Appendix 1

GENERAL MATRICES

GEBS	-	Back Solve
GECE	-	Condition Estimation
GEDC	-	DeComposition
GEFS	-	Forward Solve
GELE	-	Linear Equation solution
GELU	-	LU decomposition
GEML	-	MuLtiplication
GENM	-	NorM
GESS	-	System Solution

Purpose: GEBS (GEneral matrix Back-Solve) solves AX = B where A is an upper triangular matrix. It can be used for the back solution phase of a general linear system solution. (It is used in this way by the routines GESS and GELE.)

Usage: CALL GEBS (N, A, IA, B, IB, NB)

N	$\longrightarrow$	the number of equations
A	$\longrightarrow$	the array, dimensioned (IA, KA) in the calling program, where IA $\geq$ N and KA $\geq$ N, containing the N $\times$ N upper triangular matrix The strictly lower triangular portion of A is not used or changed.
IA	$\longrightarrow$	the row (leading) dimension of A, as dimensioned in the calling program
B	$\longrightarrow$	the matrix of right-hand sides, dimensioned (IB, KB) in the calling program, where IB $\geq$ N and KB $\geq$ NB
	$\longleftarrow$	the solution X
IB	$\longrightarrow$	the row (leading) dimension of B, as dimensioned in the calling program
NB	$\longrightarrow$	the number of right-hand sides

Note: GEFS and GEBS can be used directly on the output matrix produced by GEDC, GELU, or GECE to solve a general linear system.

Error situations: *(The user can elect to ‘recover’ from those errors marked with an asterisk — see *Error Handling*, Framework Chapter)

Number	Error
1	N < 1
2	IA < N
3	IB < N
4	NB < 1
10 + k*	singular matrix with k^{th} diagonal element 0.0

Double-precision version: DGEBS with A and B declared double precision

Complex version: CGEBS with A and B declared complex

Storage: None

Time: $(N^2/2 - N/2) \times NB$ additions
$(N^2/2 - N/2) \times NB$ multiplications
$N \times NB$ divisions

See also: GECE, GEDC, GEFS, GELE, GELU, GESS

Author: Linda Kaufman

Example: Usually the subroutine GEBS is used as part of a package designed for general matrices as in the example for GECE. However it may also be used to solve a linear system with a triangular matrix as in the following example. In this example the last column of the inverse of the triangular matrix

$$\begin{matrix} 1 & -1 & -1 & . & . & . & -1 \\ & 1 & -1 & . & . & . & -1 \\ & & 1 & . & . & . & -1 \\ & & & . & . & . & . \\ & & & & . & . & . \\ & & & & & 1 & -1 \\ & & & & & & 1 \end{matrix}$$

is computed by setting the right-hand side to the vector (0, 0, . . . , 0, 1).

Although the determinant of the matrix is 1, the inverse can have large off-diagonal elements. In fact as the size of this matrix increases, the off-diagonal elements of the inverse increase and the matrix becomes more ill-conditioned.

```
      INTEGER N, I, J, IWRITE, I1MACH
      REAL A(15,15), B(15)
      N=15
C
C FORM THE MATRIX AND SET THE RIGHT-HAND SIDE
C TO THE LAST COLUMN OF THE IDENTITY MATRIX
      DO 20 I=1,N
         DO 10 J=I,N
            A(I,J) = -1.0
 10      CONTINUE
         A(I,I) = 1.0
         B(I)   = 0.0
 20   CONTINUE
      B(N)=1.0
C FIND THE LAST COLUMN OF THE INVERSE MATRIX
      CALL GEBS(N,A,15,B,N,1)
      IWRITE=I1MACH(2)
      WRITE(IWRITE,21)(I,B(I),I=1,N)
 21   FORMAT(3H B(,I3,3H )=,F15.4)
      STOP
      END
```

Execution on the Honeywell 6000 computer at Bell Labs yields the result:

```
B(  1 )=     8192.0000
B(  2 )=     4096.0000
B(  3 )=     2048.0000
B(  4 )=     1024.0000
B(  5 )=      512.0000
B(  6 )=      256.0000
B(  7 )=      128.0000
B(  8 )=       64.0000
B(  9 )=       32.0000
B( 10 )=       16.0000
B( 11 )=        8.0000
B( 12 )=        4.0000
B( 13 )=        2.0000
B( 14 )=        1.0000
B( 15 )=        1.0000
```

GECE — LU decomposition of a general matrix with condition estimation

Purpose: GECE (GEneral matrix Condition Estimation) gives a lower bound for the condition number of a real general matrix A. It also supplies the LU decomposition of the matrix A using partial pivoting and may be used to replace GEDC or GELU in a linear equation package.

Usage: CALL GECE (N, A, IA, INTER, COND)

N	$\longrightarrow$	the order of the matrix A
A	$\longrightarrow$	the array, dimensioned (IA, KA) in the calling program, where IA $\geq$ N and KA $\geq$ N, containing the N $\times$ N coefficient matrix
	$\longleftarrow$	the LU decomposition of A (see **Note 2**)
IA	$\longrightarrow$	the row (leading) dimension of A, as dimensioned in the calling program
INTER	$\longleftarrow$	an integer vector of length N recording row interchanges performed during the decomposition (see **Note 2**)
COND	$\longleftarrow$	an estimate of the condition number of A (see **Note 1**)

Note 1: The condition number measures the sensitivity of the solution of a linear system to errors in the matrix and in the right-hand side. If the elements of the matrix and the right-hand side(s) of your linear system have **d** decimal digits of precision, the solution might have as few as $\mathbf{d} - \log_{10}$(COND) correct decimal digits. Thus if COND is greater than $10^{\mathrm{Bd}P}$, there may be no correct digits.

Note 2: INTER and the LU decomposition returned in A are suitable for input into GEFS and GEBS. The LU decomposition of A satisfies the equation PA=LU where P is a permutation matrix, L is a unit lower triangular matrix, and U is an upper triangular matrix. On return from GECE, U occupies the upper triangular portion of A, P can be obtained from INTER (see the introduction to this chapter), and the elements of L appear permuted in the strictly lower triangular portion of A. Since the diagonal elements of L are all 1, they are not stored.

