

Algorithms

J. G. HERRIOT, Editor

ALGORITHM 342

GENERATOR OF RANDOM NUMBERS SATISFYING THE POISSON DISTRIBUTION [G5]

RICHARD H. SNOW (Recd. 20 Dec. 1966, 24 Aug. 1967, 5 Feb. 1968, 26 Mar. 1968, 5 June 1968 and 9 Sept. 1968)
IIT Research Institute, Chicago, Ill. 60616

KEY WORDS AND PHRASES: Poisson distribution, random number generator, Monte Carlo
CR CATEGORIES: 5.12, 5.5

integer procedure *poisson carlo* (*np_x*, *np_{x1}*, *random*); **value** *np_x*, *random*; **real** *np_x*, *np_{x1}*, *random*;
comment The Poisson distribution gives the probability that *px* events will occur in a certain interval or volume, where the expected or mean value of events is *np_x*. Applications are described by B. W. Lindgren and G. W. McElrath [1]. For a Monte Carlo calculation we wish to generate numbers *px* that satisfy the Poisson distribution, that is to find the inverse of the Poisson function. To do this we generate a pseudo-random number in the interval 0, 1 and find the number *px* such that *random* ≤ (probability that the number is *px* or less) and *random* > (the probability that the number is *px* - 1 or less).

poisson carlo returns the value -1 to signal that the procedure was called with a value of *np_x* < 0 or too large for the precision of the computer. It is the responsibility of the user to test the calculated value if there is any possibility of the occurrence of the error condition.

In order to save computing time, values of the Poisson distribution computed at a previous entry for the same value of *np_x* are stored in the **own array** *pson*. The previous value of *np_x* is *np_{x1}*. The actual parameter corresponding to *np_{x1}* must be a real identifier, not a constant or an expression. Before the first call of *poisson carlo* the calling program must set *np_{x1}* to a value ≠ *np_x*. The number of *pson* elements that were previously computed and stored is computed. If it is desired to save storage space at the expense of computing time, the upper bound 84 of *pson* may be reduced, but then the limit of *computed* near the end of the procedure must also be decreased accordingly.

The procedure which generates *random* is preferably algorithm 266 [3] or 294 [2]. It can be called as the actual parameter in the procedure call of *poisson carlo*.

The author thanks Mr. I. D. Hill for numerous suggestions and corrections which greatly improved the algorithm.

REFERENCES:

1. LINDGREN, B. W., AND McELRATH, G. W. *Introduction to Probability and Statistics*, 2 ed. Macmillan, New York, 1966, pp. 64-68.
2. PIKE, M. C., AND HILL, I. D. Algorithm 266, pseudo-random numbers. *Comm. ACM* 8 (Oct. 1965), 605.
3. STROME, W. M. Algorithm 294, uniform random. *Comm. ACM* 10 (Jan. 1967), 40;

begin

own integer *computed*; **own real** *pnc*;
own real array *pson* [0:84];

```
integer n; real ps;  
if npx < 0 then go to error;  
if npx ≠ npx1 then  
begin  
  computed := 0;  
  pnc := pson [0] := exp (-npx);  
  if pnc = 0 then go to error;  
  comment pson [0] is the probability that poisson carlo = 0.  
  It cannot be zero unless -npx underflows the argument  
  range of procedure exp. For most computers this sets an  
  upper limit of 85 for npx;  
  npx1 := npx  
end new npx;  
ps := pson [computed];  
if random ≤ ps then  
begin  
  integer nmin, nmax;  
  comment The probability term can be found by searching  
  the stored values;  
  nmin := 0; nmax := computed + 1;  
  for n := (nmax+nmin-1) ÷ 2 while nmax - nmin > 1 do  
    if random > pson[n] then nmin := n + 1 else nmax := n + 1;  
  poisson carlo := nmin  
end search  
else  
begin  
  real psc, pn; pn := pnc;  
  comment Additional probability terms must be computed;  
  for n := computed + 1, n + 1 while random > ps do  
  begin  
    pn := pn × npx/n;  
    psc := ps; ps := ps + pn;  
    comment ps = cumulative probability of terms up to n,  
    and pn = probability of nth term;  
    if ps = psc then go to error;  
    if n ≤ 84 then begin pson[n] := ps;  
    pnc := pn; computed := n end;  
    poisson carlo := n  
  end  
end more;  
go to fn;  
error: poisson carlo := -1;  
fn:  
end poisson carlo;  
comment The following is an example of a calling program for  
the case where poisson carlo is compiled within the calling  
program rather than separately. Instead of own variables,  
non-local variables may then be used. The program is within  
the IFIP subset if this change is made, and if the expression  
(nmax+nmin-1) ÷ 2 is replaced by the less efficient expression  
.501 × (nmax+nmin-2);  
begin  
integer x, computed; real array pson [0:84];  
real pnc, npx, npx1;  
real procedure random (x);
```

```

comment Procedure body random is inserted here;
integer procedure poisson carlo (npx, npx1, random);
comment Procedure body of poisson carlo is inserted here
after deleting declarations of own variables;
ininteger (2, x); npx1 := -1;
in1: inreal (2, npx);
outinteger (1, poisson carlo (npx, npx1, random (x)));
go to in1
end

```

ALGORITHM 343
EIGENVALUES AND EIGENVECTORS OF A
REAL GENERAL MATRIX [F2]

J. GRAD AND M. A. BREBNER

(Recd. 12 Oct. 1967, 1 July 1968 and 8 July 1968)

Computer Services, University of Birmingham, Birmingham 15, England

KEY WORDS AND PHRASES: eigenvalues, eigenvectors, latent roots, latent vectors, Householder's method, QR algorithm, inverse iteration

CR CATEGORIES: 5.14

ABSTRACT:

Purpose. This subroutine finds all the eigenvalues and eigenvectors of a real general matrix. The eigenvalues are computed by the QR double-step method and the eigenvectors by inverse iteration.

Method. Firstly the following preliminary modifications are carried out to improve the accuracy of the computed results. (i) The matrix is scaled by a sequence of similarity transformations so that the absolute sums of corresponding rows and columns are roughly equal. (ii) The scaled matrix is normalized so that the value of the Euclidean norm is equal to one.

The main part of the process commences with the reduction of the matrix to an upper-Hessenberg form by means of similarity transformations (Householder's method). Then the QR double-step iterative process is performed on the Hessenberg matrix until all elements of the subdiagonal that converge to zero are in modulus less than $2^{-t} \|H\|_E$, where t is the number of significant digits in the mantissa of a binary floating-point number. The eigenvalues are then extracted from this reduced form.

Inverse iteration is performed on the upper-Hessenberg matrix until the absolute value of the largest component of the right-hand side vector is greater than the bound $2^t/(100N)$, where N is the order of the matrix. Normally after this bound is achieved, one step more is performed to obtain the computed eigenvector, but at each step the residuals are computed, and if the residuals of one particular step are greater in absolute value than the residuals of the previous step, then the vector of the previous step is accepted as the computed eigenvector.

Program. The subroutine EIGENP is completely self-contained (composed of five subroutines

EIGENP, SCALE, HESQR, REALVE, and COMPVE) and communication to it is solely through the argument list. The entrance to the subroutine is achieved by CALL EIGENP (N, NM, A, T, EVR, EVI, VECR, VECI, INDIC) The meaning of the parameters is described in the comments at the beginning of the subroutine EIGENP.

REFERENCES:

1. WILKINSON, J. H. *The Algebraic Eigenvalue Problem*. Clarendon Press, Oxford, 1965, pp. 347-353, 485-567, 619-633.

Test results. All tests have been performed on a KDF9 computer ($t = 39$). No breakdown of the method has occurred and in general very accurate computed eigenvalues and eigenvectors have been obtained.

Some examples:

(i) The matrix

$$\begin{bmatrix} -5 & -1 & -1 & -5 & -1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

has all eigenvalues with modulus equal to one. The computed eigenvalues are

$-1.00000\ 0000$, $-.25000\ 00000 \pm i.96824\ 58366$, $.50000\ 00000 \pm i.86602\ 54038$.

The computed eigenvectors are

x_1	x_2, x_3	x_4, x_5
.447213595	1.000000000	$-.500000000 \mp i.866025404$
$-.447213595$	$-.250000000 \mp i.968245837$	$-1.000000000 \mp i.16E-10$
.447213595	$-.875000000 \pm i.484122918$	$-.500000000 \pm i.866025404$
$-.447213595$	$.687500000 \pm i.726184377$	$.500000000 \pm i.866025404$
.447213595	$.531250000 \mp i.847215107$	1.000000000

and the computed residuals are in modulus less than $.3E - 10$.

(ii) The matrix

$$\begin{bmatrix} -2 & 1 & 1 & 1 \\ -7 & -5 & -2 & -4 \\ 0 & -1 & -3 & -2 \\ -1 & 0 & -1 & 0 \end{bmatrix}$$

has the eigenvalues

$-4 \pm i2$ and $-1 \pm \sqrt{2}$.

The computed eigenvalues are

$-4.000000000 \pm i2.000000000$, -2.414213562 , $.4142135624$.

The computed eigenvectors are

x_1, x_2	x_3	x_4
$-.2000000000 \mp i.4000000000$.60E-12	$-.12E-11$
1.000000000	$-.7941044878$.4759631495
$.2000000000 \pm i.4000000000$.5615166683	.3365567706
$.14E-10 \pm i.63E-11$.2325878195	$-.8125199201$

and the computed residuals are in modulus less than $.7E - 10$.

(iii) The matrix A

$$A = \begin{bmatrix} 1 & 0 & 0.01 \\ 0.1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

is transformed by the process of scaling into the form B

$$B = \begin{bmatrix} .574423 & 0 & .066333 \\ .053454 & .574423 & 0 \\ 0 & .053454 & .574423 \end{bmatrix}$$

with the elements given to six decimal places. The obtained matrix B is essentially invariant under the QR double-step process. This kind of trouble was overcome by introducing the statements

```

R = DABS(X) + DABS(Y)
IF(R.EQ.0.0)SHIFT = A(M,M-1)
IF(R.EQ.0.0)GO TO 21

```

in the subroutine HESQR.

The exact eigenvalues of A are

1.1, $0.95 \pm i0.5\sqrt{0.03}$.

The computed eigenvalues are

1.100000000, $0.9500000000 \pm i0.0866025404$.

