SPARSPAK: Waterloo Sparse Matrix Package
User's Guide for SPARSPAK-A

Eleanor Chu
Alan George
Joseph Liu
Esmond Ng

Department of Computer Science
University of Waterloo
Waterloo, Ontario, CANADA

Research Report CS-84-36

November 1984

Table of Contents

SPARSPAK: Waterloo Sparse Matrix Package

User's Guide for SPARSPAK-A

A collection of modules for solving sparse systems of linear equations

Eleanor Chu[†]

Alan George[†]

Joseph Liu[††]

Esmond Ng[†]

Research Report CS-84-36

ABSTRACT

This document describes the structure and use of SPARSPAK-A, a sparse linear equations package which is designed to efficiently solve large sparse systems of linear equations. Computer programs for solving sparse systems of linear equations typically involve fairly complicated data structures and storage management. In many cases the user of such programs simply wants to solve his problem, and should not have to understand how the storage management is done, or how the matrix components are actually stored. One of the attractive features of this package is that it effectively insulates the user from these considerations, while still allowing the package to be used in a variety of ways. Another important feature of the package is the provision of a *variety* of methods for solving sparse systems, along with convenient means by which the best method for a given problem can be selected.

---

† Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada.

†† Department of Computer Science, York University, Downsview, Ontario, Canada.

IMPORTANT NOTE

The error estimate provided by the subroutine *ERESTi* (see Section 2.6) is based on estimates of the condition number of the coefficient matrix and the error in the triangular factorization. Experience has shown that the error estimate is very good in most cases. It may occasionally overestimate the actual relative error for some ill-conditioned problems or badly scaled problems. Also, *ERESTi* may not be able to return an error estimate when the estimate of the condition number of the coefficient matrix or the estimate of the error in the triangular factors is large, even though the solution may be computed accurately.

We would appreciate receiving any comments and feedback the user may have in using the error estimators for practical problems. Such comments and feedback are important and useful for improving the error estimators. Please send comments and feedback to

Dr. Alan George
Department of Computer Science
University of Waterloo
Waterloo, Ontario
CANADA N2L 3G1

Telephone: 519-885-1211, ext 3473 (Dr. Alan George)

1. Introduction and basic structure of SPARSPAK-A

SPARSPAK-A offers a *collection* of *methods* for solving sparse systems of linear equations

$$Ax = b \quad ,$$

where A is an $n \times n$ nonsingular matrix, and x and b are vectors of length n. We assume the user is aware of the basic issues involved in solving sparse matrix equations, and the basic facts about solving systems of linear equations using Gaussian elimination. For a discussion on the initial design of this package, see [5].

For all the methods provided in SPARSPAK-A, the user and the package interact to solve the matrix problem through the following basic steps:

Step 1. The user supplies the nonzero structure of A to the package using a set of subroutines described in Section 2.2.

Step 2. The package reorders the original problem (finds a permutation P), and allocates storage for the triangular factorization of $PAP^T=LU$, as described in Section 2.3.

Step 3. The user supplies the numerical values for the matrix A to the package, as described in Section 2.4.

Step 4. The package computes the triangular factors L and U of PAP^T, as described in Section 2.5.

Step 5. The user supplies numerical values for b, as described in Section 2.4. (This step may come before Step 4, and may be intermixed with Step 3.)

Step 6. The package computes the solution x, using L, U, P and b, as described in Section 2.5.

Step 7. The user (optionally) calls a subroutine which supplies an estimate of the relative error in x. The subroutine is described in Section 2.6.

The different *methods* provided in SPARSPAK-A correspond to different algorithms for choosing P (along with appropriate storage methods), and whether or not A is symmetric. When A is symmetric, U is replaced by L^T in the above description, and of course only one of L and L^T is stored.

The user chooses a particular method by calling the appropriate subroutines in Steps 2, 3, 4, 6 and 7. The methods are distinguished by a numerical digit i, $1 \leq i \leq 6$, which is the last character of the subroutine names. The subroutines used in Steps 1 and 5 apply to all the methods. The best method to use depends very much on the particular problem, and the context in which it is being solved, so we cannot provide rigid rules as to which method to use. Some guidelines and considerations regarding the choice of method are given in Section 3.

Restrictions and assumptions:

1. SPARSPAK-A assumes that the nonzero structure of A is *symmetric*. If this is not the case, the package will still work, but if A has highly unsymmetric structure, this may lead to some inefficiencies because the matrix will be treated as though its structure is that of $A+A^T$.

2. SPARSPAK-A assumes that for *any* permutation matrix P, Gaussian elimination applied to PAP^T *without row or column interchanges* yields an acceptably accurate factorization LU. In other words, the package assumes that A can be symmetrically permuted without regard for numerical stability. This is true, for example, when A is symmetric and positive definite, or diagonally dominant. In general one can detect when the assumption above is invalid by computing an estimate of the relative error in the computed solution using the subroutine *ERESTi* (see Section 2.6).

2. Modules of SPARSPAK-A and how to use them

2.1. User mainline program and an example

SPARSPAK-A allocates all of its storage from a single one-dimensional floating-point array[1] which for purposes of discussion we will denote by S. In addition, the user must provide its size *MAXSA*, which is transmitted to the package via a common block *SPAUSR*, (SPARSPAK–A USER), which has four variables.

```
COMMON  /SPAUSR/  MSGLVA, IERRA, MAXSA, NEQNS
```

Here *MSGLVA* is the message level indicator which is used to control the amount of information printed by the package. The second variable *IERRA* is an error code, which the user can examine in his mainline program for possible errors detected by the package. Detailed discussion of the roles of *MSGLVA* and *IERRA* is provided in Section 7. The variable *NEQNS* is the number of equations.

The following program illustrates how one might use SPARSPAK-A. The various subroutines referenced are described in the subsequent parts of this section. The problem solved is a 10×10 symmetric tridiagonal system $Ax=b$ where the diagonal elements of A are all 4, the superdiagonal and subdiagonal elements are all −1, and the entries in the right hand side vector b are all ones.

```
      REAL       S(250), FOUR, ONE
      INTEGER    I, IERRA, MAXSA, MSGLVA, NEQNS
      COMMON     /SPAUSR/  MSGLVA, IERRA, MAXSA, NEQNS
C
      CALL  SPRSPK
      MAXSA = 250
C     ------------------------------------------------
C     INPUT THE MATRIX STRUCTURE.  THE DIAGONAL IS
C     ALWAYS ASSUMED TO BE NONZERO, AND SINCE THE
C     MATRIX IS SYMMETRIC, SPARSPAK-A ONLY NEEDS TO
C     KNOW THAT THE SUBDIAGONAL ELEMENTS ARE NONZERO.
C     ------------------------------------------------
      CALL  IJBEGN
      DO  100  I = 2, 10
          CALL INIJ ( I, I-1, S )
  100 CONTINUE
      CALL  IJEND ( S )
C     ---------------------------------------
C     FIND THE ORDERING AND ALLOCATE STORAGE.
C     ---------------------------------------
      CALL  ORDRA1 ( S )
C     ------------------------------------------------
C     INPUT THE NUMERICAL VALUES FOR A AND B.  SINCE
C     THE MATRIX IS SYMMETRIC, ONLY THE LOWER TRIANGLE
C     AND THE DIAGONAL ARE INPUT.
```

(1) Declared either REAL or DOUBLE PRECISION, depending on the version of SPARSPAK-A that is available. The examples in this manual assume a single precision version is being used.

```
C          ------------------------------------------------
           FOUR = 4.0E0
           ONE  = 1.0E0
           DO  200  I = 1, 10
               IF  ( I .GT. 1 )
     *             CALL  INAIJ1 ( I, I-1, (-ONE), S )
               CALL  INAIJ1 ( I, I, FOUR, S )
               CALL  INBI ( I, ONE, S )
   200     CONTINUE
C          -----------------
C          SOLVE THE SYSTEM.
C          -----------------
           CALL  SOLVE1 ( S )
C          ----------------------------------------
C          PRINT THE SOLUTION, FOUND IN THE FIRST TEN
C          LOCATIONS OF THE WORKING STORAGE ARRAY S.
C          ----------------------------------------
           WRITE (6,11)  (S(I),I=1,10)
    11     FORMAT ( / 10H SOLUTION  / (5F12.5) )
C          --------------------------------------
C          PRINT STATISTICS GATHERED BY SPARSPAK-A.
C          --------------------------------------
           CALL  STATSA
C
           STOP
       END
```

Note: If the SPARSPAK-A available to you is a double precision version, the REAL declaration in this example should be changed to DOUBLE PRECISION.

The module *SPRSPK* must be called before any part of the package is used. Its role is to initialize some system parameters (e.g. the logical unit numbers for output files), and to set default values for options (e.g. the message level indicator), and to initialize the timing routine. The routine needs only to be called once in the user program, and the FORTRAN statement is simply

CALL SPRSPK

Note that the only variable in the common block *SPAUSR* that *must* be explicitly assigned a value by the user is *MAXSA*.

It is assumed that the subroutines which comprise SPARSPAK-A have been compiled into a *library*, and that the user can reference them from his FORTRAN program just as he references the standard FORTRAN library subroutines, such as *SIN*, *COS*, etc. Normally, a user will use only a small fraction of the subroutines provided in SPARSPAK-A.

Warning:

The modules of SPARSPAK-A communicate with each other through labelled common blocks whose names are *SPKSYS*, *SPAUSR*, *SPACON*, *SPAMAP*, and *SPADTA*. Thus, the user must not use labelled common blocks with these names in his program.

If these common block names cause conflicts in your program or at your computer installation, it is possible to have the package distributed with these common blocks having *specifically requested* labels. These names should be specified when the package is acquired.

2.2. Modules for input of the matrix structure

SPARSPAK-A has to know the matrix structure before it can determine an appropriate ordering for the system. We now describe the group of routines which provide a variety of ways through which the user can inform the package where the nonzero entries are; that is, those indices (i,j) for which the (i,j)-th element of A is nonzero. Before any of these input routines is called, the user must execute an initialization routine called *IJBEGN*, which tells the package that the structure of a new matrix problem is about to be input.

CALL IJBEGN

(a) Input of a nonzero location.

To tell the package that the (i,j)-th element of A is nonzero, the user simply executes the statement

CALL INIJ (I, J, S)

where I and J are respectively the row and column indices of the nonzero, and S is the working storage array declared by the user for use by the package.

In this example,

```
.
.
.
I = 4
J = 3
CALL   INIJ ( I, J, S )
.
.
.
```

the package will record a logical nonzero in positions (4,3) and (3,4) of the matrix.

(b) Input of the structure of a row, or part of a row.

When the structure of a row or part of a row is available, it is more efficient to use the routine *INROW*. The statement to use is

CALL INROW (I, NIR, IR, S)

where I denotes the index of the row under consideration, *IR* is an array containing the column indices of some or all of the nonzeroes in the I-th row, *NIR* is the number of subscripts in *IR*, and S is the user-declared working storage.

For example, in

```
.
.
.
I = 5
IR(1) = 2
IR(2) = 7
IR(3) = 5
CALL  INROW ( I, 3, IR, S )
.
.
.
```

the package is informed of nonzeroes in locations (2,5), (5,2), (5,5), (5,7) and (7,5) of the matrix. Note that the column indices in the array *IR* can be in arbitrary order, and the rows can be input in any order.

(c) Input of a submatrix structure.

To provide greater flexibility, the package allows the user to input the structure of a submatrix. The calling statement is

CALL INIJIJ (NIJ, II, JJ, S)

where *NIJ* is the number of input index pairs, and *II* and *JJ* are the arrays containing the row and column indices.

The following example

```
.
.
.
II(1) = 1
JJ(1) = 1
II(2) = 1
JJ(2) = 3
II(3) = 2
JJ(3) = 3
CALL  INIJIJ ( 3, II, JJ, S )
.
.
.
```

informs the package that there are nonzeroes in locations (1,1), (1,3), (3,1), (2,3) and (3,2).

(d) Input of a full submatrix structure.

The structure of an entire matrix is completely specified if all the full submatrices are given. In applications where they are readily available, the routine *INCLQ* is useful. Its calling sequence is

CALL INCLQ (NCLQ, CLQ, S)

where *NCLQ* is the size of the submatrix and *CLQ* is an array containing the column (or row) indices of the submatrix.

Thus, to inform the package that the submatrix corresponding to indices 1, 3, 5 and 6 is full, we execute

```
.
.
.
CLQ(1) = 1
CLQ(2) = 3
CLQ(3) = 5
CLQ(4) = 6
CALL   INCLQ ( 4, CLQ, S )
.
.
.
```

The type of structure input routine to use depends on how the user obtains the matrix structure. Anyway, the one or ones that best suit the application can be selected; SPARSPAK-A allows *mixed use* of the routines in inputting a matrix structure. The package automatically removes duplications so the user does not have to worry about inputting duplicated index pairs.

```
.
.
.
CLQ(1) = 1
CLQ(2) = 2
CLQ(3) = 5
CALL   INCLQ ( 3, CLQ, S )
IR(1) = 2
IR(2) = 4
IR(3) = 5
CALL   INROW ( 4, 3, IR, S )
CALL   INIJ ( 1, 3, S )
.
.
.
```

The code above would input the matrix structure

$$\begin{bmatrix} * & * & * & & * \\ * & * & & * & * \\ * & & * & & \\ & * & & * & * \\ * & * & & * & * \end{bmatrix}$$

into the package. Note that the diagonal elements of the input matrix are assumed to be nonzero (see notes below).

When all pairs have been input, using one or a combination of the input routines, the user is required to tell SPARSPAK-A explicitly that structure input is complete by calling the routine *IJEND*. The statement to use is

CALL IJEND (S)

and its purpose is to transform the data from the format used during the recording phase to the standard format used by the later phases. The user does not have to concern himself with this representation or transformation.

Important Notes:

(a) SPARSPAK-A assumes that the value of *NEQNS* (the number of equations) is equal to the maximum column (or row) index supplied by the routines which transmit the (i,j) pairs to the package. Thus, it is imperative that the user supplies at least one (i,j) pair for which i or j is equal to *NEQNS*. The routine *IJEND* assigns the value of *NEQNS* found by the package to the corresponding variable in the common block *SPAUSR*.

(b) SPARSPAK-A assumes that the diagonal elements of the coefficient matrix are nonzero.

Common Errors:

The most common cause of error during matrix structure input is insufficient working storage. If we denote the number of off-diagonal nonzeroes in the matrix by *OFFDA*, then the minimum amount of storage necessary to successfully input the structure is given by

$$OFFDA + 2 \times NEQNS + 1 .$$

Of course sometimes the user does not know the value of *OFFDA*, and may guess too low. SPARSPAK-A will still accept and count the (i,j) pairs, even after running out of storage, and the user can obtain an upper bound for *OFFDA* by calling the module *STATSA*, described in Section 7, after all pairs have been input. (The number reported may be unnecessarily large because duplicate input pairs may not now be detected, and thus may be counted more than once by the package.)

For a complete list of errors which may be generated by the structure input modules, see Section 7.3.1.

2.3. Modules for ordering and storage allocation

With an internal representation of the nonzero structure of the matrix A available, SPARSPAK-A is ready to reorder the matrix problem. This is initiated by calling an ordering routine, whose name always has the form *ORDRxi*. Here i is a numerical digit between 1 and 6 that signifies the storage method. The character x can take values A or B, which denotes one of two ordering strategies tailored for storage method i.

Executing the statement

CALL ORDRA1 (S)

will imply the use of storage method 1 and the first ordering algorithm for this method. See Section 3 for a discussion of the various methods provided, and some guidance on which one to

use. Section 8 contains a list of ordering strategies provided by the package. The routine *ORDRxi* not only determines an appropriate ordering for the storage method, it sets up the data structure for the reordered matrix problem. The package is now ready for numerical input.

Common Errors:

Just as in the structure input phase, the most common cause of premature termination of the *ORDRxi* module is insufficient working storage. As mentioned above, this module performs two functions: *ordering* and *storage allocation*. The ordering step determines the permutation P, and the allocation step sets up the appropriate data structures to store the triangular factors L and U of the permuted matrix PAP^T.

In general, the ordering and allocation subroutines require different amounts of storage. Furthermore, their storage requirements are often unpredictable, because the number of data structure pointers, and the number of nonzeroes in the factors L and U, are not known until the subroutines have been executed.

