read the first column of the matrix that follows.

- RDCOL, which reads one matrix column from the data file. The subroutine call is

  CALL RDCOL(RB2,NMRC)

where RB2 is the array into which the column is read and NMRC is the number of complex numbers to be read. After each of these calls the pointer must be positioned so that the subroutine RDCOL can read the next column of the matrix. After the last column of right-hand side vectors has been read, the pointer must be positioned so that the subroutine RDHDG can read the heading before the matrix of coefficients.

After the system of equations has been solved, the program OCSOLVE produces a sequential file with the results of the solution of the linear equations. This file contains a heading that may contain the number of rows and columns of solution vectors, followed by the columns of solution vectors. Three subroutines must be provided that are designed to produce the output file in the format desired by the user and to conform to the general specifications that follow. The subroutine WTHDG is called once, and then for each solution vector the subroutine WTCOL is called once. After all the solution vectors have been written, the subroutine WTEND is called once to allow the solution file to be closed and rewound.

- WTHDG, which writes a heading on the output file, FILE 4. The heading may include NR and NC as defined above. If NR and NC are not needed by the program receiving the solution file, the heading need not be written on the solution file, but a subroutine WTHDG which at least rewinds FILE 4 must be provided. The subroutine call is

  ```
  CALL WTHDG(NR,NC)
  ```

- WTCOL, which writes one matrix column to the output file. The subroutine call is

  ```
  CALL WTCOL(AK,LSG2)
  ```

  where AK is the array from which the column is written, and LSG2 is the number of complex numbers to be written.

- WTEND, which writes an end-of-file mark and rewinds the output file. The subroutine call is

  ```
  CALL WTEND
  ```

DISCUSSION

The block solver OCSOLVE has been developed to solve large, full, unsymmetric systems of complex linear equations. It has been used as an intermediate routine in an analysis in which it received data from a sequential NASTRAN output file and returned the solution in a sequential NASTRAN input file. Block solvers have been developed previously, but the new features of this solver are:

- It determines the dimensions of the blocks of coefficients and right-hand side vectors.
- It stores the coefficient matrix and right-hand side vectors in blocked form.
- It avoids the need for extra equations by providing for blocks of different sizes.
- It solves with one call a linear system of any order having any number of right-hand sides.
- It stores data from a sequential file in the blocked configuration.

OCSOLVE reduced the elapsed time in a dedicated environment on the CDC Cyber 176 computer for the solution of a system of order 500 from the 41 wall clock minutes taken by the equation solver in NASTRAN, which was not optimized for full, complex, unsymmetric systems of equations, to approximately 2 minutes.

A linear equation solver is used in OCSOLVE to factor the diagonal blocks and reduce the off-diagonal blocks. This solver uses partial pivoting to reduce roundoff error and to avoid failure if a zero appears on the diagonal; however, the search for the largest pivot is confined to the block in memory, and it is possible that, for some column, all pivot candidates in that block would be inadequate. The block elimination process could be modified to extend the pivot search to the other blocks in the column of blocks, but the modification would be expensive to develop and expensive to run. Also the equation solver used does not equilibrate the system. If equilibration is desired, the subroutine SOLVEC could be modified or replaced with a solver that does equilibrate the system. It is expected that, for many systems of linear equations arising from physical problems, the coefficients would be of similar orders of magnitude, and there would be large entries on or near the diagonal. Thus there would be no difficulties due to the incomplete partial pivoting or the lack of equilibration. For other systems of linear equations, such as the systems with blocks of zeros that arise when the boundary integral equation method is applied to piecewise homogeneous media, these assumptions may not be valid[5] and appropriate modifications must be made or a different block solver used.

APPENDIX A. EXAMPLES OF THE USER-PROVIDED SUBROUTINES

This appendix provides two sets of subroutines for reading the data file that contains the matrix of the right-hand side vectors and the matrix of coefficients for the system of linear equations to be solved and then writes the solution matrix on the same file. The first set can be used with simple unformatted FORTRAN files and the second with unformatted NASTRAN OUTPUT2 and INPUTT2 files. In either case FILE 4 is opened by the statement

OPEN(4,FILE=DATFIL,FORM='UNFORMATTED'),

where the character string DATFIL has been set by a FORTRAN PARAMETER statement (see Appendix B).

Set 1. Unformatted FORTRAN Files

Subroutines from this set can be used to read unformatted FORTRAN data files which contain exactly the following records:

One record containing in two integer words the number of rows and columns of the right-hand side vectors,

One record for each column of the right-hand side vector,

One record containing in two integer words the number of rows and columns of the coefficient matrix, and

One record for each column of the coefficient matrix.

