Science and Technology Facilities Council

HSL_ME57

PACKAGE SPECIFICATION HSL

1 SUMMARY

To **solve a sparse complex symmetric or Hermitian system of linear equations.** Given a sparse complex symmetric or Hermitian matrix $\mathbf{A} = \{a_{ij}\}_{n\times n}$ and an n-vector $\mathbf{b}$ or a matrix $\mathbf{B} = \{b_{ij}\}_{n\times r}$, this subroutine solves the system $\mathbf{Ax} = \mathbf{b}$ or the system $\mathbf{AX} = \mathbf{B}$. The matrix $\mathbf{A}$ can be either complex symmetric or Hermitian. There is an option for iterative refinement.

The multifrontal method is used. HSL_ME57 is a complex version of the code HSL_MA57. It is a direct method based on a sparse variant of Gaussian elimination and is discussed further by Duff and Reid, ACM Trans. Math. Software **9** (1983), 302-325. A detailed discussion on the MA57 strategy and performance is given by Duff, ACM Trans. Math. Software **30** (2004), 118-144. Relevant work on pivoting and scaling strategies is given by Duff and Pralet, SIAM Journal Matrix Analysis and Applications **27** (2005), 313-340. More recent work on static pivoting is given by Duff and Pralet, SIAM Journal Matrix Analysis and Applications **29** (2007), 1007-1024.

The HSL_ME57 package has a range of options including several sparsity orderings, multiple right-hand sides, partial solutions, error analysis, scaling, a matrix modification facility, the efficient factorization of highly rank deficient systems, and a stop and restart facility. Although the default settings should work well in general, there are several parameters available to enable the user to tune the code for his or her problem class or computer architecture.

The package has facilities for automatic restarts when storage limits are exceeded, the return of information on pivots, permutations, scaling, modifications, and the possibility to alter the pivots a posteriori.

ATTRIBUTES — Version: 1.1.3 (3 May 2023). **Types:** Real (single, double). **Remark:** HSL_ME57 is a Fortran 95 encapsulation of ME57 and offers some additional facilities to the Fortran 77 version. **Calls:** ME57 (optionally using MeTiS version 4.x), _GEMM, _TPSV, FD15, MF71, HSL_ZD11. **Language:** Fortran 95 + TR 15581 (allocatable components). **Date:** Original: August 2010. **Origin:** I.S. Duff, Rutherford Appleton Laboratory.

1.1 Calling sequences

Access to the package requires a USE statement

Single precision version

```
USE HSL_ME57_SINGLE
```

Double precision version

```
USE HSL_ME57_DOUBLE
```

If it is required to use more than one module at the same time, the derived types (Section 1.2) must be renamed in one of the USE statements.

In HSL_ME57_SINGLE, all reals (complex) are default reals (complex). In HSL_ME57_DOUBLE, all reals (complex) are double precision reals (complex (KIND = 1.0d0)). In both modules, all integers are default integers.

There are five principal subroutines for user calls (see Section 1.10 for further features):

1. The subroutine ME57_INITIALIZE must be called to initialize the structure for the factors. It may also be called to set default values for the components of the control structure. If non-default values are wanted for any of the control components, the corresponding components should be altered after the call to ME57_INITIALIZE.

2. ME57_ANALYSE accepts the pattern of **A** and chooses pivots for Gaussian elimination using a selection criterion to preserve sparsity. It subsequently constructs subsidiary information for the actual factorization by ME57_FACTORIZE. The user may provide the pivot sequence, in which case only the necessary information for ME57_FACTORIZE will be generated.

3. ME57_FACTORIZE factorizes a matrix **A** using the information from a previous call to ME57_ANALYSE. The actual pivot sequence used may differ from that of ME57_ANALYSE.
4. ME57_SOLVE uses the factors generated by ME57_FACTORIZE to solve a system of equations with one ($\mathbf{Ax} = \mathbf{b}$) or several ($\mathbf{AX} = \mathbf{B}$) right-hand sides, or to improve a given solution or set of solutions by iterative refinement.
5. ME57_FINALIZE deallocates the arrays held inside the structure for the factors. It should be called when all the systems involving its matrix have been solved unless the structure is about to be used for the factors of another matrix.

1.2 The derived data types

For each problem, the user must employ derived types defined by the module to declare structures for holding the matrix, holding its factors, controlling the factorization, and providing information.

1.2.1 Derived data type for the matrix

The derived type ZD11_TYPE is used to hold the matrix. The following components are employed

N is an INTEGER scalar which holds the order n of the matrix **A**. **Restriction:** N $\geq$ 1.

NE is an INTEGER scalar which holds the number of matrix entries. **Restriction:** NE $\geq$ 0.

VAL is a COMPLEX allocatable array of length at least NE, the leading part of which holds the values of the entries. Each pair of off-diagonal entries $a_{ij} = a_{ji}$ is represented as a single entry. Duplicated entries are summed.

ROW is an INTEGER allocatable array of length at least NE, the leading part of which holds the row indices of the entries.

COL is an INTEGER allocatable array of length at least NE, the leading part of which holds the column indices of the entries.

The other components of the type are not used.

1.3 Derived data type for control of the subroutines

The module contains a derived type called ME57_CONTROL with the following components

LP is an INTEGER scalar used by the subroutines as the output stream for error messages. If it is negative, these messages will be suppressed. The default value is 6.