Thus, the interface module *ORDRxi* may terminate in several distinctly different ways.

(a) There was not enough storage to execute the ordering subroutine.

(b) The ordering was successfully obtained, but there was insufficient storage to initiate execution of the data structure set-up (storage allocation) subroutine.

(c) The data structure set-up subroutine was executed, and the amount of storage required for the data structure pointers etc. was determined, but there was insufficient storage for these pointers.

(d) The data structure was successfully generated, but there is insufficient storage for the actual numerical values, so the next step (input of the numerical values) cannot be executed.

(e) *ORDRxi* was successfully executed, and there is sufficient storage to proceed to the next step.

If any of the above conditions occurs, the user may execute *SAVEA*, and re-initiate the computation after adjusting his storage declarations (either up or down) and executing *RSTRTA*[(2)]. If (a) or (b) occurs, information is supplied indicating the minimum value of *MAXSA* needed so that (c), (d) or (e) will occur upon re-execution. If (c) occurs, the minimum value of *MAXSA* needed for (d) and (e) is provided.

When (c) or (d) occurs, after executing *SAVEA*, adjusting the storage declaration, then executing *RSTRTA*, one must again call *ORDRxi*. However, the interface will detect that the ordering and/or storage allocation have already been performed, and will skip that part of the computation. Note that if a user is simply using SPARSPAK-A to select a particular method, (c) may be an acceptable termination state. (See Example 6 in Section 9.)

---

(2) See Section 4 for details on how to use *SAVEA* and *RSTRTA*, and Examples 4, 5 and 6 in Section 9.

2.4. Modules for inputting numerical values

The modules in this group are similar to those for inputting the matrix structure. They provide a means of transmitting the actual numerical values of the matrix problem to SPARSPAK-A. Since the data structures for different storage methods are different, the package must have a different matrix input subroutine for each method. For the user's convenience, SPARSPAK-A uses the same set of subroutine names for all the methods, except for the last digit which distinguishes the method, and the parameter lists for all the methods are the same.

Important Note:

The elements of A and b transmitted to SPARSPAK-A by these routines are either *single or double precision floating-point numbers,* depending on the version of SPARSPAK-A being used. The examples in this manual assume a single precision version of the package is being used.

There are three ways of passing the numerical values to SPARSPAK-A. In all of them, indices passed to the package always refer to those of the *original given problem.* The user need not be concerned about the various permutations to the problem which may have occurred during the ordering step.

When any of the three numerical input routines is first called, the storage used for storing the numerical values is initialized to zero.

(a) Input of a single nonzero component.

The subroutine *INAIJi* is provided for this purpose and its calling sequence is

CALL INAIJi (I, J, VALUE, S)

where I and J are the row and column indices, and *VALUE* is the numerical value. The subroutine *INAIJi adds* the quantity *VALUE* to the appropriate current value in storage, rather than making an assignment. This is helpful in situations (e.g. in some finite element applications) where the numerical values are obtained in an incremental fashion.

For example, the execution of

```
      .
      .
      .
CALL  INAIJ2 ( 3, 4, 9.5, S )
CALL  INAIJ2 ( 3, 4, -4.0, S )
      .
      .
      .
```

effectively assigns 5.5 to the (3,4)-th component of A.

(b) Input of a row of nonzeroes.

The routine *INROWi* can be used to input the numerical values of a row or part of a row in the matrix. Its calling sequence is similar to that of *INROW*, described on Section 2.2.

CALL INROWi (I, NIR, IR, VALUES, S)

Here the additional parameter *VALUES* is a floating-point array containing the numerical values of the row. Again, the numerical values are added to the current values in storage.

(c) Input of a submatrix.

The routine that allows the input of a submatrix is *INMATi*. Its parameter list corresponds to that of *INIJIJ* with the additional parameter *VALUES* that stores the numerical quantities.

CALL INMATi (NIJ, II, JJ, VALUES, S)

Again, the numerical values in *VALUES* are added to those currently held by the package.

Mixed use of the routines *INAIJi*, *INROWi* and *INMATi* is permitted. Thus, the user is free to use whatever routine is most convenient.

The same convenience is provided in the input of numerical values for the right hand side vector b. SPARSPAK-A includes the routine *INBI* which inputs an entry of the right hand side vector.

CALL INBI (I, VALUE, S)

Here *I* is the index and *VALUE* is the numerical value. Alternatively, the routine *INBIBI* can be used to input a subvector, and its calling sequence is

CALL INBIBI (NI, II, VALUES, S)

where *NI* is the number of input numerical values, and *II* and *VALUES* are vectors containing the indices and numerical values respectively. In both routines, incremental calculation of the numerical values is performed.

In some situations where the entire right hand side vector is available, the user can use the routine *INRHS* which transmits the whole vector to SPARSPAK-A. It has the form

CALL INRHS (RHS, S)

where *RHS* is the vector containing the numerical values.

In all three routines, the numbers provided are added to those currently held by the package, and the use of the routines can be intermixed. (See example in (a) above.) The storage used for the right hand side by SPARSPAK-A is initialized to zero the first time any of them is executed.

Important Notes:

(a) When the matrix A is symmetric, so that method i, with i odd, is used, SPARSPAK-A requires that the elements of the *lower* triangle be provided. Thus, for example, the following statement will cause an error.

CALL INAIJ3 (3, 5, 1.3, S)

(b) The examples which we have given assume that a single precision version of SPARSPAK-A is being used. If the version is in double precision, the numerical values and numerical variables should be declared as double precision. For example:

CALL INAIJ3 (5, 3, 1.3D0, S)

2.5. Modules for numerical solution

The numerical computation of the solution vector is initiated by the FORTRAN statement

CALL SOLVEi (S)

where S is the working storage array for SPARSPAK-A. Again, the last digit i is used to distinguish between solvers for different storage methods.

Internally, the routine *SOLVEi* consists of both the factorization and forward/backward solution steps. If the factorization has been performed in a previous call to *SOLVEi*, SPARSPAK-A will automatically skip the factorization step, and perform the solution step directly. The solution vector is returned in the first *NEQNS* locations of the storage vector S. If *SOLVEi* is called before any right hand side values are input, only the factorization will be performed. The solution returned will be all zeroes. See Examples 3 and 4 in Section 9.

2.6. Modules for estimating the relative error in the computed solution

An estimate of the relative error (using the infinity norm $\|.\|_\infty$[3]) in the computed solution can be obtained by executing the following FORTRAN statement.

CALL ERESTi (RELERR, S)

Here S is the working storage array for SPARSPAK-A and *RELERR* is a variable which will contain the relative error estimate after the subroutine is invoked successfully. The last digit i

(3) $\|w\|_\infty = \max_i |w_i|$.

is used to distinguish between subroutines for different storage methods.

If the problem is too ill-conditioned with respect to the precision of the machine, or unacceptable rounding error in the factorization has occurred, a message will be printed (depending on the value of the message level indicator *MSGLVA*) and *RELERR* will be set to −1.0.

The estimate is based on estimates of the condition number of the coefficient matrix and the error incurred in its factorization [1]. Thus, the estimate is independent of the right hand side and *ERESTi* has to be called only once regardless of the number of right hand sides that are to be solved. Furthermore, *ERESTi* should be called only after the factorization has been performed; that is, after *SOLVEi* has been executed successfully.

3. Some guidelines on selecting a method

We mentioned in Section 1 that there are six basic methods, distinguished by a numerical digit i satisfying $1 \leq i \leq 6$. These six methods can be viewed as grouped into three odd-even pairs; the only distinction between method i (odd) and method $i+1$ is that method i assumes the coefficient matrix A is symmetric, and method $i+1$ assumes A is unsymmetric. Thus, we really only provide three essentially distinct methods, with each one having a symmetric and unsymmetric version. Hence, in this section we will largely confine our remarks to methods 1, 3 and 5; comparative remarks about them will also apply to their unsymmetric analogues, methods 2, 4 and 6.

The basic methods are as follows; the remarks comparing them, and the advice provided, should be regarded as at best tentative. Characteristics of sparse matrices vary a great deal.

Method	*Basic Strategy and References*
1,2	The objective of these methods is to reorder A so it has a small bandwidth or profile [7]. The well-known reverse Cuthill-McKee algorithm is used. For relatively small problems, say $n \leq 200$, they are probably the best overall methods to use.
3,4	The objective of these methods is to reduce storage requirements, but the factorization time will usually be substantially higher than any of the other methods. Their storage requirements will usually be substantially less than methods (1,2) (unless n is very large). The same remark is true about the relative solution times. Thus, these methods are often useful when storage is restricted, and/or when many problems which differ only in the right hand side must be solved (see Section 6). There are two ordering options provided: *ORDRA3* and *ORDRB3* (and similarly for the unsymmetric case). The A option is specifically tailored for "finite element problems", typical of those arising in structural analysis and the numerical solution of partial differential equations [2]. The B option is effective for less specific problems; and uses a refined quotient tree ordering described in [3].
5,6	These methods attempt to find orderings which minimize fill-in, and they exploit all zeroes. Their ordering times are almost always greater than those above, but for moderate-to-large problems the reduced factorization times usually are more than compensatory.

4. Save and restart facilities

SPARSPAK-A provides two subroutines called *SAVEA* and *RSTRTA* which allow the user to stop the calculation at some point, save the results on an external sequential file, and then restart the calculation at exactly that point some time later. To save the results of the computation done thus far, the user executes the statement

CALL SAVEA (K, S)

where K is the FORTRAN logical unit on which the results are to be written, along with other information needed to restart the computation. If execution is then terminated, the state of the computation can be re-established by executing the following statement.

CALL RSTRTA (K, S)

Examples 4, 5 and 6 provided in Section 9 illustrate the use of *SAVEA* and *RSTRTA*.

Note that executing *SAVEA* does not destroy any information; the computation can proceed just as if *SAVEA* were not executed.

When errors occur in a module, the routines *SAVEA* and *RSTRTA* are useful in saving the results of previously successfully executed modules (see Section 7.3 and Example 5 in Section 9).

Another potential use of the *SAVEA* and *RSTRTA* modules is to make the working storage array S available to the user in the middle of a sparse matrix computation. After *SAVEA* has been executed, the working storage array S can be used by some other computation.

Finally, the *SAVEA* and *RSTRTA* modules allow the user to segment the computation into several distinct phases, and thereby reduce the amount of program that must be resident in storage at any given time.

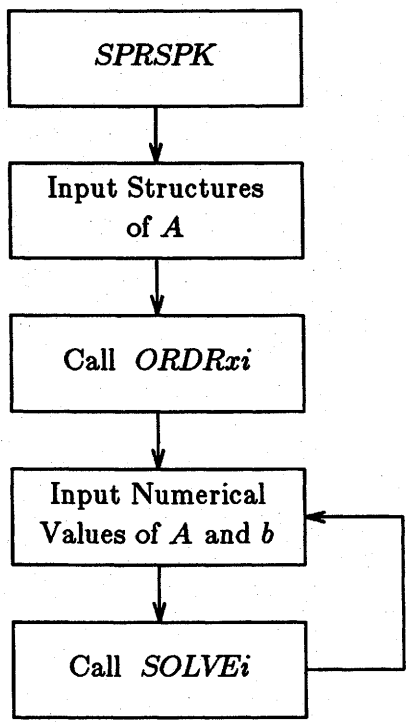
Important Notes:

(a) In the subroutines *SAVEA* and *RSTRTA*, information is either written on or read from the FORTRAN logical unit K using *binary format*.

(b) If the subroutines *SAVEA* and *RSTRTA* are used, then before the user executes his program, he must define a file for the FORTRAN logical unit K using the appropriate system control statement or command (this depends on the environment in which the program is being executed). Furthermore, this file must be preserved by the user for later access by the *RSTRTA* subroutine. Thus, the user must not write to this file.

5. Solving many problems having the same structure

In certain applications, many problems which have the same sparsity structure, but different numerical values, must be solved. In this case, the structure input, ordering, and data structure set-up needs only to be done once. This situation can be accommodated perfectly well by SPARSPAK-A. The control sequence is depicted by the following flowchart.



When the numerical input routines (*INAIJi*, *INBI*, ..., etc.) are first called after *SOLVEi* has been called, this is detected by SPARSPAK-A, and the computer storage used for A and b is initialized to zero.

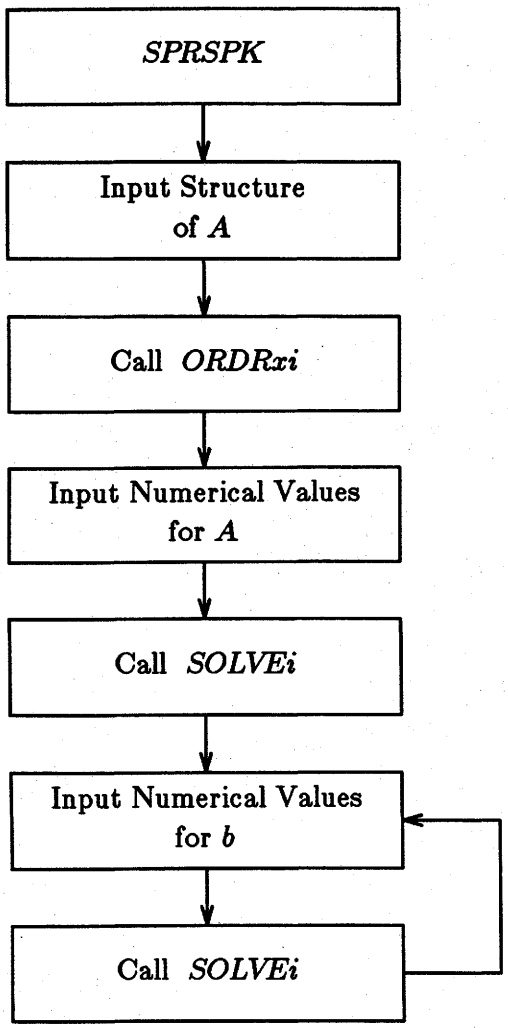
Note that if such problems must be solved over an extended time period (i.e., in different runs), the user can execute *SAVEA* after executing *ORDRxi* and thus avoiding the input of the structure of A and the execution of *ORDRxi* in subsequent equation solutions.

6. Solving many problems which differ only in their right hand side

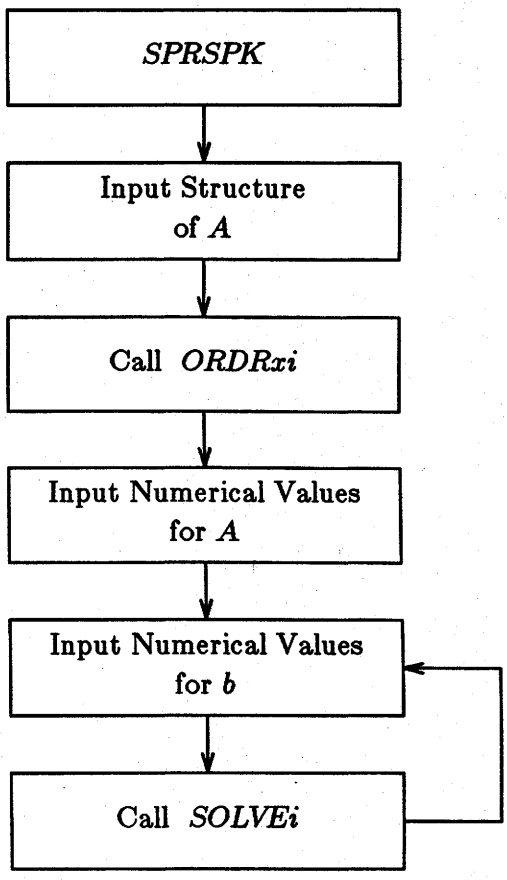
In some applications, numerous problems which differ only in their right hand sides must be solved. In this case, we only want to factor A into LU (or LL^T) once, and use the factors repeatedly in the calculation of x for each different b. Again, SPARSPAK-A can handle this situation in a straightforward manner, as illustrated by the flowcharts below.