Acknowledgments. The authors wish to thank Dr. K. A. Redish, the former director of Computer Services at the University of Birmingham, and Dr. S. H. Hollingdale, the present director of Computer Services, for their encouragement. Finally, the authors are indebted to Dr. J. H. Wilkinson, National Physical Laboratory, Teddington, for useful consultations and suggestions.

```

SUBROUTINE EIGENP(N,NM,A,T,EVR,EVI,VECR,VECI,INDIC)
DOUBLE PRECISION D1,D2,D3,PRFACT
INTEGER I,IVEC,J,K,K1,KON,L,L1,M,N,NM
REAL ENORM,EPS,EX,R,R1,T
DIMENSION A(NM,1),VECR(NM,1),VECI(NM,1),
1EVR(NM),EVI(NM),INDIC(NM)
DIMENSION IWORK(100),LOCAL(100),PRFACT(100)
1,SUBDIA(100),WORK1(100),WORK2(100),WORK(100)
C
C THIS SUBROUTINE FINDS ALL THE EIGENVALUES AND THE
C EIGENVECTORS OF A REAL GENERAL MATRIX OF ORDER N.
C
C FIRST IN THE SUBROUTINE SCALE THE MATRIX IS SCALED SO THAT
C THE CORRESPONDING ROWS AND COLUMNS ARE APPROXIMATELY
C BALANCED AND THEN THE MATRIX IS NORMALISED SO THAT THE
C VALUE OF THE EUCLIDIAN NORM OF THE MATRIX IS EQUAL TO ONE.
C
C THE EIGENVALUES ARE COMPUTED BY THE QR DOUBLE-STEP METHOD
C IN THE SUBROUTINE HESQR.
C THE EIGENVECTORS ARE COMPUTED BY INVERSE ITERATION IN
C THE SUBROUTINE REALVE, FOR THE REAL EIGENVALUES, OR IN THE
C SUBROUTINE COMPVE, FOR THE COMPLEX EIGENVALUES.
C
C THE ELEMENTS OF THE MATRIX ARE TO BE STORED IN THE FIRST N
C ROWS AND COLUMNS OF THE TWO DIMENSIONAL ARRAY A. THE
C ORIGINAL MATRIX IS DESTROYED BY THE SUBROUTINE.
C N IS THE ORDER OF THE MATRIX.
C NM DEFINES THE FIRST DIMENSION OF THE TWO DIMENSIONAL
C ARRAYS A,VECR,VECI AND THE DIMENSION OF THE ONE
C DIMENSIONAL ARRAYS EVR,EVI AND INDIC. THEREFORE THE
C CALLING PROGRAM SHOULD CONTAIN THE FOLLOWING DECLARATION
C DIMENSION A(NM,NN),VECR(NM,NN),VECI(NM,NN),
C 1EVR(NM),EVI(NM),INDIC(NM)
C WHERE NM AND NN ARE ANY NUMBERS EQUAL TO OR GREATER THAN N
C THE UPPER LIMIT FOR NM IS EQUAL TO 100 BUT MAY BE
C INCREASED TO THE VALUE MAX BY REPLACING THE DIMENSION
C STATEMENT
C DIMENSION IWORK(100),LOCAL(100), ... ,WORK(100)
C IN THE SUBROUTINE EIGENP WITH
C DIMENSION IWORK(MAX),LOCAL(MAX), ... ,WORK(MAX)
C NM AND NN ARE OF COURSE BOUNDED BY THE SIZE OF THE STORE.
C
C THE REAL PARAMETER T MUST BE SET EQUAL TO THE NUMBER OF
C BINARY DIGITS IN THE MANTISSA OF A SINGLE PRECISION
C FLOATING-POINT NUMBER.
C
C THE REAL PARTS OF THE N COMPUTED EIGENVALUES WILL BE FOUND
C IN THE FIRST N PLACES OF THE ARRAY EVR AND THE IMAGINARY
C PARTS IN THE FIRST N PLACES OF THE ARRAY EVI.
C THE REAL COMPONENTS OF THE NORMALISED EIGENVECTOR I
C (I=1,2,...,N) CORRESPONDING TO THE EIGENVALUE STORED IN
C EVR(I) AND EVI(I) WILL BE FOUND IN THE FIRST N PLACES OF
C THE COLUMN I OF THE TWO DIMENSIONAL ARRAY VECR AND THE
C IMAGINARY COMPONENTS IN THE FIRST N PLACES OF THE COLUMN I
C OF THE TWO DIMENSIONAL ARRAY VECI.
C
C THE REAL EIGENVECTOR IS NORMALISED SO THAT THE SUM OF THE
C SQUARES OF THE COMPONENTS IS EQUAL TO ONE.
C THE COMPLEX EIGENVECTOR IS NORMALISED SO THAT THE
C COMPONENT WITH THE LARGEST VALUE IN MODULUS HAS ITS REAL
C PART EQUAL TO ONE AND THE IMAGINARY PART EQUAL TO ZERO.
C
C THE ARRAY INDIC INDICATES THE SUCCESS OF THE SUBROUTINE
C EIGENP AS FOLLOWS
C VALUE OF INDIC(I) EIGENVALUE I EIGENVECTOR I
C 0 NOT FOUND NOT FOUND
C 1 FOUND NOT FOUND
C 2 FOUND FOUND
C
C IF(N.NE.1)GO TO 1
EVR(1) = A(1,1)
EVI(1) = 0.0
VECR(1,1) = 1.0
VECI(1,1) = 0.0
INDIC(1) = 2
GO TO 25
C
C 1 CALL SCALE(N,NM,A,VECI,PRFACT,ENORM)
C THE COMPUTATION OF THE EIGENVALUES OF THE NORMALISED
C MATRIX.
EX = EXP(-T*ALOG(2.0))
CALL HESQR(N,NM,A,VECI,EVR,EVI,SUBDIA,INDIC,EPS,EX)
C
C THE POSSIBLE DECOMPOSITION OF THE UPPER-HESSSENBERG MATRIX
C INTO THE SUBMATRICES OF LOWER ORDER IS INDICATED IN THE
C ARRAY LOCAL. THE DECOMPOSITION OCCURS WHEN SOME
C SUBDIAGONAL ELEMENTS ARE IN MODULUS LESS THAN A SMALL
C POSITIVE NUMBER EPS DEFINED IN THE SUBROUTINE HESQR. THE
C AMOUNT OF WORK IN THE EIGENVECTOR PROBLEM MAY BE
C DIMINISHED IN THIS WAY.
J = N
I = 1
LOCAL(1) = 1
IF(J.EQ.1)GO TO 4
2 IF(ABS(SUBDIA(J-1)).GT.EPS)GO TO 3
I = I+1
LOCAL(I)=0
3 J = J-1
LOCAL(I)=LOCAL(I)+1
IF(J.NE.1)GO TO 2
C
C THE EIGENVECTOR PROBLEM.
4 K = 1
KON = 0
L = LOCAL(1)
M = N
DO 10 I=1,N
IVEC = N-I+1
IF(I.LE.L)GO TO 5
K = K+1
M = N-L
L = L+LOCAL(K)
5 IF(INDIC(IVEC).EQ.0)GO TO 10
IF(EVI(IVEC).NE.0.0)GO TO 8
C
C TRANSFER OF AN UPPER-HESSSENBERG MATRIX OF THE ORDER M FROM
C THE ARRAYS VECI AND SUBDIA INTO THE ARRAY A.
DO 7 K1=1,M
DO 6 L1=K1,M
6 A(K1,L1) = VECI(K1,L1)
IF(K1.EQ.1)GO TO 7
A(K1,K1-1) = SUBDIA(K1-1)
7 CONTINUE
C
C THE COMPUTATION OF THE REAL EIGENVECTOR IVEC OF THE UPPER-
C HESSENBERG MATRIX CORRESPONDING TO THE REAL EIGENVALUE
C EVR(IVEC).
CALL REALVE(N,NM,M,IVEC,A,VECR,EVR,EVI,IWORK,
1 WORK,INDIC,EPS,EX)
GO TO 10
C
C THE COMPUTATION OF THE COMPLEX EIGENVECTOR IVEC OF THE
C UPPER-HESSSENBERG MATRIX CORRESPONDING TO THE COMPLEX
C EIGENVALUE EVR(IVEC) + I*EVI(IVEC). IF THE VALUE OF KON IS
C NOT EQUAL TO ZERO THEN THIS COMPLEX EIGENVECTOR HAS
C ALREADY BEEN FOUND FROM ITS CONJUGATE.
8 IF(KON.NE.0)GO TO 9
KON = 1
CALL COMPVE(N,NM,M,IVEC,A,VECR,VECI,EVR,EVI,INDIC,
1 IWORK,SUBDIA,WORK1,WORK2,WORK,EPS,EX)
GO TO 10
9 KON = 0
10 CONTINUE
C
C THE RECONSTRUCTION OF THE MATRIX USED IN THE REDUCTION OF
C MATRIX A TO AN UPPER-HESSSENBERG FORM BY HOUSEHOLDER METHOD
DO 12 I=1,N
DO 11 J=I,N
A(I,J) = 0.0
11 A(J,I) = 0.0
12 A(I,I) = 1.0
IF(N.LE.2)GO TO 15
M = N-2
DO 14 K=1,M
L = K+1
DO 14 J=2,N
D1 = 0.0
DO 13 I=L,N
D2 = VECI(I,K)
D1 = D1+ D2*A(J,I)
13 DO 14 I=L,N
A(J,I) = A(J,I)-VECI(I,K)*D1
14
C
C THE COMPUTATION OF THE EIGENVECTORS OF THE ORIGINAL NON-
C SCALED MATRIX.
15 KON = 1
DO 24 I=1,N
L = 0
IF(EVI(I).EQ.0.0)GO TO 16
L = 1
IF(KON.EQ.0)GO TO 16
KON = 0
GO TO 24
16 DO 18 J=1,N
D1 = 0.0
D2 = 0.0
DO 17 K=1,N
D3 = A(J,K)
D1 = D1+D3*VECR(K,I)
IF(L.EQ.0)GO TO 17
D2 = D2+D3*VECR(K,I-1)
17 CONTINUE
WORK(J) = D1/PRFACT(J)
IF(L.EQ.0)GO TO 18
SUBDIA(J)=D2/PRFACT(J)
18 CONTINUE
C
C THE NORMALISATION OF THE EIGENVECTORS AND THE COMPUTATION
C OF THE EIGENVALUES OF THE ORIGINAL NON-NORMALISED MATRIX.
IF(L.EQ.1)GO TO 21
D1 = 0.0
DO 19 M=1,N
D1 = D1+WORK(M)**2
D1 = DSQRT(D1)
DO 20 M=1,N
VECI(M,1) = 0.0
20 VECR(M,1) = WORK(M)/D1
EVR(I) = EVR(I)*ENORM
GO TO 24
C
21 KON = 1
EVR(I) = EVR(I)*ENORM

```

```

EVR(I-1) = EVR(I)
EVI(I) = EVI(I)*ENORM
EVI(I-1) = -EVI(I)
R = 0.0
DO 22 J=1,N
  R1 = WORK(J)**2 + SUBDIA(J)**2
  IF(R.GE.R1)GO TO 22
  R = R1
  L = J
22 CONTINUE
D3 = WORK(L)
R1 = SUBDIA(L)
DO 23 J=1,N
  D1 = WORK(J)
  D2 = SUBDIA(J)
  VECR(J,I) = (D1*D3+D2*R1)/R
  VECI(J,I) = (D2*D3-D1*R1)/R
  VECR(J,I-1) = VECR(J,I)
23 VECI(J,I-1) = -VECI(J,I)
24 CONTINUE
C
25 RETURN
END

SUBROUTINE SCALE(N,NM,A,H,PRFACT,ENORM)
DOUBLE PRECISION COLUMN,FACTOR,FNORM,PRFACT,Q,ROW
INTEGER I,J,ITER,N,NCOUNT,NM
REAL BOUND1,BOUND2,ENORM
DIMENSION A(NM,1),H(NM,1),PRFACT(NM)
C
C THIS SUBROUTINE STORES THE MATRIX OF THE ORDER N FROM THE
C ARRAY A INTO THE ARRAY H. AFTERWARD THE MATRIX IN THE
C ARRAY A IS SCALED SO THAT THE QUOTIENT OF THE ABSOLUTE SUM
C OF THE OFF-DIAGONAL ELEMENTS OF COLUMN I AND THE ABSOLUTE
C SUM OF THE OFF-DIAGONAL ELEMENTS OF ROW I LIES WITHIN THE
C VALUES OF BOUND1 AND BOUND2.
C THE COMPONENT I OF THE EIGENVECTOR OBTAINED BY USING THE
C SCALED MATRIX MUST BE DIVIDED BY THE VALUE FOUND IN THE
C PRFACT(I) OF THE ARRAY PRFACT. IN THIS WAY THE EIGENVECTOR
C OF THE NON-SCALED MATRIX IS OBTAINED.
C
C AFTER THE MATRIX IS SCALED IT IS NORMALISED SO THAT THE
C VALUE OF THE EUCLIDIAN NORM IS EQUAL TO ONE.
C IF THE PROCESS OF SCALING WAS NOT SUCCESSFUL THE ORIGINAL
C MATRIX FROM THE ARRAY H WOULD BE STORED BACK INTO A AND
C THE EIGENPROBLEM WOULD BE SOLVED BY USING THIS MATRIX.
C NM DEFINES THE FIRST DIMENSION OF THE ARRAYS A AND H. NM
C MUST BE GREATER OR EQUAL TO N.
C THE EIGENVALUES OF THE NORMALISED MATRIX MUST BE
C MULTIPLIED BY THE SCALAR ENORM IN ORDER THAT THEY BECOME
C THE EIGENVALUES OF THE NON-NORMALISED MATRIX.
C
DO 2 I=1,N
  DO 1 J=1,N
1    H(I,J) = A(I,J)
2    PRFACT(I) = 1.0
  BOUND1 = 0.75
  BOUND2 = 1.33
  ITER = 0
3  NCOUNT = 0
  DO 8 I=1,N
    COLUMN = 0.0
    ROW = 0.0
    DO 4 J=1,N
      IF(I.EQ.J)GO TO 4
      COLUMN = COLUMN+ ABS(A(J,I))
      ROW = ROW + ABS(A(I,J))
4    CONTINUE
    IF(COLUMN.EQ.0.0)GO TO 5
    IF(ROW.EQ.0.0)GO TO 5
    Q = COLUMN/ROW
    IF(Q.LT.BOUND1)GO TO 6
    IF(Q.GT.BOUND2)GO TO 6
5    NCOUNT = NCOUNT + 1
    GO TO 8
6    FACTOR = DSQRT(Q)
    DO 7 J=1,N
      IF(I.EQ.J)GO TO 7
      A(I,J) = A(I,J)*FACTOR
      A(J,I) = A(J,I)/FACTOR
7    CONTINUE
    PRFACT(I) = PRFACT(I)*FACTOR
8  CONTINUE
  ITER = ITER+1
  IF(ITER.GT.30)GO TO 11
  IF(NCOUNT.LT.N)GO TO 3
C
FNORM = 0.0
DO 9 I=1,N
  DO 9 J=1,N
    Q = A(I,J)
9  FNORM = FNORM+Q*Q
FNORM = DSQRT(FNORM)
DO 10 I=1,N
  DO 10 J=1,N
10 A(I,J) = A(I,J)/FNORM
ENORM = FNORM
GO TO 13