Error situations: *(The user can elect to 'recover' from those errors marked with an asterisk — see *Error Handling*, Framework Chapter)

Number	Error
1	N < 1
2	IA < N
10 + k*	singular matrix whose rank is at least k

Double-precision version: DGECE with A and COND declared double precision

Complex version: CGECE with A declared complex

Storage: N real (double precision for DGECE, complex for CGECE) locations of scratch storage in the dynamic storage stack

Time:
$$\frac{N^3}{3} + \frac{9}{2}N^2 + \frac{19}{6}N \text{ additions}$$
$$\frac{N^3}{3} + \frac{5}{2}N^2 + \frac{7}{6}N \text{ multiplications}$$
$$\frac{N^2}{2} + \frac{3}{2}N \text{ divisions}$$

Method: Gaussian elimination with partial pivoting.
See the reference below for the method used to estimate the condition number.
GECE calls GELU after setting EPS to 0.

See also: GEBS, GEDC, GEFS, GELE, GELU, GESS

Authors: Doris Ryan and Linda Kaufman

Reference: Cline, A. K., Moler, C. B., Stewart, G. W., and Wilkinson, J. H., An estimate for the condition number, *SIAM J. Numer. Anal. 16* (1979), 368-375.

Example: The example below is an encoding of the iterative refinement algorithm which can be used to obtain a highly accurate solution to a system of linear equations with an ill-conditioned coefficient matrix. If the condition number is not excessively high, the program usually returns a solution that is accurate to the working precision of the machine.

The iterative refinement algorithm is essentially:

(1) Solve $Ax = b$
(2) Set tol = $\varepsilon \sum |x_i|$
where ε is the precision of the machine
(3) Compute in double precision the residual
$r = Ax - b$
(4) Solve $A\,\delta x = r$
(5) Compute norm = $\sum |\delta x_i|$
(6) Set x to $x + \delta x$
(7) If norm $\le$ tol stop, else return to step 3

In our code, step (1) is accomplished using the three lower-level subroutines GECE, GEFS, and GEBS. The subroutine GECE factors A into LU where L is lower triangular and U is upper triangular. Then GEFS forward solves with L and GEBS back solves with U. Since A is overwritten by GECE and needed in step (3) of the algorithm, a copy of the A matrix is saved. In step (4) the decomposition created earlier in GECE is reused and only GEFS and GEBS are called. Since it is possible that the matrix is so ill-conditioned that the iterative refinement algorithm will diverge, steps (3) through (7) in our code are performed only a finite number of times. This number is chosen to be an upper bound on the number of bits in the mantissa of the floating-point number supported by the machine.

This algorithm is not yet included in PORT because the double-precision version of the program would require the residuals to be computed in extended precision.

```
      INTEGER N, IA, IB, NB, INTER(5), IREAD, I1MACH
      INTEGER I, J, IWRITE, ITER, IEND
      REAL A(5, 5), SAVEA(5, 5), B(5), SAVEB(5), R(5)
      REAL COND, BNORM, R1MACH, ABS, RNORM
      DOUBLE PRECISION DSDOT
C
      N=5
      IA=5
      IB=5
      NB=1
      IREAD=I1MACH(1)
C
      DO 10 I=1,N
   10    READ(IREAD,11) (A(I,J),J=1,N)
   11    FORMAT(1X,5F8.0)
      DO 20 I=1,IB
   20    READ(IREAD,21) B(I)
   21    FORMAT(F8.0)
```

```
C
C SAVE THE MATRIX AND RIGHT-HAND SIDE (WHICH WILL BE OVERWRITTEN)
C
       DO 40 I=1,N
         SAVEB(I)=B(I)
         DO 30 J=1,N
   30       SAVEA(I,J)=A(I,J)
   40  CONTINUE
C
C SOLVE AX = B USING SEPARATE CALLS TO GECE, GEFS, GEBS
C
       CALL GECE(N,A,IA,INTER,COND)
       IWRITE=I1MACH(2)
       IF (COND.GE.1.0/R1MACH(4)) WRITE(IWRITE,41)
   41  FORMAT(49H CONDITION NUMBER HIGH,ACCURATE SOLUTION UNLIKELY)
C
       CALL GEFS(N,A,IA,B,IB,NB,INTER)
C
       CALL GEBS(N,A,IA,B,IB,NB)
       WRITE(IWRITE,42)
   42  FORMAT(44H ESTIMATED CONDITION NUMBER OF THE MATRIX A,)
       WRITE(IWRITE,43) COND
   43  FORMAT(27H  USING ONE CALL TO GECE = ,E15.7)
       BNORM=0.0
       WRITE(IWRITE,44)
   44  FORMAT(/22H THE FIRST SOLUTION X,)
       WRITE(IWRITE,45)
   45  FORMAT(41H (USING CALLS TO GECE, GEFS, AND GEBS) = )
C
C COMPUTE NORM OF SOLUTION
C
       DO 50 I=1,N
          BNORM=BNORM + ABS(B(I))
   50     WRITE(IWRITE,51) B(I)
   51     FORMAT(1X, 5F20.7)
C
C REFINE THE SOLUTION DEPENDING ON THE LENGTH OF THE MANTISSA
C
              IEND=I1MACH(11)*IFIX(R1MACH(5)/ALOG10(2.0) + 1.0)
       DO 90 ITER=1,IEND
C  COMPUTE RESIDUAL R = B - AX, IN DOUBLE PRECISION
C
          WRITE(IWRITE,52)
   52     FORMAT(/27H THE RESIDUAL R = B - AX = )
          DO 70 I=1,IA
                DSDOT=0.0
                DO 60 J=1,N
   60              DSDOT = DSDOT  +  DBLE(SAVEA(I,J))*B(J)
                R(I) = SAVEB(I) - DSDOT
   70           WRITE(IWRITE,51) R(I)
C
C  SOLVE LU*(DELTA X) = R USING SEPARATE CALLS TO GEFS AND GEBS
C
          CALL GEFS(N,A,IA,R,IB,NB,INTER)
          CALL GEBS(N,A,IA,R,IB,NB)
C
C  THE NEW SOLUTION X = X  +  DELTA X
```

```
C
          WRITE(IWRITE,71)
   71     FORMAT(/36H THE NEW SOLUTION X = X  +  DELTA X = )
C
C DETERMINE NORM OF CORRECTION AND ADD IN CORRECTION
C
          RNORM=0.0
          DO 80 I=1,N
                B(I) = B(I)  +  R(I)
                RNORM=RNORM + ABS(R(I))
   80           WRITE(IWRITE,51) B(I)
C
C TEST FOR CONVERGENCE
C
          IF(RNORM.LT.R1MACH(4)*BNORM) GO TO 100
   90  CONTINUE
       WRITE(IWRITE,91)
   91  FORMAT(/29H ITERATIVE IMPROVEMENT FAILED)
  100  CONTINUE
       STOP
       END
```