When SPARSPAK-A is used as indicated by flowchart (1), the package detects that no right hand side has been provided during the first execution of *SOLVEi*, and only the factorization is performed. In subsequent calls to *SOLVEi*, SPARSPAK-A detects that the factorization has already been performed, and that part of the *SOLVEi* module is bypassed. In flowchart (2), both factorization and solution are performed during the first call to *SOLVEi*, with only the solve part performed in subsequent executions of *SOLVEi*. (See Example 3 in Section 9.)

Note that *SAVEA* can be used after *SOLVEi* has been executed, if the user wants to save the factorization for use in some future calculation.



Flowchart (1)



Flowchart (2)

7. Output from SPARSPAK-A

As noted earlier in Section 2, the user supplies a one-dimensional floating-point array *S*, from which all array storage is allocated. In particular, the interface allocates the first *NEQNS* storage locations in *S* for the solution vector of the linear system (*NEQNS* is the last variable in the common block *SPAUSR*). After all the interface modules for a particular method have been successfully executed, the user can retrieve the solution from these *NEQNS* locations.

In addition to the solution vector, SPARSPAK-A may provide other information about the computation, depending upon the value of *MSGLVA*, whether or not errors occur, and whether or not the module *STATSA* is called. This section discusses these features of SPARSPAK-A.

Note:

SPARSPAK-A writes output to two FORTRAN logical output units, whose numbers are given by *IPRNTS* and *IPRNTE*. The values for these variables are set in the module *SPRSPK* when the package is installed. Standard output requested by the user is printed on unit *IPRNTS*, while any error messages raised by SPARSPAK-A are printed on unit *IPRNTE*. In an interactive environment, *IPRNTE* is usually the user's terminal, while *IPRNTS* is some other output device on which the output of the (hopefully) successful run is recorded. In a batch oriented environment, *IPRNTS* and *IPRNTE* are usually the same. Note that the user and/or the computer installation must ensure that the files associated with *IPRNTS* and *IPRNTE* are available to the user's program before execution begins.

7.1. Message level (*MSGLVA*)

The first variable *MSGLVA* in the common block *SPAUSR* stands for "message level", and governs the amount of information printed by the interface modules. Its default value is two, and for this value a relatively small amount of summary information is printed, indicating the initiation of each phase. When *MSGLVA* is set to one by the user, only fatal error messages are printed; this option could be useful if SPARSPAK-A is being used in the "inner loop" of a large computation, where even summary information would generate excessive output. Increasing the value of *MSGLVA* (up to 4) provides increasingly detailed information about the computation. Note that the module *SPRSPK* sets *MSGLVA* to its default value; if the user wishes *MSGLVA* to be different from two, he must reset it *after SPRSPK* has been called.

In many circumstances, SPARSPAK-A will be embedded in still another "super package" which models phenomena producing sparse matrix problems. Messages printed by SPARSPAK-A may be useless or even confusing to the ultimate users of the super package, or the super package may wish to field the error conditions and perhaps take some corrective action which makes the error messages irrelevant. Thus, *all* printing by SPARSPAK-A can be prevented by setting *MSGLVA* to zero.

To summarize, we have

MSGLVA	amount of output
0	no information is provided.
1	only warnings and errors are printed.
2	warnings, errors and summary are printed.
3	warnings, errors, summary and some statistics are printed.
4	detailed information for debugging purposes.

Warning:

It should be noted that, by setting *MSGLVA* to four, a high volume of output may be generated, since the input data would also be echoed.

7.2. Statistics gathering (*STATSA*)

SPARSPAK-A gathers a number of statistics which the user will find useful if he is comparing various methods, or is going to solve numerous similar problems and wants to adjust his working storage to the minimum necessary. The package has a common block called *SPADTA* containing variables whose values can be printed by executing the following statement.

CALL STATSA

The information printed includes

the number of equations,

the number of off-diagonal nonzeroes in the matrix,

the size of the working storage,

the time used to find the ordering,

the time used for data structure set-up,

the time used for the factorization step,

the time used for the triangular solution step,

the time used for the relative error estimation step,

number of operations required by the factorization step,

number of operations required by the triangular solution step,

number of operations required by the relative error estimation step,

the storage used by the ordering subroutine,

the storage used by the data structure set-up subroutine,

the storage used by the *SOLVEi* module,

the storage used by the *ERESTi* module,

an estimate of the reciprocal of the condition number of the input matrix, and

an estimate of the relative error in the triangular factorization,

an estimate of the relative error in the computed solution.

Since the module *STATSA* can be called at any time, some of the above information may not be available, and will not be printed. The word "operations" here means multiplicative operations (multiplications and divisions). Since most of the arithmetic performed in sparse matrix computation occurs in multiply-add pairs, the number of operations (as defined here) is a useful measure of the amount of arithmetic performed.

The reader is referred to the examples in Section 9 for more discussion about the output from *STATSA*.

7.3. Error messages (*IERRA*)

When a fatal error is detected, so that the computation cannot proceed, a positive code is assigned to *IERRA*. The user can simply check the value of *IERRA* to see if the execution of module has been successful. This error flag can be used in conjunction with the save/restart feature described in Section 4 to retain the results of successfully completed parts of the computation, as shown by the program fragment below.

```
      .
      .
      .
      CALL   ORDRA1 ( S )
      IF   ( IERRA .EQ. 0 )    GO TO 100
      CALL   SAVEA ( 9, S )
      STOP
  100 CONTINUE
      .
      .
      .
```

The variable *IERRA* is set to the value $10 \times k + l$, where $0 \leq l \leq 9$ distinguishes the error, and k is determined by the type of module that sets *IERRA* positive.

k	interface modules
10	save and restart modules (*SAVEA*, *RSTRTA*)
11	matrix structure input modules (*INIJ*, *INIJIJ*, etc.)
12	matrix ordering and allocation modules (*ORDRxi*)
13	matrix numerical input modules (*INAIJi*, ..., etc.)
14	right hand side numerical input modules (*INBI*, ..., etc.)
15	factorization and solution modules (*SOLVEi*)
16	relative error estimation modules (*ERESTi*)

7.3.1. Save and restart routines

IERRA	*SAVEA, RSTRTA*
101	Output unit given to *SAVEA* is not positive.
102	Input unit given to *RSTRTA* is not positive.
103	Insufficient storage to restart the computational process. The minimum value of *MAXSA* required is printed in the error message.

7.3.2. Input of the matrix structure

IERRA	*INIJ, INROW, INIJIJ, INCLQ*
111	Incorrect execution sequence. Probable cause of error: routine *IJBEGN* was not executed successfully before (i,j) pairs input began.
112	Incorrect execution sequence. Probable cause of error: routine *IJEND* has already been called to indicate the end of structure input.
113	Insufficient storage was provided in the working storage array. The (i,j) pairs input to *INIJ*, *INROW*, *INIJIJ*, and *INCLQ* will be counted and discarded. Duplicates which are detected will not be counted, but some duplicates may be missed.
114	Input index (or subscript) is negative or zero.

IERRA	*IJEND*
115	Incorrect execution sequence. Probable cause of error: routine *IJEND* was called before new matrix structure has been input. Call *IJBEGN* to start a new problem.
116	Insufficient storage to transform matrix structure. The minimum value of *MAXSA* required is printed in the error message.
117	Number of variables (*NEQNS*) is zero.

7.3.3. Ordering and storage allocation routines

IERRA	*ORDRxi*
121	Incorrect execution sequence. Probable cause of error: routine *IJEND* was not executed successfully.
122	Incorrect execution sequence. Probable cause of error: routine *ORDRxi* was called after having already been executed successfully.
123	Incompatible ordering method. Probable cause of error: part of the ordering routine *ORDRxi* was executed, and then *SAVEA* was executed because of insufficient storage. The execution was then restarted, using *RSTRTA*, but *ORDRxj* was called with $i \neq j$.
124	Insufficient storage in working storage array to execute the ordering routine. Response: execute *SAVEA*, and restart the computation using *ORDRxi* with *MAXSA* at least as large as that indicated in the error message.
125	Insufficient storage in working storage array to execute the storage allocation routine. The ordering routine was successfully executed. Response: same as for error 124.
126	Insufficient storage in working storage array to hold the data structure pointers. The ordering and storage allocation routines were successfully executed. Response: same as for error 124.
127	Insufficient storage in working storage array to hold the numerical values. The ordering and storage allocation routines were successfully executed, and there was enough storage to hold the data structure pointers. Response: same as for error 124.

7.3.4. Input of the numerical values

IERRA	*INAIJi, INROWi, INMATi*
131	Incorrect execution sequence. Probable cause of error: routine *ORDRxi* was not executed successfully.
132	Incompatible input routine. Probable cause of error: attempt to use input routine *INAIJi*, *INROWi*, or *INMATi* after using *ORDRxj*, where $i \neq j$.

IERRA	*INAIJi, INROWi, INMATi*
133	Attempt to input the (i,j)-th element of matrix A for $i<j$. (This error occurs only for symmetric matrix methods; i.e., when method is odd). Methods for symmetric matrices expect elements of the lower triangle to be input.
134	Attempt to input an (i,j)-th element of matrix A where $i>n$, $j>n$, $i<1$, or $j<1$.
135	Attempt to input a numerical value for the (i,j)-th element of matrix A into the data structure, but the data structure has no space for it. Probable cause of error: the user has not called *INIJ, INROW, INIJIJ* or *INCLQ* with all the pairs (i,j) for which the (i,j)-th elements of A are nonzero. (SPARSPAK-A thinks A is sparser than it really is.)

IERRA	*INBI, INBIBI, INRHS*
141	Incorrect execution sequence. Probable cause of error: routine *ORDRxi* was not executed successfully.
142	Index (or subscript) out of range. Probable cause of error: attempt to input a numerical value for the i-th element of b where $i>n$ or $i<1$.

7.3.5. Factorization and solution

IERRA	*SOLVEi*
151	Incorrect execution sequence. Probable cause of error: the numerical input routines were not executed successfully.
152	Incompatible ordering and solution routines. Probable cause of error: Routines *ORDRxi* and *SOLVEj* were called, where $i \neq j$. Response: execute *SAVEA* and restart the computation using *SOLVEi* where i is the value of *METHOD* specified in the error message.

IERRA	*SOLVEi*
153	Zero pivot or negative square root was detected in the (symmetric) factorization routine. Possible cause of error: the matrix may require pivoting in order to preserve numerical stability. In this case the use of SPARSPAK-A to solve the problem is inappropriate. (See *restrictions* in Section 1.)
154	Zero pivot was detected in the (unsymmetric) factorization routine. Possible cause of error: the matrix may require pivoting in order to preserve numerical stability. In this case the use of SPARSPAK-A to solve the problem is inappropriate. (See *restrictions* in Section 1.)

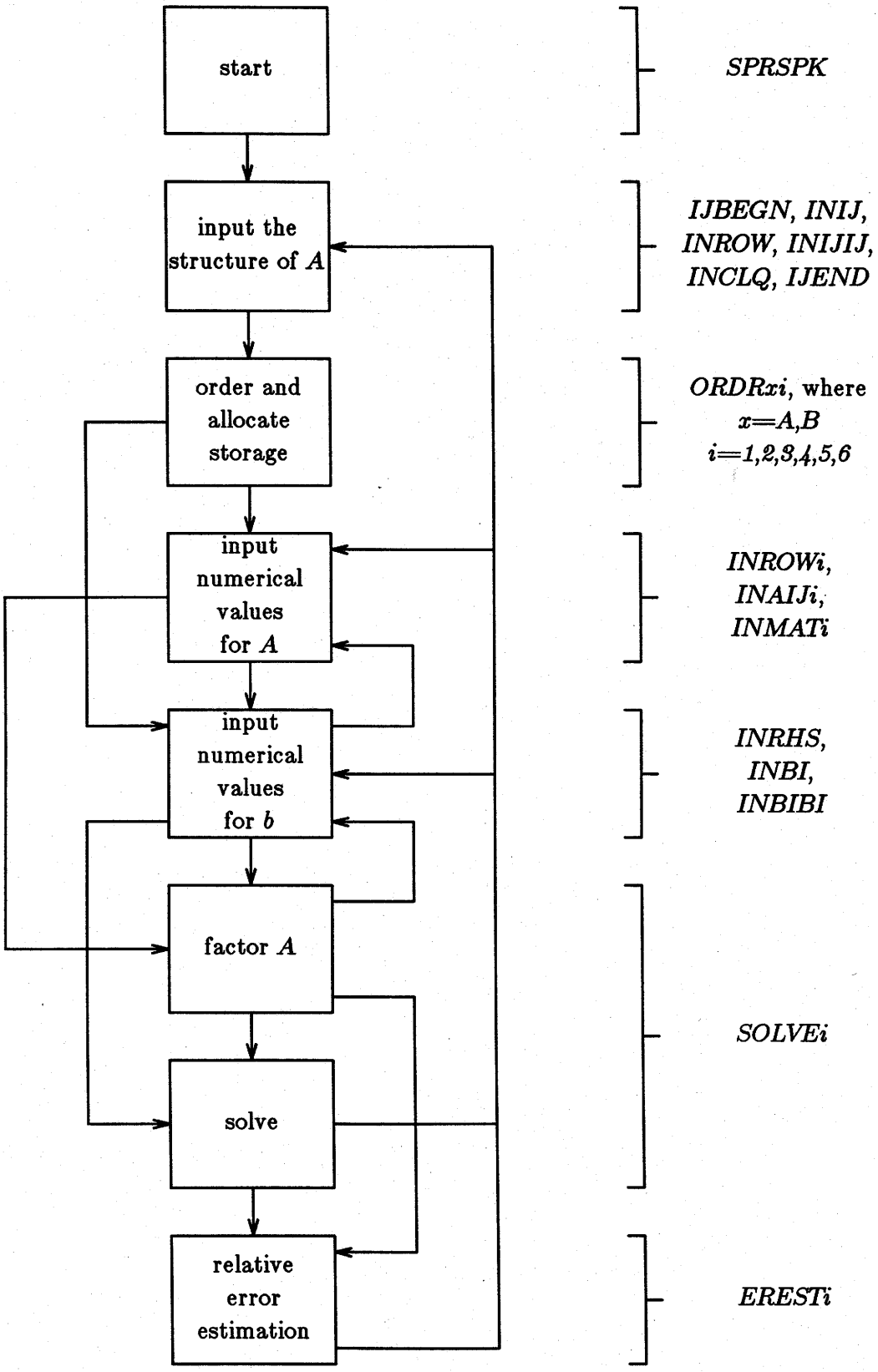
7.3.6. Relative error estimation

IERRA	*ERESTi*
161	Incorrect execution sequence. Probable cause of error: routine *SOLVEi* was not executed successfully.
162	Incompatible condition number estimation routine. Probable cause of error: Routines *ERESTi* and *SOLVEj* were called, where $i \neq j$. Response: execute *SAVEA* and restart the computation using *ERESTi* where i is the value of *METHOD* specified in the error message.
163	Insufficient storage in working storage array to compute an estimate of the relative error in the computed solution. Response: execute *SAVEA*, and restart the computation using *ERESTi* with *MAXSA* at least as large as that indicated in the error message.
164	The estimate of the relative error has a value of -1.0 which means that the computed solution may not have any correct significant digits.

8. Summary listing of interface routines

Initialization of SPARSPAK-A	*SPRSPK*
Structure input	*IJBEGN*
	INIJ (I, J, S) *INROW (I, NIR, IR, S)* *INIJIJ (NIJ, II, JJ, S)* *INCLQ (NCLQ, CLQ, S)*
	IJEND (S)
Ordering (see next table)	*ORDRxi (S)*
Matrix input	*INAIJi (I, J, VALUE, S)* *INROWi (I, NIR, IR, VALUES, S)* *INMATi (NIJ, II, JJ, VALUES, S)*
Right hand side input	*INBI (I, VALUE, S)* *INBIBI (NI, II, VALUES, S)* *INRHS (RHS, S)*
Factorization and/or Solution	*SOLVEi (S)*
Relative error estimation	*ERESTi (RELERR, S)*
Print statistics	*STATSA*
Save and Restart the computation	*SAVEA (K, S)* *RSTRTA (K, S)*

ORDRxi		Ordering Choices
x	*i*	
A	*1*	Reverse Cuthill-McKee ordering [7]; symmetric *A*
A	*2*	Reverse Cuthill-McKee ordering [7]; unsymmetric *A*
A	*3*	One-way Dissection ordering [2]; symmetric *A*
A	*4*	One-way Dissection ordering [2]; unsymmetric *A*
B	*3*	Refined quotient tree ordering [3]; symmetric *A*
B	*4*	Refined quotient tree ordering [3]; unsymmetric *A*
A	*5*	Nested Dissection ordering [4]; symmetric *A*
A	*6*	Nested Dissection ordering [4]; unsymmetric *A*
B	*5*	Minimum Degree ordering [6]; symmetric *A*
B	*6*	Minimum Degree ordering [6]; unsymmetric *A*



Sketch of possible execution paths through SPARSPAK-A modules

9. Examples

In this section, we provide several programs which illustrate how SPARSPAK-A can be used. These programs are derived from the one given in Section 2.1.