```

```

C
11 DO 12 I=1,N
    DO 12 J=1,N
12   A(I,J) = H(I,J)
ENORM = 1.0
C
13 RETURN
END

SUBROUTINE HESQR(N,NM,A,H,EVR,EVI,SUBDIA,INDIC,EPS,EX)
DOUBLE PRECISION S,SR,SR2,X,Y,Z
INTEGER I,J,K,L,M,MAXST,M1,N,NM,NS
REAL EPS,EX,R,SHIFT,T
DIMENSION A(NM,1),H(NM,1),EVR(NM),EVI(NM),SUBDIA(NM)
DIMENSION INDIC(NM)
C
C THIS SUBROUTINE FINDS ALL THE EIGENVALUES OF A REAL
C GENERAL MATRIX. THE ORIGINAL MATRIX A OF ORDER N IS
C REDUCED TO THE UPPER-HESSSENBERG FORM H BY MEANS OF
C SIMILARITY TRANSFORMATIONS(HOUSEHOLDER METHOD). THE MATRIX
C H IS PRESERVED IN THE UPPER HALF OF THE ARRAY H AND IN THE
C ARRAY SUBDIA. THE SPECIAL VECTORS USED IN THE DEFINITION
C OF THE HOUSEHOLDER TRANSFORMATION MATRICES ARE STORED IN
C THE LOWER PART OF THE ARRAY H.
C NM IS THE FIRST DIMENSION OF THE ARRAYS A AND H. NM MUST
C BE EQUAL TO OR GREATER THAN N.
C THE REAL PARTS OF THE N EIGENVALUES WILL BE FOUND IN THE
C FIRST N PLACES OF THE ARRAY EVR,AND
C THE IMAGINARY PARTS IN THE FIRST N PLACES OF THE ARRAY EVI
C THE ARRAY INDIC INDICATES THE SUCCESS OF THE ROUTINE AS
C FOLLOWS
C VALUE OF INDIC(I) EIGENVALUE I
C 0 NOT FOUND
C 1 FOUND
C EPS IS A SMALL POSITIVE NUMBER THAT NUMERICALLY REPRESENTS
C ZERO IN THE PROGRAM. EPS = (EUCLIDIAN NORM OF H)*EX ,WHERE
C EX = 2**(-T). T IS THE NUMBER OF BINARY DIGITS IN THE
C MANTISSA OF A FLOATING POINT NUMBER.
C
C
C
C REDUCTION OF THE MATRIX A TO AN UPPER-HESSSENBERG FORM H.
C THERE ARE N-2 STEPS.
  IF(N-2)14,I,2
1  SUBDIA(1) = A(2,1)
  GO TO 14
2  M = N-2
  DO 12 K=1,M
    L = K+1
    S = 0.0
    DO 3 I=L,N
      H(I,K) = A(I,K)
3    S = S+ABS(A(I,K))
    IF(S.NE.ABS(A(K+1,K)))GO TO 4
    SUBDIA(K) = A(K+1,K)
    H(K+1,K) = 0.0
    GO TO 12
4  SR2 = 0.0
    DO 5 I=L,N
      SR = A(I,K)
      SR = SR/S
5    A(I,K) = SR
      SR2 = SR2+SR*SR
    SR = DSQRT(SR2)
    IF(A(L,K).LT.0.0)GO TO 6
    SR = -SR
6  SR2 = SR2-SR*A(L,K)
    A(L,K) = A(L,K)-SR
    H(L,K) = H(L,K)-SR*S
    SUBDIA(K) = SR*S
    X = S*DSQRT(SR2)
    DO 7 I=L,N
      H(I,K) = H(I,K)/X
7  SUBDIA(I) = A(I,K)/SR2
C PREMULTIPLICATION BY THE MATRIX PR.
  DO 9 J=L,N
    SR = 0.0
    DO 8 I=L,N
      SR = SR+A(I,K)*A(I,J)
    DO 9 I=L,N
9  A(I,J) = A(I,J)-SUBDIA(I)*SR
C POSTMULTIPLICATION BY THE MATRIX PR.
  DO 11 J=1,N
    SR=0.0
    DO 10 I=L,N
      SR = SR+A(J,I)*A(I,K)
    DO 11 I=L,N
11 A(J,I) = A(J,I)-SUBDIA(I)*SR
12 CONTINUE
  DO 13 K=1,M
13 A(K+1,K) = SUBDIA(K)
C TRANSFER OF THE UPPER HALF OF THE MATRIX A INTO THE
C ARRAY H AND THE CALCULATION OF THE SMALL POSITIVE NUMBER
C EPS.
  SUBDIA(N-1) = A(N,N-1)
14 EPS = 0.0
  DO 15 K=1,N
    INDIC(K) = 0

```

```

      IF(K.NE.N)EPS = EPS+SUBDIA(K)**2
      DO 15 I=K,N
        H(K,I) = A(K,I)
15     EPS = EPS + A(K,I)**2
      EPS = EX*SQRT(EPS)
C
C THE QR ITERATIVE PROCESS. THE UPPER-HESSSENBERG MATRIX H IS
C REDUCED TO THE UPPER-MODIFIED TRIANGULAR FORM.
C
C DETERMINATION OF THE SHIFT OF THE ORIGIN FOR THE FIRST STEP OF
C THE QR ITERATIVE PROCESS.
      SHIFT = A(N,N-1)
      IF(N.LE.2)SHIFT = 0.0
      IF(A(N,N).NE.0.0)SHIFT = 0.0
      IF(A(N-1,N).NE.0.0)SHIFT = 0.0
      IF(A(N-1,N-1).NE.0.0)SHIFT = 0.0
      M = N
      NS = 0
      MAXST = N*10
C
C TESTING IF THE UPPER HALF OF THE MATRIX IS EQUAL TO ZERO.
C IF IT IS EQUAL TO ZERO THE QR PROCESS IS NOT NECESSARY.
      DO 16 I=2,N
        DO 16 K=I,N
          IF(A(I-1,K).NE.0.0)GO TO 18
16     CONTINUE
      DO 17 I=1,N
        INDIC(I)=1
        EVR(I) = A(I,I)
17     EVI(I) = 0.0
      GO TO 37
C
C START THE MAIN LOOP OF THE QR PROCESS.
18 K=M-1
      M1=K
      I = K
C FIND ANY DECOMPOSITIONS OF THE MATRIX.
C JUMP TO 34 IF THE LAST SUBMATRIX OF THE DECOMPOSITION IS
C OF THE ORDER ONE.
C JUMP TO 35 IF THE LAST SUBMATRIX OF THE DECOMPOSITION IS
C OF THE ORDER TWO.
      IF(K)37,34,19
19 IF(ABS(A(M,K)).LE.EPS)GO TO 34
      IF(M-2.EQ.0)GO TO 35
20 I = I-1
      IF(ABS(A(K,I)).LE.EPS)GO TO 21
      K = I
      IF(K.GT.1)GO TO 20
21 IF(K.EQ.M1)GO TO 35
C TRANSFORMATION OF THE MATRIX OF THE ORDER GREATER THAN TWO
      S = A(M,M)+A(M1,M1)+SHIFT
      SR = A(M,M)*A(M1,M1)-A(M,M1)*A(M1,M)+0.25*SHIFT**2
      A(K+2,K) = 0.0
C CALCULATE X1,Y1,Z1, FOR THE SUBMATRIX OBTAINED BY THE
C DECOMPOSITION.
      X = A(K,K)*(A(K,K)-S)+A(K,K+1)*A(K+1,K)+SR
      Y = A(K+1,K)*(A(K,K)+A(K+1,K+1)-S)
      R = DABS(X)+DABS(Y)
      IF(R.EQ.0.0)SHIFT = A(M,M-1)
      IF(R.EQ.0.0)GO TO 21
      Z = A(K+2,K+1)*A(K+1,K)
      SHIFT = 0.0
      NS = NS+1
C
C THE LOOP FOR ONE STEP OF THE QR PROCESS.
      DO 33 I=K,M1
        IF(I.EQ.K)GO TO 22
C CALCULATE XR,YR,ZR.
        X = A(I,I-1)
        Y = A(I+1,I-1)
        Z = 0.0
        IF(I+2.GT.M)GO TO 22
        Z = A(I+2,I-1)
22     SR2 = DABS(X)+DABS(Y)+DABS(Z)
        IF(SR2.EQ.0.0)GO TO 23
        X = X/SR2
        Y = Y/SR2
        Z = Z/SR2
23     S = DSQRT(X*X + Y*Y + Z*Z)
        IF(X.LT.0.0)GO TO 24
        S = -S
24     IF(I.EQ.K)GO TO 25
        A(I,I-1) = S*SR2
25     IF(SR2.NE.0.0)GO TO 26
        IF(I+3.GT.M)GO TO 33
        GO TO 32
26     SR = 1.0-X/S
        S = X-S
        X = Y/S
        Y = Z/S
C PREMULTIPLICATION BY THE MATRIX PR.
      DO 28 J=1,M
        S = A(I,J)+A(I+1,J)*X
        IF(I+2.GT.M)GO TO 27
        S = S+A(I+2,J)*Y
27     S = S*SR
        A(I,J) = A(I,J)-S
        A(I+1,J) = A(I+1,J)-S*X
        IF(I+2.GT.M)GO TO 28
        A(I+2,J) = A(I+2,J)-S*Y
28     CONTINUE
C POSTMULTIPLICATION BY THE MATRIX PR.
      L = I+2
      IF(I.LT.M1)GO TO 29
      L = M
29 DO 31 J=K,L
        S = A(J,I)+A(J,I+1)*X
        IF(I+2.GT.M)GO TO 30
        S = S + A(J,I+2)*Y
30     S = S*SR
        A(J,I) = A(J,I)-S
        A(J,I+1)=A(J,I+1)-S*X
        IF(I+2.GT.M)GO TO 31
        A(J,I+2)=A(J,I+2)-S*Y
31     CONTINUE
        IF(I+3.GT.M)GO TO 33
        S = -A(I+3,I+2)*Y*SR
32     A(I+3,I) = S
        A(I+3,I+1) = S*X
        A(I+3,I+2) = S*Y + A(I+3,I+2)
33     CONTINUE
C
      IF(NS.GT.MAXST)GO TO 37
      GO TO 18
C
C COMPUTE THE LAST EIGENVALUE.
34 EVR(M) = A(M,M)
      EVI(M) = 0.0
      INDIC(M) = 1
      M = K
      GO TO 18
C
C COMPUTE THE EIGENVALUES OF THE LAST 2X2 MATRIX OBTAINED BY
C THE DECOMPOSITION.
35 R = 0.5*(A(K,K)+A(M,M))
      S = 0.5*(A(M,M)-A(K,K))
      S = S*S + A(K,M)*A(M,K)
      INDIC(K) = 1
      INDIC(M) = 1
      IF(S.LT.0.0)GO TO 36
      T = DSQRT(S)
      EVR(K) = R-T
      EVR(M) = R+T
      EVI(K) = 0.0
      EVI(M) = 0.0
      M = M-2
      GO TO 18
36 T = DSQRT(-S)
      EVR(K) = R
      EVI(K) = T
      EVR(M) = R
      EVI(M) = -T
      M = M-2
      GO TO 18
C
37 RETURN
      END
      SUBROUTINE REALVE(N,NM,M,IVEC,A,VECR,EVR,EVI,
      IWORK,WORK,INDIC,EPS,EX)
      DOUBLE PRECISION S,SR
      INTEGER I,IVEC,ITER,J,K,L,M,N,NM,NS
      REAL BOUND,EPS,EVALUE,EX,PREVIS,R,R1,T
      DIMENSION A(NM,1),VECR(NM,1),EVR(NM)
      DIMENSION EVI(NM),IWORK(NM),WORK(NM),INDIC(NM)
C
C THIS SUBROUTINE FINDS THE REAL EIGENVECTOR OF THE REAL
C UPPER-HESSSENBERG MATRIX IN THE ARRAY A,CORRESPONDING TO
C THE REAL EIGENVALUE STORED IN EVR(IVEC). THE INVERSE
C ITERATION METHOD IS USED.
C NOTE THE MATRIX IN A IS DESTROYED BY THE SUBROUTINE.
C N IS THE ORDER OF THE UPPER-HESSSENBERG MATRIX.
C NM DEFINES THE FIRST DIMENSION OF THE TWO DIMENSIONAL
C ARRAYS A AND VECR. NM MUST BE EQUAL TO OR GREATER THAN N.
C M IS THE ORDER OF THE SUBMATRIX OBTAINED BY A SUITABLE
C DECOMPOSITION OF THE UPPER-HESSSENBERG MATRIX IF SOME
C SUBDIAGONAL ELEMENTS ARE EQUAL TO ZERO. THE VALUE OF M IS
C CHOSEN SO THAT THE LAST N-M COMPONENTS OF THE EIGENVECTOR
C ARE ZERO.
C IVEC GIVES THE POSITION OF THE EIGENVALUE IN THE ARRAY EVR
C FOR WHICH THE CORRESPONDING EIGENVECTOR IS COMPUTED.
C THE ARRAY EVI WOULD CONTAIN THE IMAGINARY PARTS OF THE N
C EIGENVALUES IF THEY EXISTED.
C
C THE M COMPONENTS OF THE COMPUTED REAL EIGENVECTOR WILL BE
C FOUND IN THE FIRST M PLACES OF THE COLUMN IVEC OF THE TWO
C DIMENSIONAL ARRAY VECR.
C
C IWORK AND WORK ARE THE WORKING STORES USED DURING THE
C GAUSSIAN ELIMINATION AND BACKSUBSTITUTION PROCESS.
C THE ARRAY INDIC INDICATES THE SUCCESS OF THE ROUTINE AS
C FOLLOWS
      VALUE OF INDIC(I)      EIGENVECTOR I
      1                      NOT FOUND
      2                      FOUND
C EPS IS A SMALL POSITIVE NUMBER THAT NUMERICALLY REPRESENTS
C ZERO IN THE PROGRAM. EPS = (EUCLIDIAN NORM OF A)*EX,WHERE
C EX = 2**(I-T). T IS THE NUMBER OF BINARY DIGITS IN THE