WP is an INTEGER scalar used by the subroutines as the output stream for warning messages. If it is negative, these messages will be suppressed. The default value is 6.

MP is an INTEGER scalar used by the subroutines as the output stream for diagnostic printing. If it is negative, these messages will be suppressed. The default value is 6.

SP is an INTEGER scalar used by the subroutines as the output stream for statistics. If it is negative, these messages will be suppressed. The default value is -1. This component is presently not operative.

LDIAG is an INTEGER scalar used by the subroutines to control diagnostic printing. If LDIAG is less than 1, no messages will output. If the value is 1, only error messages will be printed. If the value is 2, then error and warning messages will be printed. If the value is 3, scalar data and a few entries of array data on entry and exit from each subroutine will be printed. If the value is greater than 3, all data will be printed on entry and exit. The default value is 2.

LA is an INTEGER scalar used by ME57_FACTORIZE. If LA $\geq$ NRLNEC (see Section 1.4), the real array that holds data for the factors is reallocated to have size LA. Otherwise, the array is not reallocated unless its size is less than NRLNEC, in which case it is reallocated with size NRLTOT (see Section 1.4). The default value is 0.

LIW is an INTEGER scalar used by ME57_FACTORIZE. If LIW $\geq$ NIRNEC (see Section 1.4), the integer array that holds data for the factors is reallocated to have size LIW. Otherwise, the array is not reallocated unless its size is less than NIRNEC, in which case it is reallocated with size NIRTOT (see Section 1.4). The default value is 0.

MAXLA is an INTEGER scalar used by ME57_FACTORIZE. An error return occurs if the real array that holds data for the factors is too small and reallocating it to have size changed by the factor MULTIPLIER would make its size greater than MAXLA. The default value is HUGE(0).

MAXLIW is an INTEGER scalar used by ME57_FACTORIZE. An error return occurs if the integer array that holds data for the factors is too small and reallocating it to have size changed by the factor MULTIPLIER would make its size greater than MAXLIW. The default value is HUGE(0).

MULTIPLIER is a REAL scalar used by ME57_FACTORIZE when a real or integer array that holds data for the factors is too small. The array is reallocated with its size changed by the factor MULTIPLIER. The default value is 2.0.

REDUCE is a REAL scalar that reduces the size of previously allocated internal workspace arrays if they are larger than currently required by a factor of REDUCE or more. The default value is 2.0.

NEMIN is an INTEGER scalar used by ME57_ANALYSE for the minimum number of eliminations in a step that is automatically accepted. If two adjacent steps can be combined and each has fewer eliminations, they are combined. The default value is 16.

THRESH is an INTEGER scalar used by ME57_ANALYSE to identify dense rows during pivot selection when the minimum degree algorithm from MA27 is being used (ORDERING = 3). It is the percentage density for a row to be regarded as dense. The default value is 100.

PIVOTING is a INTEGER scalar that is used to control numerical pivoting by ME57_FACTORIZE. It must have one of the following values:

1 Numerical pivoting will be performed, with relative pivot tolerance given by the component U.

2 No pivoting will be performed and an error exit will occur immediately a sign change is detected among the pivots. This is suitable for cases when **A** is thought to be definite and is likely to decrease the factorization time while still providing a stable decomposition.

3 No pivoting will be performed and an error exit will occur if a zero pivot is detected. This is likely to decrease the factorization time, but may be unstable if there is a sign change among the pivots.

4 No pivoting will be performed but the matrix will be altered if a non-positive pivot is encountered.

The default value is 1. The options 2 to 4 are only operational if the matrix is Hermitian. For the complex symmetric case, the value is always treated as if it were 1.

U is a REAL scalar that is used by ME57_FACTORIZE when the component PIVOTING has the value 1 to hold the relative pivot tolerance. The default value is 0.01. For problems requiring greater than average numerical care a higher value than the default would be advisable. Values greater than 0.5 are treated as 0.5 and less than 0.0 as 0.0.

TOLERANCE is a REAL scalar that is used by ME57_FACTORIZE. Any entry of modulus less than or equal to TOLERANCE is treated as zero. The default value is 10^{-20}. If RANK_DEFICIENT = 1, then blocks of entries less than TOLERANCE can be discarded during the factorization and the corresponding pivots are placed at the end of the ordering. In this case, a normal value for TOLERANCE could be 10^{-12}.

CONVERGENCE is a REAL scalar that is used by ME57_SOLVE. Iterative refinement will stop if the ratio of the norm of the scaled residual in successive iterations is greater than CONVERGENCE. The default value is 0.5.

FACTORBLOCKING is an INTEGER scalar used by ME57_FACTORIZE to determine the block size used for the Level 3 BLAS. The default value is 16.

SOLVEBLOCKING is an INTEGER scalar used by ME57_SOLVE to determine when to use Level 2 and Level 3 BLAS. The default value is 10.

ORDERING is an INTEGER scalar used by ME57_ANALYSE to determine the sparsity ordering that is used. The default value is 5. Possible values are:

- 0 AMD ordering using MC47 (without the detection of dense rows).
- 2 AMD ordering using MC47.
- 3 Minimum degree ordering as generated by the MA27 code.
- 4 MeTiS ordering is used. Note that the user needs to supply the MeTiS library (see http://www-users.cs.umn.edu/∼karypis/metis/metis/download.html).
  If it is not supplied and this option is requested, the routine will return immediately with AINFO%FLAG set to -10.
- 5 If MeTiS is available, the routine will make an automatic selection of the ordering choosing either MeTiS or MC47. If MeTiS is not available, then action is taken as if ORDERING were set to 2.
- ≥ 6 Currently is equivalent to setting PIVOTING = 5, but may be used for alternative orderings in later releases of HSL_ME57.