These examples were run using a standard single precision version of SPARSPAK-A under the Berkeley f77 compiler on a DEC VAX 11/780 computer. All times reported are in seconds. It should be noted that the results will be different if a different version of SPARSPAK-A or a different computer is used.

Example 1

This is an example of the simplest use of SPARSPAK-A, with each of the modules of method 1 used in sequence. The problem solved is a 10×10 symmetric tridiagonal system $Ax=b$ where the diagonal elements of A are all 4, and the superdiagonal and subdiagonal elements are all -1. The right hand side vector b is chosen so that the entries of the solution vector x are all ones.

In the program, the nonzero structure of A is input using *IJBEGN*, *INIJ* and *IJEND*. After *ORDRA1* is executed, the interface modules *INAIJ1* and *INBI* are used to transmit the numerical values of A and b to the package respectively. The module *SOLVE1* is called to do the numerical solution, and *EREST1* is called to compute an estimate of the relative error in the computed solution. Then *STATSA* is called to print out the statistics gathered by the interface during execution. Finally, the actual error in the computed approximate solution is determined since the true solution is known.

Note that the size of the working storage provided was 250, while the maximum amount used by any of the modules was 90, which was the storage requirement for the *EREST1* module. Thus, if the user was going to solve this problem again, he could adjust his storage down to 90.

Program

```
C--- SPARSPAK-A (ANSI FORTRAN) RELEASE III --- NAME = EX1                    1SPK
C  (C)  UNIVERSITY OF WATERLOO   JANUARY 1984                                 2SPK
C*************************************************************************  3SPK
C*************************************************************************  4SPK
C**********      M A I N L I N E     P R O G R A M      **********           5SPK
C*************************************************************************  6SPK
C*************************************************************************  7SPK
C                                                                            8SPK
      INTEGER    I      , IERRA , IPRNTE, IPRNTS, MAXINT, MAXSA ,           9SPK
     1           MSGLVA, NVARS                                             10SPK
      REAL       MCHEPS, RATIOL, RATIOS, TIME                              11SPK
      REAL               S(250)                                            12SPK
      REAL               ERROR , FOUR  , ONE   , RELERR, TWO   ,           13SPK
     1                   ZERO                                              14SPK
C                                                                           15SPK
C*************************************************************************  16SPK
C                                                                           17SPK
      COMMON  /SPAUSR/  MSGLVA, IERRA , MAXSA , NVARS                       18SPK
      COMMON  /SPKSYS/  IPRNTE, IPRNTS, MAXINT, RATIOS, RATIOL,             19SPK
     1                  MCHEPS, TIME                                        20SPK
C                                                                           21SPK
C*************************************************************************  22SPK
C                                                                           23SPK
C     ----------------------                                                24SPK
C     INITIALIZE SPARSPAK-A.                                                25SPK
C     ----------------------                                                26SPK
      CALL  SPRSPK                                                          27SPK
      MAXSA = 250                                                           28SPK
C                                                                           29SPK
C     ---------------                                                       30SPK
C     INPUT STRUCTURE.                                                      31SPK
C     ---------------                                                       32SPK
      CALL  IJBEGN                                                          33SPK
      DO  100  I = 2, 10                                                    34SPK
```

```
          CALL   INIJ ( I, I-1, S )                                          95SPK
  100 CONTINUE                                                               96SPK
      CALL   IJEND ( S )                                                     97SPK
C                                                                            98SPK
C     ------------------------------                                         99SPK
C     DETERMINE SYMMETRIC ORDERING.                                          40SPK
C     ------------------------------                                         41SPK
      CALL   ORDRA1 ( S )                                                    42SPK
C                                                                            43SPK
C     ------------------------                                               44SPK
C     INPUT NUMERICAL VALUES.                                                45SPK
C     ------------------------                                               46SPK
      ZERO = 0.0E0                                                           47SPK
      ONE  = 1.0E0                                                           48SPK
      TWO  = 2.0E0                                                           49SPK
      FOUR = 4.0E0                                                           50SPK
      DO  200  I = 1, 10                                                     51SPK
          IF  ( I .GT. 1 )  CALL  INAIJ1 ( I, I-1, -ONE, S )                 52SPK
          CALL  INAIJ1 ( I, I, FOUR, S )                                     53SPK
          CALL  INBI ( I, TWO, S )                                           54SPK
  200 CONTINUE                                                               55SPK
      CALL  INBI ( 1, ONE, S )                                               56SPK
      CALL  INBI ( 10, ONE, S )                                              57SPK
C                                                                            58SPK
C     ---------------------------------------------                          59SPK
C     PERFORM NUMERICAL FACTORIZATION AND SOLUTION.                          60SPK
C     ---------------------------------------------                          61SPK
      CALL  SOLVE1 ( S )                                                     62SPK
C                                                                            63SPK
C     ----------------------------------------                               64SPK
C     COMPUTE AN ESTIMATE OF THE RELATIVE ERROR                              65SPK
C     IN THE COMPUTED SOLUTION.                                              66SPK
C     ----------------------------------------                               67SPK
      CALL  EREST1 ( RELERR, S )                                             68SPK
C                                                                            69SPK
C     ------------------                                                     70SPK
C     OBTAIN STATISTICS.                                                     71SPK
C     ------------------                                                     72SPK
      CALL  STATSA                                                           73SPK
C                                                                            74SPK
C     ------------------------------------------------------------           75SPK
C     COMPUTE THE ACTUAL RELATIVE ERROR IN THE COMPUTED SOLUTION             76SPK
C     SINCE THE TRUE SOLUTION IS KNOWN.                                      77SPK
C     ------------------------------------------------------------           78SPK
      ERROR = ZERO                                                           79SPK
      DO  300  I = 1, 10                                                     80SPK
          ERROR = AMAX1(ERROR,ABS(S(I)-ONE))                                 81SPK
  300 CONTINUE                                                               82SPK
      WRITE (IPRNTS,11)  ERROR                                               83SPK
   11 FORMAT (/10X, 35HMAXIMUM ERROR                    , 1PE15.5 )           84SPK
C                                                                            85SPK
      STOP                                                                   86SPK
      END                                                                    87SPK
```

Output

```
********** UNIVERSITY OF WATERLOO
********** SPARSE MATRIX PACKAGE
**********  ( S P A R S P A K )
**********       RELEASE   3
```

```
**********  (C) JANUARY 1984
**********  ANSI FORTRAN
**********  SINGLE PRECISION
**********  LAST UPDATE JANUARY 1984


     OUTPUT UNIT FOR ERROR MESSAGES          6
     OUTPUT UNIT FOR STATISTICS              6

IJBEGN - BEGIN STRUCTURE INPUT ...

INIJ   - INPUT OF ADJACENCY PAIRS ...

IJEND  - END OF STRUCTURE INPUT ...

ORDRA1 - RCM ORDERING ...

INAIJ1 - INPUT OF MATRIX COMPONENTS ...

INBI   - INPUT OF RIGHT HAND SIDE ...

SOLVE1 - ENVELOPE SOLVE ...

EREST1 - ERROR ESTIMATOR ...

STATSA - SYSTEM-A STATISTICS ...

     SIZE OF STORAGE ARRAY (MAXSA)          250
     NUMBER OF EQUATIONS                     10
     NUMBER OF OFF-DIAGONAL NONZEROS         18
     TIME FOR ORDERING                       0.033
     STORAGE FOR ORDERING                    60.
     TIME FOR ALLOCATION                     0.
     STORAGE FOR ALLOCATION                  60.
     STORAGE FOR SOLUTION                    70.
     TIME FOR FACTORIZATION                  0.017
     TIME FOR SOLUTION                       0.
     OPERATIONS IN FACTORIZATION             18.
     OPERATIONS IN SOLUTION                  38.
     TIME FOR ESTIMATING RELATIVE ERROR      0.050
     OPERATIONS IN ESTIMATING REL ERROR      160.
     STORAGE FOR ESTIMATING REL ERROR        90.
     ESTIMATE OF RELATIVE ERROR              1.785e-07
     TOTAL TIME REQUIRED                     0.100
     MAXIMUM STORAGE REQUIRED                90.

     MAXIMUM ERROR                           1.19209e-07
```

Example 2

This is the same as Example 1, except that the matrix A is unsymmetric. The diagonal elements of A are all 4, the superdiagonal elements are all 1, and the subdiagonal elements are all -1. The right hand side vector b is chosen so that the entries of the solution vector x are all ones.

Program

```
C--- SPARSPAK-A (ANSI FORTRAN) RELEASE III --- NAME = EX2                    1SPK
C  (C)  UNIVERSITY OF WATERLOO   JANUARY 1984                                2SPK
C***************************************************************************3SPK
C***************************************************************************4SPK
C**********     M A I N L I N E     P R O G R A M     **********             5SPK
C***************************************************************************6SPK
C***************************************************************************7SPK
C                                                                           8SPK
      INTEGER    I      , IERRA , IPRNTE, IPRNTS, MAXINT, MAXSA ,          9SPK
     1           MSGLVA, NVARS                                            10SPK
      REAL       MCHEPS, RATIOL, RATIOS, TIME                             11SPK
      REAL               S(250)                                           12SPK
      REAL               ERROR , FOUR  , ONE   , RELERR, ZERO             13SPK
C                                                                          14SPK
C***************************************************************************15SPK
C                                                                          16SPK
      COMMON  /SPAUSR/  MSGLVA, IERRA , MAXSA , NVARS                      17SPK
      COMMON  /SPKSYS/  IPRNTE, IPRNTS, MAXINT, RATIOS, RATIOL,            18SPK
     1                  MCHEPS, TIME                                       19SPK
C                                                                          20SPK
C***************************************************************************21SPK
C                                                                          22SPK
C     ----------------------                                               23SPK
C     INITIALIZE SPARSPAK-A.                                               24SPK
C     ----------------------                                               25SPK
      CALL  SPRSPK                                                         26SPK
      MAXSA = 250                                                          27SPK
C                                                                          28SPK
C     ----------------                                                     29SPK
C     INPUT STRUCTURE.                                                     30SPK
C     ----------------                                                     31SPK
      CALL  IJBEGN                                                         32SPK
      DO  100  I = 2, 10                                                   33SPK
          CALL  INIJ ( I, I-1, S )                                         34SPK
  100 CONTINUE                                                             35SPK
      CALL  IJEND ( S )                                                    36SPK
C                                                                          37SPK
C     ----------------------------                                         38SPK
C     DETERMINE SYMMETRIC ORDERING.                                        39SPK
C     ----------------------------                                         40SPK
      CALL  ORDRA2 ( S )                                                   41SPK
C                                                                          42SPK
C     ----------------------                                               43SPK
C     INPUT NUMERICAL VALUES.                                              44SPK
C     ----------------------                                               45SPK
      ZERO = 0.0E0                                                         46SPK
      ONE  = 1.0E0                                                         47SPK
      FOUR = 4.0E0                                                         48SPK
      DO  200  I = 1, 10                                                   49SPK
          IF  ( I .GT. 1 )  CALL  INAIJ2 ( I, I-1, -ONE, S )               50SPK
          IF  ( I .LT. 10 )  CALL  INAIJ2 ( I, I+1, ONE, S )               51SPK
          CALL  INAIJ2 ( I, I, FOUR, S )                                   52SPK
```

```
            CALL  INBI ( I, FOUR, S )                                    59SPK
    200 CONTINUE                                                         54SPK
        CALL  INBI ( 1, ONE, S )                                         55SPK
        CALL  INBI ( 10, -ONE, S )                                       56SPK
C                                                                        57SPK
C       ------------------------------------------                       58SPK
C       PERFORM NUMERICAL FACTORIZATION AND SOLUTION.                    59SPK
C       ------------------------------------------                       60SPK
        CALL  SOLVE2 ( S )                                               61SPK
C                                                                        62SPK
C       ---------------------------------------                          63SPK
C       COMPUTE AN ESTIMATE OF THE RELATIVE ERROR                        64SPK
C       IN THE COMPUTED SOLUTION.                                        65SPK
C       ---------------------------------------                          66SPK
        CALL  EREST2 ( RELERR, S )                                       67SPK
C                                                                        68SPK
C       -----------------                                                69SPK
C       OBTAIN STATISTICS.                                               70SPK
C       -----------------                                                71SPK
        CALL  STATSA                                                     72SPK
C                                                                        73SPK
C       ------------------------------------------------------           74SPK
C       COMPUTE THE ACTUAL RELATIVE ERROR IN THE COMPUTED SOLUTION       75SPK
C       SINCE THE TRUE SOLUTION IS KNOWN.                                76SPK
C       ------------------------------------------------------           77SPK
        ERROR = ZERO                                                     78SPK
        DO  300  I = 1, 10                                               79SPK
            ERROR = AMAX1(ERROR,ABS(S(I)-ONE))                           80SPK
    300 CONTINUE                                                         81SPK
        WRITE (IPRNTS,11)  ERROR                                         82SPK
     11 FORMAT (/10X, 35HMAXIMUM ERROR                    , 1PE15.5 )    83SPK
C                                                                        84SPK
        STOP                                                             85SPK
        END                                                              86SPK
```

Output

```
********** UNIVERSITY OF WATERLOO
********** SPARSE MATRIX PACKAGE
**********  ( S P A R S P A K )
**********       RELEASE   3
**********  (C) JANUARY 1984
********** ANSI FORTRAN
********** SINGLE PRECISION
********** LAST UPDATE JANUARY 1984


     OUTPUT UNIT FOR ERROR MESSAGES          6
     OUTPUT UNIT FOR STATISTICS              6

IJBEGN - BEGIN STRUCTURE INPUT ...

INIJ   - INPUT OF ADJACENCY PAIRS ...

IJEND  - END OF STRUCTURE INPUT ...

ORDRA2 - RCM ORDERING ...

INAIJ2 - INPUT OF MATRIX COMPONENTS ...
```

```
INBI    - INPUT OF RIGHT HAND SIDE ...

SOLVE2  - ENVELOPE SOLVE ...

EREST2  - ERROR ESTIMATOR ...

STATSA  - SYSTEM-A STATISTICS ...

      SIZE OF STORAGE ARRAY (MAXSA)              250
      NUMBER OF EQUATIONS                         10
      NUMBER OF OFF-DIAGONAL NONZEROS             18
      TIME FOR ORDERING                           0.017
      STORAGE FOR ORDERING                        60.
      TIME FOR ALLOCATION                         0.
      STORAGE FOR ALLOCATION                      60.
      STORAGE FOR SOLUTION                        79.
      TIME FOR FACTORIZATION                      0.017
      TIME FOR SOLUTION                           0.
      OPERATIONS IN FACTORIZATION                 18.
      OPERATIONS IN SOLUTION                      28.
      TIME FOR ESTIMATING RELATIVE ERROR          0.033
      OPERATIONS IN ESTIMATING REL ERROR          160.
      STORAGE FOR ESTIMATING REL ERROR            99.
      ESTIMATE OF RELATIVE ERROR                  1.197e-07
      TOTAL TIME REQUIRED                         0.067
      MAXIMUM STORAGE REQUIRED                    99.

      MAXIMUM ERROR                               1.19209e-07
```

Example 3

This is similar to Example 1, except that method 3 is used (with the *A* ordering option), and two problems differing only in their right hand sides are solved. After solving the problem whose solution vector contains all ones, a new right hand side is input which corresponds to a different problem whose solution vector contains all twos. When the module *SOLVE3* is called a second time, the interface detects that the factorization has already been done, and only the triangular solution is performed.

