Chapter 2: Band Matrices

Subroutines: SGBCO, SGBFA, SGBSL, SGBDI

1. Overview

Purpose. The LINPACK subroutines in this chapter operate on general nonsymmetric band matrices. The operations performed include the triangular factorization of matrices, the estimation of the matrix condition number, the solution of simultaneous linear equations and the calculation of determinants.

Background. A band matrix is a matrix whose nonzero elements are all fairly near the main diagonal, specifically $a_{ij} = 0$ if $i-j > m_\ell$ or $j-i > m_u$. The integers m_ℓ and m_u are called the lower and upper band widths and $m = m_\ell + m_u + 1$ is the total band width. The subroutines for band matrices use less time and storage than the subroutines for full matrices if $2m_\ell + m_u < n$. Numerically, when the same matrix is input to the two sets of subroutines, the computed results are identical, but the elements are stored in different locations.

Tridiagonal matrices are the special case $m_\ell = m_u = 1$. They can be handled by the subroutines in this chapter, but the special subroutines in Chapter 7 are more efficient.

Let A be a real or complex band matrix of order n . There is an upper triangular band matrix U and a matrix L which is the product of elementary lower triangular band matrices and permutation matrices such that A = LU . This factorization can be used to solve linear equations Ax = b by solving successively L(Ux) = b and to compute the determinant of A as det(L)·det(U) . If needed the inverse of A can also be obtained.

The condition number $\kappa(A)$ is a quantity which measures the sensitivity of the solution x to errors in the matrix A and the right hand side b . If the relative error in A is of size ε , then the resulting relative error in x can be as large as $\kappa(A)\varepsilon$. Errors in A can arise in many ways. In particular, the effect of the roundoff error introduced by the subroutines in this chapter can usually be assessed by taking ε to be a small multiple of the rounding unit.

It is possible to efficiently compute a quantity RCOND which is an estimate of the reciprocal condition, $1/\kappa(A)$. If RCOND is approximately 10^{-d} then the elements of x can usually be expected to have d fewer significant figures of accuracy than the elements of A . Consequently, if RCOND is so small that in floating point arithmetic it is negligible compared to 1.0 , then x may have no significant figures. On most computers, this condition may be tested by the logical expression

1.0 + RCOND .EQ. 1.0 .

When this expression is true, the matrix can be considered to be "singular to working precision." As a special case, if exact singularity is detected, RCOND may be set to 0.0 .

If A is badly scaled, then the interpretation of RCOND is more delicate. For a detailed discussion of RCOND , including the consequences of unusual scaling in A and errors in b , see Chapter 1, sections 4 and 6.

2. Usage

When Gaussian elimination is performed on a band matrix, the pivoting causes the introduction of nonzero elements outside the band. If m_ℓ and m_u are the lower and upper band widths of the original matrix, then its two triangular factors have band widths m_ℓ and $m_\ell + m_u$ and storage must be provided for the extra m_ℓ diagonals. This is illustrated by the following example with $n = 9$, $m_\ell = 2$ and $m_u = 3$. The original matirx is

$$\begin{matrix}
11 & 12 & 13 & 14 & 0 & 0 & 0 & 0 & 0 \\
21 & 22 & 23 & 24 & 25 & 0 & 0 & 0 & 0 \\
31 & 32 & 33 & 34 & 35 & 36 & 0 & 0 & 0 \\
0 & 42 & 43 & 44 & 45 & 46 & 47 & 0 & 0 \\
0 & 0 & 53 & 54 & 55 & 56 & 57 & 58 & 0 \\
0 & 0 & 0 & 64 & 65 & 66 & 67 & 68 & 69 \\
0 & 0 & 0 & 0 & 75 & 76 & 77 & 78 & 79 \\
0 & 0 & 0 & 0 & 0 & 86 & 87 & 88 & 89 \\
0 & 0 & 0 & 0 & 0 & 0 & 97 & 98 & 99
\end{matrix}$$