SCALING is an INTEGER scalar used by ME57_FACTORIZE to control the scaling. This has default value 1 and indicates that the matrix will be scaled using a symmetrized version of the HSL code MC64. Setting SCALING to any other value will suppress this option although this parameter may be used to call other scalings in future releases.

STATIC_TOLERANCE is a REAL scalar that is used by ME57_FACTORIZE to control the static pivoting (see Section 3). It has default value 0.0. If the value is positive, we will only accept delayed pivots to a level defined by STATIC_LEVEL and small pivots will be increased so that the factorization could be inaccurate. When static pivots are chosen, HSL_ME57_SOLVE will automatically use iterative refinement.

STATIC_LEVEL is a REAL scalar that is used by ME57_FACTORIZE. Static pivoting is only invoked if STATIC_TOLERANCE is greater than zero and the accumulated number of delayed pivots exceeds STATIC_LEVEL*N. The default value is zero.

RANK_DEFICIENT is an INTEGER scalar used by ME57_FACTORIZE that has default value 0. If RANK_DEFICIENT is set to 1, then when small entries (defined by TOLERANCE) are detected, they are removed and the corresponding pivots placed at the end of the factorization. This can be particularly efficient if the matrix is highly rank deficient.

1.4 Derived data type for information from ME57_ANALYSE

The module contains a derived type called ME57_AINFO with the following components:

FLAG is an INTEGER scalar. The value zero indicates that the subroutine has performed successfully. For nonzero values, see Section 1.9.1.

MORE is an INTEGER scalar that provides further information in the case of an error, see Section 1.9.1.

OOR is an INTEGER scalar which is set to the number of entries with one or both indices out of range.

DUP is an INTEGER scalar which is set to the number of duplicate off-diagonal entries.

STAT is an INTEGER scalar. In the case of the failure of an allocate or deallocate statement, it is set to the STAT value.

NSTEPS is an INTEGER scalar which is set to the number of nodes in the assembly tree (number of major steps in the factorization).

MAXFRT is an INTEGER scalar that holds the largest front size.

OPSA is a REAL scalar which is set to the number of floating-point additions required by the assembly of frontal matrices if no pivoting is performed. Numerical pivoting may increase the number of operations.

OPSE is a REAL scalar which is set to the number of floating-point operations required by the factorization if no pivoting is performed. Numerical pivoting may increase the number of operations.

NRLTOT and NIRTOT are INTEGER scalars that give the total amount of REAL and INTEGER words respectively required for a successful factorization without the need for data compression, provided no numerical pivoting is performed.

NRLNEC and NIRNEC are INTEGER scalars that give the total amount of REAL and INTEGER words required respectively for successful factorization allowing data compression, provided no numerical pivoting is performed.

NRLADU and NIRADU are INTEGER scalars that give the number of REAL and INTEGER words required respectively to hold the matrix factors if no numerical pivoting is performed.

NCMPA is an INTEGER scalar that holds the number of compresses of the internal data structure performed by ME57_ANALYSE.

ORDERING is an INTEGER scalar that records the actual ordering used by ME57_ANALYSE using the same numbering as for CONTROL%ORDERING.

1.5 Derived data type for information from ME57_FACTORIZE

The module contains a derived type called ME57_FINFO with the following components:

FLAG is an INTEGER scalar. The value of zero indicates that the subroutine has performed successfully. For nonzero values, see Section 1.9.2.

MORE is an INTEGER scalar that provides further information in the case of an error, see Section 1.9.2.

STAT is an INTEGER scalar. In the case of the failure of an allocate or deallocate statement, it is set to the STAT value.

MAXFRT is an INTEGER scalar that holds the largest front size.

OPSA is a REAL scalar which is set to the number of floating-point additions performed during assembly.

OPSE is a REAL scalar which is set to the number of floating-point operations performed during factorization.

OPSB is a REAL scalar which is set to the number of additional floating-point operations performed during factorization because of use of the BLAS.

NRLTOT and NIRTOT are INTEGER scalars that give the total amount of REAL and INTEGER words respectively required for a successful factorization without the need for data compression, provided the same pivots are used.

NRLNEC and NIRNEC are INTEGER scalars that give the amount of REAL and INTEGER words required respectively for successful factorization allowing data compression, provided the same pivots are used.

NEBDU is an INTEGER scalar that gives the total number of entries in the factorization.

NRLBDU and NIRBDU are INTEGER scalars that give the amount of REAL and INTEGER words used respectively to hold the factorization.

NCMPBR and NCMPBI are INTEGER scalars that hold the number of compresses of the real and integer data structure respectively required by the factorization.

NTWO is an INTEGER scalar that holds the number of 2×2 pivots used during the factorization.

NEIG is an INTEGER scalar that holds the number of negative eigenvalues of $\mathbf{A}$.

RANK is an INTEGER scalar that holds the rank of the original factorization. Note that, if static pivoting is used (see CONTROL%STATIC_TOLERANCE in Section 1.3), then the rank of the factorized matrix will always be n.