Program

```
C--- SPARSPAK-A (ANSI FORTRAN) RELEASE III --- NAME = EX3                    1SPK
C  (C)  UNIVERSITY OF WATERLOO    JANUARY 1984                               2SPK
C***************************************************************           3SPK
C***************************************************************           4SPK
C**********     M A I N L I N E     P R O G R A M     **********           5SPK
C***************************************************************           6SPK
C***************************************************************           7SPK
C                                                                           8SPK
      INTEGER    I      , IERRA , IPRNTE, IPRNTS, MAXINT, MAXSA ,           9SPK
     1           MSGLVA, NVARS                                             10SPK
      REAL       MCHEPS, RATIOL, RATIOS, TIME                              11SPK
      REAL               S(250)                                            12SPK
      REAL               ERROR , FOUR  , ONE   , RELERR, TWO   ,           13SPK
     1                   ZERO                                              14SPK
C                                                                           15SPK
C***************************************************************           16SPK
C                                                                           17SPK
      COMMON  /SPAUSR/  MSGLVA, IERRA , MAXSA , NVARS                       18SPK
      COMMON  /SPKSYS/  IPRNTE, IPRNTS, MAXINT, RATIOS, RATIOL,             19SPK
     1                  MCHEPS, TIME                                        20SPK
C                                                                           21SPK
C***************************************************************           22SPK
C                                                                           23SPK
C     ----------------------                                                24SPK
C     INITIALIZE SPARSPAK-A.                                                25SPK
C     ----------------------                                                26SPK
      CALL  SPRSPK                                                          27SPK
      MAXSA = 250                                                           28SPK
C                                                                           29SPK
C     ---------------                                                       30SPK
C     INPUT STRUCTURE.                                                      31SPK
C     ---------------                                                       32SPK
      CALL  IJBEGN                                                          33SPK
      DO  100  I = 2, 10                                                    34SPK
          CALL  INIJ ( I, I-1, S )                                          35SPK
  100 CONTINUE                                                              36SPK
      CALL  IJEND ( S )                                                     37SPK
C                                                                           38SPK
C     ----------------------------                                          39SPK
C     DETERMINE SYMMETRIC ORDERING.                                         40SPK
C     ----------------------------                                          41SPK
      CALL  ORDRA3 ( S )                                                    42SPK
C                                                                           43SPK
C     ------------------------                                              44SPK
C     INPUT NNUMERICAL VALUES.                                              45SPK
C     ------------------------                                              46SPK
      ZERO = 0.0E0                                                          47SPK
      ONE  = 1.0E0                                                          48SPK
      TWO  = 2.0E0                                                          49SPK
```

```
      FOUR = 4.0E0                                                        50SPK
      DO  200  I = 1, 10                                                  51SPK
          IF  ( I .GT. 1 )  CALL  INAIJS ( I, I-1, -ONE, S )              52SPK
          CALL  INAIJS ( I, I, FOUR, S )                                  53SPK
          CALL  INBI ( I, TWO, S )                                        54SPK
  200 CONTINUE                                                            55SPK
      CALL  INBI ( 1, ONE, S )                                            56SPK
      CALL  INBI ( 10, ONE, S )                                           57SPK
C                                                                         58SPK
C     ------------------------------------------------                    59SPK
C     PERFORM NUMERICAL FACTORIZATION AND SOLUTION.                       60SPK
C     ------------------------------------------------                    61SPK
      CALL  SOLVES ( S )                                                  62SPK
C                                                                         63SPK
C     ----------------------------------------                            64SPK
C     COMPUTE AN ESTIMATE OF THE RELATIVE ERROR                           65SPK
C     IN THE COMPUTED SOLUTION.                                           66SPK
C     ----------------------------------------                            67SPK
      CALL  ERESTS ( RELERR, S )                                          68SPK
C                                                                         69SPK
C     ------------------                                                  70SPK
C     OBTAIN STATISTICS.                                                  71SPK
C     ------------------                                                  72SPK
      CALL  STATSA                                                        73SPK
C                                                                         74SPK
C     ------------------------------------------------------              75SPK
C     COMPUTE THE ACTUAL RELATIVE ERROR IN THE SOLUTION                   76SPK
C     SINCE THE TRUE SOLUTION IS KNOWN.                                   77SPK
C     ------------------------------------------------------              78SPK
      ERROR = ZERO                                                        79SPK
      DO  300  I = 1, 10                                                  80SPK
          ERROR = AMAX1(ERROR,ABS(S(I)-ONE))                              81SPK
  300 CONTINUE                                                            82SPK
      WRITE (IPRNTS,11)  ERROR                                            83SPK
   11 FORMAT (/10X, 35HMAXIMUM ERROR                    , 1PE15.5 )       84SPK
C                                                                         85SPK
C     ------------------------------------                                86SPK
C     INPUT A NEW RIGHT HAND SIDE VECTOR.                                 87SPK
C     ------------------------------------                                88SPK
      DO  400  I = 1, 10                                                  89SPK
          CALL  INBI ( I, FOUR, S )                                       90SPK
  400 CONTINUE                                                            91SPK
      CALL  INBI ( 1, TWO, S )                                            92SPK
      CALL  INBI ( 10, TWO, S )                                           93SPK
C                                                                         94SPK
C     ----------------------                                              95SPK
C     PERFORM ANOTHER SOLVE.                                              96SPK
C     ----------------------                                              97SPK
      CALL  SOLVES ( S )                                                  98SPK
C                                                                         99SPK
C     ------------------                                                 100SPK
C     OBTAIN STATISTICS.                                                 101SPK
C     ------------------                                                 102SPK
      CALL  STATSA                                                       103SPK
C                                                                        104SPK
C     ----------------------------------                                 105SPK
C     COMPUTE THE ACTUAL RELATIVE ERROR.                                 106SPK
C     ----------------------------------                                 107SPK
      ERROR = ZERO                                                       108SPK
      DO  500  I = 1, 10                                                 109SPK
          ERROR = AMAX1(ERROR,ABS(S(I)-TWO))                             110SPK
  500 CONTINUE                                                           111SPK
      WRITE (IPRNTS,11)  ERROR                                           112SPK
C                                                                        113SPK
      STOP                                                               114SPK
      END                                                                115SPK
```

Output

```
********** UNIVERSITY OF WATERLOO
********** SPARSE MATRIX PACKAGE
**********  ( S P A R S P A K )
**********       RELEASE   3
**********  (C) JANUARY 1984
********** ANSI FORTRAN
********** SINGLE PRECISION
********** LAST UPDATE JANUARY 1984


     OUTPUT UNIT FOR ERROR MESSAGES          6
     OUTPUT UNIT FOR STATISTICS              6

IJBEGN - BEGIN STRUCTURE INPUT ...

INIJ   - INPUT OF ADJACENCY PAIRS ...

IJEND  - END OF STRUCTURE INPUT ...

ORDRA3 - ONE-WAY DISSECTION ORDERING ...

INAIJ3 - INPUT OF MATRIX COMPONENTS ...

INBI   - INPUT OF RIGHT HAND SIDE ...

SOLVE3 - IMPLICIT BLOCK SOLVE ...

EREST3 - ERROR ESTIMATOR ...

STATSA - SYSTEM-A STATISTICS ...

     SIZE OF STORAGE ARRAY (MAXSA)          250
     NUMBER OF EQUATIONS                     10
     NUMBER OF OFF-DIAGONAL NONZEROS         18
     TIME FOR ORDERING                       0.033
     STORAGE FOR ORDERING                    91.
     TIME FOR ALLOCATION                     0.017
     STORAGE FOR ALLOCATION                  94.
     STORAGE FOR SOLUTION                    94.
     TIME FOR FACTORIZATION                  0.
     TIME FOR SOLUTION                       0.017
     OPERATIONS IN FACTORIZATION             18.
     OPERATIONS IN SOLUTION                  38.
     TIME FOR ESTIMATING RELATIVE ERROR      0.067
     OPERATIONS IN ESTIMATING REL ERROR      157.
     STORAGE FOR ESTIMATING REL ERROR        144.
     ESTIMATE OF RELATIVE ERROR              1.785e-07
     TOTAL TIME REQUIRED                     0.133
     MAXIMUM STORAGE REQUIRED                144.

     MAXIMUM ERROR                           1.19209e-07

INBI   - INPUT OF RIGHT HAND SIDE ...

SOLVE3 - IMPLICIT BLOCK SOLVE ...
     FACTORIZATION ALREADY DONE.

STATSA - SYSTEM-A STATISTICS ...
```

```
SIZE OF STORAGE ARRAY (MAXSA)              250
NUMBER OF EQUATIONS                         10
NUMBER OF OFF-DIAGONAL NONZEROS             18
TIME FOR ORDERING                          0.033
STORAGE FOR ORDERING                       91.
TIME FOR ALLOCATION                        0.017
STORAGE FOR ALLOCATION                     94.
STORAGE FOR SOLUTION                       94.
TIME FOR FACTORIZATION                      0.
TIME FOR SOLUTION                           0.
OPERATIONS IN FACTORIZATION                18.
OPERATIONS IN SOLUTION                     38.
TOTAL TIME REQUIRED                        0.050
MAXIMUM STORAGE REQUIRED                   94.

MAXIMUM ERROR                          2.38419e-07
```

Example 4

This example illustrates the use of the save/restart feature of SPARSPAK-A. After the factorization is computed, *SAVEA* is executed, which writes the current state of the computation on FORTRAN logical unit 3. In a second program the module *RSTRTA* is executed to read the information from unit 3, and the computation resumes at the point at which *SAVEA* was invoked.

Program 1

```
C--- SPARSPAK-A (ANSI FORTRAN) RELEASE III --- NAME = EX4A                    1SPK
C  (C)  UNIVERSITY OF WATERLOO   JANUARY 1984                                 2SPK
C*************************************************************              3SPK
C*************************************************************              4SPK
C**********     M A I N L I N E     P R O G R A M     **********              5SPK
C*************************************************************              6SPK
C*************************************************************              7SPK
C                                                                            8SPK
C     FILE REQUIREMENT :                                                     9SPK
C        --- UNIT 3 - FOR SAVEA (SHOULD NOT BE DESTROYED).                  10SPK
C                                                                           11SPK
C*************************************************************             12SPK
C                                                                           13SPK
      INTEGER     I       , IERRA , MAXSA , MSGLVA, NVARS                   14SPK
      REAL                  S(250)                                          15SPK
      REAL                  FOUR  , ONE   , RELERR                          16SPK
C                                                                           17SPK
C*************************************************************             18SPK
C                                                                           19SPK
      COMMON  /SPAUSR/  MSGLVA, IERRA , MAXSA , NVARS                       20SPK
C                                                                           21SPK
C*************************************************************             22SPK
C                                                                           23SPK
C     ----------------------                                                24SPK
C     INITIALIZE SPARSPAK-A.                                                25SPK
C     ----------------------                                                26SPK
      CALL  SPRSPK                                                          27SPK
      MAXSA = 250                                                           28SPK
C                                                                           29SPK
C     ---------------                                                       30SPK
C     INPUT STRUCTURE.                                                      31SPK
C     ---------------                                                       32SPK
      CALL  IJBEGN                                                          33SPK
      DO  100  I = 2, 10                                                    34SPK
          CALL  INIJ ( I, I-1, S )                                          35SPK
  100 CONTINUE                                                              36SPK
      CALL  IJEND ( S )                                                     37SPK
C                                                                           38SPK
C     ----------------------------                                          39SPK
C     DETERMINE SYMMETRIC ORDERING.                                         40SPK
C     ----------------------------                                          41SPK
      CALL  ORDRA1 ( S )                                                    42SPK
C                                                                           43SPK
C     ----------------------                                                44SPK
C     INPUT NUMERICAL VALUES.                                               45SPK
C     ----------------------                                                46SPK
      ONE  = 1.0E0                                                          47SPK
      FOUR = 4.0E0                                                          48SPK
      DO  200  I = 1, 10                                                    49SPK
          IF  ( I .GT. 1 )  CALL  INAIJ1 ( I, I-1, -ONE, S )                50SPK
```

```
            CALL   INAIJ1 ( I, I, FOUR, S )                          51SPK
    200 CONTINUE                                                     52SPK
C                                                                    53SPK
C       -------------------------------                              54SPK
C       PERFORM NUMERICAL FACTORIZATION.                             55SPK
C       -------------------------------                              56SPK
        CALL   SOLVE1 ( S )                                          57SPK
C                                                                    58SPK
C       ----------------------------------------                     59SPK
C       COMPUTE AN ESTIMATE OF THE RELATIVE ERROR                    60SPK
C       IN THE COMPUTED SOLUTION.                                    61SPK
C       ----------------------------------------                     62SPK
        CALL   EREST1 ( RELERR, S )                                  63SPK
C                                                                    64SPK
C       ------------------                                           65SPK
C       OBTAIN STATISTICS.                                           66SPK
C       ------------------                                           67SPK
        CALL   STATSA                                                68SPK
C                                                                    69SPK
C       --------------------------------------                       70SPK
C       SAVE CURRENT STATE OF THE COMPUTATION.                       71SPK
C       --------------------------------------                       72SPK
        CALL   SAVEA ( 3, S )                                        73SPK
C                                                                    74SPK
        STOP                                                         75SPK
        END                                                          76SPK
```

Output

```
********** UNIVERSITY OF WATERLOO
********** SPARSE MATRIX PACKAGE
**********  ( S P A R S P A K )
**********       RELEASE   3
**********  (C) JANUARY 1984
********** ANSI FORTRAN
********** SINGLE PRECISION
********** LAST UPDATE JANUARY 1984


     OUTPUT UNIT FOR ERROR MESSAGES          6
     OUTPUT UNIT FOR STATISTICS              6

IJBEGN - BEGIN STRUCTURE INPUT ...

INIJ   - INPUT OF ADJACENCY PAIRS ...

IJEND  - END OF STRUCTURE INPUT ...

ORDRA1 - RCM ORDERING ...

INAIJ1 - INPUT OF MATRIX COMPONENTS ...

SOLVE1 - ENVELOPE SOLVE ...
     NO RIGHT HAND SIDE PROVIDED,
     SOLUTION WILL BE ALL ZEROS.

EREST1 - ERROR ESTIMATOR ...

STATSA - SYSTEM-A STATISTICS ...
```

```
      SIZE OF STORAGE ARRAY (MAXSA)                250
      NUMBER OF EQUATIONS                           10
      NUMBER OF OFF-DIAGONAL NONZEROS               18
      TIME FOR ORDERING                             0.
      STORAGE FOR ORDERING                          60.
      TIME FOR ALLOCATION                           0.017
      STORAGE FOR ALLOCATION                        60.
      STORAGE FOR SOLUTION                          70.
      TIME FOR FACTORIZATION                        0.017
      TIME FOR SOLUTION                             0.
      OPERATIONS IN FACTORIZATION                   18.
      OPERATIONS IN SOLUTION                        0.
      TIME FOR ESTIMATING RELATIVE ERROR            0.033
      OPERATIONS IN ESTIMATING REL ERROR            160.
      STORAGE FOR ESTIMATING REL ERROR              90.
      ESTIMATE OF RELATIVE ERROR                 1.785e-07
      TOTAL TIME REQUIRED                           0.067
      MAXIMUM STORAGE REQUIRED                      90.

 SAVEA  - SAVE STORAGE VECTOR ...
```