The band storage requires $2m_\ell + m_u + 1 = 8$ rows arranged as follows. The * indicates elements which are never referenced, but for which the space must be provided. The + indicates elements which may be filled in during the elimination.

$$\begin{matrix}
* & * & * & * & * & + & + & + & + \\
* & * & * & * & + & + & + & + & + \\
* & * & * & 14 & 25 & 36 & 47 & 58 & 69 \\
* & * & 13 & 24 & 35 & 46 & 57 & 68 & 79 \\
* & 12 & 23 & 34 & 45 & 56 & 67 & 78 & 89 \\
11 & 22 & 33 & 44 & 55 & 66 & 77 & 88 & 99 \\
21 & 32 & 43 & 54 & 65 & 76 & 87 & 98 & * \\
31 & 42 & 53 & 64 & 75 & 86 & 97 & * & *
\end{matrix}$$

The elements in position (i,j) of the original matrix is stored in position (k,j) of the band array where $k = i-j+m$ and $m = m_\ell + m_u + 1$. The following program segment will transfer a band matrix from conventional full matrix storage in A to band storage in ABD.

```
      M = ML + MU + 1
      DO 20 J = 1, N
         I1 = MAX0(1, J-MU)
         I2 = MIN0(N, J+ML)
         DO 10 I = I1, I2
            K = I - J + M
            ABD(K,J) = A(I,J)
   10    CONTINUE
   20 CONTINUE
```

Single precision, general band matrices. The four subroutines for single precision, general band matrices are SGBCO , SGBFA , SGBSL , and SGBDI . Ordinarily, SGBCO or SGBFA will be called once to factor a particular matrix and then SGBSL and SGBDI will be called to apply the factorization as many times as needed.

SGBCO uses Gaussian elimination with partial pivoting to compute the LU factorization of a band matrix and then estimates its condition. The calling sequence is

CALL SGBCO(ABD,LDA,N,ML,MU,IPVT,RCOND,Z) .

On entry,

ABD is a doubly subscripted array with dimension (LDA,N) which contains the band of the matrix A whose factorization is to be computed. The columns of A are stored in the columns of ABD and the diagonals of A are stored in rows ML + 1 through 2*ML + MU + 1 of ABD . The first ML rows are used for work space and output.

LDA is the leading dimension of the array ABD , which must satisfy LDA $\geq$ 2*ML + MU + 1 .

N is the order of A and the number of elements in the vectors IPUT and Z.

ML is the number of diagonals below the main diagonal in the band, which must satisfy $0 \leq ML < N$.

MU is the number of diagonals above the main diagonal in the band, which must satisfy $0 \leq MU < N$ and preferably should satisfy $ML \leq MU$.

On return,

ABD contains in its first ML + MU + 1 rows an upper triangular band matrix U and in its next ML rows the multipliers necessary to construct a matrix L so that A = LU . The ML + MU by ML + MU upper left triangle and the

ML by ML lower right triangle are never referenced.

IPVT is a singly subscripted integer array of dimension N which contains the pivot information necessary to construct the permutations in L . Specifically, IPVT(K) is the index of the K-th pivot row.

RCOND is an estimate of the reciprocal condition, $1/\kappa(A)$. If RCOND is so small that the logical expression 1.0 + RCOND .EQ. 1.0 is true, then A can usually be regarded as singular to working precision. If RCOND is exactly zero, then SGBSL will divide by zero.

Z is a singly subscripted array of dimension N used for work space. If A is close to a singular matrix, then Z will contain an approximate null vector in the sense that $\|Az\| = \text{RCOND}\cdot\|A\|\cdot\|z\|$ (see Section 4).

SGBFA should be used in place of SGBCO if the condition estimate is not needed. The calling sequence is

```
CALL SGBFA(ABD,LDA,N,ML,MU,IPVT,INFO)
```

On entry,

ABD is a doubly subscripted array with dimension (LDA,N) which contains the band of the matrix A whose factorization is to be computed. The columns of A are stored in the columns of ABD and the diagonals of A are stored in rows ML + 1 through 2*ML + MU + 1 of ABD . The first ML rows are used for work space and output.