For the input matrix

1.	–2.	3.	7.	–9.
–2.	8.	–6.	2.	50.
3.	–6.	18.	–15.	–18.
7.	2.	–15.	273.	174.
–9.	50.	–18.	173.	1667.

with the following right-hand side:

78.
–320.
–81.
215.
–10856.

the following results were obtained on the Honeywell 6000 computer at Bell Labs:

```
ESTIMATED CONDITION NUMBER OF THE MATRIX A,
 USING ONE CALL TO GECE =   0.7263499E 07

THE FIRST SOLUTION X,
(USING CALLS TO GECE, GEFS, AND GEBS) =
```

−5.9872369
−4.9984942
−8.0019742
4.9995200
−6.9999477

```
THE RESIDUAL R = B − AX =
```

0.0000020
−0.0000190
0.0000215
−0.0000073
−0.0000473

```
THE NEW SOLUTION X = X + DELTA X =
```

−6.0000002
−5.0000000
−7.9999999
5.0000000
−7.0000000

```
THE RESIDUAL R = B − AX =
```

0.0000001
−0.0000001
−0.0000004
0.0000026
−0.0000011

```
THE NEW SOLUTION X = X + DELTA X =
```

−6.0000000
−5.0000000
−8.0000000
5.0000000
−7.0000000

The first solution above is inaccurate, as would have been expected from the estimate of the condition number for the matrix. The iterative refinement algorithm successfully improved the solution to this problem because the matrix and the right-hand side could be represented exactly in the machine. (Also the condition number was not high.) Often the input matrix cannot be represented exactly and the iterative refinement algorithm produces a very accurate, but worthless, solution to a slightly incorrect problem.

GEDC — LU decomposition of a general matrix

Purpose: GEDC (GEneral matrix DeComposition) computes the LU decomposition of a dense general matrix using partial pivoting. It is called by GELE as the first step of the solution of a general linear system.

Usage: CALL GEDC (N, A, IA, INTER)

N	$\longrightarrow$	the order of the matrix A
A	$\longrightarrow$	the array, dimensioned (IA, KA) in the calling program, where IA ≥ N and KA ≥ N, containing the N × N coefficient matrix
	$\longleftarrow$	the LU decomposition of A (see **Note**)
IA	$\longrightarrow$	the row (leading) dimension of A, as dimensioned in the calling program
INTER	$\longleftarrow$	an integer vector of length N recording row interchanges performed during the decomposition (see **Note**)

Note: INTER and the LU decomposition returned in A are suitable for input into GEFS and GEBS. The LU decomposition of A satisfies the equation PA=LU where P is a permutation matrix, L is a unit lower triangular matrix, and U is an upper triangular matrix. On return from GEDC, U occupies the upper triangular portion of A, P can be obtained from INTER (see the introduction to this chapter), and the elements of L appear permuted in the strictly lower triangular portion of A. Since the diagonal elements of L are all 1, they are not stored.

Error situations: *(The user can elect to ‘recover’ from those errors marked with an asterisk — see *Error Handling*, Framework Chapter)

Number	Error
1	N < 1
2	IA < N
10 + k*	singular matrix whose rank is at least k

Double-precision version: DGEDC with A declared double precision

Complex version: CGEDC with A declared complex

Storage: None

Time:

$\frac{N^3}{3} + \frac{N^2}{2} + \frac{N}{6}$ additions

$\frac{N^3}{3} - \frac{N^2}{2} + \frac{N}{6}$ multiplications

$\frac{(N^2 - N)}{2}$ divisions

Method: Gaussian elimination with partial pivoting. GEDC calls GELU after setting EPS = $||A||\varepsilon$, where ε is machine precision, i.e. the value returned by R1MACH(4) (or, for double precision, by D1MACH(4)).

See also: GEBS, GECE, GEFS, GELE, GELU, GESS

Author: Linda Kaufman

Example: In this example, we illustrate, for a special case, how the building blocks, GEDC, GEFS and GEBS of our linear equation solver can be used to circumvent a limitation in memory space.

We consider a matrix A which has the special form

$$A = \begin{bmatrix} C & D \\ E & F \end{bmatrix}$$

where C and F are dense $n \times n$ matrices, and D and E are $n \times n$ diagonal matrices. (Thus D and E can be stored in vectors of length n.) If n is 200, and the problem is to be solved on a computer with only 100K words of data space, the set of linear equations cannot be solved by either the general subprogram GELE or the band package, BALE, because too much additional storage would be required. The sparse matrix package is also eliminated because those subroutines demand additional storage for additional column indices for each nonzero element.

However, if C^{-1} exists then one can solve $AX = B$ using the fact that

$$\begin{bmatrix} C & D \\ E & F \end{bmatrix} = \begin{bmatrix} I & 0 \\ EC^{-1} & I \end{bmatrix} \begin{bmatrix} C & D \\ 0 & F - EC^{-1} D \end{bmatrix} .$$

If B is partitioned into

$$B = \begin{bmatrix} B_U \\ \hline B_L \end{bmatrix}$$

where B_U has length n, then, applying the inverse,

$$\begin{bmatrix} I & 0 \\ EC^{-1} & I \end{bmatrix}^{-1} = \begin{bmatrix} I & 0 \\ -EC^{-1} & I \end{bmatrix} ,$$

to $AX = B$, we see that X is the solution to

$$\begin{bmatrix} C & D \\ 0 & F - E\ C^{-1} D \end{bmatrix} X = \begin{bmatrix} B_U \\ B_L - E\ C^{-1} B_U \end{bmatrix}$$

Finally, if X is partitioned into

$$X = \begin{bmatrix} X_U \\ \hline X_L \end{bmatrix}$$

where X_U has length n, then X can be found by the following algorithm:

(1) replace F by $F - EC^{-1} D$
(2) replace B_L by $B_L - EC^{-1} B_U$
(3) solve $FX_L = B_L$
(4) replace B_U by $B_U - DX_L$
(5) solve $CX_U = B_U$

Of course in steps (1) and (2) we do not form C^{-1} explicitly, but solve a system of equations with C as the coefficient matrix. Thus steps (1),(2), and (5) solve systems with the same coefficient matrix but different right-hand sides. Moreover, the right-hand side for step (5) is not known until after the first two systems have been solved so that GELE, the general linear equation solver, cannot be used to solve all three simultaneously. The correct approach is to compute the LU factorization of C once, and to use it three times. Then subroutine GEFS forward solves with the L portion and GEBS back solves with the U portion.