Program 2

```
C--- SPARSPAK-A (ANSI FORTRAN) RELEASE III --- NAME = EX4B                1SPK
C  (C)  UNIVERSITY OF WATERLOO   JANUARY 1984                              2SPK
C*************************************************************           3SPK
C*************************************************************           4SPK
C**********     M A I N L I N E     P R O G R A M     **********           5SPK
C*************************************************************           6SPK
C*************************************************************           7SPK
C                                                                         8SPK
C     FILE REQUIREMENT:                                                   9SPK
C        --- UNIT 9 - FOR RSTRTA.                                        10SPK
C                                                                        11SPK
C*************************************************************          12SPK
C                                                                        13SPK
      INTEGER    I      , IERRA , IPRNTE, IPRNTS, MAXINT, MAXSA ,       14SPK
     1           MSGLVA, NVARS                                         15SPK
      REAL       MCHEPS, RATIOL, RATIOS, TIME                          16SPK
      REAL               S(250)                                        17SPK
      REAL               ERROR , ONE   , TWO   , ZERO                  18SPK
C                                                                       19SPK
C*************************************************************         20SPK
C                                                                       21SPK
      COMMON  /SPAUSR/  MSGLVA, IERRA , MAXSA , NVARS                   22SPK
      COMMON  /SPKSYS/  IPRNTE, IPRNTS, MAXINT, RATIOS, RATIOL,         23SPK
     1                  MCHEPS, TIME                                    24SPK
C                                                                       25SPK
C*************************************************************         26SPK
C                                                                       27SPK
C     ----------------------                                            28SPK
C     INITIALIZE SPARSPAK-A.                                            29SPK
C     ----------------------                                            30SPK
      CALL  SPRSPK                                                      31SPK
      MAXSA = 250                                                       32SPK
C                                                                       33SPK
C     ----------------------------                                      34SPK
C     RESTORE STATE OF COMPUTATION.                                     35SPK
C     ----------------------------                                      36SPK
      CALL  RSTRTA ( 9, S )                                             37SPK
```

```
C                                                                       38SPK
C     ------------------------------                                    39SPK
C     INPUT RIGHT HAND SIDE VECTOR.                                     40SPK
C     ------------------------------                                    41SPK
      ZERO = 0.0E0                                                      42SPK
      ONE  = 1.0E0                                                      43SPK
      TWO  = 2.0E0                                                      44SPK
      DO  100  I = 1, 10                                                45SPK
          CALL  INBI ( I, TWO, S )                                      46SPK
  100 CONTINUE                                                          47SPK
      CALL  INBI ( 1, ONE, S )                                          48SPK
      CALL  INBI ( 10, ONE, S )                                         49SPK
C                                                                       50SPK
C     ---------------------------                                       51SPK
C     PERFORM NUMERICAL SOLUTION.                                       52SPK
C     ---------------------------                                       53SPK
      CALL  SOLVE1 ( S )                                                54SPK
C                                                                       55SPK
C     -----------------                                                 56SPK
C     OBTAIN STATISTICS.                                                57SPK
C     -----------------                                                 58SPK
      CALL  STATSA                                                      59SPK
C                                                                       60SPK
C     ------------------------------------------------------------      61SPK
C     COMPUTE THE ACTUAL RELATIVE ERROR IN THE COMPUTED SOLUTION        62SPK
C     SINCE THE TRUE SOLUTION IS KNOWN.                                 63SPK
C     ------------------------------------------------------------      64SPK
      ERROR = ZERO                                                      65SPK
      DO  200  I = 1, 10                                                66SPK
          ERROR = AMAX1(ERROR,ABS(S(I)-ONE))                            67SPK
  200 CONTINUE                                                          68SPK
      WRITE (IPRNTS,11)  ERROR                                          69SPK
   11 FORMAT (/10X, 35HMAXIMUM ERROR                , 1PE15.5 )        70SPK
C                                                                       71SPK
      STOP                                                              72SPK
      END                                                               73SPK
```

Output

```
********** UNIVERSITY OF WATERLOO
********** SPARSE MATRIX PACKAGE
**********  ( S P A R S P A K )
**********      RELEASE   3
**********  (C) JANUARY 1984
********** ANSI FORTRAN
********** SINGLE PRECISION
********** LAST UPDATE JANUARY 1984


     OUTPUT UNIT FOR ERROR MESSAGES          6
     OUTPUT UNIT FOR STATISTICS              6

RSTRTA - RESTART SYSTEM-A ...

INBI   - INPUT OF RIGHT HAND SIDE ...

SOLVE1 - ENVELOPE SOLVE ...
     FACTORIZATION ALREADY DONE.

STATSA - SYSTEM-A STATISTICS ...
```

```
SIZE OF STORAGE ARRAY (MAXSA)              250
NUMBER OF EQUATIONS                         10
NUMBER OF OFF-DIAGONAL NONZEROS             18
TIME FOR ORDERING                           0.
STORAGE FOR ORDERING                        60.
TIME FOR ALLOCATION                         0.017
STORAGE FOR ALLOCATION                      60.
STORAGE FOR SOLUTION                        70.
TIME FOR FACTORIZATION                      0.017
TIME FOR SOLUTION                           0.017
OPERATIONS IN FACTORIZATION                 18.
OPERATIONS IN SOLUTION                      38.
TOTAL TIME REQUIRED                         0.050
MAXIMUM STORAGE REQUIRED                    70.

MAXIMUM ERROR                        1.19209e-07
```

Example 5

This example consists of four runs of essentially the same program, illustrating how the *SAVEA* and *RSTRTA* modules can be used to avoid repeating successfully completed computations when the execution cannot proceed further because of lack of working storage. In the first run, *MAXSA* was too small to accommodate the structure, and a message was printed indicating that *MAXSA* must be at least 999 in order to input the structure. A second run with *MAXSA*=999 was executed, and the structure was successfully input; however, the *ORDRA5* module could not execute because *MAXSA* was less than 1400. The module *SAVEA* was then executed and the run terminated.

The third run had *MAXSA*=2500, and the ordering and storage allocation were successfully performed. However, *ORDRA5* terminated with an error because it detected that too little storage was available for the numerical computation (*SOLVE5*), so *SAVEA* was again executed. Finally, the last run was executed with *MAXSA* set to 2509 (the maximum value, printed in the third run), and the solution to the problem was obtained.

Note:

The following examples were run using a single precision version of SPARSPAK-A on a DEC VAX 11/780 computer. The working storage required will therefore be different if a different version of SPARSPAK-A or a different computer is used.

Program 1

```
C--- SPARSPAK-A (ANSI FORTRAN) RELEASE III --- NAME = EX5A                    1SPK
C  (C) UNIVERSITY OF WATERLOO   JANUARY 1984                                  2SPK
C*************************************************************************  3SPK
C*************************************************************************  4SPK
C**********     M A I N L I N E     P R O G R A M     **********             5SPK
C*************************************************************************  6SPK
C*************************************************************************  7SPK
C                                                                            8SPK
C     FILE REQUIRMENT :                                                      9SPK
C        --- UNIT 9 - FOR SAVEA AND RSTRTA (SHOULD NOT BE                   10SPK
C                     DESTROYED).                                           11SPK
C                                                                           12SPK
C*************************************************************************  13SPK
C                                                                           14SPK
      INTEGER    I      , IERRA , IPRNTE, IPRNTS, MAXINT, MAXSA ,           15SPK
     1           MSGLVA, NVARS                                             16SPK
      REAL       MCHEPS, RATIOL, RATIOS, TIME                              17SPK
      REAL               S(900)                                            18SPK
      REAL               ERROR , FOUR  , ONE   , TWO   , ZERO              19SPK
C                                                                           20SPK
C*************************************************************************  21SPK
C                                                                           22SPK
      COMMON  /SPAUSR/  MSGLVA, IERRA , MAXSA , NVARS                       23SPK
      COMMON  /SPKSYS/  IPRNTE, IPRNTS, MAXINT, RATIOS, RATIOL,             24SPK
     1                  MCHEPS, TIME                                        25SPK
C                                                                           26SPK
C*************************************************************************  27SPK
C                                                                           28SPK
C     ----------------------                                                29SPK
C     INITIALIZE SPARSPAK-A.                                                30SPK
```

```
C      ----------------------                                          91SPK
       CALL  SPRSPK                                                   92SPK
       MAXSA = 900                                                    93SPK
C                                                                     94SPK
C      ---------------                                                95SPK
C      INPUT STRUCTURE.                                               96SPK
C      ---------------                                                97SPK
       CALL  IJBEGN                                                   98SPK
       DO  100  I = 2, 200                                            99SPK
           CALL  INIJ ( I, I-1, S )                                   40SPK
  100  CONTINUE                                                       41SPK
       CALL  IJEND ( S )                                              42SPK
       IF  ( IERRA .EQ. 0 )  GO TO 200                                43SPK
           CALL  STATSA                                               44SPK
           STOP                                                       45SPK
C                                                                     46SPK
  200  CONTINUE                                                       47SPK
C      ----------------------------                                   48SPK
C      DETERMINE SYMMETRIC ORDERING.                                  49SPK
C      ----------------------------                                   50SPK
       CALL  ORDRA5 ( S )                                             51SPK
       IF  ( IERRA .EQ. 0 )  GO TO 300                                52SPK
           CALL  SAVEA ( 3, S )                                       53SPK
           CALL  STATSA                                               54SPK
           STOP                                                       55SPK
C                                                                     56SPK
  300  CONTINUE                                                       57SPK
C      ----------------------                                         58SPK
C      INPUT NUMERICAL VALUES.                                        59SPK
C      ----------------------                                         60SPK
       ZERO = 0.0E0                                                   61SPK
       ONE  = 1.0E0                                                   62SPK
       TWO  = 2.0E0                                                   63SPK
       FOUR = 4.0E0                                                   64SPK
       DO  400  I = 1, 200                                            65SPK
           IF  ( I .GT. 1 )  CALL  INAIJ5 ( I, I-1, -ONE, S )         66SPK
           CALL  INBI ( I, TWO, S )                                   67SPK
  400  CONTINUE                                                       68SPK
       CALL  INBI ( 1, ONE, S )                                       69SPK
       CALL  INBI ( 200, ONE, S )                                     70SPK
C                                                                     71SPK
C      ------------------------------------------------               72SPK
C      PERFORM NUMERICAL FACTORIZATION AND SOLUTION.                  73SPK
C      ------------------------------------------------               74SPK
       CALL  SOLVE5 ( S )                                             75SPK
C                                                                     76SPK
C      -----------------                                              77SPK
C      OBTAIN STATISTICS.                                             78SPK
C      -----------------                                              79SPK
       CALL  STATSA                                                   80SPK
C                                                                     81SPK
C      ------------------------------------------------------------   82SPK
C      COMPUTE THE ACTUAL RELATIVE ERROR IN THE COMPUTED SOLUTION     83SPK
C      SINCE THE TRUE SOLUTION IS KNOWN.                              84SPK
C      ------------------------------------------------------------   85SPK
       ERROR = ZERO                                                   86SPK
       DO  500  I = 1, 200                                            87SPK
           ERROR = AMAX1(ERROR,ABS(S(I)-ONE))                         88SPK
  500  CONTINUE                                                       89SPK
       WRITE (IPRNTS,11)  ERROR                                       90SPK
   11  FORMAT (/10X, 35HMAXIMUM ERROR                , 1PE15.5 )      91SPK
C                                                                     92SPK
       STOP                                                           93SPK
       END                                                            94SPK
```

Output

```
********** UNIVERSITY OF WATERLOO
********** SPARSE MATRIX PACKAGE
**********  ( S P A R S P A K )
**********       RELEASE   3
**********  (C) JANUARY 1984
********** ANSI FORTRAN
********** SINGLE PRECISION
********** LAST UPDATE JANUARY 1984


     OUTPUT UNIT FOR ERROR MESSAGES              6
     OUTPUT UNIT FOR STATISTICS                  6

IJBEGN - BEGIN STRUCTURE INPUT ...

INIJ   - INPUT OF ADJACENCY PAIRS ...

IJEND  - END OF STRUCTURE INPUT ...

EMSGA  - SYSTEM-A ERROR ...

     IJEND  - ERROR NUMBER                     116
     INSUFF. STORAGE FOR ADJ. STRUCTURE.
     MAXSA MUST AT LEAST BE                    999

STATSA - SYSTEM-A STATISTICS ...

     NO STATISTICS AVAILABLE.
```

Program 2

```
C--- SPARSPAK-A (ANSI FORTRAN) RELEASE III --- NAME = EX5B                  1SPK
C  (C)  UNIVERSITY OF WATERLOO    JANUARY 1984                               2SPK
C******************************************************************        3SPK
C******************************************************************        4SPK
C**********     M A I N L I N E     P R O G R A M     **********            5SPK
C******************************************************************        6SPK
C******************************************************************        7SPK
C                                                                           8SPK
C     FILE REQUIREMENT :                                                    9SPK
C        --- UNIT 3 - FOR SAVEA AND RSTRTA (SHOULD NOT BE                  10SPK
C                     DESTROYED).                                          11SPK
C                                                                          12SPK
C******************************************************************       13SPK
C                                                                          14SPK
      INTEGER    I      , IERRA , IPRNTE, IPRNTS, MAXINT, MAXSA ,          15SPK
     1           MSGLVA, NVARS                                            16SPK
      REAL       MCHEPS, RATIOL, RATIOS, TIME                             17SPK
      REAL               S(999)                                           18SPK
      REAL               ERROR , FOUR  , ONE   , TWO   , ZERO             19SPK
C                                                                          20SPK
C******************************************************************       21SPK
```

```
C                                                                       22SPK
      COMMON  /SPAUSR/  MSGLVA, IERRA , MAXSA , NVARS                   23SPK
      COMMON  /SPKSYS/  IPRNTE, IPRNTS, MAXINT, RATIOS, RATIOL,         24SPK
     1                  MCHEPS, TIME                                    25SPK
C                                                                       26SPK
C***********************************************************************27SPK
C                                                                       28SPK
C     ----------------------                                            29SPK
C     INITIALIZE SPARSPAK-A.                                            30SPK
C     ----------------------                                            31SPK
      CALL  SPRSPK                                                      32SPK
      MAXSA = 999                                                       33SPK
C                                                                       34SPK
C     ----------------                                                  35SPK
C     INPUT STRUCTURE.                                                  36SPK
C     ----------------                                                  37SPK
      CALL  IJBEGN                                                      38SPK
      DO  100  I = 2, 200                                               39SPK
          CALL  INIJ ( I, I-1, S )                                      40SPK
  100 CONTINUE                                                          41SPK
      CALL  IJEND ( S )                                                 42SPK
      IF  ( IERRA .EQ. 0 )  GO TO 200                                   43SPK
          CALL  STATSA                                                  44SPK
          STOP                                                          45SPK
C                                                                       46SPK
  200 CONTINUE                                                          47SPK
C     ----------------------------                                      48SPK
C     DETERMINE SYMMETRIC ORDERING.                                     49SPK
C     ----------------------------                                      50SPK
      CALL  ORDRA5 ( S )                                                51SPK
      IF  ( IERRA .EQ. 0 )  GO TO 300                                   52SPK
          CALL  SAVEA ( 3, S )                                          53SPK
          CALL  STATSA                                                  54SPK
          STOP                                                          55SPK
C                                                                       56SPK
  300 CONTINUE                                                          57SPK
C     ----------------------                                            58SPK
C     INPUT NUMERICAL VALUES.                                           59SPK
C     ----------------------                                            60SPK
      ZERO = 0.0E0                                                      61SPK
      ONE  = 1.0E0                                                      62SPK
      TWO  = 2.0E0                                                      63SPK
      FOUR = 4.0E0                                                      64SPK
      DO  400  I = 1, 200                                               65SPK
          IF  ( I .GT. 1 )  CALL  INAIJ5 ( I, I-1, -ONE, S )            66SPK
          CALL  INBI ( I, TWO, S )                                      67SPK
  400 CONTINUE                                                          68SPK
      CALL  INBI ( 1, ONE, S )                                          69SPK
      CALL  INBI ( 200, ONE, S )                                        70SPK
C                                                                       71SPK
C     ---------------------------------------------                     72SPK
C     PERFORM NUMERICAL FACTORIZATION AND SOLUTION.                     73SPK
C     ---------------------------------------------                     74SPK
      CALL  SOLVE5 ( S )                                                75SPK
C                                                                       76SPK
C     -----------------                                                 77SPK
C     OBTAIN STATISTICS.                                                78SPK
C     -----------------                                                 79SPK
      CALL  STATSA                                                      80SPK
C                                                                       81SPK
C     ------------------------------------------------------------      82SPK
C     COMPUTE THE ACTUAL RELATIVE ERROR IN THE COMPUTED SOLUTION        83SPK
C     SINCE THE TRUE SOLUTION IS KNOWN.                                 84SPK
C     ------------------------------------------------------------      85SPK
      ERROR = ZERO                                                      86SPK
      DO  500  I = 1, 200                                               87SPK
```

```
            ERROR = AMAX1(ERROR,ABS(S(I)-ONE))                          88SPK
  500 CONTINUE                                                          89SPK
      WRITE (IPRNTS,11)  ERROR                                          90SPK
   11 FORMAT (/10X, 35HMAXIMUM ERROR                    , 1PE15.5 )     91SPK
C                                                                       92SPK
      STOP                                                              93SPK
      END                                                               94SPK
```