LDA is the leading dimension of the array ABD , which must satisfy $\text{LDA} \geq 2*\text{ML} + \text{MU} + 1$.

N is the order of A and the number of elements in the vector IPVT .

ML is the number of diagonals below the main diagonal in the band, which must satisfy $0 \leq \text{ML} < \text{N}$.

MU is the number of diagonals above the main diagonal in the band, which must satisfy $0 \leq \text{MU} < \text{N}$ and preferably should satisfy $\text{ML} \leq \text{MU}$.

On return,

ABD contains in its first ML + MU + 1 rows an upper triangular band matrix U and in its next ML rows the multipliers necessary to construct a matrix L so that $A = LU$. The $ML + MU$ by $ML + MU$ upper left triangle and the ML by ML lower right triangle are never referenced.

IPVT is a singly subscripted integer array of dimension N which contains the pivot information necessary to construct the permutations in L . Specifically, IPVT(K) is the index of the K-th pivot row.

INFO is an integer returned by SGBFA which, if it is 0 , indicates that SGBSL can be safely used. If $\text{INFO} = K \neq 0$, then SGBSL will divide by $U(K,K) = 0.0$. If U has several zero diagonal elements, K will be the index of the last one. Although a nonzero INFO technically indicates singularity, RCOND is a more reliable indicator.

SGBCO is usually called first to factor the matrix and estimate its condition. The actual factorization is done by SGBFA which can be called in place of SGBCO if the condition estimate is not needed. The time required by SGBCO is roughly $(1 + 6/ML)$ times the time required by SGBFA .

Since any matrix has an LU factorization, there is no error return from SGBCO or SGBFA . However, the factors can be singular and consequently unusable by SGBSL . Either RCOND or INFO should be tested before calling SGBSL .

SGBSL uses the LU factorization of a band matrix A to solve linear systems of the form

$$Ax = b$$

or

$$A^T x = b$$

where A^T is the transpose of A . The calling sequence is

```
CALL SGBSL(ABD,LDA,N,ML,MU,IPVT,B,JOB) .
```

On entry,

ABD is a doubly subscripted array with dimension (LDA,N) which contains the factorization computed by SGBCO or SGBFA . It is not changed by SGBSL .

LDA is the leading dimension of the array ABD .

N is the order of the matrix A and the number of elements in the vectors B and IPVT .

ML is the number of diagonals below the main diagonal.

MU is the number of diagonals above the main diagonal.

IPVT is a singly subscripted integer array of dimension N which contains the pivot information from SGBCO or SGBFA .

B is a singly subscripted array of dimension N which contains the right hand side b of a system of simultaneous linear equations $Ax = b$ or $A^Tx = b$.

JOB indicates what is to be computed. If JOB is 0 , the system $Ax = b$ is solved and if JOB is nonzero, the system $A^Tx = b$ is solved.

On return,

B contains the solution, x .

If the upper triangular factor of A has a zero element on the diagonal (a situation that will cause INFO ≠ 0 in SGBFA or RCOND = 0.0 in SGBCO), a division by zero will occur in SGBSL . Technically this means that the matrix A is singular, but it is often caused by incorrect setting of LDA or other improper use of the subroutines.

SGBDI uses the LU factorization of a matrix to compute its determinant. The calling sequence is

```
CALL SGBDI(ABD,LDA,N,ML,MU,IPVT,DET) .
```

On entry,

ABD is a doubly subscripted array with dimension (LDA,N) which contains the factorization computed by SGBCO or SGBFA . It is not changed by SGBDI .

LDA is the leading dimension of the array ABD .

N is the order of the matrix A and the number of elements in the vector IPVT.

ML is the number of diagonals below the main diagonal.

MU is the number of diagonals above the main diagonal.

IPVT is a singly subscripted integer array of dimension N which contains the pivot information from SGBCO or SGBFA .