In the code below, X_U is left in the array B_U and X_L is left in the array B_L.

```
        INTEGER INTER(200), I, J, N
        REAL C(200, 200), D(200), E(200), F(200, 200)
        REAL TEMP(200), BL(200), BU(200)
        N=200
        .
        code for filling in C,D,E,F,BL, and BU belongs here
        .
C DO AN LU DECOMPOSITION OF C
        CALL GEDC(N,C,200,INTER)
C
C FORM F - EC(INVERSE)D IN F
C
        DO 30 J=1,N
           DO 10 I=1,N
              TEMP(I)=0
  10       CONTINUE
           TEMP(J)=D(J)
           CALL GEFS(N,C,200,TEMP,200,1,INTER)
           CALL GEBS(N,C,200,TEMP,200,1)
C TEMP CONTAINS THE JTH COLUMN OF
C C(INVERSE)D
           DO 20 I=1,N
              F(I,J)=F(I,J) - E(I)*TEMP(I)
  20       CONTINUE
  30    CONTINUE
C
C FORM BL - EC(INVERSE)BU
C
        DO 40 I=1,N
           TEMP(I)=BU(I)
  40    CONTINUE
        CALL GEFS(N,C,200,TEMP,200,1,INTER)
        CALL GEBS(N,C,200,TEMP,200,1)
        DO 50 I=1,N
           BL(I)=BL(I) - E(I)*TEMP(I)
  50    CONTINUE
C
C SOLVE FOR LOWER PART OF X
C
        CALL GELE(N,F,200,BL,200,1)
C
C FORM RIGHT HAND SIDE TO SOLVE FOR UPPER PART OF X
C
        DO 60 I=1,N
           BU(I)=BU(I) - D(I)*BL(I)
  60    CONTINUE
C
C SOLVE FOR UPPER PART OF X
C
        CALL GEFS(N,C,200,BU,200,1,INTER)
        CALL GEBS(N,C,200,BU,200,1)
        .
        .
```

GEFS — lower (unit) triangular linear system solution

Purpose: GEFS (GEneral matrix Forward-Solve) solves AX = PB where A is a unit lower triangular matrix, (i.e. 1's on the diagonal), and P is a permutation matrix. It can be used for the forward solution phase of a general linear system solver. (It is used in this way by the routines GESS and GELE.)

Usage: CALL GEFS (N, A, IA, B, IB, NB, INTER)

N	$\longrightarrow$	the number of equations
A	$\longrightarrow$	the array, dimensioned (IA, KA) in the calling program, where IA ≥ N and KA ≥ N, containing the N × N coefficient matrix The upper triangular portion of A(including the main diagonal) is not used or changed.
IA	$\longrightarrow$	the row (leading) dimension of A, as dimensioned in the calling program
B	$\longrightarrow$	the matrix of right-hand sides, dimensioned (IB, KB) in the calling program, where IB ≥ N and KB ≥ NB
	$\longleftarrow$	the solution X
IB	$\longrightarrow$	the row (leading) dimension of B, as dimensioned in the calling program
NB	$\longrightarrow$	the number of right-hand sides
INTER	$\longrightarrow$	the integer vector of length N recording interchanges performed in GELU, GEDC, or GECE. To solve a unit lower triangular system, set INTER(J) = J, J = 1,...,N.

Note 1: GEFS and GEBS can be used directly on the output matrix produced by GEDC, GELU, or GECE to solve a general linear system.

Note 2: Users who have to solve a sequence of problems with the same coefficient matrix, but different right-hand sides, *not all known in advance*, should not call GESS or GELE repeatedly, but should use the sequence shown in the example on page 3.

Error situations: (All errors in this subprogram are fatal — see *Error Handling*, Framework Chapter)

Number	Error
1	N < 1
2	IA < N
3	IB < N
4	NB < 1
5	elements of INTER out of range

Double-precision version: DGEFS with A and B declared double precision

Complex version: CGEFS with A and B declared complex

Storage: None

Time: $(N^2 - N) \times NB/2$ additions
$(N^2 - N) \times NB/2$ multiplications

See also: GEBS, GECE, GEDC, GELE, GELU, GESS

Author: Linda Kaufman

Example: In this example we give a general outline of a program to solve a sequence of problems with the same coefficient matrix but different right-hand sides.

The call to GEDC computes the LU decomposition of the matrix A. This decomposition can be then used repeatedly (and efficiently) for the sequence of forward solutions (using GEFS) and back solutions (using GEBS) for each set of right-hand sides.

```
     declare matrix with leading dimension IA
     declare right-hand side vector
     declare integer vector INTER
     assign appropriate values to N and IA
     .
     compute or read in the coefficient matrix
     .
     CALL GEDC(N,A,IA,INTER)

10   compute a right-hand side

     CALL GEFS(N,A,IA,B,N,1,INTER)
     CALL GEBS(N,A,IA,B,N,1)

     if sequence of problems is not finished go to 10
     .
     .
     .
```

GELE — general linear system solution

Purpose: GELE (GEneral Linear Equation solution) solves the system AX = B where A is a dense general matrix.

Usage: CALL GELE (N, A, IA, B, IB, NB)

N	$\longrightarrow$	the number of equations
A	$\longrightarrow$	the array, dimensioned (IA, KA) in the calling program, where IA $\geq$ N and KA $\geq$ N, containing the N $\times$ N coefficient matrix A is overwritten during the solution.
IA	$\longrightarrow$	the row (leading) dimension of A, as dimensioned in the calling program
B	$\longrightarrow$	the matrix of right-hand sides, dimensioned (IB, KB) in the calling program, where IB $\geq$ N and KB $\geq$ NB
	$\longleftarrow$	the solution X
IB	$\longrightarrow$	the row (leading) dimension of B, as dimensioned in the calling program
NB	$\longrightarrow$	the number of right-hand sides

Note 1: Unless the given matrix, A, is known in advance to be well-conditioned, the user should use GESS instead of GELE.