```

```

C MANTISSA OF A FLOATING POINT NUMBER.
  VECR(1,IVEC) = 1.0
  IF(M.EQ.1)GO TO 24
C SMALL PERTURBATION OF EQUAL EIGENVALUES TO OBTAIN A FULL
C SET OF EIGENVECTORS.
  EVALUE = EVR(IVEC)
  IF(IVEC.EQ.M)GO TO 2
  K = IVEC+1
  R = 0.0
  DO 1 I=K,M
    IF(EVALUE.NE.EVR(I))GO TO 1
    IF(EVI(I).NE.0.0)GO TO 1
    R = R+3.0
  1 CONTINUE
  EVALUE = EVALUE+R*EX
  2 DO 3 K=1,M
  3 A(K,K) = A(K,K)-EVALUE
C
C GAUSSIAN ELIMINATION OF THE UPPER-HESSBERG MATRIX A. ALL
C ROW INTERCHANGES ARE INDICATED IN THE ARRAY IWORK.ALL THE
C MULTIPLIERS ARE STORED AS THE SUBDIAGONAL ELEMENTS OF A.
  K = M-1
  DO 8 I=1,K
    L = I+1
    IWORK(I) = 0
    IF(A(I+1,I).NE.0.0)GO TO 4
    IF(A(I,I).NE.0.0)GO TO 8
    A(I,I) = EPS
    GO TO 8
  4 IF(ABS(A(I,I)).GE.ABS(A(I+1,I)))GO TO 6
  IWORK(I) = 1
  DO 5 J=I,M
    R = A(I,J)
    A(I,J) = A(I+1,J)
  5 A(I+1,J) = R
  6 R = -A(I+1,I)/A(I,I)
  A(I+1,I) = R
  DO 7 J=L,M
  7 A(I+1,J) = A(I+1,J)+R*A(I,J)
  8 CONTINUE
  IF(A(M,M).NE.0.0)GO TO 9
  A(M,M) = EPS
C
C THE VECTOR {1,1,...,1} IS STORED IN THE PLACE OF THE RIGHT
C HAND SIDE COLUMN VECTOR.
  9 DO 11 I=1,N
    IF(I.GT.M)GO TO 10
    WORK(I) = 1.0
    GO TO 11
  10 WORK(I) = 0.0
  11 CONTINUE
C
C THE INVERSE ITERATION IS PERFORMED ON THE MATRIX UNTIL THE
C INFINITE NORM OF THE RIGHT-HAND SIDE VECTOR IS GREATER
C THAN THE BOUND DEFINED AS 0.01/(N*EX).
  BOUND = 0.01/(EX * FLOAT(N))
  NS = 0
  ITER = 1
C
C THE BACKSUBSTITUTION.
  12 R = 0.0
  DO 15 I=1,M
    J = M-I+1
    S = WORK(J)
    IF(J.EQ.M)GO TO 14
    L = J+1
    DO 13 K=L,M
      SR = WORK(K)
    13 S = S - SR*A(J,K)
  14 WORK(J) = S/A(J,J)
    T = ABS(WORK(J))
    IF(R.GE.T)GO TO 15
    R = T
  15 CONTINUE
C
C THE COMPUTATION OF THE RIGHT-HAND SIDE VECTOR FOR THE NEW
C ITERATION STEP.
  DO 16 I=1,M
  16 WORK(I) = WORK(I)/R
C
C THE COMPUTATION OF THE RESIDUALS AND COMPARISON OF THE
C RESIDUALS OF THE TWO SUCCESSIVE STEPS OF THE INVERSE
C ITERATION.IF THE INFINITE NORM OF THE RESIDUAL VECTOR IS
C GREATER THAN THE INFINITE NORM OF THE PREVIOUS RESIDUAL
C VECTOR THE COMPUTED EIGENVECTOR OF THE PREVIOUS STEP IS
C TAKEN AS THE FINAL EIGENVECTOR.
  R1 = 0.0
  DO 18 I=1,M
    T = 0.0
    DO 17 J=I,M
      T = T+A(I,J)*WORK(J)
    17 T = ABS(T)
    IF(R1.GE.T)GO TO 18
    R1 = T
  18 CONTINUE
  IF(ITER.EQ.1)GO TO 19
  IF(PREVIS.LE.R1)GO TO 24
  19 DO 20 I=1,M
  20 VECR(I,IVEC) = WORK(I)
  PREVIS = R1
  IF(NS.EQ.1)GO TO 24
  IF(ITER.GT.6)GO TO 25
  ITER = ITER+1
  IF(R.LT.BOUND)GO TO 21
  NS = 1
C
C GAUSSIAN ELIMINATION OF THE RIGHT-HAND SIDE VECTOR.
  21 K = M-1
  DO 23 I=1,K
    R = WORK(I+1)
    IF(IWORK(I).EQ.0)GO TO 22
    WORK(I+1)=WORK(I)+WORK(I+1)*A(I+1,I)
    WORK(I) = R
    GO TO 23
  22 WORK(I+1)=WORK(I+1)+WORK(I)*A(I+1,I)
  23 CONTINUE
  GO TO 12
C
  24 INDIC(IVEC) = 2
  25 IF(M.EQ.N)GO TO 27
  J = M+1
  DO 26 I=J,N
  26 VECR(I,IVEC) = 0.0
  27 RETURN
  END
SUBROUTINE COMPVE(N,NM,M,IVEC,A,VECR,H,EVR,EVI,INDIC,
  IWORK,SUBDIA,WORK1,WORK2,WORK,EPS,EX)
  DOUBLE PRECISION D,D1
  INTEGER I,I1,I2,ITER,IVEC,J,K,L,M,N,NM,NS
  REAL B,BOUND,EPS,ETA,EX,FKSI,PREVIS,R,S,U,V
  DIMENSION A(NM,1),VECR(NM,1),H(NM,1),EVR(NM),EVI(NM),
  IINDIC(NM),IWORK(NM),SUBDIA(NM),WORK1(NM),WORK2(NM),
  2WORK(NM)
C
C THIS SUBROUTINE FINDS THE COMPLEX EIGENVECTOR OF THE REAL
C UPPER-HESSBERG MATRIX OF ORDER N CORRESPONDING TO THE
C COMPLEX EIGENVALUE WITH THE REAL PART IN EVR(IVEC) AND THE
C CORRESPONDING IMAGINARY PART IN EVI(IVEC). THE INVERSE
C ITERATION METHOD IS USED MODIFIED TO AVOID THE USE OF
C COMPLEX ARITHMETIC.
C THE MATRIX ON WHICH THE INVERSE ITERATION IS PERFORMED IS
C BUILT UP IN THE ARRAY A BY USING THE UPPER-HESSBERG
C MATRIX PRESERVED IN THE UPPER HALF OF THE ARRAY H AND IN
C THE ARRAY SUBDIA.
C NM DEFINES THE FIRST DIMENSION OF THE TWO DIMENSIONAL
C ARRAYS A,VECR AND H. NM MUST BE EQUAL TO OR GREATER
C THAN N.
C M IS THE ORDER OF THE SUBMATRIX OBTAINED BY A SUITABLE
C DECOMPOSITION OF THE UPPER-HESSBERG MATRIX IF SOME
C SUBDIAGONAL ELEMENTS ARE EQUAL TO ZERO. THE VALUE OF M IS
C CHOSEN SO THAT THE LAST N-M COMPONENTS OF THE COMPLEX
C EIGENVECTOR ARE ZERO.
C
C THE REAL PARTS OF THE FIRST M COMPONENTS OF THE COMPUTED
C COMPLEX EIGENVECTOR WILL BE FOUND IN THE FIRST M PLACES OF
C THE COLUMN WHOSE TOP ELEMENT IS VECR(1,IVEC) AND THE
C CORRESPONDING IMAGINARY PARTS OF THE FIRST M COMPONENTS OF
C THE COMPLEX EIGENVECTOR WILL BE FOUND IN THE FIRST M
C PLACES OF THE COLUMN WHOSE TOP ELEMENT IS VECR(1,IVEC-1).
C
C THE ARRAY INDIC INDICATES THE SUCCESS OF THE ROUTINE AS
C FOLLOWS
C VALUE OF INDIC(I) EIGENVECTOR I
C 1 NOT FOUND
C 2 FOUND
C THE ARRAYS IWORK,WORK1,WORK2 AND WORK ARE THE WORKING
C STORES USED DURING THE INVERSE ITERATION PROCESS.
C EPS IS A SMALL POSITIVE NUMBER THAT NUMERICALLY REPRESENTS
C ZERO IN THE PROGRAM. EPS = (EUCLIDIAN NORM OF H)*EX, WHERE
C EX = 2**(-T). T IS THE NUMBER OF BINARY DIGITS IN THE
C MANTISSA OF A FLOATING POINT NUMBER.
C
  FKSI = EVR(IVEC)
  ETA = EVI(IVEC)
C THE MODIFICATION OF THE EIGENVALUE (FKSI + I*ETA) IF MORE
C EIGENVALUES ARE EQUAL.
  IF(IVEC.EQ.M)GO TO 2
  K = IVEC+1
  R = 0.0
  DO 1 I=K,M
    IF(FKSI.NE.EVR(I))GO TO 1
    IF(ABS(ETA).NE.ABS(EVI(I)))GO TO 1
    R = R + 3.0
  1 CONTINUE
  R = R*EX
  FKSI = FKSI+R
  ETA = ETA +R
C
C THE MATRIX ((H-FKSI*I)*(H-FKSI*I) + (ETA*ETA)*I) IS
C STORED INTO THE ARRAY A.
  2 R = FKSI*FKSI + ETA*ETA
  S = 2.0*FKSI
  L = M-1
  DO 5 I=1,M
    DO 4 J=I,M
      D = 0.0

