Printed at the Mathematical Centre, 413 Kruislaan, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

MC SYLLABUS 47.4

NUMAL

NUMERICAL PROCEDURES IN ALGOL 60

VOLUME 3B, LINEAR ALGEBRA, PART 2

P.W. HEMKER (ed.)

MATHEMATISCH CENTRUM AMSTERDAM 1981

1980 Mathematics subject classification: 65XX04, 68B99

ISBN 90 6196 217 X

INDEX	PROCEDURE	CODE	MNT/YR	RECORD NUMBER
VOLUME 3B.				
3.LINEAR ALGEBRA				
1.LINEAR SYSTEMS (SEE VOLUME 3A)				
2.TRANSFORMATION TO SPECIAL FORM				
1.SIMILARITY TRANSFORMATIONS				
1.EQUILIBRATION				
1.REAL MATRICES				
	EQILBR	34173	JUN/74	97
	BAKLBR	34174	JUN/74	97
2.COMPLEX MATRICES				
	EQILBRCOM	34361	JUN/74	99
	BAKLBRCOM	34362	JUN/74	99
2.TRANSF TO HESSENBERG FORM				
1.REAL MATRICES				
1.SYMMETRIC MATRICES				
	TFMSYMTRI2	34140	JUN/74	101
	BAKSYMTRI2	34141	JUN/74	101
	TFMPREVEC	34142	JUN/74	101
	TFMSYMTRI1	34143	JUN/74	101
	BAKSYMTRI1	34144	JUN/74	101
2.ASYMMETRIC MATRICES				
	TFMREAHES	34170	JUN/74	103
	BAKREAHES1	34171	JUN/74	103
	BAKREAHES2	34172	JUN/74	103
2.COMPLEX MATRICES				
1.HERMITIAN MATRICES				
	HSHHRMTRI	34363	JUN/74	105
	HSHHRMTRIVAL	34364	JUN/74	105
	BAKHRMTRI	34365	JUN/74	105
2.NON-HERMITIAN MATRICES				
	HSHCOMHES	34366	JUN/74	107
	BAKCOMHES	34367	JUN/74	107
2.OTHER TRANSFORMATIONS				
1.TRANSF TO BIDIAGONAL FORM				
1.REAL MATRICES				
	HSHREABID	34260	JUN/74	109
	PSTTFMMAT	34261	JUN/74	109
	PRETFMMAT	34262	JUN/74	109
2.COMPLEX MATRICES				
3.THE (ORDINARY) EIGENV PROBLEM				
1.REAL MATRICES				
1.SYMMETRIC MATRICES				
1.TRIDIAGONAL MATRICES				
	VALSYMTRI	34151	JUL/74	111
	VECSYMTRI	34152	JUL/74	111
	QRIVALSYMTRI	34160	JUL/74	111
	QRISYMTRI	34161	JUL/74	111
2.FULL MATRICES				
	EIGVALSYM2	34153	JUL/74	113
	EIGSYM2	34154	JUL/74	113
	EIGVALSYM1	34155	JUL/74	113
	EIGSYM1	34156	JUL/74	113
	QRIVALSYM2	34162	JUL/74	113
	QRISYM	34163	JUL/74	113
	QRIVALSYM1	34164	JUL/74	113
3. 1. 1. 3.ITERATIVE IMPROVEMENT				

INDEX	PROCEDURE	CODE	MNT/YR	RECORD NUMBER
3. 3. 1. 1. 3. 1.AUXILIARY PROCEDURES				
	MERGESORT	36405	NOV/76	297
	VECPERM	36404	NOV/76	297
	ROWPERM	36403	NOV/76	297
2.ORTHOGONALISATION				
	ORTHOG	36402	NOV/76	299
3.IMPROVEMENT AND ERRORBOUNDS				
	SYMEIGINP	36401	NOV/76	301
2.ASYMMETRIC MATRICES				
1.MATRICES IN HESSENBERG FORM				
	REAVALQRI	34180	JUL/74	115
	REAVECHES	34181	JUL/74	115
	REAQRI	34186	JUL/74	115
	COMVALQRI	34190	JUL/74	115
	COMVECHES	34191	JUL/74	115
2.FULL MATRICES				
	REAEIGVAL	34182	JUL/74	117
	REAEIG1	34184	JUL/74	117
	REAEIG3	34187	JUL/74	117
	COMEIGVAL	34192	JUL/74	117
	COMEIG1	34194	JUL/74	117
2.COMPLEX MATRICES				
1.HERMITIAN MATRICES				
	EIGVALHRM	34368	JUL/74	119
	EIGHRM	34369	JUL/74	119
	QRIVALHRM	34370	JUL/74	119
	QRIHRM	34371	JUL/74	119
2.NON-HERMITIAN MATRICES				
1.MATRICES IN HESSENBERG FORM				
	VALQRICOM	34372	JUL/74	121
	QRICOM	34373	JUL/74	121
2.FULL MATRICES				
	EIGVALCOM	34374	JUL/74	123
	EIGCOM	34375	JUL/74	123
4.THE GENERALIZED EIGENV PROBLEM				
1.REAL MATRICES				
1.SYMMETRIC MATRICES				
2.ASYMMETRIC MATRICES				
	QZIVAL	34600	JAN/76	267
	QZI	34601	JAN/76	267
	HSHDECMUL	34602	JAN/76	267
	HESTGL3	34603	JAN/76	267
	HESTGL2	34604	JAN/76	267
	HSH2COL	34605	JAN/76	267
	HSH3COL	34606	JAN/76	267
	HSH2ROW3	34607	JAN/76	267
	HSH2ROW2	34608	JAN/76	267
	HSH3ROW3	34609	JAN/76	267
	HSH3ROW2	34610	JAN/76	267
5.SINGULAR VALUES				
1.REAL MATRICES				
1.BIDIAGONAL MATRICES				
	QRISNGVALBID	34270	JUL/74	125
	QRISNGVALDECBID	34271	JUL/74	125
3. 5. 1. 2.FULL MATRICES				

ERSION: 791231

AUTHORS : T.J.DEKKER AND W.HOFFMANN.