DELAY is an INTEGER scalar that holds the number of pivots passed up the tree because of numerical pivoting considerations.

SIGNC is an INTEGER scalar that holds the number of sign changes of pivot when ME57_CONTROL%PIVOTING is set to 3.

MODSTEP is an INTEGER scalar that holds the pivot step at which matrix modification is first performed when ME57_CONTROL%PIVOTING is set to 4. If no matrix modification is performed, MODSTEP is set to 0.

MAXCHANGE is a REAL scalar that is set to the value of the largest change made to a pivot when ME57_AINFO%MODSTEP is positive.

SMAX is a REAL scalar that is set to the value of the largest scaling factor.

SMIN is a REAL scalar that is set to the value of the smallest scaling factor.

STATIC is an INTEGER scalar that is set to 1 if static pivots have been chosen and is otherwise set to 0.

1.6 Derived data type for information from ME57_SOLVE

The module contains a derived type called ME57_SINFO with the following components:

FLAG is an INTEGER scalar. The value of zero indicates that the subroutine has performed successfully. For nonzero values, see Section 1.9.3.

COND and COND2 are REAL scalars that hold the condition number of the matrix if the optional parameter COND was included in the call to ME57_SOLVE (see Sections 1.8.4 and 1.10.4).

BERR and BERR2 are REAL scalars that hold the backward error (scaled residual) for the matrix if the optional parameter COND was included in the call to ME57_SOLVE (see Sections 1.8.4 and 1.10.4).

ERROR is a REAL scalar that holds an estimate of the error in the solution if the optional parameter COND was included in the call to ME57_SOLVE (see Sections 1.8.4 and 1.10.4).

STAT is an INTEGER scalar. In the case of the failure of an allocate or deallocate statement, it is set to the STAT value.

1.7 Derived data type for the factors of a matrix

The module contains a derived type called ME57_FACTORS. The components of ME57_FACTORS are used to pass data between the subroutines of the package and must not be altered by the user.

1.8 Argument lists

We use square brackets [] to indicate optional arguments.

1.8.1 The initialization subroutine

The initialization subroutine must be called for each structure used to hold the factors. It may also be called for a structure used to control the subroutines. Each argument is optional. A call with no arguments has no effect.

```
CALL ME57_INITIALIZE([FACTORS][,CONTROL])
```

FACTORS is optional, scalar, of INTENT(OUT) and of type ME57_FACTORS. On exit, its allocatable array components will be deallocated.

CONTROL is optional, scalar, of INTENT(OUT) and of type ME57_CONTROL. On exit, its components will have been given the default values specified in Section 1.3.

1.8.2 To analyse the sparsity pattern

```
CALL ME57_ANALYSE(CMPLX,MATRIX,FACTORS,CONTROL,AINFO[,PERM])
```

CMPLX is scalar, of INTENT(IN) and of type INTEGER. It must be set by the user to indicate whether the matrix is complex symmetric (CMPLX $=$ 1) or complex Hermitian (CMPLX $\neq$ 1).

MATRIX is scalar, of INTENT(IN) and of type ZD11_TYPE. The user must set the components N, NE, ROW, and COL. **Restrictions:** MATRIX%N $\geq$ 1 and MATRIX%NE $\geq$ 0.

FACTORS is scalar, of INTENT(INOUT) and of type ME57_FACTORS. It must have been initialized by a call to ME57_INITIALIZE or have been used for a previous calculation. In the latter case, the previous data will be lost but the allocatable arrays will not be reallocated unless they are found to be too small.

CONTROL is scalar, of INTENT(IN) and of type ME57_CONTROL. Its components control the action, as explained in Section 1.3.

AINFO is scalar, of INTENT(OUT) and of type ME57_AINFO. Its components provide information about the execution, as explained in Section 1.4.

PERM is an array of shape (n), optional, of INTENT(IN) and of type INTEGER. If present, PERM(i), i = 1, 2, ..., n, should be set to the position of variable i in the pivotal sequence.

1.8.3 To perform a factorization

```
CALL ME57_FACTORIZE(CMPLX,MATRIX,FACTORS,CONTROL,FINFO)
```

CMPLX is scalar, of INTENT(IN) and of type INTEGER. It must be set by the user to indicate whether the matrix is complex symmetric (CMPLX $=$ 1) or complex Hermitian (CMPLX $\neq$ 1).

MATRIX is scalar, of INTENT(IN) and of type ZD11_TYPE. The components N and NE must be unaltered since the call to ME57_ANALYSE. The user must set the component VAL to hold the real values of the entries. None of the components are altered by the subroutine.

FACTORS is scalar, of INTENT(INOUT) and of type ME57_FACTORS. It must be unaltered since the call to ME57_ANALYSE or a subsequent call to ME57_FACTORIZE.

CONTROL is scalar, of INTENT(IN) and of type ME57_CONTROL. Its components control the action, as explained in Section 1.3.

FINFO is scalar, of INTENT(OUT) and of type ME57_FINFO. Its components provide information about the execution, as explained in Section 1.5.

1.8.4 To solve a set of equations

```
CALL ME57_SOLVE(CMPLX,MATRIX,FACTORS,X,CONTROL,SINFO[,RHS,ITER,COND])
```

CMPLX is scalar, of INTENT(IN) and of type INTEGER. It must be set by the user to indicate whether the matrix is complex symmetric (CMPLX $=$ 1) or complex Hermitian (CMPLX $\neq$ 1).