Output

```
********** UNIVERSITY OF WATERLOO
********** SPARSE MATRIX PACKAGE
**********  ( S P A R S P A K )
**********      RELEASE   3
**********  (C) JANUARY 1984
********** ANSI FORTRAN
********** SINGLE PRECISION
********** LAST UPDATE JANUARY 1984


     OUTPUT UNIT FOR ERROR MESSAGES          6
     OUTPUT UNIT FOR STATISTICS              6

IJBEGN - BEGIN STRUCTURE INPUT ...

INIJ   - INPUT OF ADJACENCY PAIRS ...

IJEND  - END OF STRUCTURE INPUT ...

ORDRA5 - NESTED DISSECTION ORDERING ...

EMSGA  - SYSTEM-A ERROR ...

     ORDRXI  (X=A,B AND I=1,2,3,4,5,6)
             - ERROR NUMBER                124
     INSUFF. STORAGE FOR ORDERING.
     MAXSA MUST AT LEAST BE               1400

SAVEA  - SAVE STORAGE VECTOR ...

STATSA - SYSTEM-A STATISTICS ...

     SIZE OF STORAGE ARRAY (MAXSA)         999
     NUMBER OF EQUATIONS                   200
     NUMBER OF OFF-DIAGONAL NONZEROS       398
     TOTAL TIME REQUIRED                    0.
     MAXIMUM STORAGE REQUIRED               0.
```

Program 3

```
C--- SPARSPAK-A (ANSI FORTRAN) RELEASE III --- NAME = EX5C           1SPK
C  (C)  UNIVERSITY OF WATERLOO    JANUARY 1984                        2SPK
C*****************************************************************  3SPK
```

```
C***********************************************************          4SPK
C**********     M A I N L I N E     P R O G R A M     **********          5SPK
C***********************************************************          6SPK
C***********************************************************          7SPK
C                                                                      8SPK
C     FILE REQUIRMENT :                                                9SPK
C        --- UNIT 3 - FOR SAVEA AND RSTRTA (SHOULD NOT BE             10SPK
C                     DESTROYED).                                      11SPK
C                                                                      12SPK
C***********************************************************          13SPK
C                                                                      14SPK
      INTEGER    I      , IERRA , IPRNTE, IPRNTS, MAXINT, MAXSA ,     15SPK
     1           MSGLVA, NVARS                                        16SPK
      REAL       MCHEPS, RATIOL, RATIOS, TIME                         17SPK
      REAL               S(2500)                                      18SPK
      REAL               ERROR , FOUR  , ONE   , TWO   , ZERO         19SPK
C                                                                      20SPK
C***********************************************************          21SPK
C                                                                      22SPK
      COMMON  /SPAUSR/  MSGLVA, IERRA , MAXSA , NVARS                  23SPK
      COMMON  /SPKSYS/  IPRNTE, IPRNTS, MAXINT, RATIOS, RATIOL,        24SPK
     1                  MCHEPS, TIME                                 25SPK
C                                                                      26SPK
C***********************************************************          27SPK
C                                                                      28SPK
C     ----------------------                                           29SPK
C     INITIALIZE SPARSPAK-A.                                           30SPK
C     ----------------------                                           31SPK
      CALL  SPRSPK                                                     32SPK
      MAXSA = 2500                                                     33SPK
C                                                                      34SPK
C     ----------------------------                                     35SPK
C     RESTORE STATE OF COMPUTATION.                                    36SPK
C     ----------------------------                                     37SPK
      CALL  RSTRTA ( 3, S )                                            38SPK
C                                                                      39SPK
C     ----------------------------                                     40SPK
C     DETERMINE SYMMETRIC ORDERING.                                    41SPK
C     ----------------------------                                     42SPK
      CALL  ORDRA5 ( S )                                               43SPK
      IF  ( IERRA .EQ. 0 )  GO TO 100                                  44SPK
          CALL  SAVEA ( 3, S )                                         45SPK
          CALL  STATSA                                                 46SPK
          STOP                                                         47SPK
C                                                                      48SPK
  100 CONTINUE                                                         49SPK
C     ----------------------                                           50SPK
C     INPUT NUMERICAL VALUES.                                          51SPK
C     ----------------------                                           52SPK
      ZERO = 0.0E0                                                     53SPK
      ONE  = 1.0E0                                                     54SPK
      TWO  = 2.0E0                                                     55SPK
      FOUR = 4.0E0                                                     56SPK
      DO  200  I = 1, 200                                              57SPK
          IF  ( I .GT. 1 )  CALL  INAIJ5 ( I, I-1, -ONE, S )           58SPK
          CALL  INAIJ5 ( I, I, FOUR, S )                               59SPK
          CALL  INBI ( I, TWO, S )                                     60SPK
  200 CONTINUE                                                         61SPK
      CALL  INBI ( 1, ONE, S )                                         62SPK
      CALL  INBI ( 200, ONE, S )                                       63SPK
C                                                                      64SPK
C     ---------------------------------------------                    65SPK
C     PERFORM NUMERICAL FACTORIZATION AND SOLUTION.                    66SPK
C     ---------------------------------------------                    67SPK
      CALL  SOLVE5 ( S )                                               68SPK
C                                                                      69SPK
```

```
C       -----------------                                                   70SPK
C       OBTAIN STATISTICS.                                                  71SPK
C       -----------------                                                   72SPK
        CALL  STATSA                                                        73SPK
C                                                                           74SPK
C       ------------------------------------------------------------        75SPK
C       COMPUTE THE ACTUAL RELATIVE ERROR IN THE COMPUTED SOLUTION          76SPK
C       SINCE THE TRUE SOLUTION IS KNOWN.                                   77SPK
C       ------------------------------------------------------------        78SPK
        ERROR = ZERO                                                        79SPK
        DO  300  I = 1, 200                                                 80SPK
            ERROR = AMAX1(ERROR,ABS(S(I)-ONE))                              81SPK
  300   CONTINUE                                                            82SPK
        WRITE (IPRNTS,11)  ERROR                                            83SPK
   11   FORMAT (/10X, 35HMAXIMUM ERROR                    , 1PE15.5 )       84SPK
C                                                                           85SPK
        STOP                                                                86SPK
        END                                                                 87SPK
```

Output

```
********** UNIVERSITY OF WATERLOO
********** SPARSE MATRIX PACKAGE
**********  ( S P A R S P A K )
**********       RELEASE   3
**********  (C) JANUARY 1984
********** ANSI FORTRAN
********** SINGLE PRECISION
********** LAST UPDATE JANUARY 1984


     OUTPUT UNIT FOR ERROR MESSAGES            6
     OUTPUT UNIT FOR STATISTICS                6

RSTRTA - RESTART SYSTEM-A ...

ORDRA5 - NESTED DISSECTION ORDERING ...

EMSGA  - SYSTEM-A ERROR ...

     ORDRXI  (X=A,B AND I=1,2,3,4,5,6)
             - ERROR NUMBER                  127
     INSUFF. STORAGE FOR SOLVEI.
     MAXSA MUST AT LEAST BE                 2509

SAVEA  - SAVE STORAGE VECTOR ...

STATSA - SYSTEM-A STATISTICS ...

     SIZE OF STORAGE ARRAY (MAXSA)          2500
     NUMBER OF EQUATIONS                     200
     NUMBER OF OFF-DIAGONAL NONZEROS         398
     TIME FOR ORDERING                         0.167
     STORAGE FOR ORDERING                   1400.
     TIME FOR ALLOCATION                       0.050
     STORAGE FOR ALLOCATION                 2324.
     TOTAL TIME REQUIRED                       0.217
     MAXIMUM STORAGE REQUIRED               2324.
```

Program 4

```
C--- SPARSPAK-A (ANSI FORTRAN) RELEASE III --- NAME = EX5D                    1SPK
C  (C) UNIVERSITY OF WATERLOO   JANUARY 1984                                  2SPK
C*************************************************************              3SPK
C*************************************************************              4SPK
C**********     M A I N L I N E     P R O G R A M     **********              5SPK
C*************************************************************              6SPK
C*************************************************************              7SPK
C                                                                            8SPK
C     FILE REQUIREMENT :                                                     9SPK
C        --- UNIT 3 - FOR SAVEA AND RSTRTA.                                 10SPK
C                                                                           11SPK
C*************************************************************             12SPK
C                                                                           13SPK
      INTEGER    I      , IERRA , IPRNTE, IPRNTS, MAXINT, MAXSA ,          14SPK
     1           MSGLVA, NVARS                                             15SPK
      REAL       MCHEPS, RATIOL, RATIOS, TIME                              16SPK
      REAL               S(2509)                                           17SPK
      REAL               ERROR , FOUR  , ONE   , TWO   , ZERO              18SPK
C                                                                           19SPK
C*************************************************************             20SPK
C                                                                           21SPK
      COMMON  /SPAUSR/  MSGLVA, IERRA , MAXSA , NVARS                       22SPK
      COMMON  /SPKSYS/  IPRNTE, IPRNTS, MAXINT, RATIOS, RATIOL,             23SPK
     1                  MCHEPS, TIME                                        24SPK
C                                                                           25SPK
C*************************************************************             26SPK
C                                                                           27SPK
C     ----------------------                                                28SPK
C     INITIALIZE SPARSPAK-A.                                                29SPK
C     ----------------------                                                30SPK
      CALL  SPRSPK                                                          31SPK
      MAXSA = 2509                                                          32SPK
C                                                                           33SPK
C     ----------------------------                                          34SPK
C     RESTORE STATE OF COMPUTATION.                                         35SPK
C     ----------------------------                                          36SPK
      CALL  RSTRTA ( 3, S )                                                 37SPK
C                                                                           38SPK
C     ----------------------------                                          39SPK
C     DETERMINE SYMMETRIC ORDERING.                                         40SPK
C     ----------------------------                                          41SPK
      CALL  ORDRA5 ( S )                                                    42SPK
      IF  ( IERRA .EQ. 0 )  GO TO 100                                       43SPK
          CALL  SAVEA ( 3, S )                                              44SPK
          CALL  STATSA                                                      45SPK
          STOP                                                              46SPK
C                                                                           47SPK
  100 CONTINUE                                                              48SPK
C     ----------------------                                                49SPK
C     INPUT NUMERICAL VALUES.                                               50SPK
C     ----------------------                                                51SPK
      ZERO = 0.0E0                                                          52SPK
      ONE  = 1.0E0                                                          53SPK
      TWO  = 2.0E0                                                          54SPK
      FOUR = 4.0E0                                                          55SPK
      DO  200  I = 1, 200                                                   56SPK
          IF  ( I .GT. 1 )  CALL  INAIJ5 ( I, I-1, -ONE, S )                57SPK
          CALL  INAIJ5 ( I, I, FOUR, S )                                    58SPK
```

```
          CALL  INBI ( I, TWO, S )                                    59SPK
  200 CONTINUE                                                        60SPK
      CALL  INBI ( 1, ONE, S )                                        61SPK
      CALL  INBI ( 200, ONE, S )                                      62SPK
C                                                                     63SPK
C     ------------------------------------------                      64SPK
C     PERFORM NUMERICAL FACTORIZATION AND SOLUTION.                   65SPK
C     ------------------------------------------                      66SPK
      CALL  SOLVE5 ( S )                                              67SPK
C                                                                     68SPK
C     ------------------                                              69SPK
C     OBTAIN STATISTICS.                                              70SPK
C     ------------------                                              71SPK
      CALL  STATSA                                                    72SPK
C                                                                     73SPK
C     ------------------------------------------------------          74SPK
C     COMPUTE THE ACTUAL RELATIVE ERROR IN THE COMPUTED SOLUTION      75SPK
C     SINCE THE TRUE SOLUTION IS KNOWN.                               76SPK
C     ------------------------------------------------------          77SPK
      ERROR = ZERO                                                    78SPK
      DO  300  I = 1, 200                                             79SPK
          ERROR = AMAX1(ERROR,ABS(S(I)-ONE))                          80SPK
  300 CONTINUE                                                        81SPK
      WRITE (IPRNTS,11)  ERROR                                        82SPK
   11 FORMAT (/10X, 35HMAXIMUM ERROR                    , 1PE15.5 )   83SPK
C                                                                     84SPK
      STOP                                                            85SPK
      END                                                             86SPK
```

Output

```
********** UNIVERSITY OF WATERLOO
********** SPARSE MATRIX PACKAGE
**********  ( S P A R S P A K )
**********       RELEASE   3
**********  (C) JANUARY 1984
********** ANSI FORTRAN
********** SINGLE PRECISION
********** LAST UPDATE JANUARY 1984


     OUTPUT UNIT FOR ERROR MESSAGES          6
     OUTPUT UNIT FOR STATISTICS              6

RSTRTA - RESTART SYSTEM-A ...

ORDRA5 - NESTED DISSECTION ORDERING ...

INAIJ5 - INPUT OF MATRIX COMPONENTS ...

INBI   - INPUT OF RIGHT HAND SIDE ...

SOLVE5 - GENERAL SPARSE SOLVE ...

STATSA - SYSTEM-A STATISTICS ...

     SIZE OF STORAGE ARRAY (MAXSA)          2509
     NUMBER OF EQUATIONS                     200
     NUMBER OF OFF-DIAGONAL NONZEROS         398
     TIME FOR ORDERING                         0.167
```

```
STORAGE FOR ORDERING                1400.
TIME FOR ALLOCATION                    0.050
STORAGE FOR ALLOCATION              2924.
STORAGE FOR SOLUTION                2509.
TIME FOR FACTORIZATION                 0.100
TIME FOR SOLUTION                      0.050
OPERATIONS IN FACTORIZATION          953.
OPERATIONS IN SOLUTION              1168.
TOTAL TIME REQUIRED                    0.367
MAXIMUM STORAGE REQUIRED            2509.

MAXIMUM ERROR                       2.38419e-07
```

Example 6

This is a program to illustrate how one might use SPARSPAK-A to choose a method. The matrix is 300×300, it has nonzeroes on the diagonal, the first column and the last row. The structure of the matrix is input using *IJBEGN*, *INIJ* and *IJEND*, and then saved on FORTRAN unit 3. The modules *ORDRA1*, *ORDRA3* and *ORDRA5* are then executed, each one followed by a call to *STATSA* to obtain the storage information. Note that *RSTRTA* is called after execution of *ORDRA1* and *ORDRA3*, to restore the package to the state that existed immediately after the structure inputting routines were executed. Note also that *SAVEA* could have been used after each ordering module (with different output unit numbers). After one of the methods was chosen, *RSTRTA* (with the appropriate unit number) could be used to initiate the computation, avoiding re-executing the ordering module corresponding to the method chosen.