```

```

      A(J,I) = 0.0
      DO 3 K = 1,J
3        D = D+H(I,K)*H(K,J)
4        A(I,J) = D-S*H(I,J)
5        A(I,I) = A(I,I)+R
      DO 9 I=1,L
        R = SUBDIA(I)
        A(I+1,I) = -S*R
        I1 = I+1
        DO 6 J=1,I1
9          A(J,I) = A(J,I)+R*H(J,I+1)
          IF(I.EQ.1)GO TO 7
          A(I+1,I-1) = R*SUBDIA(I-1)
6          DO 8 J=I,M
8            A(I+1,J) = A(I+1,J)+R*H(I,J)
          CONTINUE
C
C THE GAUSSIAN ELIMINATION OF THE MATRIX
C ((H-FKSI*I)*(H-FKSI*I) + (ETA*ETA)*I) IN THE ARRAY A. THE
C ROW INTERCHANGES THAT OCCUR ARE INDICATED IN THE ARRAY
C IWORK. ALL THE MULTIPLIERS ARE STORED IN THE FIRST AND IN
C THE SECOND SUBDIAGONAL OF THE ARRAY A.
      K = M-1
      DO 18 I=1,K
        I1 = I+1
        I2 = I+2
        IWORK(I) = 0
        IF(I.EQ.K)GO TO 10
        IF(A(I+2,I).NE.0.0)GO TO 11
10       IF(A(I+1,I).NE.0.0)GO TO 11
        IF(A(I,I).NE.0.0)GO TO 18
        A(I,I) = EPS
        GO TO 18
C
11       IF(I.EQ.K)GO TO 12
        IF(ABS(A(I+1,I1)).GE.ABS(A(I+2,I1)))GO TO 12
        IF(ABS(A(I,I)).GE.ABS(A(I+2,I1)))GO TO 16
        L = I+2
        IWORK(I) = 2
        GO TO 13
12       IF(ABS(A(I,I)).GE.ABS(A(I+1,I1)))GO TO 15
        L = I+1
        IWORK(I) = 1
C
13       DO 14 J=I,M
14         R = A(I,J)
        A(I,J) = A(L,J)
        A(L,J) = R
15       IF(I.NE.K)GO TO 16
        I2 = I1
16       DO 17 L=I1,I2
        R = -A(L,I)/A(I,I)
        A(L,I) = R
        DO 17 J=I1,M
17         A(L,J) = A(L,J)+R*A(I,J)
18       CONTINUE
        IF(A(M,M).NE.0.0)GO TO 19
        A(M,M) = EPS
C
C THE VECTOR (1,1,...,1) IS STORED INTO THE RIGHT-HAND SIDE
C VECTORS VECR( ,IVEC) AND VECR( ,IVEC-1) REPRESENTING THE
C COMPLEX RIGHT-HAND SIDE VECTOR.
19       DO 21 I=1,N
        IF(I.GT.M)GO TO 20
        VECR(I,IVEC) = 1.0
        VECR(I,IVEC-1) = 1.0
        GO TO 21
20       VECR(I,IVEC) = 0.0
        VECR(I,IVEC-1) = 0.0
21       CONTINUE
C
C THE INVERSE ITERATION IS PERFORMED ON THE MATRIX UNTIL THE
C INFINITE NORM OF THE RIGHT-HAND SIDE VECTOR IS GREATER
C THAN THE BOUND DEFINED AS 0.01/(N*EX).
        BOUND = 0.01/(EX*FLOAT(N))
        NS = 0
        ITER = 1
        DO 22 I=1,M
22       WORK(I) = H(I,I)-FKSI
C
C THE SEQUENCE OF THE COMPLEX VECTORS Z(S) = P(S)+I*Q(S) AND
C W(S+1) = U(S+1)+I*V(S+1) IS GIVEN BY THE RELATIONS
C (A - (FKSI-I*ETA)*I)*W(S+1) = Z(S) AND
C Z(S+1) = W(S+1)/MAX(|W(S+1)|).
C THE FINAL W(S) IS TAKEN AS THE COMPUTED EIGENVECTOR.
C
C THE COMPUTATION OF THE RIGHT-HAND SIDE VECTOR
C (A-FKSI*I)*P(S)-ETA*Q(S). A IS AN UPPER-HESSSENBERG MATRIX.
23       DO 27 I=1,M
        D = WORK(I)*VECR(I,IVEC)
        IF(I.EQ.1)GO TO 24
        D = D+SUBDIA(I-1)*VECR(I-1,IVEC)
24       L = I+1
        IF(L.GT.M)GO TO 26
        DO 25 K=L,M
25         D = D+H(I,K)*VECR(K,IVEC)
26       VECR(I,IVEC-1) = D-ETA*VECR(I,IVEC-1)
27       CONTINUE
C
C GAUSSIAN ELIMINATION OF THE RIGHT-HAND SIDE VECTOR.
      K = M-1
      DO 28 I=1,K
        L = I+IWORK(I)
        R = VECR(L,IVEC-1)
        VECR(L,IVEC-1) = VECR(I,IVEC-1)
        VECR(I,IVEC-1) = R
        VECR(I+1,IVEC-1) = VECR(I+1,IVEC-1)+A(I+1,I)*R
        IF(I.EQ.K)GO TO 28
        VECR(I+2,IVEC-1) = VECR(I+2,IVEC-1)+A(I+2,I)*R
28       CONTINUE
C
C THE COMPUTATION OF THE REAL PART U(S+1) OF THE COMPLEX
C VECTOR W(S+1). THE VECTOR U(S+1) IS OBTAINED AFTER THE
C BACKSUBSTITUTION.
      DO 31 I=1,M
        J = M-I+1
        D = VECR(J,IVEC-1)
        IF(J.EQ.M)GO TO 30
        L = J+1
        DO 29 K=L,M
          D1 = A(J,K)
29         D = D-D1*VECR(K,IVEC-1)
30       VECR(J,IVEC-1) = D/A(J,J)
31       CONTINUE
C
C THE COMPUTATION OF THE IMAGINARY PART V(S+1) OF THE VECTOR
C W(S+1), WHERE V(S+1) = (P(S)-(A-FKSI*I)*U(S+1))/ETA.
      DO 35 I=1,M
        D = WORK(I)*VECR(I,IVEC-1)
        IF(I.EQ.1)GO TO 32
        D = D+SUBDIA(I-1)*VECR(I-1,IVEC-1)
32       L = I+1
        IF(L.GT.M)GO TO 34
        DO 33 K=L,M
33         D = D+H(I,K)*VECR(K,IVEC-1)
34       VECR(I,IVEC) = (VECR(I,IVEC)-D)/ETA
35       CONTINUE
C
C THE COMPUTATION OF (INFIN. NORM OF W(S+1))**2 .
      L = 1
      S = 0.0
      DO 36 I=1,M
        R = VECR(I,IVEC)**2 + VECR(I,IVEC-1)**2
        IF(R.LE.S)GO TO 36
        S = R
        L = I
36       CONTINUE
C
C THE COMPUTATION OF THE VECTOR Z(S+1),WHERE Z(S+1)= W(S+1)/
C (COMPONENT OF W(S+1) WITH THE LARGEST ABSOLUTE VALUE) .
      U = VECR(L,IVEC-1)
      V = VECR(L,IVEC)
      DO 37 I=1,M
        B = VECR(I,IVEC)
        R = VECR(I,IVEC-1)
        VECR(I,IVEC) = (R*U + B*V)/S
        VECR(I,IVEC-1) = (B*U-R*V)/S
37       CONTINUE
C
C THE COMPUTATION OF THE RESIDUALS AND COMPARISON OF THE
C RESIDUALS OF THE TWO SUCCESSIVE STEPS OF THE INVERSE
C ITERATION. IF THE INFINITE NORM OF THE RESIDUAL VECTOR IS
C GREATER THAN THE INFINITE NORM OF THE PREVIOUS RESIDUAL
C VECTOR THE COMPUTED VECTOR OF THE PREVIOUS STEP IS TAKEN
C AS THE COMPUTED APPROXIMATION TO THE EIGENVECTOR.
      B = 0.0
      DO 41 I=1,M
        R = WORK(I)*VECR(I,IVEC-1) - ETA*VECR(I,IVEC)
        U = WORK(I)*VECR(I,IVEC) + ETA*VECR(I,IVEC-1)
        IF(I.EQ.1)GO TO 38
        R = R+SUBDIA(I-1)*VECR(I-1,IVEC-1)
        U = U+SUBDIA(I-1)*VECR(I-1,IVEC)
38       L = I+1
        IF(L.GT.M)GO TO 40
        DO 39 J=L,M
          R = R+H(I,J)*VECR(J,IVEC-1)
          U = U+H(I,J)*VECR(J,IVEC)
39         U = R*R + U*U
40       IF(B.GE.U)GO TO 41
        B = U
41       CONTINUE
        IF(ITER.EQ.1)GO TO 42
        IF(PREVIS.LE.B)GO TO 44
42       DO 43 I=1,N
        WORK1(I) = VECR(I,IVEC)
        WORK2(I) = VECR(I,IVEC-1)
        PREVIS = B
        IF(NS.EQ.1)GO TO 46
        IF(ITER.GT.6)GO TO 47
        ITER = ITER+1
        IF(BOUND.GT.SQRT(S))GO TO 23
        NS = 1
        GO TO 23
C
44       DO 45 I=1,N
        VECR(I,IVEC) = WORK1(I)
        VECR(I,IVEC-1) = WORK2(I)
45       INDIC(IVEC-1) = 2
46       INDIC(IVEC) = 2
47       RETURN
      END

```

ADDED IN PROOF. A small alteration to the program is desirable. The four statements in the subroutine SCALE, page 822, lines 3-6, should be replaced by the four statements below. The alteration is necessary so that the program will also give correct eigenvectors for the case when no convergence of the process of scaling occurs.

```

PRFACT (I) = 1.0
DO 12 J = 1, N
12      A (I, J) = H (I, J)
ENORM = 1.0

```

REMARKS ON ALGORITHM 32 [D1]

MULTINT [R. Don Freeman, Jr., *Comm. ACM* 4 (Feb. 1961), 106]

AND

CERTIFICATION OF ALGORITHM 32 [Henry C. Thacher, Jr., *Comm. ACM* 6 (Feb. 1963), 69]

K. S. KÖLBIG

Data Handling Division, European Organization for Nuclear Research (CERN), 1211 Geneva 23, Switzerland

KEY WORDS AND PHRASES: numerical integration, multi-dimensional integration, Gaussian integration

CR CATEGORIES: 5.16

The real procedure *MULTINT* was corrected according to the certification. It was then compiled on a CDC 3800 computer and tested on the second integral given in the certification. It became apparent that

(i) Equation (2) of the certification should read

$$\int_{-1}^1 \int_{-\sqrt{1-x^2}}^{\sqrt{1-x^2}} \int_{-\sqrt{1-x^2-y^2}}^{\sqrt{1-x^2-y^2}} \frac{dz dy dx}{x^2 + y^2 + (z - k)^2} = \pi \left(2 + \left(\frac{1}{k} - k \right) \log \left| \frac{1+k}{1-k} \right| \right). \quad (2)$$

It should be noted that the right-hand side of equation (2) as printed in the certification does not correspond either to the original limits or to those given above.

(ii) the statement

$$Low := 0;$$

in the real procedure *Low* should be replaced by

$$Low := -Upp(j, x);$$

(iii) the second line of the **for** statement in the real procedure *Upp* should read

$$temp := temp - x[i] \times x[i];$$

After making these corrections, it is possible to obtain results corresponding to a permuted version of the table given in the certification, which should be replaced by the following:

<i>k</i>	$\frac{1}{2}$	2	
true	11.46027375	1.10609686	
<i>s</i>	1	2	1
<i>P</i> = 2	5.454466	9.361670	1.0368787
<i>P</i> = 3	11.838664	12.408983	1.1343568

In addition, since several compilers require specifications, it would be desirable

(i) to change the last specification in the heading of *MULTINT* to read

integer *n, P*;

(ii) to insert the specifications

integer *j*; **array** *x*;

in the heading of the real procedures *Low*, *Upp*, and *Funev*.

Some of these additions were necessary in order to ensure correct results with the compiler used for the tests.



Howard and Tashjian—cont'd from page 818

and the invariance of tensor quantities with respect to coordinate transformations, it is possible to reduce the derivations to routine computer operations. As was shown in [3 and 4], this technique can be applied with equal facility to the problem of deriving the equations of motion of a particle in any curvilinear coordinate system. In fact, any equation or system of equations which can be expressed in tensor form is amenable to automatic formulation by the methods described.

RECEIVED NOVEMBER, 1967; REVISED MAY, 1968

REFERENCES

1. WALTON, JOHN J. Tensor calculations on the computer. *Comm. ACM* 9, 12 (Dec. 1966), 864.
2. ——. Tensor calculations on computers: Appendix. *Comm. ACM* 10, 3 (Mar. 1967), 167-168.
3. HOWARD, JAMES C. Application of computers to the formulation of problems in curvilinear coordinate systems. *NASA TN D-3939*, 1967.
4. ——. Computer formulation of the equations of motion using tensor notation. *Comm. ACM* 10, 9 (Sept. 1967), 543-548.
5. SOKOLNIKOFF, I. S. *Tensor Analysis: Theory and Applications*. Wiley, New York, 1951, p. 324.
6. SCHLICHTING, H. *Boundary Layer Theory*. McGraw-Hill, New York, 1955, p. 49.

Index By Subject To Algorithms, 1960-1968

ALGORITHMS NOT IN CACM HAVE BEEN INCLUDED, WHEN KNOWN TO US.

Key—1st column: A1, B1, B3, etc. is the key to the underlined Modified Share Classification heading each group of algorithms; 2d column: number of the algorithm in *CACM*; 3d column: title of algorithm; 4th column: month, year and page (in parens) in *CACM*, or reference elsewhere. This Index by Subject to Algorithms is cumulative to date (1960-1968) and replaces all previously published versions.