MATRIX is scalar, of INTENT(IN) and of type ZD11_TYPE. It must be unaltered since the call to ME57_FACTORIZE.

FACTORS is scalar, of INTENT(IN) and of type ME57_FACTORS. It must be unaltered since the call to ME57_FACTORIZE.

X is an assumed-shape array with 1 or 2 dimensions, of INTENT(INOUT) and of type COMPLEX. If RHS is absent, X must be set by the user to the vector **b** or the matrix **B** and on return it holds the solution **x** or **X**. If RHS is present, X must be set by the user to an approximate solution and on return it holds an improved solution, obtained by one cycle of iterative refinement without any use of arithmetic with additional precision.

CONTROL is scalar, of INTENT(IN) and of type ME57_CONTROL. Its components control the action, as explained in Section 1.3.

SINFO is scalar, of INTENT(OUT) and of type ME57_SINFO. Its components provide information about the execution, as explained in Section 1.6.

RHS is an assumed-shape array of the same shape as X, optional, of INTENT(IN) and of type COMPLEX. If present, it must be set by the user to the vector **b** or the matrix **B**.

ITER is a scalar, optional, of INTENT(IN) and of type INTEGER. If RHS is not present, this parameter is not operational. If present, iterative refinement is performed (see Section 1.10.4).

COND is a scalar, optional, of INTENT(IN) and of type INTEGER. It is only operational if ITER and RHS are also present. If present, the condition number of the matrix is computed and is returned in SINFO%COND and SINFO%COND2, the backward error in SINFO%BERR and SINFO%BERR2, and an estimate of the error in SINFO%ERROR (see Section 1.10.4).

1.8.5 The finalization subroutine

```
CALL ME57_FINALIZE(FACTORS,CONTROL,INFO)
```

FACTORS is scalar, of INTENT(INOUT) and of type ME57_FACTORS. On exit, its allocatable array components will be deallocated. Without such finalization, the storage occupied is unavailable for other purposes.

CONTROL is scalar, of INTENT(IN) and of type ME57_CONTROL. Its components control the action, as explained in Section 1.3.

INFO is scalar, of INTENT(OUT) and of type INTEGER. On return, the value 0 indicates success. Any other value is the STAT value of a DEALLOCATE statement that has failed.

1.9 Error diagnostics

1.9.1 When analysing the sparsity pattern

A successful return from ME57_ANALYSE is indicated by AINFO%FLAG having the value zero. A negative value is associated with an error message which will be output on unit CONTROL%LP. Possible negative values are:

-1 Value of MATRIX%N out of range. MATRIX%N $<$ 1. AINFO%MORE is set to value of MATRIX%N.

-2 Value of MATRIX%NE out of range. MATRIX%NE $<$ 0. AINFO%MORE is set to value of MATRIX%NE.

-3 Failure of an allocate or deallocate statement. AINFO%STAT is set to the STAT value.

-9 The array PERM does not hold a permutation. AINFO%MORE holds first component at which an error was detected.

-10 The ordering from MeTiS was requested but the package was not available.

A positive flag value is associated with a warning message which will be output on unit AINFO%MP. Possible positive values are:

+1 Index (in MATRIX%ROW or MATRIX%COL) out of range. Action taken by subroutine is to ignore any such entries and continue. AINFO%OOR is set to the number of such entries. Details of the first ten are printed on unit CONTROL%MP.

+2 Duplicate indices. Action taken by subroutine is to keep the duplicates and then to sum corresponding reals when ME57_FACTORIZE is called. AINFO%DUP is set to the number of faulty entries. Details of the first ten are printed on unit CONTROL%MP.

+3 Both out-of-range indices and duplicates exist.

1.9.2 When factorizing the matrix

A successful return from ME57_FACTORIZE is indicated by FINFO%FLAG having the value zero. A negative value is associated with an error message which will be output on unit CONTROL%LP. In this case, no factorization will have been calculated. Possible negative values are:

-1 Value of MATRIX%N differs from the ME57_ANALYSE value. FINFO%MORE holds value of MATRIX%N.

-2 Value of MATRIX%NE out of range. MATRIX%NE $<$ 0. FINFO%MORE holds value of MATRIX%NE.

-3 Failure of an allocate or deallocate statement. FINFO%STAT is set to the STAT value.

-5 Zero pivot detected (CONTROL%PIVOTING has the value 2 or 3). FINFO%MORE is set to the pivot step at which this was detected.

-6 A change of sign of pivots has been detected (CONTROL%PIVOTING has the value 2). FINFO%MORE is set to the pivot step at which this was detected.

-7 The real array that holds data for the factors needs to be bigger than CONTROL%MAXLA.

-8 The integer array that holds data for the factors needs to be bigger than CONTROL%MAXLIW.

A positive flag value is associated with a warning message which will be output on unit CONTROL%MP. In this case, a factorization will have been calculated.

+4 Matrix is rank deficient. In this case, FINFO%RANK will be set to the rank of the original factorization, but the factorization is altered by changing all the zero pivots to one. This will enable the subsequent solution of consistent sets of equations.

+5 Pivots have different signs when CONTROL%PIVOTING has the value 3. FINFO%NEIG is set to the number of negative eigenvalues. Details of the first ten are printed on unit CONTROL%MP. FINFO%MORE is set to the number of sign changes.