CONTRIBUTORS: W.HOFFMANN, J.G.VERWER.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 731022.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS TWO PROCEDURES.
A) EQILBR EQUILIBRATES A MATRIX BY MEANS OF A DIAGONAL SIMILARITY TRANSFORMATION,
B) BAKLBR PERFORMS THE CORRESPONDING BACK TRANSFORMATION ON THE COLUMNS OF A MATRIX AND SHOULD BE CALLED AFTER EQILBR.

KEYWORDS:

SIMILARITY TRANSFORMATION,
EQUILIBRATION.

SUBSECTION: EQILBR.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" EQILBR(A, N, EM, D, INT); "VALUE" N;
"INTEGER" N; "ARRAY" A, EM, D; "INTEGER" "ARRAY" INT;

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

A: <ARRAY IDENTIFIER>;
"ARRAY"A[1:N,1:N];
ENTRY: THE MATRIX TO BE EQUILIBRATED;
EXIT: THE EQUILIBRATED MATRIX;

EM: <ARRAY IDENTIFIER>;
"ARRAY"EM[0:0];
ENTRY: EM[0], THE MACHINE PRECISION;

D: <ARRAY IDENTIFIER>;
"ARRAY"D[1:N];
EXIT: THE MAIN DIAGONAL OF THE TRANSFORMING DIAGONAL MATRIX;

INT: <ARRAY IDENTIFIER>;
"INTEGER""ARRAY" INT[1:N];
EXIT: INFORMATION DEFINING THE POSSIBLE INTERCHANGING OF SOME ROWS AND THE CORRESPONDING COLUMNS;

PROCEDURES USED:

TAMMAT	=	CP34014.
MATTAM	=	CP34015.
ICHCOL	=	CP34031.
ICHROW	=	CP34032.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N SQUARED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

THE MATRIX IS EQUILIBRATED BY MEANS OF OSBORNE'S DIAGONAL SIMILARITY TRANSFORMATION POSSIBLY WITH INTERCHANGES [2]. THE TRANSFORMING DIAGONAL MATRIX AND THE EQUILIBRATED MATRIX ARE CALCULATED ITERATIVELY:

IN EACH STEP A CERTAIN COLUMN OF THE MATRIX IS MULTIPLIED BY, AND THE CORRESPONDING ROW DIVIDED BY, A FACTOR WHICH IS CHOSEN IN SUCH A WAY THAT THE CONSIDERED COLUMN AND ROW OBTAIN ROUGHLY THE SAME EUCLIDEAN NORM (IN FACT, THE FACTOR IS ROUNDED TO THE NEAREST INTEGRAL POWER OF 2, IN ORDER TO PREVENT ROUNDING ERRORS); THE COLUMNS AND ROWS ARE HANDLED IN CYCLIC ORDER. IF THE MATRIX DOES NOT CONTAIN COLUMNS OR ROWS WHOSE OFF-DIAGONAL ELEMENTS ARE 0 OR NEARLY 0, THEN THE PROCESS (WITH UNROUNDED FACTORS) CONVERGES, AND IN PRACTICE A FEW STEPS ARE NEEDED TO OBTAIN A REASONABLY EQUILIBRATED MATRIX [2].

IF ALL OFF-DIAGONAL ELEMENTS OF SOME CONSIDERED COLUMN (ROW) ARE 0 OR NEARLY 0, THEN THIS COLUMN (ROW) IS INTERCHANGED WITH THE FIRST NONZERO COLUMN (LAST NONZERO ROW) OF THE MATRIX, AND, IN ORDER TO HAVE A SIMILARITY TRANSFORMATION, THE CORRESPONDING ROWS (COLUMNS) ARE ALSO INTERCHANGED; THEN FOR THE FURTHER EQUILIBRATION, THE SUBMATRIX IS CONSIDERED WHICH DOES NOT CONTAIN SUCH ZERO COLUMNS AND ROWS AND THE CORRESPONDING ROWS AND COLUMNS. THE EQUILIBRATION PROCESS IS CONTINUED UNTIL, IN A WHOLE CYCLE NO FACTOR > 2 OR < 0.5 AND NO ZERO COLUMN OR ROW IS FOUND, OR UNTIL (N + 1) * N ** 2 ROWS AND COLUMNS HAVE BEEN CONSIDERED.