Subroutines from this set can also be used to produce an unformatted solution file which contains the number of rows and columns of the solution vectors in the first record and one column of the solution matrix in each of the following records. The subroutines in Set 1 are

```
SUBROUTINE RDLBL
REWIND 4
RETURN
END
```

```
      SUBROUTINE RDHDG(NR,NC)
      READ(4) NR,NC
      RETURN
      END

      SUBROUTINE RDCOL(NMRC,RB)
      COMPLEX RB(*)
      READ(4) (RB(I), I = 1,NMRC)
      RETURN
      END

      SUBROUTINE WRTHDG(NR,NC)
      REWIND 4
      WRITE(4) NR,NC
      RETURN
      END

      SUBROUTINE WTCOL(AK,LSG2)
      COMPLEX AK(*)
      WRITE(4) (AK(I),I = 1,LSG2)
      RETURN
      END

      SUBROUTINE WTEND
      ENDFILE 4
      REWIND 4
      RETURN
```

Set 2. NASTRAN OUTPUT2 and INPUTT2 Files

Subroutines from this set can be used to read the unformatted NASTRAN files produced by OUTPUT2. The subroutines RDLBL, RDHDG, and RDCOL are provided to read the label, heading, and columns of a NASTRAN unformatted OUTPUT2 file. Similarly the subroutines WTHDG and WTCOL are provided to write an unformatted NASTRAN INPUTT2 file[7]. The subroutines in Set 2 are

```
      SUBROUTINE RDLBL
      REWIND 4
      DO 100 I = 1,8
  100 READ(4)
      RETURN
      END

       SUBROUTINE RDHDG(NR,NC)
       COMMON /HDG/ N1,N2,N3,KNT1,KNT2,KNT3,NCL,NRW,IFORM,ITYPE,NZW,
       1          IDENS,IFLG,NAME(2)
```

```
C   THE INPUT AND OUTPUT FILES ARE STANDARD NASTRAN UT1 FILES, WHERE
C     THE INPUT FILE HAS A LABEL AND THE OUTPUT FILE DOES NOT.
C     THE SAME FILE (TAPE4=UT1) IS USED FOR BOTH INPUT AND OUTPUT.
C     ON INPUT, UT1 CONTAINS THE RHS MATRIX AND THE COEFFICIENT
C     MATRIX IN THAT ORDER.  ON OUTPUT, UT1 CONTAINS THE SOLUTION
C     MATRIX.
C
C   THE NASTRAN DMAP INSTRUCTIONS TO INTERFACE WITH 'OCSOLVE' ARE
C       OUTPUT2 RHS,COEF,,,//-1 $  TO GENERATE THE MATRICES
C                   AND
C       INPUTT2 /SOLN,,,, $  TO READ THE SOLUTION

      READ(4) N1
      READ(4) NAME
      READ(4) KNT1

      READ(4) N2
      READ(4) KF,NCL,NRW,IFORM,ITYPE,NZW,IDENS,IFLG
      READ(4) KNT2

      READ(4) N3
      READ(4) NAME
      READ(4) KNT3

      NR = NRW
      NC = NCL
      RETURN
      END

      SUBROUTINE WRTHDG
      COMMON /HDG/ N1,N2,N3,KNT1,KNT2,KNT3,NCL,NRW,IFORM,ITYPE,NZW,
     1            IDENS,IFLG,NAME(2)

      REWIND 4

      WRITE(4) N1
      WRITE(4) NAME
      WRITE(4) KNT1

      KFRST = 201
      WRITE(4) N2
      WRITE(4) KFRST,NCL,NRW,IFORM,ITYPE,NZW,IDENS,IFLG
      WRITE(4) KNT2

      WRITE(4) N3
      WRITE(4) NAME
      WRITE(4) KNT3

      RETURN
      END

      SUBROUTINE RDCOL(NMRC,RB)
```

```
 COMPLEX RB(*)
 READ(4) IB
 READ(4) (RB(I), I = 1,NMRC)
 READ(4) KNT
 RETURN
 END

SUBROUTINE WTCOL(AK,LSG2)
COMPLEX AK(*)
COMMON /KOUNT/ KNT
NMRC2 = 2*LSG2
WRITE(4) NMRC2
WRITE(4) (AK(I),I = 1,LSG2)
WRITE(4) KNT
KNT = KNT-1
RETURN
END

SUBROUTINE WTEND
KNT = 0
WRITE(4) KNT
ENDFILE 4
REWIND 4
RETURN
```

APPENDIX B. USING OCSOLVE

The program OCSOLVE must be prepared by setting the user-provided parameters and including the user-provided subroutines from one of the two examples in Appendix A, or by providing routines that satisfy the requirements of Appendix A. Specifically, the following steps must be taken:

1. Provide the parameters on the FORTRAN PARAMETER statement located at the beginning of the source code for the program OCSOLVE.

```
PARAMETER(MACHNR=m,LGTHBC=n,DATFIL='name',NPREC=i)
```

- n is the length of blank COMMON in terms of real words; the length can be calculated by subtracting the number of words of memory occupied by all coding and variables from the total field length.