1.9.3 When solving the system

A successful return from ME57_SOLVE is indicated by SINFO%FLAG having the value zero. A negative value is associated with an error message which will be output on unit CONTROL%LP. In this case, no solution will have been calculated. Possible negative values are:

-3 Failure of an allocate or deallocate statement. SINFO%STAT is set to the STAT value.

-11 Iterative refinement has failed to converge.

1.10 Further features

In this section, we describe features for enquiring about and manipulating the parts of the factorization constructed. These features will not be needed by a user who wants simply to solve systems of equations with matrix **A**.

The algorithm produces an $\mathbf{LDL}^*$ factorization of a permutation of a scaled version of **A**, where **L** is a unit lower triangular matrix and **D** is a block diagonal matrix with blocks of order 1 and 2. L^* will be equal to L^T if the matrix is complex symmetric and equal to L^H if the matrix is Hermitian. It is convenient to write this factorization in the form

$$\mathbf{PSASP^T} = \mathbf{LDL}^*,$$

where **P** is a permutation matrix and **S** the diagonal scaling matrix. The following subroutines are provided:

ME57_ENQUIRE returns **P**, **D**, or **S**.

ME57_ALTER_D alters **D**. Note that this means that we no longer have a factorization of the given matrix **A**.

ME57_PART_SOLVE solves one of the systems of equations $\mathbf{LSx} = \mathbf{PSb}$, $\mathbf{S^{-1}DS^{-1}x} = \mathbf{b}$, or $\mathbf{SL^*P^TS^{-1}x} = \mathbf{b}$, for one or more right-hand sides (**b** or **B**, respectively).

1.10.1 To return information on the analysis and factorization

```
CALL ME57_ENQUIRE(FACTORS[,PERM,PIVOTS,D,PERTURBATION,SCALING])
```

FACTORS is scalar, of INTENT(IN) and of type ME57_FACTORS. It must be unaltered since the call to ME57_FACTORIZE or a subsequent call to ME57_ALTER_D.

PERM is an array of shape (n), optional, of INTENT(OUT) and of type INTEGER. If present, PERM will be set to the pivot permutation selected by ME57_ANALYSE.

PIVOTS is an array of shape (n), optional, of INTENT(OUT) and of type INTEGER. If present, the index of pivot i will be placed in PIVOTS(i), i = 1, 2, ..., n, with its sign negative if it is the index of a 2 x 2 block.

D is an array of shape (2, n), optional, of INTENT(OUT) and of type COMPLEX. If present, the diagonal entries of $\mathbf{D^{-1}}$ will be placed in D(1,i), i = 1, 2, ..., n and the off-diagonal entries of $\mathbf{D^{-1}}$ will be placed in D(2,i), i = 1, 2, ..., $n-1$.

PERTURBATION is an array of shape (n), optional, of INTENT(OUT) and of type COMPLEX. If present and ME57_CONTROL%PIVOTING is set to 4, PERTURBATION will be set to the perturbation to the diagonal of the matrix (see Section 1.3). If present and the matrix is non-Hermitian or ME57_CONTROL%PIVOTING is not set to 4, PERTURBATION will be set to zero.

SCALING is an array of shape (n), optional, of INTENT(OUT) and of type REAL. If present and the parameter ME57_CONTROL%SCALING is set to 1, SCALING will be set to the values of the diagonal scaling matrix **S**. If present and ME57_CONTROL%SCALING is not set to 1, SCALING will be set to the vector with all entries equal to one.

1.10.2 To alter D

```
CALL ME57_ALTER_D(FACTORS,D,INFO)
```

FACTORS is scalar, of INTENT(INOUT) and of type ME57_FACTORS. It must be unaltered since the previous call to ME57_FACTORIZE or a subsequent call to ME57_ALTER_D.

D is an array of shape (2, n), of INTENT(IN) and of type COMPLEX. The diagonal entries of $\mathbf{D^{-1}}$ will be altered to D(1,i), i = 1, 2, ..., n and the off-diagonal entries of $\mathbf{D^{-1}}$ will be altered to D(2,i), i = 1, 2, ..., $n-1$.

INFO is scalar, of INTENT(OUT) and of type INTEGER. On return, the value 0 indicates success and the value $i > 0$ indicates that D(2, i) is nonzero, but is not part of a block of order 2 of **D**.

1.10.3 To perform a partial solution

```
CALL ME57_PART_SOLVE(CMPLX,FACTORS,CONTROL,PART,X,INFO)
```

CMPLX is scalar, of INTENT(IN) and of type INTEGER. It must be set by the user to indicate whether the matrix is complex symmetric (CMPLX $=$ 1) or complex Hermitian (CMPLX $\neq$ 1).

FACTORS is scalar, of INTENT(IN) and of type ME57_FACTORS. It must be unaltered since the call to ME57_FACTORIZE or a subsequent call to ME57_ALTER_D.

CONTROL is scalar, of INTENT(IN) and of type ME57_CONTROL. Its components control the action, as explained in Section 1.3.

PART is scalar, of INTENT(IN) and of type CHARACTER. It must have one of the values

L for solving $\mathbf{LSx = PSb}$ or $\mathbf{LSX = PSB}$

D for solving $\mathbf{S^{-1}DS^{-1}x = b}$ or $\mathbf{S^{-1}DS^{-1}X = B}$, or

U for solving $\mathbf{SL^{*}P^{T}S^{-1}x = b}$ or $\mathbf{SL^{*}P^{T}S^{-1}X = B}$.