SUBSECTION: BAKLBR.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" BAKLBR(N, N1, N2, D, INT, VEC); "VALUE" N, N1, N2;
"INTEGER" N, N1, N2; "ARRAY" D, VEC; "INTEGER" "ARRAY" INT;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE LENGTH OF THE VECTORS TO BE TRANSFORMED;
N1, N2: <ARITHMETIC EXPRESSION>;
THE SERIAL NUMBERS OF THE FIRST AND LAST VECTOR TO BE TRANSFORMED;
VEC: <ARRAY IDENTIFIER>;
"ARRAY"VEC[1:N,N1:N2];
ENTRY: THE N2 - N1 + 1 VECTORS OF LENGTH N TO BE TRANSFORMED;
EXIT: THE N2 - N1 + 1 VECTORS OF LENGTH N RESULTING FROM THE BACK TRANSFORMATION;
D: <ARRAY IDENTIFIER>;
"ARRAY"D[1:N];
ENTRY: THE MAIN DIAGONAL OF THE TRANSFORMING DIAGONAL MATRIX OF ORDER N, AS PRODUCED BY EQILBR;
INT: <ARRAY IDENTIFIER>;
"INTEGER""ARRAY" INT[1:N];
ENTRY: INFORMATION DEFINING THE POSSIBLE INTERCHANGING OF SOME ROWS AND COLUMNS, AS PRODUCED BY EQILBR.

PROCEDURES USED:
ICHROW = CP34032.

RUNNING TIME: ROUGHLY PROPORTIONAL TO (N2 - N1 + 1) * N.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE BACK TRANSFORMATION, WHICH CORRESPONDS WITH THE DIAGONAL SIMILARITY TRANSFORMATION AS PERFORMED BY EQILBR, TRANSFORMS A VECTOR X INTO A VECTOR DX AND PERFORMS THE CORRESPONDING INTERCHANGES. THE MATRIX D IS THE DIAGONAL MATRIX OF THE DIAGONAL SIMILARITY TRANSFORMATION.

REFERENCES:
[1] DEKKER, T. J. AND HOFFMANN, W.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2,
MATHEMATICAL CENTRE TRACTS 23,
MATHEMATISCH CENTRUM, AMSTERDAM, 1968;
[2] OSBORNE, E. E, ON PRECONDITIONING OF MATRICES,
J. ACM 7(1960) 338-354.

EXAMPLES OF USE:
EXAMPLES OF USE OF EQILBR AND BAKLBR CAN BE FOUND IN THE PROCEDURES FOR CALCULATING EIGENVALUES AND EIGENVECTORS AS DESCRIBED IN SECTION 3.3.1.2.2.