<u>CLASSIFICATION SYSTEM (MODIFIED SHARE)</u>			
A1	REAL ARITHMETIC, NUMBER THEORY	B1	<u>TRIG AND INVERSE TRIG FUNCTIONS</u>
A2	COMPLEX ARITHMETIC	B1 206	ARCCOS(SIN) 9-63(519),2-65(104)
B1	TRIG AND INVERSE TRIG FUNCTIONS	B1 229	ELEMENTARY FCNS. BY CONT. FRACT. 5-64(296)
B2	HYPERBOLIC FUNCTIONS	B1 241	ARCTAN(Z) 9-64(546)
B3	EXPONENTIAL AND LOGARITHMIC FUNCTIONS	B1	ARCSIN(Z) BIT 1962(236)
B4	ROOTS AND POWERS	B1	ARCCOS(Z) BIT 1962(236)
C1	OPERATIONS ON POLYNOMIALS AND POWER SERIES	B1	ARCTAN(Z) BIT 1962(236)
C2	ZEROS OF POLYNOMIALS	B1	SIN FCN. BY CHEBYSHEV EXPANSION NUM. MATH. V4(411),
C5	ZEROS OF ONE OR MORE TRANSCENDENTAL EQUATIONS	B1	V7(194)
C6	SUMMATION OF SERIES, CONVERGENCE ACCELERATION	B1	COS FCN. BY CHEBYSHEV EXPANSION NUM. MATH. V4(411),
D1	QUADRATURE	B1	V7(195)
D2	ORDINARY DIFFERENTIAL EQUATIONS	B1	TAN FCN. BY CHEBYSHEV EXPANSION NUM. MATH. V4(412),
D3	PARTIAL DIFFERENTIAL EQUATIONS	B1	V7(195)
D4	DIFFERENTIATION	B1	ARCSIN BY CHEBYSHEV EXPANSION NUM. MATH. V4(412)
D5	INTEGRAL EQUATIONS	B1	ARCTAN BY CHEBYSHEV EXPANSION NUM. MATH. V4(412)
E1	INTERPOLATION	B2	<u>HYPERBOLIC FUNCTIONS</u>
E2	CURVE AND SURFACE FITTING	B2	SINH(X) BIT 1962(235)
E3	SMOOTHING	B2	COSH(X) BIT 1962(235)
E4	MINIMIZING OR MAXIMIZING A FUNCTION		
F1	MATRIX OPERATIONS, INCLUDING INVERSION	B3	<u>EXPONENTIAL AND LOGARITHMIC FUNCTIONS</u>
F2	EIGENVALUES AND EIGENVECTORS OF MATRICES	B3	46 EXP(Z), Z COMPLEX 4-61(178),6-62(347)
F3	DETERMINANTS	B3	48 LOG(Z), Z COMPLEX 4-61(179),6-62(347),
F4	SIMULTANEOUS LINEAR EQUATIONS	B3	48 7-62(391),8-64(485)
F5	ORTHOGONALIZATION	B3	243 LOGARITHM OF COMPLEX NUMBER 11-64(660),5-65(279)
G1	SIMPLE CALCULATIONS ON STATISTICAL DATA	B3	EXP FCN. BY CHEBYSHEV EXPANSION NUM. MATH. V4(410)
G2	CORRELATION AND REGRESSION ANALYSIS	B3	LOG FCN. BY CHEBYSHEV EXPANSION NUM. MATH. V4(411)
G5	RANDOM NUMBER GENERATORS		
G6	PERMUTATIONS AND COMBINATIONS	B4	<u>ROOTS AND POWERS</u>
G7	SUBSET GENERATORS	B4	53 ROOTS OF COMPLEX NUMBERS 4-61(180),7-61(322)
H	OPERATIONS RESEARCH, GRAPH STRUCTURES	B4	106 POWERS OF COMPLEX NUMBER 7-62(388),11-62(557)
I5	INPUT - COMPOSITE	B4	190 POWERS OF COMPLEX NUMBERS 7-63(388)
J6	PLOTTING		
K2	RELOCATION	C1	<u>OPERATIONS ON POLYNOMIALS AND POWER SERIES</u>
L2	COMPILING	C1	29 POLYNOMIAL SHIFTER 11-60(604)
M1	SORTING	C1	131 DIVIDE POWER SERIES 11-62(551)
M2	DATA CONVERSION AND SCALING	C1	134 EXPONENTIAL POWER SERIES 11-62(553),7-63(390)
O2	SIMULATION OF COMPUTING STRUCTURE	C1	158 EXPONENTIAL POWER SERIES 3-63(104),7-63(390),
R2	SYMBOL MANIPULATION	C1	158 9-63(522)
S	APPROXIMATION OF SPECIAL FUNCTIONS...	C1	193 REVERT POWER SERIES 7-63(388),12-63(745)
S	FUNCTIONS ARE CLASSIFIED SO1 TO S22, FOLLOWING	C1	273 SOLN. OF EQNS. BY REVERSION 1-66(11)
S	FLETCHER-MILLER-ROSENHEAD, INDEX OF MATH. TABLES	C1	305 SYMMETRIC POLYNOMIALS 7-67(450),4-68(272)
Z	ALL OTHERS	C1	337 POLY. AND DERIV. BY HORNER SCHEME 9-68(633)
		C1	CALCULATION OF GRAM POLYS. COMP. J. V9(323)
		C1	EVALUATION OF CONTINUED FRACTNS CHIFFRES V9(327)
A1	<u>REAL ARITHMETIC, NUMBER THEORY</u>		
A1	7 EUCLIDEAN ALGORITHM 4-60(240)		
A1	35 SIEVE OF ERATOSTHENES 3-61(151),4-62(209),		
A1	35 8-62(438),9-67(570)		
A1	61 RANGE ARITHMETIC 7-61(319)		
A1	66 AUGMENTATION 8-61(339),11-61(498)		
A1	72 COMPOSITIONS 11-61(498),8-62(439)		
A1	93 GENERALIZED ARITHMETIC 6-62(344),10-62(514)		
A1	95 PARTITIONS 6-62(344)		
A1	99 JACOBI SYMBOL 6-62(345),11-62(557)		
A1	114 PARTITIONS 8-62(434)		
A1	139 DIOPHANTINE EQUATION 11-62(556),3-65(170)		
A1	223 PRIME TWINS 4-64(243)		
A1	237 GREATEST COMMON DIVISOR 8-64(481),12-64(702)		
A1	262 RESTRICTED PARTITIONS OF N 8-65(493)		
A1	263 PARTITION GENERATOR 8-65(493)		
A1	264 MAP OF PARTITIONS INTO INTEGERS 8-65(493)		
A1	337 SYMMETRIC GROUP CHARACTERS 7-67(451),1-68(14)		
A1	310 PRIME NUMBER GENERATOR 1 9-67(569),9-67(570)		
A1	311 PRIME NUMBER GENERATOR 2 9-67(570),9-67(570)		
A1	313 MULTI-DIMENSION PARTITION GEN. 10-67(666)		
A1	SUM OF FACTORS OF N (SUMFAC) COMP. J. V9(416)		
A2	<u>COMPLEX ARITHMETIC</u>		
A2	116 COMPLEX DIVIDE 8-62(435)		
A2	186 COMPLEX ARITHMETIC 7-63(386)		
A2	312 COMPLEX ABS, SQRT 10-67(665)		
A2	COMPLEX ARITHMETIC BIT 1962(233)		
A2	ADD, SUB, MULT, DIVO-COMPLEX COMP. J. V10(112),		
A2	V10(208)		
		C2	<u>ZEROS OF POLYNOMIALS</u>
		C2	3 BAIRSTOW 2-60(74),6-60(354),
		C2	3 2-61(105),3-61(153),4-61(181)
		C2	30 BAIRSTOW-NEWTON 12-60(643),5-61(238),
		C2	30 1-62(50), 5-67(293)
		C2	59 RESULTANT METHOD 5-61(236)
		C2	75 RATIONAL ROOTS-INTEG. COEFF. 1-62(48),7-62(392),
		C2	75 8-62(439)
		C2	78 RATIONAL ROOTS-INTEG. COEFF. 2-62(97),3-62(168),
		C2	78 8-62(440)
		C2	105 NEWTON-MAEHLI 7-62(387),7-63(389)
		C2	174 BOUNDS ON ZEROS 6-63(311)
		C2	256 MODIFIED GRAEFFE METHOD 6-65(379),9-66(687)
		C2	283 REAL SIMPLE ROOTS 4-66(273)
		C2	326 ROOTS OF LOW ORDER POLY EQNS 4-68(269)
		C2	340 RT-SQUARING AND RESULTANT METH. 11-68(779)
		C2	ZEROS IN THE RIGHT HALF PLANE ZH. VYCH. MAT. MAT. FIZ. -
		C2	1963(364)
		C2	LEHMERS METHOD BIT 1964(255)
		C2	BAIRSTOW COMP. J. V10(207)
		C5	<u>ZEROS OF ONE OR MORE TRANSCENDENTAL EQUATIONS</u>
		C5	ZEROS BY INTERP. CR BISECTION BIT 1963(205)
		C5	2 REGULA FALSI 2-60(74),6-60(354),
		C5	2 8-60(475),3-61(153)

C5	4	BISECTION	3-60(174),3-61(153)	E1	168	INTERPOLATION-DIVIDED DIFFCS.	4-63(165),9-63(523)
C5	15	REGULA FALSI	8-60(475),11-60(602),	E1	169	INTERPOLATION-DIVIDED DIFFCS.	4-63(165),9-63(523)
C5	15	3-61(153)		E1	187	DIFFCS.AND DERIVS.-RECURSIVE	7-63(387)
C5	25	REAL ZEROS	11-60(602),3-61(153),	E1	210	LAGRANGE INTERPOLATION	10-63(616)
C5	25	3-61(154)		E1	211	HERMITE INTERPOLATION	10-63(617),10-63(619)
C5	26	REGULA FALSI	11-60(603),3-61(153)	E1	264	INTERPOLATION IN A TABLE	10-65(602)
C5	196	MULLERS METHOD	8-63(442),1-68(12)	E1		AITKEN INTERPOLATION	COMP.J.V9(211)
C5	314	N FUNCTIONAL EQNS.IN N UNKNOWNS	11-67(726)	E1		NEVILLE INTERPOLATION	COMP.J.V9(212)
C5	315	DAMPED TAYLR SERIES-NONLIN.SYS.	11-67(726)				
C5	316	NON-LINEAR SYSTEM	11-67(728)				
C6	<u>SUMMATION OF SERIES, CONVERGENCE ACCELERATION</u>						
C6	8	EULER SUM	5-60(311),11-63(663)	E2	28	LEAST SQUARES BY ORTHOG. POLYN.	11-60(604),
C6	128	FOURIER SERIES SUMMATION	10-62(513),7-64(421)	E2	28	12-61(544),5-67(293)	
C6	157	FOURIER SERIES APPROXIMATION	3-63(103),9-63(521),	E2	37	ECONOMIZATION	3-61(151),8-62(438),
C6	157	10-63(618)		E2	37	8-63(445)	
C6	215	EPSILON ALGORITHM	11-63(662),5-64(297)	E2	38	ECONOMIZATION	3-61(151),8-63(445)
C6	255	FOURIER COEFFICIENTS	5-65(279)	E2	74	LEAST SQUARES WITH CONSTRAINTS	1-62(47),6-63(316)
C6	277	CHEBYSHEV SERIES COEFFICIENTS	2-66(86)	E2	91	CHEBYSHEV FIT	5-62(281),4-63(167),
C6	320	HARMONIC ANAL.-SYM DISTR DATA	2-68(114)	E2	91	5-64(296),12-67(803)	
C6	338	FAST FOURIER TRANSFORM	11-68(773)	E2	164	SURFACE FIT	4-63(162),8-63(450)
C6	339	FAST FOURIER WITH ARB. FACTORS	11-68(776)	E2	176	SURFACE FIT	6-63(313)
C6		FIND LIMIT OF SEQUENCE	BIT 1961(64)	E2	177	LEAST SQUARES WITH CONSTRAINTS	6-63(313),7-63(390)
C6		SUM FOURIER SERIES	COMP.J.V6(248)	E2	275	EXPONENTIAL CURVE FIT	2-66(85)
C6		EPSILON ALGORITHM	BIT 1962(240)	E2	276	CONSTRAINED EXPONENTIAL FIT	2-66(85)
C6		EPSILON ALGORITHM	NUM.MATH.V6(22)	E2	295	EXPONENTIAL CURVE FIT	2-67(87)
C6		EPSILON ALG.-CONTINUED FRACTNS	CHIFFRES V9(327)	E2	296	LEAST SQ.FIT-ORTHOG.POLYS.	2-67(87),6-67(377)
C6		COMPLEX FOURIER ANALYSIS	COMP.J.V10(414)	E2	318	CHEBYSHEV CURVE FIT(REVISED)	12-67(801)
C6		V11(115)		E2		CONTINUED FRACTION EXPANSION	BIT 1962(245)
				E2		RATIONAL CHEBYSHEV APPROX.	JACM-1964(66)
				E2		L1 APPROX. ON A DISCRETE SET	NUM.MATH.V8(299)
				E2		CHEBYSHEV APPROX.-DISCRETE SET	NUM.MATH.V8(303)
				E2		REMES ALGORITHM-GENERALIZED	NUM.MATH.V10(203)
				E2		EXPONENTIAL FIT	COMP.J.V11(114)
				E2		RATIONAL CHEBYSHEV APPROX	NUM.MATH.V10(291)
D1	<u>QUADRATURE</u>						
D1	1	QUADRATURE	2-60(74)	E3	<u>SMOOTHING</u>		
D1	32	MULTIPLE INTEGRAL	2-61(106),2-63(69),	E3	188	SMOOTHING	7-63(387)
D1	32	12-68(826)		E3	189	SMOOTHING	7-63(387)
D1	60	ROMBERG METHOD	6-61(255),3-62(168),	E3	216	SMOOTHING	11-63(663)
D1	60	5-62(281),7-64(420)		E3		SMOOTHING BY SPLINE FCNS	NUM.MATH.V10(182)
D1	84	SIMPSONS RULE	4-62(208),7-62(392),				
D1	84	8-62(440),11-62(557)		E4	<u>MINIMIZING OR MAXIMIZING A FUNCTION</u>		
D1	98	COMPLEX LINE INTEGRAL	6-62(345)	E4	129	MINIMIZE FUNCT. OF N VARIABLES	11-62(556) 9-63(521)
D1	133	SIMPSONS RULE	6-62(347)	E4	178	MINIMIZE FUNCT. OF N VARIABLES	6-63(313),9-66(684),
D1	125	GAUSSIAN COEFFICIENTS	10-62(510)	E4	178	7-68(498)	
D1	145	ADAPTIVE SIMPSON	12-62(604),4-63(167),	E4	203	MINIMIZE FUNCT.OF N VARIABLES	9-63(517),10-64(585),
D1	145	3-65(171)		E4	203	3-65(171)	
D1	146	MULTIPLE INTEGRAL	12-62(604),5-64(296)	E4	204	MINIMIZE FUNCT.OF N VARIABLES	9-63(519)
D1	192	ADAPTIVE SIMPSON	6-63(315),4-64(244)	E4	205	MINIMIZE FUNCT.OF N VARIABLES	9-63(519),3-65(171)
D1	198	ADAPTIVE,MULTIPLE INTEGRAL	8-63(443)	E4	251	FUNCTION MINIMIZATION	3-65(169),9-66(686)
D1	233	MULTIPLE INTEG.-SIMPSONS RULE	6-64(348)	E4	315	MINIMIZING SUM OF SQUARES	11-67(726)
D1	257	HAVIE INTEGRATOR	6-65(381),11-66(795),	E4		MINIMIZING FCN.-CONJ.GRAD.	COMP.J.V7(151)
D1	257	12-66(871)		E4		FIBONACCI SEARCH	COMP.GULL.V8(147),
D1	279	CHEBYSHEV QUADRATURE	4-66(270),6-66(434),	E4		V9(15),COMP.J.V9(414),V9(416)	
D1	279	5-67(294),10-67(666)		E4		MIN.OF UNIMODAL FCN.OF 1 VAR.	COMP.BULL.V9(104),
D1	280	GREGORY QUADRATURE COEFFICIENTS	4-66(271)	E4		COMP.J.V9(414)	
D1	291	ROMBERG QUADRATURE COEFFICIENTS	4-66(271),3-67(188)	E4		MINIMIZING ARG.OF UNIMODAL FCN.	COMP.J.V9(415)
D1	303	ADAPTIVE QUAD.-RANDOM PANEL SIZE	6-67(373)	E4		MIN.OF A UNIMODAL FCN.OF 1 VAR.	COMP.J.V9(415)
D1	331	GAUSSIAN QUADRATURE FORMULAS	6-68(432)				
D1		ADAPTIVE SIMPSONS RULE	BIT 1961(290)				
D1		MONTE CARLO QUADRATURE	COMP.J.V6(281)				
D1		ROMBERG METHOD	NUM.MATH.V6(15)				
D1		ROMBERG METHOD	COLL.ANAL.NUM.(1961)				
D1		ROMBERG METHOD	BIT 1964(58)				
D1		QUADRATURE BY EXTRAPOLATION	NUM.MATH.V9(274)				
D2	<u>ORDINARY DIFFERENTIAL EQUATIONS</u>						
D2	9	RUNGE-KUTTA	5-60(312),4-66(273)	F1	<u>MATRIX OPERATIONS, INCLUDING INVERSION</u>		
D2	194	ZEROS OF O.D.E. SYSTEM	8-63(441)	F1	42	INVERSION	4-61(176),11-61(498),
D2	218	KUTTA-MERSON	12-63(737),10-64(585),	F1	42	1-63(38),8-63(445)	
D2	218	4-66(273)		F1	50	INVERSE OF HILBERT MATRIX	4-61(179),1-62(50),
D2		EXTRAPOLATION METHOD	NUM.MATH.V8(10)	F1	50	1-63(38)	
				F1	51	INVERSE OF PERTURBED MATRIX	4-61(180),7-62(391)
				F1	52	INVERSE OF TEST MATRIX	4-61(180),8-61(339),
				F1	52	11-61(498),8-62(438),1-63(39),8-63(446)	
				F1	58	INVERSION-GAUSSIAN ELIMINATION	5-61(236),6-62(347),
				F1	58	8-62(438),12-62(606)	
				F1	66	INVERSION-SORT METHOD	7-61(322),1-62(50),
				F1	66	6-62(348)	
				F1	67	CRAM MATRIX	7-61(322),6-62(348)
				F1	120	INVERSION-GAUSSIAN ELIMINATION	8-62(437),1-63(40),
				F1	120	8-63(445)	
				F1	140	INVERSION	11-62(556),8-63(448)
				F1	150	INVERSE OF SYMMETRIC MATRIX	2-63(67),7-63(390),
				F1	150	3-64(148)	
				F1	166	MONTE CARLO INVERSE	4-63(164),9-63(523)
				F1	197	MATRIX DIVISION	8-63(443),3-64(148)
				F1	230	MATRIX PERMUTATION	6-64(347)
				F1	231	INVERSION-GAUSS.ELIM.-COMP.PIV.	6-64(347),4-65(220)
				F1	274	HILBERT DERIVED TEST MATRIX	1-66(11)
				F1	287	INTEGER MATRIX TRIANGULATION	7-66(513)
				F1	298	SQ.RT.OF A POS.DEFINITE MATRIX	3-67(182)
				F1	319	TRIANG FCTRS OF MODIFIED MATRIX	1-68(12)
				F1	325	ADJUST INVERSE OF SYM MATRIX	2-68(118)
				F1		INVERSE-CONFL. VANDERMONDE MTX	NUM.MATH.V5(429)
				F1		EQUIVALENCE OF MATRICES	ICC BULL.-1964(62)
				F1		SYMM.DECOMP.OF POS.DEF.BAND MTX	NUM.MATH.V7(357)
				F1		SYMM.DECOMP.OF POS.DEF.MTX.	NUM.MATH.V7(368)
				F1		SYMM.DECOMP.OF POS.DEF.BAND MTX	COMPUTING V1(77)
				F1		INVERSION-SYMM.POS.DEF.MTX.	COMP.J.V9(321)
				F1		SMITH NORMAL FORM	BIT 1967(163)
				F1		PERMUTATIONS OF ROWS AND COLS.	COMP.J.V10(206)
				F1		ADJUST INVERSE OF SYM MATRIX	COMPUTING V3(76)
				F1		HOUSEHOLDER TRIANG OF SYM MTRX	NUM.MATH.V11(184)
E1	<u>INTERPOLATION</u>						
E1	18	RATIONAL INTERP.-CONT.FRACT.	9-60(508),8-62(437)				
E1	70	AITKEN INTERPOLATION	11-61(497),7-62(392)				
E1	77	INTERPOLATION,DIFFN.,INTEGRN.	2-62(96),6-62(348),				
E1	77	8-63(446),11-63(663)					
E1	167	CONFLUENT DIVIDED DIFFERENCES	4-63(164),9-63(523)				