- m is a parameter that indicates the type of computer used, and is given by

COMPUTER	MACHNR
CRAY	m = 1
CDC	m = 2
IBM	m = 3
VAX	m = 4
APOLLO	m = 5

- 'name' is a character string that is is the logical name of the file that contains the matrix of coefficients and the right-hand side vectors that are input to OCSOLVE and contains the solution vectors that are output from OCSOLVE.

- i is set by the user to either 1 or 2 when implementing single or double precision arithmetic. (Also see additional steps given below for changing to double precision.)

2. Provide the subroutines RDLBL, RDHDG, and RDCOL to read the data file (see Appendix A)

3. Provide the subroutines WTHDG, WTCOL, and WTEND to write the solution file (see Appendix A)

To use OCSOLVE the user merely provides the necessary data and input files and then executes OCSOLVE. The specific steps are

1. Provide a data file with the logical name specified by the preceding PARAMETER statement

2. Execute OCSOLVE

3. Get the solution from the file that has the same logical name as the input file.

If the program OCSOLVE encounters a singular diagonal block, it will stop with the message:

SINGULARITY IN KNN IN POSITION n

where n is the number of the equation for which the solver could not obtain a nonzero pivot.

To change from single to double precisicn complex arithmetic, the parameter i on the PARAMETER statement must be set to 2, and all TYPE COMPLEX declaration statements must be changed to TYPE COMPLEX*16 statements.

REFERENCES

1. Everstine, G.C., F.M. Henderson, E.A. Schroeder, and R.R. Lipman, "A General Low Frequency Acoustic Radiation Capability for NASTRAN," *Fourteenth NASTRAN Users' Colloquium*, NASA CP-2419, National Aeronautical and Space Admin., Washington, D.C., pp.293-310 (1986).

2. Everstine, G.C., F.M. Henderson, and L.S. Schuetz, "Coupled NASTRAN/Boundary Element Formulation for Acoustic Scattering," *Fifteenth NASTRAN Users' Colloquium*, NASA CP-2481, National Aeronautical and Space Admin., Washington, D.C., pp.250-265 (1987).

3. Cantin, G., "An Equation Solver of Very Large Capacity," *International Journal for Numerical Methods in Engineering*, Vol. 3, pp. 379-388 (1971).

4. Hofmeister, L.D., "An Out-of-Core Equation Solver for Full, Unsymmetric Matrices," *International Journal for Numerical Methods in Engineering*, Vol. 12, pp. 721-731 (1978).

5. Crotty, J.M., "A Block Equation Solver for Large Unsymmetric Matrices Arising in the Boundary Integral Equation Method," *International Journal for Numerical Methods in Engineering*, Vol. 18, pp. 997-1017 (1982).

6. Vandergraft, J.S., *Introduction to Numerical Computations*, Second Edition, Academic Press, New York (1983).

7. *The NASTRAN User's Manual*, NASA SP-222(06), Washington, D.C. (Sep 1983).

INITIAL DISTRIBUTION

Copies	
3	CONR
	1 432 (A. Kushner)
	1 432 (A. Tucker)
	1 Library
1	CNO/OP-21T3 (A. Bisson)
5	NRL
	1 2312 (R. Perlut)
	1 5134 (L. Schuetz)
	1 5134 (R. Menton)
	1 8233 (L. Turner)
	1 Library
1	NRL (Orlando)/Library
2	NORDA
	1 251 (K. Kwork)
	1 Library
1	Naval Academy/Library
2	NAVPGSCOL
	1 G. Cantin (M. E. Dept.)
	1 Library
1	NAVSSES (Phila.)/Library
1	NAVWPNCEN/Library
1	NADC/Library
1	NCEL/Library
1	NCSC/Library
3	NOSC
	1 701 (H. Schenck)
	1 733 (E. McDaid)
	1 Library
3	NSWC (WO)
	1 R14 (H. Huang)
	1 R31 (G. Gaunaurd)
	1 Library
1	NSWC (DL)/Library
2	NUSC (NL)
	1 44 (A. Carlson)
	1 Library
1	NUSC (NPT)/Library
1	NWC/Library
1	Charleston NSY/Library
1	Long Beach NSY/Library
1	Mare Island NSY/Library
1	Norfolk NSY/Library
1	Pearl Harbor NSY/Library
1	Philadelphia NSY/Library
1	Portsmouth NSY/Library
1	Puget Sound NSY/Library
17	NAVSEA
	1 05R
	1 05R26 (J. Gagorik)
	1 08
	1 502
	1 55N
	1 55W33
	1 55Y
	1 55Y2 (R. Provencher)
	1 55Y22 (D. Nichols)
	1 55Y3 (R. Chiu)
	1 55Y32
	1 921N
	1 PMS 39
	1 PMS 393 (J. Denham)
	1 PMS 394 (M. Montrol)
	1 PMS 396 (M. Bourkland)
	1 Library
12	DTIC
1	Alcoa Defense Systems/ J. Bock
1	ARL/JHU/Library