SOURCE TEXT(S) :

```
"CODE" 34173;
    "COMMENT" MCA 2405;
    "PROCEDURE" EQILBR(A, N, EM, D, INT); "VALUE" N; "INTEGER" N;
     "ARRAY" A, EM, D; "INTEGER" "ARRAY" INT;
    "BEGIN" "INTEGER" I, IM, I1, P, Q, J, T, COUNT, EXPONENT, NI;
        "REAL" C, R, EPS, OMEGA, FACTOR;

        "PROCEDURE" MOVE(K); "VALUE" K; "INTEGER" K;
        "BEGIN" "REAL" DI;
            NI:= Q - P; T:= T + 1; "IF" K ^= I "THEN"
            "BEGIN" ICHCOL(1, N, K, I, A); ICHROW(1, N, K, I, A);
                DI:= D[I]; D[I]:= D[K]; D[K]:= DI
            "END"
        "END" MOVE;

        "REAL" "PROCEDURE" TAMMAT(L, U, I, J, A, B); "CODE" 34014;
        "REAL" "PROCEDURE" MATTAM(L, U, I, J, A, B); "CODE" 34015;
        "PROCEDURE" ICHCOL(L, U, I, J, A); "CODE" 34031;
        "PROCEDURE" ICHROW(L, U, I, J, A); "CODE" 34032;

        FACTOR:= 1 / (2 * LN(2)); "COMMENT" MORE GENERALLY: LN(BASE);
        EPS:= EM[0]; OMEGA:= 1 / EPS; T:= P:= 1; Q:= NI:= I:= N;
        COUNT:= (N + 1) * N // 2;
        "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" D[J]:= 1; INT[J]:= 0 "END";
        "FOR" I:= "IF" I < Q "THEN" I + 1 "ELSE" P
        "WHILE" COUNT > 0 "AND" NI > 0 "DO"
        "BEGIN" COUNT:= COUNT - 1; IM:= I - 1; I1:= I + 1;
            C:= SQRT(TAMMAT(P, IM, I, I, A, A) +
            TAMMAT(I1, Q, I, I, A, A));
            R:= SQRT(MATTAM(P, IM, I, I, A, A) +
            MATTAM(I1, Q, I, I, A, A));
            "IF" C * OMEGA <= R * EPS "THEN"
            "BEGIN" INT[T]:= I; MOVE(P); P:= P + 1 "END"
            "ELSE" "IF" R * OMEGA <= C * EPS "THEN"
            "BEGIN" INT[T]:= -I; MOVE(Q); Q:= Q - 1 "END"
            "ELSE"
            "BEGIN" EXPONENT:= LN(R / C) * FACTOR;
                "IF" ABS(EXPONENT) > 1 "THEN"
                "BEGIN" NI:= Q - P; C:= 2 ** EXPONENT; R:= 1 / C;
                    D[I]:= D[I] * C;
                    "FOR" J:= 1 "STEP" 1 "UNTIL" IM,
                    I1 "STEP" 1 "UNTIL" N "DO"
                    "BEGIN" A[J,I]:= A[J,I] * C;
                        A[I,J]:= A[I,J] * R
                    "END"
                "END" "ELSE" NI:= NI - 1
            "END"
        "END"
    "END" EQILBR;
        "EOP"
```

```
"CODE" 34174;
    "COMMENT" MCA 2406;
    "PROCEDURE" BAKLBR(N, N1, N2, D, INT, VEC); "VALUE" N, N1, N2;
    "INTEGER" N, N1, N2; "ARRAY" D, VEC; "INTEGER" "ARRAY" INT;
    "BEGIN" "INTEGER" I, J, K, P, Q; "REAL" DI;

        "PROCEDURE" ICHROW(L, U, I, J, A); "CODE" 34032;

        P:= 1; Q:= N;
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" DI:= D[I]; "IF" DI ^= 1 "THEN"
            "FOR" J:= N1 "STEP" 1 "UNTIL" N2 "DO" VEC[I,J]:=
            VEC[I,J] * DI; K:= INT[I];
            "IF" K > 0 "THEN" P:= P + 1 "ELSE"
            "IF" K < 0 "THEN" Q:= Q - 1
        "END";
        "FOR" I:= P - 1 + N - Q "STEP" -1 "UNTIL" 1 "DO"
        "BEGIN" K:= INT[I]; "IF" K > 0 "THEN"
            "BEGIN" P:= P - 1; "IF" K ^= P "THEN"
                ICHROW(N1, N2, K, P, VEC)
            "END" "ELSE"
            "BEGIN" Q:= Q + 1; "IF" -K ^= Q "THEN"
                ICHROW(N1, N2, -K, Q, VEC)
            "END"
        "END"
    "END" BAKLBR;
        "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

CONTRIBUTORS : H.FIOLET, C.G. VAN DER LAAN.

INSTITUTE : MATHEMATICAL CENTRE.

RECEIVED: 731008.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS THE PROCEDURES EQILBRCOM AND BAKLBRCOM.
EQILBRCOM EQUILIBRATES A GIVEN MATRIX.
BAKLBRCOM TRANSFORMS THE EIGENVECTORS OF THE EQUILIBRATED MATRIX
INTO THE EIGENVECTORS OF THE ORIGINAL MATRIX.

KEYWORDS :

COMPLEX MATRIX,
EIGENVECTORS,
EQUILIBRATION.

SUBSECTION: EQILBRCOM.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" EQILBRCOM(A1, A2, N, EM, D, INT); "VALUE" N;
"INTEGER" N; "ARRAY" A1, A2, EM, D; "INTEGER" "ARRAY" INT;

THE MEANING OF THE FORMAL PARAMETERS IS:

A1,A2: <ARRAY IDENTIFIER>;
"ARRAY" A1,A2[1:N,1:N];
ENTRY:
THE REAL PART AND IMAGINARY PART OF THE MATRIX TO BE
EQUILIBRATED MUST BE GIVEN IN THE ARRAYS A1 AND A2,
RESPECTIVELY;
EXIT:
THE REAL PART AND THE IMAGINARY PART OF THE EQUILIBRATED
MATRIX ARE DELIVERED IN THE ARRAYS A1 AND A2,
RESPECTIVELY;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

EM: <ARRAY IDENTIFIER>;
"ARRAY"EM[0:7];
ENTRY:
EM[0]: THE MACHINE PRECISION;
EM[6]: THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
EXIT:
EM[7]: THE NUMBER OF ITERATIONS PERFORMED;

D: <ARRAY IDENTIFIER>:
"ARRAY" D[1:N];
EXIT:
THE SCALING FACTORS OF THE DIAGONAL SIMILARITY
TRANSFORMATION;
INT: <ARRAY IDENTIFIER>;
"INTEGER""ARRAY" INT[1:N];
EXIT:
INFORMATION CONCERNING THE POSSIBLE INTERCHANGING OF
SOME ROWS AND CORRESPONDING COLUMNS.

PROCEDURES USED:

ICHCOL = CP34031,
ICHROW = CP34032,
TAMMAT = CP34014,
MATTAM = CP34015.

RUNNING TIME: PROPORTIONAL TO N * NUMBER OF ITERATIONS.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE BAKLBRCOM.

SUBSECTION: BAKLBRCOM.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" BAKLBRCOM(N, N1, N2, D, INT, VR, VI);
"VALUE" N, N1, N2; "INTEGER" N, N1, N2; "ARRAY" D, VR, VI;
"INTEGER" "ARRAY" INT;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE MATRIX OF WHICH THE EIGENVECTORS ARE
CALCULATED;
N1,N2: <ARITHMETIC EXPRESSION>;
THE EIGENVECTORS CORRESPONDING TO THE EIGENVALUES WITH
INDICES N1,...,N2 ARE TO BE TRANSFORMED;
D: <ARRAY IDENTIFIER>:
"ARRAY" D[1:N];
ENTRY: THE SCALING FACTORS OF THE DIAGONAL SIMILARITY
TRANSFORMATION AS DELIVERED BY EQILBRCOM;
INT: <ARRAY IDENTIFIER>;
"INTEGER""ARRAY" INT[1:N];
ENTRY: INFORMATION DEFINING THE INTERCHANGING OF SOME
ROWS AND COLUMNS, AS DELIVERED BY EQILBRCOM;

VR,VI: <ARRAY IDENTIFIER>;
"ARRAY" VR,VI[1:N,N1:N2];
ENTRY:
THE BACK TRANSFORMATION IS PERFORMED ON THE EIGENVECTORS WITH THE REAL PARTS GIVEN IN ARRAY VR AND THE IMAGINARY PARTS GIVEN IN ARRAY VI;
EXIT:
THE REAL PARTS AND IMAGINARY PARTS OF THE RESULTING EIGENVECTORS ARE DELIVERED IN THE COLUMNS OF THE ARRAYS VR AND VI, RESPECTIVELY.

PROCEDURES USED: BAKLBR = CP34174.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N * (N2-N1).

LANGUAGE: ALGOL 60.

THE FOLLOWING HOLDS FOR BOTH PROCEDURES:

METHOD AND PERFORMANCE:

A MATRIX M IS SAID TO BE EQUILIBRATED, WHEN THE DIAGONAL ELEMENTS OF M''M - MM'' ARE ZERO, WHERE '' STANDS FOR CONJUGATING AND TRANSPOSING. IN EQILBRCOM THE MATRIX M IS EQUILIBRATED BY MEANS OF OSBORNE'S DIAGONAL SIMILARITY TRANSFORMATION WITH POSSIBLE INTERCHANGES (OSBORNE, 1960).
BAKLBRCOM PERFORMS THE CORRESPONDING BACK TRANSFORMATION.
LET THE EIGENVECTORS OF THE EQUILIBRATED MATRIX BE GIVEN IN THE COLUMNS OF MATRIX V. THE EIGENVECTORS OF THE ORIGINAL MATRIX ARE OBTAINED BY MULTIPLYING (OR POSSIBLE INTERCHANGING) THE ROWS OF THE MATRIX V WITH THE SCALING FACTORS. AS THE SCALING FACTORS ARE REAL QUANTITIES, THE TRANSFORMATION IS PERFORMED BY CALLING THE PROCEDURE BAKLBR FOR BOTH VR AND VI (DEKKER AND HOFFMANN, 1968).

REFERENCES:

DEKKER, T.J. AND W.HOFFMANN (1968),
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2,
MATH. CENTRE TRACTS 23, MATHEMATISCH CENTRUM;

OSBORNE, E.E. (1960),
ON PRECONDITIONING OF MATRICES,
JACM., 7, P.338-354;

PARLETT, B.N. AND C.REINSCH (1969),
BALANCING A MATRIX FOR CALCULATION OF EIGENVALUES AND EIGENVECTORS,
NUM. MATH., 13, P.293-304;

EXAMPLE OF USE:

BAKLBRCOM IS USED IN THE PROCEDURE EIGCOM (SEE SECTION 3.3.2.2.2.).
AS A FORMAL TEST OF THE PROCEDURE EQILBRCOM, THE FOLLOWING MATRIX
WAS USED:

```
  1      0   1024*I
  0      1     0
I/1024   0     2
```

```
"BEGIN" "INTEGER" I,J;
"REAL" "ARRAY" A1,A2[1:3,1:3],EM[0:7],D,INT[1:3];
"PROCEDURE" INIMAT(LR,UR,LC,UC,A,X):"CODE" 31011;
"PROCEDURE" EQILBRCOM(A1,A2,N,EM,D,INT):"CODE" 34361;
EM[0]:=5"-14;EM[6]:=10;
INIMAT(1,3,1,3,A1,0);INIMAT(1,3,1,3,A2,0);
A1[1,1]:=A1[2,2]:=1;A1[3,3]:=2;
A2[1,3]:=2**10;A2[3,1]:=2**(-10);
EQILBRCOM(A1,A2,3,EM,D,INT);
OUTPUT(61,"(""(""EQUILIBRATED MATRIX:")",/")");
OUTPUT(61,"("3(D2B),/,2(D2B),"("I")",/,D2B,"("I")",2BD/")",
          A1[1,1],A1[1,2],A1[1,3],A1[2,1],A1[2,2],A1[3,1],A1[3,3]);
OUTPUT(61,"("/,"("EM[7]:")",5BD/")",EM[7]);
OUTPUT(61,"("""("D[1:3]: ")",3(3ZD2B),/")",D[1],D[2],D[3]);
OUTPUT(61,"("""("INT[1:3]: ")",BD,3B,2BD,4BD")",
                INT[1],INT[2],INT[3])
"END"
```

```
OUTPUT:
EQUILIBRATED MATRIX:
1  0  0
0  1  I
0  I  2

EM[7]:      4
D[1:3]:     1   1024      1
INT[1:3]:   2      0      0
```

SOURCE TEXT(S) :

```
"CODE" 34361:
    "PROCEDURE" EQILBRCOM(A1, A2, N, EM, D, INT); "VALUE" N;
    "INTEGER" N; "ARRAY" A1, A2, EM, D; "INTEGER" "ARRAY" INT;
    "BEGIN" "INTEGER" I, P, Q, J, T, COUNT, EXPONENT, NI, IM, I1;
        "REAL" C, R, EPS;
        "PROCEDURE" ICHCOL(L,U,I,J,A);"CODE" 34031;
        "PROCEDURE" ICHROW(L,U,I,J,A);"CODE" 34032;
        "REAL" "PROCEDURE" TAMMAT(L,U,I,J,A,B);"CODE" 34014;
        "REAL" "PROCEDURE" MATTAM(L,U,I,J,A,B);"CODE" 34015;
                                                        "COMMENT"
```