Program

```
C--- SPARSPAK-A (ANSI FORTRAN) RELEASE III --- NAME = EX6                    1SPK
C  (C)  UNIVERSITY OF WATERLOO   JANUARY 1984                                2SPK
C***************************************************************           3SPK
C***************************************************************           4SPK
C**********     M A I N L I N E     P R O G R A M     **********           5SPK
C***************************************************************           6SPK
C***************************************************************           7SPK
C                                                                           8SPK
C     FILE REQUIREMENT :                                                    9SPK
C        --- UNIT 3 - FOR SAVEA AND RSTRTA.                                10SPK
C                                                                          11SPK
C***************************************************************          12SPK
C                                                                          13SPK
      INTEGER     I       , IERRA , MAXSA , MSGLVA, NVARS                  14SPK
      REAL                  S(7500)                                        15SPK
C                                                                          16SPK
C***************************************************************          17SPK
C                                                                          18SPK
      COMMON  /SPAUSR/  MSGLVA, IERRA , MAXSA , NVARS                      19SPK
C                                                                          20SPK
C***************************************************************          21SPK
C                                                                          22SPK
C     ----------------------                                               23SPK
C     INITIALIZE SPARSPAK-A.                                               24SPK
C     ----------------------                                               25SPK
      CALL  SPRSPK                                                         26SPK
      MAXSA = 7500                                                         27SPK
C                                                                          28SPK
C     ----------------                                                     29SPK
C     INPUT STRUCTURE.                                                     30SPK
C     ----------------                                                     31SPK
      CALL  IJBEGN                                                         32SPK
      DO  100  I = 1, 300                                                  33SPK
          CALL  INIJ ( I, 1, S )                                           34SPK
          CALL  INIJ ( 300, I, S )                                         35SPK
  100 CONTINUE                                                             36SPK
      CALL  IJEND ( S )                                                    37SPK
C                                                                          38SPK
C     ---------------------------                                          39SPK
C     SAVE STRUCTURE INFORMATION.                                          40SPK
C     ---------------------------                                          41SPK
```

```
      CALL  SAVEA ( 3, S )                                              42SPK
C                                                                       43SPK
C     ------------------------------------------                        44SPK
C     DETERMINE REVERSE CUTHILL-MCKEE ORDERING AND                      45SPK
C     OBTAIN STATISTICS.                                                46SPK
C     ------------------------------------------                        47SPK
      CALL  ORDRA1 ( S )                                                48SPK
      CALL  STATSA                                                      49SPK
C                                                                       50SPK
C     ------------------------------------------------                  51SPK
C     RESTORE STRUCTURE INFORMATION, DETERMINE ONE-WAY                  52SPK
C     DISSECTION ORDERING AND OBTAIN STATISTICS.                        53SPK
C     ------------------------------------------------                  54SPK
      CALL  RSTRTA ( 3, S )                                             55SPK
      CALL  ORDRA3 ( S )                                                56SPK
      CALL  STATSA                                                      57SPK
C                                                                       58SPK
C     ------------------------------------------------                  59SPK
C     RESTORE STRUCTURE INFORMATION, DETERMINE NESTED                   60SPK
C     DISSECTION ORDERING AND OBTAIN STATISTICS.                        61SPK
C     ------------------------------------------------                  62SPK
      CALL  RSTRTA ( 3, S )                                             63SPK
      CALL  ORDRA5 ( S )                                                64SPK
      CALL  STATSA                                                      65SPK
C                                                                       66SPK
      STOP                                                              67SPK
      END                                                               68SPK
```

Output

```
********** UNIVERSITY OF WATERLOO
********** SPARSE MATRIX PACKAGE
**********  ( S P A R S P A K )
**********      RELEASE   3
**********  (C) JANUARY 1984
********** ANSI FORTRAN
********** SINGLE PRECISION
********** LAST UPDATE JANUARY 1984


     OUTPUT UNIT FOR ERROR MESSAGES          6
     OUTPUT UNIT FOR STATISTICS              6

IJBEGN - BEGIN STRUCTURE INPUT ...

INIJ   - INPUT OF ADJACENCY PAIRS ...

IJEND  - END OF STRUCTURE INPUT ...

SAVEA  - SAVE STORAGE VECTOR ...

ORDRA1 - RCM ORDERING ...

STATSA - SYSTEM-A STATISTICS ...

     SIZE OF STORAGE ARRAY (MAXSA)          7500
     NUMBER OF EQUATIONS                     300
     NUMBER OF OFF-DIAGONAL NONZEROS        1194
     TIME FOR ORDERING                      0.217
     STORAGE FOR ORDERING                  2996.
```

```
      TIME FOR ALLOCATION                        0.017
      STORAGE FOR ALLOCATION                  2396.
      STORAGE FOR SOLUTION                    2398.
      TOTAL TIME REQUIRED                        0.233
      MAXIMUM STORAGE REQUIRED                2396.

RSTRTA - RESTART SYSTEM-A ...

ORDRA3 - ONE-WAY DISSECTION ORDERING ...

STATSA - SYSTEM-A STATISTICS ...

      SIZE OF STORAGE ARRAY (MAXSA)            7500
      NUMBER OF EQUATIONS                       300
      NUMBER OF OFF-DIAGONAL NONZEROS          1194
      TIME FOR ORDERING                          0.300
      STORAGE FOR ORDERING                    3297.
      TIME FOR ALLOCATION                        0.117
      STORAGE FOR ALLOCATION                  3300.
      STORAGE FOR SOLUTION                    3002.
      TOTAL TIME REQUIRED                        0.417
      MAXIMUM STORAGE REQUIRED                3300.

RSTRTA - RESTART SYSTEM-A ...

ORDRA5 - NESTED DISSECTION ORDERING ...

STATSA - SYSTEM-A STATISTICS ...

      SIZE OF STORAGE ARRAY (MAXSA)            7500
      NUMBER OF EQUATIONS                       300
      NUMBER OF OFF-DIAGONAL NONZEROS          1194
      TIME FOR ORDERING                          0.217
      STORAGE FOR ORDERING                    2696.
      TIME FOR ALLOCATION                        0.083
      STORAGE FOR ALLOCATION                  3599.
      STORAGE FOR SOLUTION                    3301.
      TOTAL TIME REQUIRED                        0.300
      MAXIMUM STORAGE REQUIRED                3599.
```

10. Appendix – implementation overview

In this section, we describe briefly the use of labelled common blocks in the internal implementation of SPARSPAK-A and the various methods of communication between modules.

10.1. User/module communication

As described in previous sections of this user guide, the user supplies a one-dimensional floating-point array *S*, from which all array storage is allocated. In particular, the interface allocates the first *NEQNS* storage locations in *S* for the solution vector of the linear system of equations. After all the interface modules for a particular method have been successfully executed, the user can retrieve the solution from these *NEQNS* locations.

There is one labelled common block *SPAUSR* that the user must provide, having four variables.

```
COMMON  /SPAUSR/  MSGLVA, IERRA, MAXSA, NEQNS
```

The variable *MAXSA* is the declared size of the one-dimensional floating-point array *S* and it must be set by user at the beginning of his program. For each module in the interface that allocates storage (e.g. *INIJ*, *IJEND*, *ORDRxi*), *MAXSA* is used to make sure that there is enough storage to carry out the particular phase.

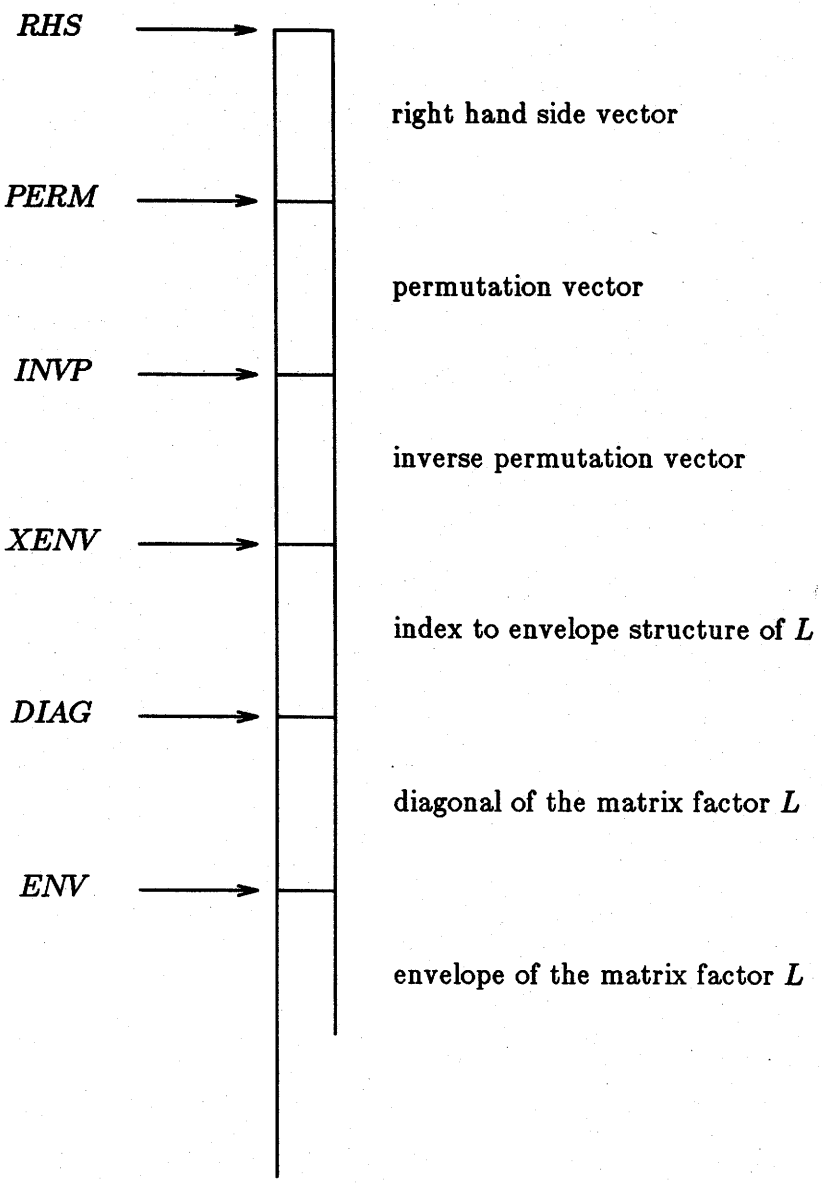
10.2. Module/module communication

There are several labelled common blocks used for communication among modules within the interface. Two important ones are the control block *SPACON* and the storage map block *SPAMAP*.

```
COMMON  /SPACON/  STAGE, MXUSED, MXREQD, NVARS, NEDGES, METHOD,
                  {and other method-related control variables}


COMMON  /SPAMAP/  PERM, INVP, RHS,
                  {and other method-related data
                  structure pointers}
```

The control block has fifty integer variables and contains control information about the specific problem being solved. There are fifty variables in the storage map block, which keep the locations (origins in *S*) of the various arrays used in the particular storage scheme. These storage schemes differ in complexity across the methods, so the same storage map block must be used in the corresponding routines *ORDRxi*, *INAIJi*, *INROWi*, *INMATi*, *SOLVEi*, and *ERESTi*. An example is given below.



Storage allocation for the symmetric envelope method (*ORDRA1*)

10.3. Save and restart implementation

The *SAVEA* routine saves the control information in the control block, the storage pointers in the storage map block, as well as the storage vector S. In this way, the state of the computation can be re-established by executing the module *RSTRTA*, which restores the control block and the storage map block, and the storage vector S.

The variable *MXUSED* in the control block is used to avoid saving irrelevant data from S. After the successful completion of each phase, *MXUSED* is set to the maximum number of storage locations in S used thus far. It is then only necessary to save the first *MXUSED* locations of S whenever the routine *SAVEA* is called.

Some operating systems allow a program to change the space it occupies in main storage during execution. Thus, in some installations the user of SPARSPAK-A may be able to dynamically increase or decrease the size of the working storage *S*. He can determine what the value of *MAXSA* should be by declaring the labelled common block *SPACON* in his mainline program, and examining the value of *MXREQD*. At the end of each successfully executed phase of the computation, *MXREQD* is set to the minimum value of *MAXSA* required to successfully execute the next phase of the computation.

It is often the case that when this dynamic growing of program space is provided, the effect is to increase the space allocated to the unlabelled COMMON, which is usually assigned the highest memory locations in the user's program area. In such a circumstance, the array *S* in the user's program would have to be declared in blank common.

10.4. Method checking

As we discussed in the introduction, using a particular "method" means calling the appropriate interface routines *ORDRxi*, *INAIJi*, *INROWi*, *INMATi*, *SOLVEi*, and *ERESTi*, where the last character is a numerical digit denoting the method. These ordering, input, solve, and relative error estimation modules cannot be mixed since they in general involve different data structures. In order to ensure that these modules are not inadvertently mixed by the user, *ORDRxi* sets the variable *METHOD* in the control block *SPACON* equal to $(10 \times i + k)$, where k is an integer that distinguishes orderings *A* and *B*. This variable is checked by subsequently executed input and solve modules.

10.5. Stage (sequence) checking

Another control variable that deserves comment is *STAGE*. As its name implies, it is used to keep track of the current step or stage of the execution. This variable is particularly important in connection with *SAVEA* and *RSTRTA* modules. In restarting the system using the *RSTRTA* routine, the variable *STAGE* in the control block *SPACON* is restored, and it indicates the last successfully completed stage or phase before the routine *SAVEA* was called. In this way, the execution can be restarted without repeating already successfully completed steps.

Another function of this variable is to enforce the correct execution sequence of the various interface routines. Before the actual execution of each interface routines, the variable *STAGE* is used to check that all previous interface modules have been successfully completed. This avoids producing erroneous results due to an improper processing sequence, or accidental omission of steps.

The content of the variable *STAGE* is only changed after a phase has been successfully executed. When an error occurs during the execution of the phase, the variable *STAGE* remains unchanged. This prevents the execution of all the subsequent phases, even if they are invoked by the user. The variable *STAGE* is also used by the modules to determine whether some initialization is necessary in a module, or whether part of the module has already successfully executed during a previous call to it.

10.6. Storage allocation of integer and floating-point arrays

The ANSI FORTRAN standard specifies that the number of bits used to represent integers and floating-point numbers are the same. However, some vendors provide the user with the option of specifying "short" integers, either explicitly in the declarations such as "INTEGER*2", or via a parameter to the FORTRAN processor which automatically represents all integers using fewer bits than used for floating-point numbers. Since a significant portion of the storage used in sparse matrix computations involves integer data for pointers, subscripts etc., it is desirable to try to exploit these "short" integer features whenever it makes sense to do so.

SPARSPAK-A contains parameters *RATIOS* and *RATIOL*, set in the module *SPRSPK*[4]. which specify the ratios of the number of bits used for floating-point numbers to the number used for "short" and "long" integers. For example, in a double precision IBM version of the package which exploits "short" integers, *RATIOS* is 4 and *RATIOL* is 2. Let $\lceil x \rceil$ be the smallest integer such that $\lceil x \rceil \geq x$. The package then uses *RATIOS* {*RATIOL*} to allocate only $\lceil p/RATIOS \rceil$ {$\lceil p/RATIOL \rceil$} elements of S for "short" {"long"} integer arrays of length p.

SPARSPAK-A assumes that the declaration of S that the user makes in his program is of the same type as that used for floating-point computation. We also make the reasonable assumption that $RATIOS \geq 1$ and $RATIOL \geq 1$.

10.7. Statistics gathering

SPARSPAK-A contains a labelled common block called *SPADTA* which appears below. These variables are used to provide the output described in Section 7.2.

```
COMMON /SPADTA/ ORDTIM, ALOCTM, FCTIME, SLVTIM, ERRTIM,
                FCTOPS, SLVOPS, ERROPS, ORDSTR, ALOSTR,
                SLVSTR, ERRSTR, OVERHD, ANORM , RCONDA,
                ERRFCT, RELEST, SVPAD(33)
```

In order to supply timing information, SPARSPAK-A assumes the existence of a real function *DTIME* which returns the processor execution time that has elapsed since *DTIME* was last referenced. Thus, the *DTIME* function is also installation dependent.

(4) Thus *SPRSPK* is an installation dependent subroutine.

11. References

[1] E.C.H. Chu and J.A. George, "An algorithm to estimate the error in Gaussian elimination without pivoting", Research report CS-84-21, Department of Computer Science, University of Waterloo (1984).

[2] J.A. George, "An automatic one-way dissection algorithm for irregular finite element problems", SIAM J. Numer. Anal., 17 (1980), pp. 740-751.

[3] J.A. George and J.W.H. Liu, "Algorithms for matrix partitioning and the numerical solution of finite element systems", SIAM J. Numer. Anal., 15 (1978), pp. 297-327.

[4] J.A. George and J.W.H. Liu, "An automatic nested dissection algorithm for irregular finite element problems", SIAM J. Numer. Anal., 15 (1978), pp. 1053-1069.

[5] J.A. George and J.W.H. Liu, "The design of a user interface for a sparse matrix package", ACM Trans. on Math. Software, 5 (1979), pp. 134-162.

[6] J.W.H. Liu, "On multiple elimination in the minimum degree algorithm", Technical Report No. 83-03, Department of Computer Science, York University, Downsview, Ontario (1983).

[7] J.W.H. Liu and A.H. Sherman, "Comparative analysis of the Cuthill-McKee and Reverse Cuthill-McKee ordering algorithms for sparse matrices", SIAM J. Numer. Anal., 13 (1976), pp. 198-213.