On return,

DET is a singly subscripted array with 2 elements which contains the determinant of A in the form det(A) = DET(1)*10.0**DET(2) , although this expression may underflow or overflow if evaluated. DET(1) is normalized so that

$1.0 \leq |\mathrm{DET}(1)| < 10.0$ or $\mathrm{DET}(1) = 0.0$. DET(2) contains an integer stored as a real number.

No direct provision is made for computing the inverse of a band matrix because the inverse is usually a full n by n matrix which cannot be stored in the band storage. Moreover, calculations formulated in terms of matrix inverses are invariably more efficient when expressed in terms of the solution of sets of linear equations. However, an example in section 3 shows how the inverse can be obtained using SGBSL .

Double precision, general band matrices. The calling sequences of the double precision, general band subroutines DGBCO , DGBFA , DGBSL and DGBDI are the same as those of the corresponding single precision "S" subroutines except that A , B , RCOND , DET and Z are DOUBLE PRECISION variables.

Complex, general band matrices. The calling sequences of the complex, general band subroutines CGBCO , CGBFA , CGBSL and CGBDI are the same as those of the corresponding single precision "S" subroutines except that A , B , DET and Z are COMPLEX variables, RCOND is a REAL variable and the system solved by CGBSL when JOB is nonzero involves the complex conjugate transpose of A .

Double precision complex, general band matrices. In those computing systems where they are available, the calling sequences of the double precision complex, general band subroutines ZGBCO , ZGBFA , ZGBSL and ZGBDI are the same as those of the corresponding single precision "S" subroutines except that A , B , DET and Z are COMPLEX*16 varibles, RCOND is a DOUBLE PRECISION variable and the system solved by ZGBSL when JOB is nonzero involves the complex conjugate transpose of A .

3. Examples

The first example is a complete program which factors a pentadiagonal matrix, tests for near singularity, and then solves a single system of linear equations. The system is

$$\begin{bmatrix} 11 & 12 & 13 & & & & \\ 21 & 22 & 23 & 24 & & 0 & \\ 31 & 32 & 33 & 34 & 35 & & \\ & . & . & . & . & . & \\ & & . & . & . & . & . \\ & 0 & & 86 & 87 & 88 & 89 \\ & & & & 97 & 98 & 99 \end{bmatrix} X = \begin{bmatrix} 1 \\ 1 \\ 1 \\ . \\ . \\ 1 \\ 1 \end{bmatrix}$$

```
      REAL ABD(7,9),B(9),Z(9),RCOND,T
      INTEGER IPVT(9)
      DATA LDA /7/
      N = 9
      ML = 2
      MU = 2
      M = ML + MU + 1
      DO 20 J = 1, N
         I1 = MAXO(1, J-MU)
         I2 = MINO(N, J+ML)
         DO 10 I = I1, I2
            K = I - J + M
            ABD(K,J) = 10 * I + J
   10    CONTINUE
   20 CONTINUE
C
      CALL SGBCO(ABD,LDA,N,ML,MU,IPVT,RCOND,Z)
      WRITE(6,30) RCOND
   30 FORMAT(9H RCOND = , E15.5)
      T = 1.0 + RCOND
      IF (T .EQ. 1.0) GO TO 90
C
      DO 40 I = 1, N
         B(I) = 1.0
   40 CONTINUE
      CALL SGBSL(ABD,LDA,N,ML,MU,IPVT,B,0)
      DO 50 I = 1, N
         WRITE(6,60) B(I)
   50 CONTINUE
   60 FORMAT(F15.6)
      STOP
C
   90 WRITE(6,99)
   99 FORMAT(40H MATRIX IS SINGULAR TO WORKING PRECISION)
      STOP
      END
```