```
        "PROCEDURE" MOVE(K); "VALUE" K; "INTEGER" K;
        "BEGIN" "REAL" DI;
            NI:= Q - P; T:= T + 1; "IF" K ^= I "THEN"
            "BEGIN" ICHCOL(1, N, K, I, A1); ICHROW(1, N, K, I, A1);
                ICHCOL(1, N, K, I, A2); ICHROW(1, N, K, I, A2);
                DI:= D[I]; D[I]:= D[K]; D[K]:= DI
            "END"
        "END" MOVE;
        EPS:= EM[0] ** 4; T:= P:= 1; Q:= NI:= I:= N;
        COUNT:= EM[6];
        "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" D[J]:= 1; INT[J]:= 0 "END";
        "FOR" I:= "IF" I < Q "THEN" I + 1 "ELSE" P "WHILE" COUNT > 0
        "AND" NI > 0 "DO"
        "BEGIN" COUNT:= COUNT - 1; IM:= I - 1; I1:= I + 1;
            C:= TAMMAT(P, IM, I, I, A1, A1) + TAMMAT(I1, Q, I,
            I, A1, A1) + TAMMAT(P, IM, I, I, A2, A2) +
            TAMMAT(I1, Q, I, I, A2, A2);
            R:= MATTAM(P, IM, I, I, A1, A1) + MATTAM(I1, Q, I,
            I, A1, A1) + MATTAM(P, IM, I, I, A2, A2) +
            MATTAM(I1, Q, I, I, A2, A2); "IF" C / EPS <= R "THEN"
            "BEGIN" INT[T]:= I; MOVE(P); P:= P + 1 "END"
            "ELSE" "IF" R / EPS <= C "THEN"
            "BEGIN" INT[T]:= - I; MOVE(Q); Q:= Q - 1 "END"
            "ELSE"
            "BEGIN" EXPONENT:= LN(R / C) * 0.36067;
                "IF" ABS(EXPONENT) > 1 "THEN"
                "BEGIN" NI:= Q - P; C:= 2 ** EXPONENT;
                    D[I]:= D[I] * C;
                    "FOR" J:= 1 "STEP" 1 "UNTIL" IM, I1 "STEP" 1
                    "UNTIL" N "DO"
                    "BEGIN" A1[J,I]:= A1[J,I] * C;
                        A1[I,J]:= A1[I,J] / C;
                        A2[J,I]:= A2[J,I] * C;
                        A2[I,J]:= A2[I,J] / C
                    "END"
                "END"
                "ELSE" NI:= NI - 1
            "END"
        "END";
        EM[7]:= EM[6] - COUNT
    "END" EQILBRCOM;
        "EOP"