Note 2: Users who wish to solve a sequence of problems with the same coefficient matrix, but different right-hand sides *not all known in advance*, should not use GELE, but should call subprograms GEDC, GEFS and GEBS. (See the example in GEDC.) GEDC is called once to get the LU decomposition (see the introduction to this chapter) and then the pair, GEFS (forward solve) and GEBS (back solve), is called for each new right-hand side.

Error situations: *(The user can elect to ‘recover’ from those errors marked with an asterisk — see *Error Handling*, Framework Chapter)

Number	Error
1	N < 1
2	IA < N
3	IB < N
4	NB < 1
10 + k*	singular matrix whose rank is at least k

Double-precision version: DGELE with A and B declared double precision

Complex version: CGELE with A and B declared complex

Storage: N integer locations of scratch storage in the dynamic storage stack

Time: $N^3/3 + N^2/2 + N/6 + NB\times(N^2-N)$ additions
$N^3/3-N^2/2 + N/6 + NB\times(N^2-N)$ multiplications
$(N^2-N)/2 + N\times NB$ divisions

Method: Gaussian elimination with partial pivoting.
GELE calls GEDC, GEFS, and GEBS.

See also: GEBS, GECE, GEDC, GEFS, GELU, GESS

Authors: Linda Kaufman and Doris Ryan

Example: Two things are illustrated in the following example. First, the relative efficiencies of GELE and GESS are compared as a function of the size of the system. The example indicates that the extra cost associated with computing the condition number (GESS) decreases as the size of the system increases. When N = 90, GELE is only about 10% faster that GESS. For small systems with one right-hand side, however, GESS takes almost twice the time required by GELE.

Secondly, the example illustrates the increase of the error in the solution as the condition number grows. Notice that as the condition number increases from 1.3×10^3 to 1×10^6, the error in the solution grows from 3×10^{-7} to 6×10^{-4}.

The $N \times N$ matrix used in the example

$$a_{ij} = \begin{cases} j-i & \text{for } i<j \\ i-j+1 & \text{for } i \geq j \end{cases}$$

becomes more ill-conditioned as N increases. The right-hand side was chosen to make the solution vector all 1's, and the maximum error is computed as $\max_I |X(I) - 1.0|$, for the solution, X.

The timing subroutine, ILAPSZ, on the Honeywell 6000 system has about 1% accuracy.

```
        INTEGER IA, IB, I1MACH, N, I, J, IT, ILAPSZ, IWRITE
        REAL A(100, 100), AA(100, 100), B(100), BB(100)
        REAL SUM, ERR, COND, ABS, TIME, TIMES, AMAX1
        IA=100
        IB =100
C
C GENERATE THE MATRIX AND RIGHT-HAND SIDE
C
        DO 40 N=10,90,40
           DO 20 I=1,N
              SUM=0.0
              DO 10 J=1,N
                 A(I,J)=ABS(I-J)
                 IF (I.GE.J) A(I,J)=A(I,J) + 1.0
                 AA(I,J)=A(I,J)
                 SUM=SUM + AA(I,J)
  10          CONTINUE
              B(I)=SUM
              BB(I)=SUM
  20       CONTINUE
C
C CALL  GELE AND TIME IT
           IT =ILAPSZ(0)
           CALL GELE(N,A,IA,B,IB,1)
           TIME=FLOAT(ILAPSZ(0)-IT)/64.0
C
C COMPUTE THE MAXIMUM ERROR
C
           ERR=0.0
           DO 30 I=1,N
              ERR=AMAX1(ERR, ABS(B(I)-1.0))
  30       CONTINUE
C
C CALL GESS
C
           IT =ILAPSZ(0)
```

```
          CALL GESS(N,AA,IA,BB,IB,1,COND)
          TIMES=FLOAT(ILAPSZ(0)-IT)/64.0
          IWRITE=I1MACH(2)
          WRITE(IWRITE,31)N,COND
 31       FORMAT(8H FOR N= ,I4,20H CONDITION NUMBER = ,E15.7)
          WRITE(IWRITE,32)ERR
 32       FORMAT(30H MAXIMUM ERROR IN SOLUTION IS ,F15.7)
          WRITE(IWRITE,33)TIME
 33       FORMAT(34H TIME IN MILLISECONDS FOR GELE IS ,F10.2)
          WRITE(IWRITE,34)TIMES
 34       FORMAT(34H TIME IN MILLISECONDS FOR GESS IS ,F10.2)
 40    CONTINUE
       STOP
       END
```

When the program above was run on the Honeywell 6000 machine at Bell Laboratories, the following was printed.

```
FOR N=   10 CONDITION NUMBER =   0.1349031E 04
MAXIMUM ERROR IN SOLUTION IS       0.0000003
TIME IN MILLISECONDS FOR GELE IS      13.27
TIME IN MILLISECONDS FOR GESS IS      22.70

FOR N=   50 CONDITION NUMBER =   0.1674143E 06
MAXIMUM ERROR IN SOLUTION IS       0.0000704
TIME IN MILLISECONDS FOR GELE IS     499.30
TIME IN MILLISECONDS FOR GESS IS     597.75

FOR N=   90 CONDITION NUMBER =   0.9745692E 06
MAXIMUM ERROR IN SOLUTION IS       0.0005805
TIME IN MILLISECONDS FOR GELE IS    2602.00
TIME IN MILLISECONDS FOR GESS IS    2919.06
```

GELU — LU decomposition of a general matrix

Purpose: GELU (GEneral matrix LU decomposition) finds the LU decomposition of a dense general matrix using partial pivoting. It allows the user to specify a threshold for considering a matrix singular. GELU is called by the LU decomposition routines GECE and GEDC.

Usage: CALL GELU (N, A, IA, INTER, EPS)

N	$\longrightarrow$	the order of the matrix A
A	$\longrightarrow$	the array, dimensioned (IA, KA) in the calling program, where IA $\geq$ N and KA $\geq$ N, containing the N $\times$ N coefficient matrix
	$\longleftarrow$	the LU decomposition of A (see **Note 2**)
IA	$\longrightarrow$	the row (leading) dimension of A, as dimensioned in the calling program
INTER	$\longleftarrow$	an integer vector of length N recording row interchanges performed during the decomposition (see **Note 2**)
EPS	$\longrightarrow$	if A = LU and $\lvert u_{kk} \rvert$ < EPS, for some $1 \leq k \leq N$, the matrix is considered singular.

Note 1: After the execution of GELU, (if the matrix has not been found singular), the value of the determinant is INTER(N) $\times$ A(1,1) $\times$ A(2,2) $\times \cdots \times$ A(N,N) where INTER(N) contains the sign of the permutation (the number of row interchanges).