<u>EIGENVALUES AND EIGENVECTORS OF MATRICES</u>		G5	133	RANDOM FLAT	11-62(553),12-62(606),
F2	85 JACOBI METHOD	G5	133	3-63(105),4-63(167)	
F2	85 3-63(447)	G5	200	RANDOM NORMAL	8-63(444),9-65(556)
F2	104 REDUCTION-BAND TO TRIDIAGONAL	G5	247	QUASI-RANDOM POINT SEQUENCE	12-64(701)
F2	122 GIVENS TRIDIAGONAL REDUCTION	G5	266	PSEUDO-RANDOM NUMBERS	10-65(605),9-66(687)
F2	183 REDUCTION-BAND TO TRIDIAGONAL	G5	267	RANDOM NORMAL DEVIATES	10-65(606)
F2	253 SYMMETRIC QR-EIGENVALUES	G5	294	UNIFORM RANDOM	1-67(40)
F2	254 SYMMETRIC QR-EIGENVALUES, EIGENVECTORS	G5	334	NORMAL RANDOM	7-68(498)
F2	270 EIGENVECTORS BY GAUSSIAN ELIM.	G5	342	POISSON RANDOM NUMBERS	12-68(819)
F2	297 SYM. SYS. (A-LAMBDA=B) X, EIGENVALUES-VECS.	G5		RANDOM SAMPLES, VARIOUS DISTRIBUTIONS	COMP. J. V6(279)
F2	343 EIGENVALUES-VECS. OF REAL GEN. MTRX	G5		RANDOM UNIFORM	COMP. BULL. V9(105)
F2	HOUSEHOLDERS METHOD				
F2	EIGENVALUES OF TRIDIAG. MATRIX				
F2	EIGENVECTORS OF TRIDIAG. MATRIX				
F2	LR TRANSFORMATION METHOD				
F2	EIGENVALUES-LAGUERRES METHOD				
F2	APPL. MATH. STAT. REP. NONR 225(37)NO. 21				
F2	HOUSEHOLDERS METHOD				
F2	APPL. MATH. STAT. REP. NONR 225(37)NO. 18				
F2	EIGENVALUES BY QR-ALGORITHM				
F2	TRIDIAGONAL SIMIL. BY ELIM.				
F2	EIGENVALUES-LAGUERRES METHOD				
F2	1966(437)				
F2	SYMMETRIC-BISECTION, INV. ITN.				
F2	HOUSEHOLDER RED. - COMPLEX MAT.				
F2	SYMM. MAT. - LLT AND STURM SEQ.				
F2	JACOBI METHOD				
F2	EIGENVECTORS OF BAND MATRICES				
F2	EIGENVALUES OF SYMM. TRIDIAG. MATRIX				
F2	EIGENVALUES-EIGENVECTORS OF REAL MTRX				
F2	SYM EIPROBLEM A. X=LAMBDA. B. X				
F2	RATIONAL QR FOR SYM TRIDIAG				
F2	EIGENVALUES-REAL SYM MTRX-DBL QR STP				
F3	<u>DETERMINANTS</u>				
F3	41 DETERMINANT EVALUATION				
F3	41 3-64(144),9-66(686)				
F3	159 DETERMINANT EVALUATION				
F3	170 DETERMINANT-POLYNOMIAL ELEMENTS				
F3	170 7-64(421)				
F3	224 EVALUATION OF DETERMINANT				
F3	269 DETERMINANT BY GAUSSIAN ELIM.				
F4	<u>SIMULTANEOUS LINEAR EQUATIONS</u>				
F4	16 CROUT WITH PIVOTING				
F4	16 3-61(154)				
F4	17 SOLVE TRIDIAGONAL MATRIX				
F4	24 SOLVE TRIDIAGONAL MATRIX				
F4	43 CROUT WITH PIVOTING				
F4	43 8-63(445)				
F4	92 SIMULT. EQNS. - ITERATIVE SOLN.				
F4	107 GAUSSIAN ELIMINATION				
F4	107 8-63(445)				
F4	126 GAUSSIAN ELIMINATION				
F4	135 CROUT WITH EQUILIBRATION				
F4	135 7-64(421),2-65(104)				
F4	195 BAND SOLVE				
F4	220 GAUSS-SEIDEL				
F4	238 CONJUGATE GRADIENT METHOD				
F4	288 LINEAR DIOPHANTINE EQUATIONS				
F4	290 EXACT SOLUTION OF LINEAR EQNS.				
F4	328 CHEBY SOLN-OVERDETERMINED LINEAR SYS				
F4	GAUSSIAN ELIMINATION				
F4	BIT 1963(60)				
F4	LINEAR SYSTEM WITH BAND MATRIX				
F4	CONJUGATE GRADIENT METHOD				
F4	LEAST SQUARES SOLUTION				
F4	GAUSSIAN ELIMINATION				
F4	ELIM. WITH WEIGHTED ROW COMB.				
F4	ITER. REFIN. - SOLN. OF POS. DEF. MTX				
F4	REAL AND COMPLEX LINEAR SYSTEM				
F4	SYMM. AND UNSYMM. BAND EQUATIONS				
F4	IT REFINEMENT-LEAST SQR SOLN				
F4	SOLUTION WITH REL ERR ESTIMATE				
F5	<u>ORTHOGONALIZATION</u>				
F5	127 ORTHONORMALIZATION				
F5	SCHMIDT ORTHONORMALIZATION				
G1	<u>SIMPLE CALCULATIONS ON STATISTICAL DATA</u>				
G1	208 DISCRETE CONVOLUTION				
G1	212 DETERMINE DISTRIB. FCN. FROM DATA				
G1	289 CONFIDENCE INTERVAL FOR A RATIO				
G1	330 FACTORIAL ANALYSIS OF VARIANCE				
G1	TAIL AREA PROB. FOR 2X2 TABLE				
G1	COMP. J. V9(212), V9(436)				
G2	<u>CORRELATION AND REGRESSION ANALYSIS</u>				
G2	39 CORRELATION COEFFICIENTS				
G2	142 TRIANGULAR REGRESSION				
G5	<u>RANDOM NUMBER GENERATORS</u>				
G5	121 RANDOM NORMAL				
G6	<u>PERMUTATIONS AND COMBINATIONS</u>				
G6	71 PERMUTATIONS				
G6	71 8-62(439)				
G6	86 PERMUTATIONS				
G6	86 8-62(440)				
G6	87 PERMUTATION GENERATOR				
G6	87 10-62(514),7-67(452)				
G6	94 COMBINATIONS				
G6	94 12-62(606)				
G6	132 PERMUTATIONS IN LEXIC. ORDER				
G6	102 10-62(514), 7-67(452)				
G6	130 PERMUTE				
G6	202 PERMUTATIONS				
G6	202 7-67(452)				
G6	235 RANDOM PERMUTATION				
G6	242 PERMUTATIONS WITH REPETITIONS				
G6	250 INVERSE PERMUTATION				
G6	306 PERMUTATIONS WITH REPETITIONS				
G6	308 PERMUT. IN PSEUDOLEXIC. ORDER				
G6	317 PERMUTATION				
G6	323 PERMUTATIONS IN LEXIC. ORDER				
G6	329 DISTR. OF INDISTINGUISHABLE OBJ				
G6	ALL PERMUTATIONS OF N OBJECTS				
G6	PERMUTNS OF VECTOR-LEXIC. ORDER				
G6	PERMUTN. OF VECTOR				
G6	FAST PERMUTN. OF VECTOR				
G7	<u>SUBSET GENERATORS</u>				
G7	81 SUBSEQUENCES				
G7	82 SUBSEQUENCES				
H	<u>OPERATIONS RESEARCH, GRAPH STRUCTURES</u>				
H	27 ASSIGNMENT PROBLEM				
H	27 12-63(739)				
H	40 CRITICAL PATH SCHEDULING				
H	40 10-62(513),6-64(349)				
H	69 CHAIN TRACING				
H	83 CLASSIFICATIONS				
H	96 ANCESTOR				
H	97 SHORTEST PATH				
H	119 PERT NETWORK				
H	141 FIND PATH				
H	153 INTEGER PROGRAMMING				
H	217 MIN. EXCESS COST CURVE				
H	219 TOPLOGICAL ORDERING				
H	248 NETFLOW				
H	248 9-68(633)				
H	258 TRANSPORT				
H	258 7-67(453)				
H	263 INTEGER PROGRAMMING-GOMORY I				
H	285 MUTUAL PRIMAL-DUAL METHOD				
H	286 EXAMINATION SCHEDULING				
H	293 TRANSPORTATION PROBLEM				
H	293 4-68(271)				
H	324 MAXFLOW				
H	332 MINIT ALGORITHM FOR LIN PROG				
H	336 NETFLOW				
H	341 LINEAR PGMS. IN 0-1 VARIABLES				
H	MINIMAL SPANNING TREE				
H	V8(109), V8(147), V9(19)				
H	SIMPLEX METHOD				
H	1966(82)				
H	PROCESSING EVENT NETWORK				
H	SHORTEST PATH-START TO END				
H	SHORTEST PATH-START TO ANY				
H	NODES ON SHORTEST PATH				
I5	<u>INPUT - COMPOSITE</u>				
I5	239 FREE-FIELD READ				
I5	249 OUTREAL M				
I5	335 BASIC I/O PROCEDURES				
I5	OPTICAL SCANNING OF NUMBERS				
I5	1962(236)				
J6	<u>PLOTTING</u>				
J6	162 XY PLOTTER				
J6	162 8-64(482)				
J6	278 GRAPH PLOTTER				
K2	<u>RELOCATION</u>				
K2	173 TRANSFER ARRAY VALUES				