The next segment finds the inverse of a pentadiagonal matrix of order 50 by successively solving $Ax_j = e_j$ where e_j is the j-th column of the identify and x_j is the j-th column of A^{-1} . Note that different dimensions are used for ABD and AINV . This program is primarily for illustration; its actual use should be avoided whenever possible.

```
    DIMENSION ABD(7,50),AINV(50,50),IPVT(50)
    DATA LDA/7/, M/5/, M2/2/
    N = 50
    DO 15 J = 1, N
       I1 = MAX0(1, J-M2)
       I2 = MIN0(N, J+M2)
       DO 10 I = I1, I2
          K = I - J + M
          ABD(K,J) = (I,J)-th element of A
10     CONTINUE
15  CONTINUE
    CALL SGBFA(ABD,LDA,N,M2,M2,IPVT,INFO)
    IF (INFO .NE. 0) GO TO . . .
    DO 30 J = 1, N
       DO 20 I = 1, N
          AINV(I,J) = 0.0
20     CONTINUE
       AINV(J,J) = 1.0
       CALL SGBSL(ABD,LDA,N,M2,M2,IPVT,AINV(1,J),0)
30  CONTINUE
       .
       .
       .
```

4. Programming Details

The algorithms employed by the subroutines for band matrices are essentially the same as those employed by the subroutines for full matrices. Only the storage arrangement and the ranges of the DO-loops are different. Except for the operations on zeros, the two sets of routines do the same arithmetic operations in the same order. Thus anyone attempting to understand the band subroutines in detail is advised to first study the full subroutines.

SGBFA

The principal integer variables have the following meanings:

K	index of the pivot column.
L	index in the band structure of the pivot row.
IPVT(K)	index in the full structure of the pivot row.
M	row index in the band structure of the diagonal elements.
JZ	column index of the last nonzero element in the last active row.
JU	column index of the last nonzero element of the pivot row.
J	column index during elimination.

MM index in the band structure of the K-th row.

LM number of nonzero elements below the diagonal in the K-th column.

The following example, taken from section 2, illustrates how these variables are used. It has N = 8 , MU = 3 and ML = 2 .

$$
\begin{array}{r|ccccccccc}
 & \overset{\text{K}}{\downarrow} & & & & \overset{\text{JU}}{\downarrow} & \overset{\text{JZ}}{\downarrow} & & & \\
 & * & * & * & * & * & 0_1 & + & + & + \\
 & * & * & * & * & 0_0 & 0_1 & + & + & + \\
 & * & * & * & 14 & \underline{25} & 36 & 47 & 58 & 69 \\
 & * & * & 13 & \underline{24} & 35 & 46 & 57 & 68 & 79 \\
 & * & 12 & \underline{23} & 34 & 45 & 56 & 67 & 78 & 89 \\
\text{M} \rightarrow & 11 & \underline{22} & 33 & 44 & 55 & 66 & 77 & 88 & 99 \\
\text{LM} \left\{ \text{MM} \nearrow \right. & \underline{21} & 32 & 43 & 54 & 65 & 76 & 87 & 98 & * \\
\text{L} \nearrow & 31 & 42 & 53 & 64 & 75 & 86 & 97 & * & *
\end{array}
$$

Elements labeled with integers are input and the integers indicate their position in the original matrix. Elements labeled * are never referenced, elements labeled + are the auxiliary storage that is filled in by the pivoting. The element shown as 0_0 is set to zero before the elimination begins. The situation during the first step of the elimination, that is K = 1 , is shown. The diagonal pointer remains at M = 6 throughout. With JZ = 6 , the elements shown as 0_1 are set to zero. The pivot search involves the LM+1 = 2+1 = 3 nonzero elements in the first column. Suppose that the second of these, labeled 21, is the largest and is chosen as pivot. Then the underlined elements constitute the pivot row and so L = 7 , IPVT(1) = 2 and JU = 5 . The elimination involves the column index J = 2,...,5 and the two row pointers, MM = 5,...,2 and L = 6,...,3 . The underlined elements are exchanged with the elements directly above them. This causes one element of fill-in, in the position labeled 0_0 . Note that the two 0_1's would have been filled in if 31 had been chosen as pivot. Later values of K will lead to the remaining +'s being zeroed and then possibly overwritten with nonzeros.