1 ARL/Penn State U/Library

1 Adv. Tech. & Res./Library

2 Bettis
 1 E. Zanoni
 1 Library

1 Boeing Computer Services
 (Vienna, Va)/ S. Chander

3 Bolt, Beranek, & Newman
 Laboratories/(New London)
 1 R. Haberman
 1 H. Allik
 1 N. Martin

1 Butler Analyses

1 Cambridge Acoust. Assoc./
 M. Junger

1 Cambridge Collaborative

1 Catholic Univ./H. Uberall

1 Colorado Univ./T. Geers

1 CSIRO, Mount Waverly, Victoria,
 Australia/J. Crotty

1 Donanco, Inc/D. Ross

4 General Dynamics (EB)
 1 443 (J. Wilder)
 1 443 (S. Gordon)
 1 471 (J. Dimitri)
 1 Library

1 Georgia Tech Univ./
 A. Pierce (M. E. Dept.)

1 J. G. Engr. Res. Assoc.

1 JPL/M. El-Raheb

2 Knolls Atomic Power Lab.
 1 M. Pollack
 1 Library

1 Lockheed Palo Alto Res.
 Lab/J. DeRuntz

1 Magnavox Elect. Sys./
 D. Scheiber

1 Martin-Marietta/Baltimore

1 Mechanics Research/
 L. Hofmeister

3 Newport News Shipbuilding
 1 E-12 (T. Heldreth)
 1 E-12 (R. Moses)
 1 Library

2 NKF Engineering Assoc.
 1 M. Pakstys
 1 R. Miller

1 Polytechnic Inst. of NY/
 J. Klosner

2 RpkCorp.
 1 P. Pamidi
 1 K. Brown

2 Weidlinger Assoc.
 1 M. Baron
 1 J. McCormick

CENTER DISTRIBUTION

Copies	Code	Name
1	011	
1	0113	
1	0114	
1	0117	
1	0118	
1	06	
1	12	
1	121	
1	1211	O. Ritter
1	14	
1	15	
1	1504	
1	1508	
1	1522	M. Wilson
1	154	
1	1542	Y-T. Lee
1	1544	

1	16	
1	17	
1	1702	
1	1703	
1	172	
1	1720	R. F. Jones, Jr.
1	1720.1	
1	1720.1	D. Lesar
1	1720.1	C. McNamara
1	1720.1	M. Costello
1	173	
1	174	
1	175	
1	177	
1	18	
1	1802	
1	1805	
1	1808	
1	182	
1	184	
1	1843	
1	1844	
1	1844	G. Everstine
1	1844	S. Hambric
1	1844	R. Lipman
1	1844	M. Marcus
1	1844	A. Quezon
5	1844	E. Schroeder
1	185	
1	187	
1	189	
1	19	
1	1901	
1	1904	
1	1905.1	
1	1905.2	
1	1905.3	
1	1905.4	
1	1906	
1	1908	
1	191	
1	192	
1	193	
1	194	
1	1942	
1	1942	Y-F. Hwang
1	196	
1	1960.1	M. Rumerman
1	1961	
1	1961	Y-N. Liu
1	1961	E. Unver
1	1961	R. Ratcliffe
1	1962	
1	1965	
1	1965	D. Photiadis
1	27	
1	2704	
1	274	
1	274	J. Dickey
1	274	D. Goldsmith
1	2742	
1	2742	H. Neilson
1	28	
1	29	
10	5211.1	Reports Control
1	522.1	TIC (C)
1	522.2	TIC (A)
1	93	

DTNSRDC ISSUES THREE TYPES OF REPORTS:

1. **DTNSRDC reports, a formal series,** contain information of permanent technical value. They carry a consecutive numerical identification regardless of their classification or the originating department.

2. **Departmental reports, a semiformal series,** contain information of a preliminary, temporary, or proprietary nature or of limited interest or significance. They carry a departmental alphanumerical identification.

3. **Technical memoranda, an informal series,** contain technical documentation of limited use and interest. They are primarily working papers intended for internal use. They carry an identifying number which indicates their type and the numerical code of the originating department. Any distribution outside DTNSRDC must be approved by the head of the originating department on a case-by-case basis.