"CODE" 34362;
    "PROCEDURE" BAKLBRCOM(N, N1, N2, D, INT, VR, VI);
    "VALUE" N, N1, N2; "INTEGER" N, N1, N2; "ARRAY" D, VR, VI;
    "INTEGER" "ARRAY" INT;
    "BEGIN"
        "PROCEDURE" BAKLBR(N,N1,N2,D,INT,VEC);"CODE" 34174;
        BAKLBR(N, N1, N2, D, INT, VR);
        BAKLBR(N, N1, N2, D, INT, VI)
    "END" BAKLBRCOM;
        "EOP"
```

AUTHORS: T.J.DEKKER, W.HOFFMANN.

CONTRIBUTORS: W.HOFFMANN, J.G.VERWER.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730705.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS FIVE PROCEDURES.
A) TFMSYMTRI2 AND TFMSYMTRI1 TRANSFORM A REAL SYMMETRIC MATRIX INTO A SIMILAR TRIDIAGONAL ONE BY MEANS OF HOUSEHOLDER'S TRANSFORMATION,
B) BAKSYMTRI2 AND BAKSYMTRI1 PERFORM THE CORRESPONDING BACK TRANSFORMATION AND FINALLY,
C) TFMPREVEC (WHICH IS TO BE USED IN COMBINATION WITH TFMSYMTRI2) CALCULATES THE TRANSFORMING MATRIX.
TFMSYMTRI2 AND BAKSYMTRI2 USE THE UPPER TRIANGLE OF A TWO-DIMENSIONAL ARRAY FOR THE UPPER TRIANGLE OF THE GIVEN SYMMETRIC MATRIX (TFMSYMTRI2) OR FOR THE DATA FOR HOUSEHOLDER'S BACK TRANSFORMATION (BAKSYMTRI2). THE OTHER ELEMENTS ARE NEITHER USED NOR CHANGED.
TFMSYMTRI1 AND BAKSYMTRI1 USE AN ARRAY A[1:(N+1)*N//2] FOR THE GIVEN SYMMETRIC MATRIX (TFMSYMTRI1) OR FOR THE DATA FOR HOUSEHOLDER'S TRANSFORMATION (BAKSYMTRI1).

KEYWORDS:

HOUSEHOLDER'S TRANSFORMATION,
TRIANGULARIZATION.

SUBSECTION: TFMSYMTRI2.

CALLING SEQUENCE:
THE HEADING OF THIS PROCEDURE IS:
"PROCEDURE" TFMSYMTRI2(A, N, D, B, BB, EM); "VALUE" N;
"INTEGER" N; "ARRAY" A, D, B, BB, EM; "CODE" 34140;

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

A: <ARRAY IDENTIFIER>;
"ARRAY"A[1:N,1:N];
ENTRY: THE UPPER TRIANGLE OF THE SYMMETRIC MATRIX MUST BE GIVEN IN THE UPPER TRIANGULAR PART OF A (THE ELEMENTS A[I,J],I<=J);
EXIT: THE DATA FOR HOUSEHOLDER'S BACK TRANSFORMATION IS DELIVERED IN THE UPPER TRIANGULAR PART OF A. THE ELEMENTS A[I,J], I>J ARE NEITHER USED NOR CHANGED;

D: <ARRAY IDENTIFIER>;
"ARRAY"D[1:N];
EXIT: THE MAIN DIAGONAL OF THE SYMMETRIC TRIDIAGONAL MATRIX T (SAY), PRODUCED BY HOUSEHOLDER'S TRANSFORMATION;

B: <ARRAY IDENTIFIER>;
"ARRAY" B[1:N];
EXIT: THE CODIAGONAL ELEMENTS OF T ARE DELIVERED IN B[1] THROUGH B[N-1]; B[N] IS SET EQUAL TO ZERO;

BB: <ARRAY IDENTIFIER>;
"ARRAY"BB[1:N];
EXIT: THE SQUARES OF THE CODIAGONAL ELEMENTS OF T ARE DELIVERED IN BB[1] THROUGH BB[N-1]; BB[N] IS SET EQUAL TO ZERO;

EM: <ARRAY IDENTIFIER>;
"ARRAY"EM[0:1];
ENTRY: EM[0], THE MACHINE PRECISION;
EXIT: EM[1], THE INFINITY NORM OF THE ORIGINAL MATRIX.

PROCEDURES USED:

TAMVEC	=	CP34012,
MATMAT	=	CP34013,
TAMMAT	=	CP34014,
ELMVECCOL	=	CP34021,
ELMCOLVEC	=	CP34022,
ELMCOL	=	CP34023.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

METHOD AND PERFORMANCE:
A GIVEN SYMMETRIC MATRIX M IS TRANSFORMED INTO A TRIDIAGONAL ONE BY MEANS OF N-1 ORTHOGONAL SIMILARITY TRANSFORMATIONS; THE P-TH TRANSFORMATION IS CHOSEN IN SUCH A WAY THAT IN THE (N-P+1)-TH COLUMN AND ROW OF M THE DESIRED ZEROES ARE INTRODUCED. HOWEVER, IF, IN THIS COLUMN AND ROW, ALL ELEMENTS OUTSIDE THE MAIN DIAGONAL AND THE ADJACENT CODIAGONALS ARE SMALLER IN ABSOLUTE VALUE THAN THE INFINITY NORM OF M TIMES THE MACHINE PRECISION, THEN THE P-TH TRANSFORMATION IS SKIPPED.
FOR FURTHER DETAILS SEE REF[1] AND REF[2].

SUBSECTION: BAKSYMTRI2.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" BAKSYMTRI2(A, N, N1, N2, VEC); "VALUE" N, N1, N2;
"INTEGER" N, N1, N2; "ARRAY" A, VEC; "CODE" 34141;

THE MEANING OF THE FORMAL PARAMETERS IS:

A:	<ARRAY IDENTIFIER>;	
	"ARRAY"A[1:N,1:N];	
	ENTRY:	THE DATA FOR THE BACK TRANSFORMATION, AS PRODUCED BY TFMSYMTRI2, MUST BE GIVEN IN THE UPPER TRIANGULAR PART OF A;
N:	<ARITHMETIC EXPRESSION>;	
	THE ORDER OF THE GIVEN MATRIX;	
N1,N2:	<ARITHMETIC EXPRESSION>;	
	LOWER AND UPPER BOUND, RESPECTIVELY, OF THE COLUMN NUMBERS OF VEC;	
VEC:	<ARRAY IDENTIFIER>;	
	"ARRAY"VEC[1:N,N1:N2];	
	ENTRY:	THE VECTORS ON WHICH THE BACK TRANSFORMATION HAS TO BE PERFORMED;
	EXIT:	THE TRANSFORMED VECTORS.

PROCEDURES USED:
TAMMAT = CP34014,