Note 2: INTER and the LU decomposition returned in A are suitable for input into GEFS and GEBS. The LU decomposition of A satisfies the equation PA=LU where P is a permutation matrix, L is a unit lower triangular matrix, and U is an upper triangular matrix. On return from GELU, U occupies the upper triangular portion of A, P can be obtained from INTER (see the introduction to this chapter), and the elements of L appear permuted in the strictly lower triangular portion of A. Since the diagonal elements of L are all 1, they are not stored.

Error situations: *(The user can elect to 'recover' from those errors marked with an asterisk — see *Error Handling*, Framework Chapter)

Number	Error
1	N < 1
2	IA < N
10 + k*	singular matrix whose rank is at least k

Double-precision version: DGELU with A and EPS declared double precision

Complex version: CGELU with A declared complex

Storage: None

Time: $\frac{N^3}{3} - \frac{N^2}{2} + \frac{N}{6}$ additions

$\frac{N^3}{3} - \frac{N^2}{2} + \frac{N}{6}$ multiplications

$\frac{(N^2 - N)}{2}$ divisions

Method: Gaussian elimination with partial pivoting

See also: GEBS, GECE, GEDC, GEFS, GELE, GESS

Author: Linda Kaufman

Example: The following subroutine uses GELU to compute a determinant. The subroutine uses the stack to obtain space for the integer vector INTER. Care is taken to avoid overflow and underflow during the calculation. The subroutine UMKFL is used to decompose a floating point number, F, into a mantissa, M, and an exponent E such that

$$F = Mb^E$$

where b is the base of the machine and $1/b \le M < 1$.

```
      SUBROUTINE DET(N,A,IA,DETMAN,IDETEX)
C
C THIS SUBROUTINE COMPUTES THE DETERMINANT OF A
C THE RESULT IS GIVEN BY DETMAN*BETA**IDETEX
C WHERE BETA IS THE BASE OF THE MACHINE
C AND DETMAN IS BETWEEN 1/BETA AND 1
C
      INTEGER N, IA, IDETEX
      INTEGER E, IPOINT, ISTKGT, I1MACH, ISIGN, I
      INTEGER IN(1000)
      REAL A(IA, N), DETMAN, BETA, FLOAT, ONOVBE, M, ABS
      DOUBLE PRECISION D(500)
      COMMON /CSTAK/ D
      EQUIVALENCE(D(1),IN(1))
C
C ALLOCATE SPACE FROM THE STACK FOR THE PIVOT ARRAY
C
      IPOINT=ISTKGT(N,2)
      CALL GELU(N,A,IA,IN(IPOINT),0.0)
C
C THE DETERMINANT IS THE PRODUCT OF THE DIAGONAL ELEMENTS
C AND THE LAST ELEMENT OF THE INTERCHANGE ARRAY
C WE TRY TO COMPUTE THIS PRODUCT IN A WAY THAT WILL
C AVOID UNDERFLOW AND OVERFLOW
C
      BETA=FLOAT(I1MACH(10))
      ONOVBE=1.0/BETA
      ISIGN=IPOINT + N-1
      DETMAN=IN(ISIGN)*ONOVBE
      IDETEX=1
      DO 10 I=1,N
          CALL UMKFL(A(I,I),E,M)
          DETMAN=DETMAN*M
          IDETEX=IDETEX+E
          IF(ABS(DETMAN).GE.ONOVBE) GO TO 10
             IDETEX=IDETEX-1
             DETMAN=DETMAN*BETA
  10   CONTINUE
      RETURN
      END
```

GEML — matrix - vector multiplication

Purpose: GEML (GEneral matrix MuLtiplication) forms the product *Ax* where *A* is a general matrix.

Usage: CALL GEML(N, A, IA, X, B)

N	$\longrightarrow$	the length of *x*
A	$\longrightarrow$	the array, dimensioned (IA, KA) in the calling program, where IA ≥ N and KA ≥ N, containing the N × N coefficient matrix
IA	$\longrightarrow$	the row (leading) dimension of A, as dimensioned in the calling program
X	$\longrightarrow$	the vector *x* to be multiplied
B	$\longleftarrow$	the vector A*x*

Error situations: (All errors in this subprogram are fatal — see *Error Handling*, Framework Chapter)

Number	Error
1	N < 1
2	IA < N

Double-precision version: DGEML with A, X, and B declared double precision.

Complex version: CGEML with A, X, and B declared complex

Time: N^2 additions
N^2 multiplications

See also: GECE, GEDC, GELU, GELE, GESS

Author: Linda Kaufman

Example: This example checks the consistency of GEML and GESS, the linear system solver.

First the example uses GEML to compute for a given vector x and matrix A, the vector

$$b = Ax.$$

Then the problem is inverted, i.e., GESS is used to find the vector x which satisfies

$$Ax = b$$

This x is then compared with the original vector. The 10×10 matrix A is chosen so that

$$a_{ij} = \begin{cases} j-i & \text{for } i<j \\ i-j+1 & \text{for } i\geq j \end{cases}$$

The vector x is chosen randomly.

```
      INTEGER I, J, IWRITE, I1MACH, N
      REAL A(10, 10), X(10), B(10)
      REAL ERR, SASUM, UNI, COND
      N=10
C
C CONSTRUCT A MATRIX
C
      DO 20 I=1,N
         DO 10 J=I,N
            A(I,J)=J-I
            A(J,I)=J-I + 1
 10      CONTINUE
 20   CONTINUE
C
C CONSTRUCT A RANDOM VECTOR X
C
      DO 30 I=1,N
         X(I)=UNI(0)
 30   CONTINUE
C
C FIND THE VECTOR B=AX
C
     CALL GEML(N,A,10,X,B)
C
C SOLVE THE SYSTEM AX=B
C
     CALL GESS(N,A,10,B,N,1,COND)
C
C PRINT THE COMPUTED AND TRUE SOLUTION
C
     IWRITE=I1MACH(2)
     WRITE(IWRITE,31)
 31  FORMAT(34H TRUE SOLUTION   COMPUTED SOLUTION)
     WRITE(IWRITE,32)(X(I),B(I),I=1,N)
 32  FORMAT(1H ,2E17.8)
C
```

```
C COMPUTE THE RELATIVE ERROR
C
      ERR=0.0
      DO 40 I=1,N
         ERR=ERR + ABS(B(I)-X(I))
 40   CONTINUE
      ERR=ERR/SASUM(N,X,1)
      WRITE(IWRITE,41)ERR
 41   FORMAT(19H RELATIVE ERROR IS ,1PE15.7)
      WRITE(6,42)COND
 42   FORMAT(21H CONDITION NUMBER IS ,1PE15.7)
      STOP
      END
```