K2	284	INTERCHANGE 2 BLOCKS OF DATA	5-66(326)	S15	209	8-64(482), 6-67(377)	
K2	302	TRANSPOSE VECTOR STORED ARRAY	5-67(292)	S15	226	NORMAL DISTRIBUTION FUNCTION	5-64(295),6-67(377)
				S15	272	NORMAL DISTRIBUTION FUNCTION	12-65(789),6-67(377),
				S15	272	7-68(493)	
L2		COMPILING		S15	299	CHI-SQUARED INTEGRAL	4-67(243),4-68(270)
L2	265	FIND PRECEDENCE FUNCTIONS	10-65(604)	S15	304	NORMAL CURVE INTEGRAL	6-67(374),6-67(377),
L2		EVALUATION OF FCNAL EXPRESSION	BIT 1965(133)	S15	304	4-68(271)	
				S15		ERF(X) BY CHEBYSHEV EXPANSION	NUM.MATH.V4(414)
				S15		DERIV.OF BOYS ERROR FCN.	COMP.BULL.V9(165)
				S15		COMPL.ERROR INT.-COMPLEX ARG.	BIT 1965(290)
				S15		NORMAL DISTRIBUTION CURVE	COMP.J.V9(322),
				S15		V10(113)	
M1		SORTING		S16	13	LEGENDRE POLYNOMIAL	6-60(353),2-61(105),
M1	23	SORT	11-60(601),5-61(238)	S16	13	4-61(181)	
M1	63	SORT	7-61(321),8-62(439),	S16	47	ASSOCIATED LEGENDRE FUNCTION	4-61(178),8-63(446)
M1	63	8-63(446)		S16	62	ASSOCIATED LEGENDRE FUNCTION	7-61(320),12-61(544)
M1	64	SORT	7-61(321),8-62(439),	S16	259	LEGENDRE FUNCTION	8-65(488)
M1	64	8-63(446)		S17	21	BESSEL FUNCTION	11-60(600),4-65(219)
M1	65	SORT	7-61(321),8-62(439),	S17	22	RICCATI-BESSEL FUNCTION	11-60(600)
M1	65	8-63(446)		S17	44	BESSEL FUNCTION	4-61(177)
M1	76	SORT	1-62(48),6-62(348)	S17	49	SPHERICAL NEUMANN FUNCTION	4-61(179)
M1	113	TREESORT	8-62(434)	S17	124	HANKEL FUNCTION	9-62(483),12-65(790)
M1	143	TREESORT	12-62(604)	S17	163	HANKEL FUNCTION	4-63(161),9-63(522)
M1	144	TREESORT	12-62(604)	S17	236	BESSCL FCNS OF FIRST KIND	8-64(479),2-65(135)
M1	151	LOCATE IN A LIST	2-63(68)	S18	5	BESSEL FUNCTION	4-60(240)
M1	175	SHUTTLE SORT	6-63(312),10-63(619),	S18	6	BESSEL FUNCTION	4-60(240)
M1	175	12-63(739),5-64(296)		S18	214	BESSEL FUNCTION	11-63(662),6-64(349)
M1	201	SHELL SORT	8-63(445),6-64(349)	S18	228	Q-BESSEL FUNCTION	5-64(295)
M1	207	STRING SORT	10-63(615),10-64(585)	S19	57	BERBEI FUNCTION	4-61(181),7-62(392),
M1	232	HEAPSORT	6-64(347)	S19	57	8-62(438)	
M1	245	TREESORT 3	12-64(701),7-65(445)	S20	88	FRESNEL INTEGRALS	5-62(280),10-63(618)
M1	271	QUICKERSORT	11-65(669),5-66(354)	S20	89	FRESNEL SINE INTEGRAL	5-62(280),10-63(618)
M1		SEARCH IN A LIST	J.ACM-1962(23)	S20	90	FRESNEL COSINE INTEGRAL	5-62(281),10-63(618)
M1		INSERTION IN A LIST	J.ACM-1962(23)	S20	213	FRESNEL INTEGRALS	10-63(617),11-64(661)
M1		DELETION FROM A LIST	J.ACM-1962(24)	S20	244	FRESNEL INTEGRALS	11-64(660)
M1		SORTING WITH MINIMUM STORAGE	J.ACM-1962(27)	S20	301	AIRY FUNCTIONS	5-67(291),7-67(453)
M1		SORTING OF INTEGERS	COMP.BULL.V9(63)	S20		WEBER FUNCTION	BIT 1962(239)
M1		SORT BY RANKING ELEMENTS	COMP.J.V10(308)	S20		COMPLEMENTARY FRESNEL INTEGRAL	BIT 1962(192)
M1		ORDER SUBSCRIPTS BY ELEMNT SIZE	COMP.J.V10(309)	S20		FRESNEL INTEGRALS S(X),C(X)	NUM.MATH.V9(382)
M1		SORT ON PERMUTN OF SUBSCRIPTS	COMP.J.V10(310)	S21	55	ELLIPTIC INTEGRAL-FIRST KIND	4-61(180),4-63(166)
				S21	56	ELLIPTIC INTEGRAL-SECOND KIND	4-61(180),1-66(12)
M2		DATA CONVERSION AND SCALING		S21	73	INCOMPLETE ELLIPTIC INTEGRAL	12-61(543),12-61(544),
M2		DATA PROCESSING-VECTORCARDIOGRM CACM	2-62(121)	S21	73	10-62(514),2-63(69),4-63(167)	
				S21	149	ELLIPTIC INTEGRAL	12-62(605),4-63(166)
O2		SIMULATION OF COMPUTING STRUCTURE		S21	165	ELLIPTIC INTEGRAL	4-63(163)
O2	100	PROCESSING OF CHAIN-LINKED LIST	6-62(346)	S21		COMPLETE ELL.INT.-FIRST KIND(K)	NUM.MATH.V5(296)
O2	131	PROCESSING OF CHAIN-LINKED LIST	6-62(346)	S21		COMPLETE ELL.INT.-SECOND KIND(E)	NUM.MATH.V5(297)
O2	137	NESTED FOR STATEMENT	11-62(555)	S21		COMPLETE ELL.INT.(B)	NUM.MATH.V5(297)
O2	138	NESTED FOR STATEMENT	11-62(555)	S21		INCOMPL.ELL.INT.-FIRST KIND(K)	NUM.MATH.V5(297)
O2		EVALUATION OF FCNAL EXPRESSION	BIT 1965(137)	S21		INCOMPL.ELL.INT.-SECOND KIND(E)	NUM.MATH.V5(298)
				S21		INCOMPL.ELL.INT.(B)	NUM.MATH.V5(299)
R2		SYMBCL MANIPULATION		S21		JACOBIAN ELLIPTIC SIN FCN.(SN)	NUM.MATH.V5(299)
R2	268	ALGOL 60 REF.LANG.EDITOR	11-65(667)	S21		JACOBIAN ELLIPTIC COS FCN.(CN)	NUM.MATH.V5(300)
R2		BASIC LIST PROCESSING	BIT 1966(166)	S21		JACOBIAN ELLIPTIC FCN.(DN)	NUM.MATH.V5(301)
R2		SIMPLIFYING BOOLEAN EXPRESSIONS	BIT 1966(260)	S21		ELLIPTIC INTEGRALS-KINDS 1,2,3	NUM.MATH.V7(85),
				S21		V7(353)	
				S21		JACOBIAN ELLIPTIC FUNCTIONS	NUM.MATH.V7(89)
S		APPROXIMATION OF SPECIAL FUNCTIONS...		S22	10	CHEBYSHEV POLYNOMIAL	6-60(353),4-61(181)
S		FUNCTIONS ARE CLASSIFIED S01 TO S22, FOLLOWING		S22	12	LAGUERRE POLYNOMIAL	6-60(353)
S		FLETCHER-MILLER-ROSENHEAD, INDEX OF MATH. TABLES		S22	36	CHEBYSHEV POLYNOMIAL	3-61(151)
S03	19	BINOMIAL COEFFICIENTS	10-60(540),6-62(347),	S22	110	PHYSICS INTEGRALS	7-62(389),7-62(393)
S03	19	8-62(438)		S22	111	PHYSICS INTEGRALS	7-62(390)
S03	33	FACTORIAL N	2-61(106)	S22	132	PHYSICS INTEGRALS	11-62(551)
S07		POLAR TRANSF. BY CHEBYSHEV EXP.	NUM.MATH.V4(413)	S22	184	ERLANG PROBABILITY FUNCTION	7-63(386)
S13	14	COMPLEX EXPONENTIAL INTEGRAL	7-60(406)	S22	191	HYPERGEOMETRIC FCN.(COMPLEX)	7-63(388),4-64(244)
S13	20	REAL EXPONENTIAL INTEGRAL	10-60(540),2-61(105),	S22	192	CONFLUENT HYPERG.FCN.(COMPLEX)	7-63(388),4-64(244)
S13	20	4-61(182)		S22	227	CHEBYSHEV POLYNOMIAL COEFF.	5-64(295)
S13	108	EXPONENTIAL INTEGRAL	7-62(388),7-62(393)	S22	282	DERIVATIVES OF EXP(X OR IX)/X	4-66(272)
S13	109	EXPONENTIAL INTEGRAL	7-62(388),7-62(393)	S22	292	REGULAR COULOMB WAVE FCNS.	11-66(793)
S13		EXPONENTIAL INTEGRAL EXPANSION	CHIFFRES-V6(187)	S22	300	COULOMB WAVE FUNCTIONS	4-67(244)
S13		EI(X) BY CHEBYSHEV EXPANSION	NUM.MATH.V4(413)	S22	332	JACOBI POLYNOMIALS	6-68(436)
S13		SIN INTEGRAL SI(X)	NUM.MATH.V9(381)	S22	327	DILOGARITHM	4-68(270)
S13		COS INTEGRAL CI(X)	NUM.MATH.V9(382)	S22		CONFLUENT HYPERG.FCN.(COMPLEX)	BIT 1962(237)
S14	31	GAMMA FUNCTION	2-61(105),12-62(605)	S22		FERMI FUNCTION	BIT 1963(141)
S14	34	GAMMA FUNCTION	2-61(106),7-62(391),	S22		RIEMANN ZETA FUNCTION	BIT 1965(141)
S14	34	9-66(685)		S23	234	POISSON-CHARLIER POLYNOMIALS	7-64(420),2-65(105)
S14	54	GAMMA FUNCTION	4-61(180),9-66(685)				
S14	80	GAMMA FUNCTION	3-62(166),9-66(685)				
S14	147	DERIVATIVE OF GAMMA FUNCTION	12-62(605),4-63(168)	Z		ALL OTHERS	
S14	179	BETA RATIO	6-63(314),6-67(375)	Z	45	INTEREST REFINEMENT	4-61(178),9-63(520)
S14	221	GAMMA FUNCTION	3-64(143),10-64(586)	Z	112	POINT INSIDE POLYGON	8-62(434),12-62(606)
S14	221	9-66(685)		Z	117	MAGIC SQUARE	8-62(435),8-62(440),
S14	222	INCOMPLETE BETA FCN.RATIOS	3-64(143),4-64(244)	Z	117	1-63(39),3-63(105)	
S14	225	GAMMA FCN WITH CONTROLLED ACCY.	5-64(295),10-64(586)	Z	118	MAGIC SQUARE	8-62(436),8-62(440),
S14	309	GAMMA FCN.-ARBITRARY PRECISION	8-67(511)	Z	119	12-62(606),1-63(39),3-63(105)	
S14	291	LOGARITHM OF GAMMA FCN.	9-66(684),9-66(685),	Z	136	ENLARGE A GROUP	11-62(555)
S14	291	1-68(14)		Z	148	MAGIC SQAPE	12-62(605),4-63(168)
S14	321	T-TEST PROBABILITIES	2-68(115)	Z	199	CALENDAR CONVERSION	8-63(444),11-64(661)
S14	322	F-DISTRIBUTION	2-68(116)	Z	240	COORDINATES ON AN ELLIPSOID	9-64(546)
S14		GAMMA FUNCTION	BIT 1962(238)	Z	246	GRAYCODE	12-64(701),6-65(382)
S14		GAMMA FCN. BY CHEBYSHEV EXP.	NUM.MATH.V4(413)	Z	252	VECTOR COUPLING COEFFICIENTS	4-65(217)
S15	11	HERMITE POLYNOMIAL	6-60(353)	Z	260	6-J SYMBCLS	8-65(492)
S15	123	REAL ERROR FUNCTION, ERF(X)	9-62(483),6-63(316),	Z	261	9-J SYMBCLS	8-65(492)
S15	123	10-63(618),3-64(145),6-67(377)		Z		CALCULATION OF EASTER	4-62(209),11-62(556)
S15	180	ERROR FUNCTION-LARGE X	6-63(314),6-67(377)	Z		GRADER PROGRAM	CACM 5-65(277)
S15	181	COMPLEMENTARY ERR.FCN.-LARGE X	6-63(315),	Z		CALCULATION OF EASTER	COMP.BULL.V9(18)
S15	181	12-64(702), 6-67(377)		Z		MANY-ELECTRON WAVEFUNCTIONS	CACM 4-66(278)
S15	185	ERROR FUNCTION	7-63(386)	Z		SEASONAL ADJ-FORECASTING	COMP.J.V10(148),
S15	209	ERROR FUNCTION	10-63(616),3-64(148),	Z		V11(25)	