X is an assumed-shape array with 1 or 2 dimensions, of INTENT(INOUT) and of type COMPLEX. It must be set by the user to the vector **b** or the matrix **B** and on return it holds the solution **x** or **X**.

INFO is scalar, of INTENT(OUT) and of type INTEGER. On return, the value 0 indicates success. Any other value is the STAT value of an ALLOCATE or DEALLOCATE statement that has failed.

1.10.4 Iterative refinement, condition number, and error estimates

If the parameter ITER is included in a call to ME57_SOLVE, then iterative refinement is invoked. If COND is also included, information is returned on the condition number and errors. We use the theory developed by Arioli, Demmel, and Duff (1989). We use the notation $\bar{\mathbf{x}}$ for the computed solution and a modulus sign on a vector or matrix to indicate the vector or matrix obtained by replacing all entries by their moduli. We define two scaled residuals:

$$\omega_1 \;=\; \max_i \left(\frac{|\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}|_i}{(|\mathbf{b}| + |\mathbf{A}||\bar{\mathbf{x}}|)_i} \right)$$

where the max is over all equations except those for which the numerator is nonzero and the denominator is small. For the exceptional equations, we define

$$\omega_2 \;=\; \max_i \left(\frac{|\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}|_i}{(|\mathbf{A}||\bar{\mathbf{x}}|)_i + ||\mathbf{A}_i||_\infty ||\bar{\mathbf{x}}||_\infty} \right).$$

ω_1 and ω_2 are returned in SINFO%BERR and SINFO%BERR2, respectively.

The computed solution $\bar{\mathbf{x}}$ is the exact solution of the equation

$$(\mathbf{A} + \delta\mathbf{A})\mathbf{x} \;=\; (\mathbf{b} + \delta\mathbf{b})$$

where $\delta\mathbf{A}_{ij} \leq \max(\texttt{BERR}, \texttt{BERR2})|\mathbf{A}|_{ij}$ and $\delta\mathbf{b}_i \leq \max(\texttt{BERR}|\mathbf{b}|_i, \texttt{BERR2}||\mathbf{A}_i||_\infty ||\bar{\mathbf{x}}||_\infty)$. Note that $\delta\mathbf{A}$ respects the sparsity in $\mathbf{A}$. An upper bound for the forward error is returned in ERROR that is computed as BERR*COND + BERR2*COND2 where COND and COND2 are condition numbers corresponding to κ_{ω_1} and κ_{ω_2}, respectively as defined in equation (15) of Arioli, Demmel, and Duff (1989).

Reference

[1] Arioli, M. Demmel, J. W., and Duff, I. S. (1989). Solving sparse linear systems with sparse backward error. *SIAM J. Matrix Anal. Appl.* **10**, 165-190.

2 GENERAL INFORMATION

Other routines called directly: ME57A/AD, ME57B/BD, ME57C/CD, ME57D/DD, ME57E/ED, ME57I/ID, FD15A/AD, and MC71A/AD. The package HSL_ZD11 is also used.

Input/output: Error, warning and diagnostic messages only. Error messages on unit CONTROL%LP and warning and diagnostic messages on unit CONTROL%WP and CONTROL%MP, respectively. These have default value 6, and printing of these messages is suppressed if the relevant unit number is set negative. These messages are also suppressed if ME57_CONTROL%LDIAG is less than 1.

Restrictions: MATRIX%N $\geq$ 1, MATRIX%NE $\geq$ 0.

MeTiS: The ME57 package uses the MeTiS graph partitioning library available from the University of Minnesota website. If MeTiS is not available then the user must compile with the supplied replacement subroutine METIS_NodeND. At present, ME57 only supports MeTiS version 4, not the latest version 5 releases.

3 METHOD

A version of sparse Gaussian elimination is used.

The ME57_ANALYSE entry (with CONTROL%ORDERING $\neq$ 1) chooses a pivot ordering based on either nested dissection or minimum degree using a generalized element model of the elimination to avoid storing the filled-in pattern explicitly.

The elimination is represented as an assembly tree with the order of elimination determined by a depth-first search of the tree.

The ME57_FACTORIZE entry factorizes the matrix by using the assembly and elimination ordering generated by ME57_ANALYSE. By default, the input matrix is first scaled using a symmetrized version of the HSL code MC64. At each stage in the multifrontal approach, pivoting and elimination are performed on full submatrices and, when diagonal 1×1 pivots would be numerically unstable, 2×2 diagonal blocks are used. The operations on the full submatrices are performed using the Level 3 BLAS. ME57_FACTORIZE can thus be used to factor systems that are not Hermitian definite and will perform well on machines with caches or levels of memory hierarchy.

If CONTROL%STATIC_TOLERANCE is set to a value greater than zero, static pivoting is invoked. This means that if, at any stage of the multifrontal elimination, there are fully summed rows and columns from which it is not possible to choose pivots because of the threshold criterion defined by CONTROL%U, then we first try to get pivots using a weaker threshold by successively trying values one tenth of the previous until a threshold of

$$\sqrt{(\texttt{CONTROL\%U*CONTROL\%STATIC\_TOLERANCE})}$$

is reached. If there are still fully summed rows and columns left, then the diagonal entries are replaced by CONTROL%STATIC_TOLERANCE times the largest modulus of an entry in the scaled matrix and are used as 1×1 pivots in the factorization. If, furthermore, CONTROL%STATIC_LEVEL is greater than zero, then CONTROL%STATIC_TOLERANCE is treated as 0 (uneliminated variables are delayed) until CONTROL%STATIC_LEVEL*N fully-summed variables have been delayed.