When the above program was executed on the Honeywell 6000 machine at Bell Laboratories, the following was printed:

```
TRUE SOLUTION COMPUTED SOLUTION
0.22925607E 00 0.22925687E 00
0.76687502E 00 0.76687336E 00
0.68317685E 00 0.68317838E 00
0.50919111E 00 0.50918986E 00
0.87455959E 00 0.87456071E 00
0.64464101E 00 0.64463982E 00
0.84746840E 00 0.84746962E 00
0.35396343E 00 0.35396226E 00
0.39889160E 00 0.39889258E 00
0.45709422E 00 0.45709377E 00
RELATIVE ERROR IS 1.9705190E-06
CONDITION NUMBER IS 1.3490306E 03
```

The condition number of the matrix and the precision of the Honeywell computer suggest that even in the absence of roundoff error in GEML, a relative error of 10^{-5} would not be surprising. The value computed above is quite reasonable.

GENM — norm of a general matrix

Purpose: GENM (GEneral matrix NorM) computes the norm of a general matrix A. The norm is defined as $\max_{1 \le j \le n} \sum_{i=1}^{n} |a_{ij}|$

Type: Real function

Usage: <answer> = GENM (N, A, IA)

N $\longrightarrow$ the number of rows in A

A $\longrightarrow$ the array, dimensioned (IA, KA) in the calling program, where IA ≥ N and KA ≥ N, containing the N × N coefficient matrix

$\longleftarrow$ the LU decomposition of A (see **Note 2**)

IA $\longrightarrow$ the row (leading) dimension of A, as dimensioned in the calling program

<answer> $\longleftarrow$ $\max_{1 \le j \le n} \sum_{i=1}^{n} |a_{ij}|$

Error situations: (All errors in this subprogram are fatal — see *Error Handling*, Framework Chapter)

Number	Error
1	N < 1
2	IA < N

Double precision version: DGENM with A and DGENM declared double precision

Complex version: CGENM with A declared complex

Storage: None

Time: N^2 additions
N comparisons

See also: GEDC, GELU, GELE, GESS, GECE

Author: Linda Kaufman

Example: The subroutines in the PORT library for solving $Ax = b$ are designed to return computed solutions x such that the residual $r = Ax - b$ satisfies

$$\frac{||r||}{||A||\ ||x||} \le \varepsilon$$

where ε is the machine precision. In this example we show that if A is ill-conditioned, then the computed solution need not be very close to the true solution even though equation (1.1) is satisfied. The subroutine GENM is used to compute the left-hand side of (1.1). The matrix in this example is given by

$$a_{ij} = \begin{cases} j-i & \text{for } i<j \\ i-j+1 & \text{for } i \ge j \end{cases}$$

and the true solution is $x_i = i$. The right hand side is generated using GEML and the computed solution is obtained using GELE. The function SAMAX is used to compute the 1-norm of a vector; i.e. $\max\limits_{1 \le i \le n} |x_i|$

```
        INTEGER I, J, L, N, IA, IWRITE, I1MACH
        REAL A(50, 50), AA(50, 50), B(50), X(50)
        REAL RELERR, RELRES, XNORM, RNORM, ERR, R(50)
        REAL GENM, SAMAX
        IA = 50
C
C GENERATE MATRIX
C
        N=50
        DO 20 I=1,N
           DO 10 J=I,N
              A(I,J)=J-I
              A(J,I)=J-I + 1
              AA(I,J)=A(I,J)
              AA(J,I)=A(J,I)
 10        CONTINUE
           B(I)=I
 20     CONTINUE
C
C GENERATE RIGHT HAND SIDE
C
```

```
      CALL GEML(N,A,IA,B,X)
C
C MAKE COPY OF RIGHT HAND SIDE
C
      CALL MOVEFR(N,X,B)
C
C SOLVE THE SYSTEM
C
      CALL GELE(N,A,IA,B,N,1)
C
C COMPUTE THE RELATIVE ERROR AND THE RELATIVE RESIDUAL
C
      CALL GEML(N,AA,IA,B,R)
      ERR=0.0
      DO 30 I=1,N
         ERR=AMAX1(ERR,ABS(B(I)-FLOAT(I)))
         R(I)=R(I)-X(I)
 30   CONTINUE
      XNORM=SAMAX(N,X,1)
      RNORM=SAMAX(N,R,1)
      RELERR=ERR/XNORM
      RELRES=RNORM/(XNORM*GENM(N,AA,IA))
      IWRITE=I1MACH(2)
      WRITE(IWRITE,31)RELERR,RELRES
 31   FORMAT(16H RELATIVE ERROR=,E15.5,19H RELATIVE RESIDUAL=,
     1  E15.5)
      STOP
      END
```

When the above program was executed on the Honeywell 6000 machine at Bell Laboratories, the following was printed:

```
RELATIVE ERROR=    0.13554E-06 RELATIVE RESIDUAL=    0.22987E-10
```

The condition number of the matrix(see the example in GELE) is about 10^5, and the machine precision on the Honeywell computer is about 10^{-8}. Thus even in the absence of roundoff error in GEML, a relative error of 10^{-3} would not be surprising. The relative error given above is quite within reason. The relative residual, as promised, satisfies (1.1) even though the problem is ill-conditioned.

GESS — general linear system solution with condition estimation

Purpose: GESS (GEneral System Solution) solves the system AX = B where A is a general matrix. An estimate of the condition of A is provided.

Usage: CALL GESS (N, A, IA, B, IB, NB, COND)

N	$\longrightarrow$	the number of equations
A	$\longrightarrow$	the array, dimensioned (IA, KA) in the calling program, where IA ≥ N and KA ≥ N, containing the N × N coefficient matrix A is overwritten during the solution.
IA	$\longrightarrow$	the row (leading) dimension of A, as dimensioned in the calling program
B	$\longrightarrow$	the matrix of right-hand sides, dimensioned (IB, KB) in the calling program, where IB ≥ N and KB ≥ NB
	$\longleftarrow$	the solution X
IB	$\longrightarrow$	the row (leading) dimension of B, as dimensioned in the calling program
NB	$\longrightarrow$	the number of right-hand sides
COND	$\longleftarrow$	an estimate of the condition number of A (see **Note 1**)

Note 1: The condition number measures the sensitivity of the solution of a linear system to errors in the matrix and in the right-hand side. If the elements of the matrix and the right-hand side(s) of your linear system have **d** decimal digits of precision, the solution might have as few as **d** $-\log_{10}$ (COND) correct decimal digits. Thus if COND is greater than $10^{\mathrm{Bd}P}$, there may be no correct digits.