SGBCO and SGBSL

These are versions of SGECO and SGESL , modified to handle the band storage. It is assumed that complete fill-in has occurred during the factorization so that the upper triangular factor U has ML + MU diagonals above the main diagonal even though, for some matrices, portions of these diagonals may be zero. The variable LM contains the length of the vectors processed in the inner loop, that is by SAXPY and SDOT . For the systems involving U or U^T , LM usually has the value ML+MU , except near the bottom of U where the vectors become shorter. For the systems involving L or L^T , LM usually has the value ML , except near the top of L where the vectors become shorter.

SGBDI

The inverse of a band matrix is usually a full matrix which cannot be stored in the band storage. Consequently, SGBDI does not compute the inverse. It consists merely of the determinant part of SGEDI .

5. Performance

Accuracy. In Chapter 1, section 6, the following roundoff error bounds for computations involving full matrices are discussed:

$$\|A-LU\| \leq \rho_n \|A\| \varepsilon_M$$
$$\|Ax-b\| \leq \sigma_n \|A\| \, \|x\| \varepsilon_M$$
$$\|x-x_*\| \leq \sigma_n \kappa(A) \|x\| \varepsilon_M \ .$$

These bounds apply as well to band matrices. It is possible to obtain smaller growth coefficients ρ_n and σ_n that reflect the band structure, but we will not do so here because these bounds are rarely used in any quantitative way.

For any particular problem, the actual roundoff error, as opposed to the above roundoff error bounds, is the same for the full and band routines.

Timing. To obtain a rough estimate for the execution time required by a particular subroutine on a particular computer, let μ be the time required by that computer to execute the following Fortran statement once

```
Y(I) = Y(I) + T*X(I)  .
```

This involves one floating-point multiplication, one floating-point addition, a few one-dimensional indexing operations, and a few storage references. Let n be the order of the band matrix and let m_ℓ and m_u be the lower and upper band widths. Assume $m_\ell \leq m_u$.

The time for SGBFA depends upon the amount of pivoting required. Let $m_{u'}$ be an unknown quantity somewhere in the range

$$m_u \leq m_{u'} \leq \min(m_\ell + m_u, n) \ .$$

If the matrix is diagonally dominant, no pivoting will be needed and so $m_{u'} = m_u$. Otherwise pivoting will usually be required and so $m_{u'} = m_\ell + m_u$, unless this is greater than n . The operation counts predict an execution time of

$$\left(nm_{\ell}m_{u'} - \frac{1}{2}\, m_{\ell}m_{u'}^{2} - \frac{1}{6}\, m_{\ell}^{3}\right)\mu \quad .$$

The last two terms in this estimate come from the "end effects" near the bottom of the band. In the common situation where m_{ℓ} and m_u are both much smaller than n , the predicted time is somewhere between

$$nm_{\ell}m_u \quad \text{and} \quad nm_{\ell}(m_{\ell}+m_u)$$

depending upon how much pivoting is done. Note that the estimate is a linear function of the order of the matrix and a quadratic function of the band width.

The time for SGBSL does not depend upon the amount of pivoting required. Let $m_{u''} = \min(m_{\ell}+m_u, n)$. The operation counts predict an execution time of

$$\left(nm_{\ell} + nm_{u''} - \frac{1}{2}\, m_{\ell}^{2} - \frac{1}{2}\, m_{u''}^{2}\right)\mu \quad .$$

If m_{ℓ} and m_u are both much smaller than n , the predicted time is

$$n(2m_{\ell}+m_u)\mu \quad .$$

Note that this is linear in both the order and the band width.

The estimated time for SGBCO , if m and m_u are much smaller than n , is the time for SGBFA plus $n(8m_{\ell}+3m_u)$.

To summarize and simplify, assume that pivoting is required and that $m_{\ell} = m_u = m \ll n$. Then estimates for the execution times can be obtained from the following table. The estimates may be quite inaccurate for very small m , small n , or m nearly equal to n .

Subroutine	Time
SGBCO	$(2nm^2 + 11nm)\mu$
SGBFA	$2nm^2\mu$
SGBSL	$3nm\mu$

Estimates for the execution times of the double precision and complex subroutines can be obtained by changing μ appropriately.