The ME57_SOLVE entry uses the factors from ME57_FACTORIZE to solve systems of equations either by loading the appropriate parts of the vectors into an array of the current front-size and using full matrix code employing the Level 2 and Level 3 BLAS or by indirect addressing at each stage, depending on the value of CONTROL%SOLVEBLOCKING and the size of the frontal matrix. If static pivots were chosen, then iterative refinement is automatically performed.

A fuller account of the original version is given by Duff and Reid (AERE-R.10533, 1982) and Duff and Reid, ACM Trans. Math. Software **9** (1983), 302-325.

A description of Version 1.0.0 of the MA57 package is given by Duff, ACM Trans. Math. Software **30** (2004), 118-144.

Some of the added features are described in "Strategies for scaling and pivoting for sparse symmetric indefinite problems" by Duff and Pralet (SIAM Journal Matrix Analysis and Applications **27** (2005), 313-340). A fuller discussion of our static pivoting strategy is given in the paper "Towards a stable static pivoting strategy for the sequential and parallel solution of sparse symmetric indefinite systems" by Duff and Pralet (SIAM Journal Matrix Analysis and Applications **29** (2007), 1007-1024).

4 EXAMPLE OF USE

We illustrate the use of the package on the solution of the single set of equations

$$\begin{pmatrix} 2+i & 3-i & & & \\ 3-i & 0 & 4 & & 6i \\ & 4 & i & 5+2i & \\ & & 5+2i & 0 & \\ & 6i & & & 1-3i \end{pmatrix} \mathbf{x} = \begin{pmatrix} 9+7i \\ -14+44i \\ 17+39i \\ 9+21i \\ 8+2i \end{pmatrix}$$

Note that this example does not illustrate all the facilities.

Program

```
! Simple example of use of HSL_ME57
PROGRAM MAIN
```

```
   USE HSL_ZD11_DOUBLE_COMPLEX
   USE HSL_ME57_DOUBLE
   IMPLICIT NONE
   TYPE(ZD11_TYPE) MATRIX
   TYPE(ME57_CONTROL) CONTROL
   TYPE(ME57_AINFO) AINFO
   TYPE(ME57_FINFO) FINFO
   TYPE(ME57_SINFO) SINFO
   TYPE(ME57_FACTORS) FACTORS

   integer, parameter :: wp = kind(0.0d0)

   COMPLEX(wp), ALLOCATABLE  :: B(:),X(:)
   INTEGER CMPLX,I,INFO,N,NE

! Matrix is complex symmetric
      CMPLX = 1

! Read matrix order and number of entries
      READ (5,*) N,NE
      MATRIX%N = N
      MATRIX%NE  = NE

! Allocate arrays of appropriate sizes
      ALLOCATE(MATRIX%VAL(NE), MATRIX%ROW(NE), MATRIX%COL(NE))
      ALLOCATE(B(N), X(N))

! Read matrix and right-hand side
      READ (5,*) (MATRIX%ROW(I),MATRIX%COL(I),MATRIX%VAL(I),I=1,NE)
      READ (5,*) B

! Initialize the structures
      CALL ME57_INITIALIZE(FACTORS,CONTROL)

! Analyse
      CALL ME57_ANALYSE(CMPLX,MATRIX,FACTORS,CONTROL,AINFO)
      IF(AINFO%FLAG .LT. 0) THEN
         WRITE(6,'(A,I2)') &
            ' Failure of ME57_ANALYSE with AINFO%FLAG=', AINFO%FLAG
         STOP
      END IF

! Factorize
      CALL ME57_FACTORIZE(CMPLX,MATRIX,FACTORS,CONTROL,FINFO)
      IF(FINFO%FLAG .LT. 0) THEN
         WRITE(6,'(A,I2)') &
            ' Failure of ME57 FACTORIZE with FINFO%FLAG=', FINFO%FLAG
         STOP
      END IF
```

```
! Solve
      X = B
      CALL ME57_SOLVE(CMPLX,MATRIX,FACTORS,X,CONTROL,SINFO)
      IF(SINFO%FLAG==0)WRITE(6,'(A,/,(5F14.10))')  &
         ' Solution is',X

! Clean up
      DEALLOCATE(MATRIX%VAL, MATRIX%ROW, MATRIX%COL, B, X)
      CALL ME57_FINALIZE(FACTORS,CONTROL,INFO)

END PROGRAM MAIN
```

we have as input

```
 5 7
 1 1  (2.0,1.0)
 1 2  (3.0,-1.0)
 2 3  (4.0,0.0)
 2 5  (0.0,6.0)
 3 3  (0.0,1.0)
 3 4  (5.0,2.0)
 5 5  (1.0,-3.0)
   (9.0,7.0) (-14.0,44.0)  (17.0,39.0)   (9.0,21.0)   (8.0,2.0)
```

and the output would be

```
Solution is
 1.0000000000  1.0000000000  2.0000000000  2.0000000000  3.0000000000
 3.0000000000  4.0000000000  4.0000000000  5.0000000000  5.0000000000
```