If the given matrix, A, is known in advance to be well-conditioned, then the user may wish to use the routine GELE, which is a little faster than GESS. Ordinarily, however, the user is strongly urged to choose GESS, and to follow it by a test of the condition estimate.

Note 2: Users who wish to solve a sequence of problems with the same coefficient matrix, but different right-hand sides *not all known in advance*, should not use GESS, but should call subprograms GECE, GEFS and GEBS. (See the example of GEDC.) GECE is called once to get the LU decomposition (see the introduction to this chapter) and then the pair, GEFS (forward solve) and GEBS (back solve), is called for each new right-hand side.

Error situations: *(The user can elect to ‘recover’ from those errors marked with an asterisk — see *Error Handling*, Framework Chapter)

Number	Error
1	N < 1
2	IA < N
3	IB < N
4	NB < 1
10 + k*	singular matrix whose rank is at least k

Double-precision version: DGESS with A, B, and COND declared double precision

Complex version: CGESS with A and B declared complex

Storage: N integer locations and
N real (double precision for DGESS, complex for CGESS) locations of scratch storage in the dynamic storage stack

Time:

$$\frac{N^3}{3} + N^2 \times (\frac{9}{2} + NB) + N \times (\frac{19}{6} + NB) \text{ additions}$$

$$\frac{N^3}{3} + N^2 \times (\frac{5}{2} + NB) + N \times (\frac{7}{6} + NB) \text{ multiplications}$$

$$\frac{N^2}{2} + N \times (\frac{3}{2} + NB) \text{ divisions}$$

Method: Gaussian elimination with partial pivoting.
See the reference below for the method used to estimate the condition number.
GESS calls GECE, GEFS, and GEBS.

See also: GEBS, GECE, GEDC, GEFS, GELE, GELU

Authors: Linda Kaufman and Doris Ryan

Reference: Cline, A. K., Moler, C. B., Stewart, G. W., and Wilkinson, J. H., An estimate for the condition number, *SIAM J. Numer. Anal. 16* (1979), 368-375.

Example: The following program solves a 5×5 system with two right-hand sides

```
      INTEGER N, IREAD, I1MACH, I, NB, IWRITE, J
      REAL A(5,5), B(5,2), COND
      N=5
      IREAD=I1MACH(1)
C
      DO 10 I=1,N
          READ(IREAD,1) (A(I,J),J=1,N)
 1        FORMAT(1X,5F10.0)
 10   CONTINUE
C
      NB=2
      DO 20 I=1,N
          READ(IREAD,11) (B(I,J),J=1,NB)
 11       FORMAT(1X,2F10.3)
 20   CONTINUE
C
C SOLVE AX = B  BY CALLING GESS
C
      CALL GESS(N,A,N,B,N,NB,COND)
      IWRITE=I1MACH(2)
      WRITE(IWRITE,21) COND
 21   FORMAT(52H AN ESTIMATE OF THE CONDITION NUMBER OF THE MATRIX =,
     1         E14.7)
C
      WRITE(IWRITE,22)
 22   FORMAT(27H THE COMPUTED SOLUTION X IS,//)
      DO 30 I=1,N
          WRITE(IWRITE,23) (B(I,J),J=1,NB)
 23       FORMAT(1H,5F20.7)
 30   CONTINUE
C
      STOP
      END
```

For the input matrix given by:

1.	–2.	3.	7.	–9.
–2.	8.	–6.	9.	50.
11.	–6.	18.	–15.	–18.
7.	2.	–15.	273.	173.
–9.	50.	–18.	6.	1667.

and the following right-hand sides:

30.	29.419
–191.	–190.994
133.	133.072
–986.	–985.775
–6496.	–6495.553

the following results were obtained on the Honeywell 6000 computer at Bell Labs:

```
AN ESTIMATE OF THE CONDITION NUMBER OF THE MATRIX = 0.2759414E 04
THE COMPUTED SOLUTION X IS
```

2.0000004	2.4800003
4.9999970	4.8709986
2.9999988	2.6439993
–1.0000001	–1.0320001
–3.9999999	–3.9970000

The true solution to this problem is:

2.	2.48
5.	4.871
3.	2.644
–1.	–1.032
–4.	–3.997

Notice that a seemingly slight change in the right-hand side causes the solution to change noticeably. Furthermore, the relative error in the solution is about 2×10^{-7}. On the Honeywell computer, which has about 8 decimal digits for single-precision numbers, this represents the loss of about 1.5 decimal digits. A loss of up to 2×10^{-5} could be expected in light of the analysis given below.

Let Δb represent a perturbation in the right-hand side of a linear system.
If $Ax = b$ then

$$A(x + \Delta x) = b + \Delta b$$

where

$$\frac{||\Delta x||}{||x||} \leq K(A) \left[\frac{||\Delta b||}{||b||} \right]$$

where $K(A)$ is the condition number of A, $K(A) = ||A||\,||A^{-1}||$ and $||\cdot||$, is some norm, e.g., $||x||_1 = \sum_{i=1}^{n} |x_i|$ if x is a vector.

The methods used in our linear equation package are guaranteed to provide an accurate answer to a slightly perturbed problem. If we assume that our method produces the correct answer to a problem where $||\Delta b|| \leq \varepsilon ||b||$, where ε is the machine precision, then on the Honeywell 6000 where ε is about 10^{-8}, a relative error for the above example of 2×10^{-5} would not be surprising.

In our example one may consider the first column of B as b in (1.1), and the second column of B as $b + \Delta b$, so that $||\Delta b|| / ||b||$ is approximately .00015 using the $||\cdot||_1$ norm. If we look at the second solution as $x + \Delta x$ in (1.1) and the first solution as x, then $||\Delta x|| / ||x||$ is approximately .07. Thus equation (1.1) indicates that the condition number is at least 400, and the condition estimate, which at first appeared to be conservative, was in fact quite realistic.