ELMCOL = CP34023.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N SQUARED TIMES (N2-N1+1).

METHOD AND PERFORMANCE: SEE REF[1].

SUBSECTION: TFMPREVEC.

CALLING SEQUENCE:
THE HEADING OF THIS PROCEDURE IS:
"PROCEDURE" TFMPREVEC(A, N); "VALUE" N; "INTEGER" N; "ARRAY" A;
"CODE" 34142;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARRAY IDENTIFIER>;
"ARRAY"A[1:N,1:N];
ENTRY: THE DATA FOR THE BACK TRANSFORMATION, AS PRODUCED BY TFMSYMTRI2, MUST BE GIVEN IN THE UPPER TRIANGULAR PART OF A;
EXIT: THE MATRIX WHICH TRANSFORMS THE ORIGINAL MATRIX INTO A SIMILAR TRIDIAGONAL ONE.
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE MATRIX;

PROCEDURES USED:
TAMMAT = CP34014,
ELMCOL = CP34023.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

METHOD AND PERFORMANCE: SEE REF[1].

SUBSECTION: TFMSYMTRI1.

CALLING SEQUENCE:
THE HEADING OF THIS PROCEDURE IS:
"PROCEDURE" TFMSYMTRI1(A, N, D, B, BB, EM); "VALUE" N;
"INTEGER" N; "ARRAY" A, D, B, BB, EM; "CODE" 34143;

THE MEANING OF THE FORMAL PARAMETERS IS:

A: <ARRAY IDENTIFIER>;
"ARRAY"A[1:(N+1)*N//2];
ENTRY: THE UPPER TRIANGLE OF THE GIVEN MATRIX MUST BE GIVEN IN SUCH A WAY THAT THE (I,J)-TH ELEMENT OF THE MATRIX IS A[(J-1)*J//2+I], 1<=I<=J<=N;
EXIT: THE DATA FOR HOUSEHOLDER'S BACK TRANSFORMATION AS USED BY BAKSYMTRI1;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

D: <ARRAY IDENTIFIER>;
"ARRAY"D[1:N];
EXIT: THE MAIN DIAGONAL OF THE SYMMETRIC TRIDIAGONAL MATRIX T (SAY), PRODUCED BY HOUSEHOLDER'S TRANSFORMATION;

B: <ARRAY IDENTIFIER>;
"ARRAY" B[1:N];
EXIT: THE CODIAGONAL ELEMENTS OF T ARE DELIVERED IN B[1] THROUGH B[N-1]; B[N] IS SET EQUAL TO ZERO;

BB: <ARRAY IDENTIFIER>;
"ARRAY"BB[1:N];
EXIT: THE SQUARES OF THE CODIAGONAL ELEMENTS OF T ARE DELIVERED IN BB[1] THROUGH BB[N-1]; BB[N] IS SET EQUAL TO ZERO;

EM: <ARRAY IDENTIFIER>;
"ARRAY"EM[0:1];
ENTRY: EM[0], THE MACHINE PRECISION;
EXIT: EM[1], THE INFINITY NORM OF THE ORIGINAL MATRIX.

PROCEDURES USED:
VECVEC = CP34010,
SEQVEC = CP34016,
ELMVEC = CP34020.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

METHOD AND PERFORMANCE: SEE TFMSYMTRI2 (THIS SECTION).

SUBSECTION: BAKSYMTRI1.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" BAKSYMTRI1(A, N, N1, N2, VEC); "VALUE" N, N1, N2;
"INTEGER" N, N1, N2; "ARRAY" A, VEC; "CODE" 34144;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARRAY IDENTIFIER>;
"ARRAY"A[1:(N+1)*N//2];
ENTRY: THE DATA FOR THE BACK TRANSFORMATION, AS PRODUCED BY TFMSYMTRI1;
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;
N1,N2: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND, RESPECTIVELY, OF THE COLUMN NUMBERS OF VEC;
VEC: <ARRAY IDENTIFIER>;
"ARRAY"VEC[1:N,N1:N2];
ENTRY: THE VECTORS ON WHICH THE BACK TRANSFORMATION HAS TO BE PERFORMED;
EXIT: THE TRANSFORMED VECTORS.

PROCEDURES USED:
VECVEC = CP34010,
ELMVEC = CP34020.

REQUIRED CENTRAL MEMORY:
AN AUXILIARY ONE-DIMENSIONAL REAL ARRAY OF LENGTH N IS USED.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N SQUARED TIMES (N2-N1+1).

METHOD AND PERFORMANCE: SEE REF[1].

REFERENCES:
[1] DEKKER, T.J. AND HOFFMANN, W.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2,
MATHEMATICAL CENTRE TRACTS 23,
MATHEMATISCH CENTRUM, AMSTERDAM, 1968;

[2] WILKINSON, J.H.
THE ALGEBRAIC EIGENVALUE PROBLEM,
CLARENDON PRESS, OXFORD 1965.

EXAMPLE OF USE:

THE FIVE PROCEDURES OF THIS SECTION ARE USED IN SECTION 3.3.1.1.2:
EIGSYM2 USES TFMSYMTRI2 AND BAKSYMTRI2;
EIGSYM1 USES TFMSYMTRI1 AND BAKSYMTRI1;
QRISYM USES TFMSYMTRI2 AND TFMPREVEC.

SOURCE TEXT(S):

```
"CODE" 34140;
    "COMMENT" MCA 2300;
    "PROCEDURE" TFMSYMTRI2(A, N, D, B, BB, EM); "VALUE" N;"INTEGER" N;
     "ARRAY" A, B, BB, D, EM;
    "BEGIN" "INTEGER" I, J, R, R1;
        "REAL" W, X, A1, BO, BBJ, DO, MACHTOL, NORM;

        "REAL" "PROCEDURE" TAMMAT(L, U, I, J, A, B); "CODE" 34014;
        "REAL" "PROCEDURE" MATMAT(L, U, I, J, A, B); "CODE" 34013;
        "PROCEDURE" ELMVECCOL(L, U, I, A, B, X); "CODE" 34021;
        "REAL" "PROCEDURE" TAMVEC(L, U, I, A, B); "CODE" 34012;
        "PROCEDURE" ELMCOL(L, U, I, J, A, B, X); "CODE" 34023;
        "PROCEDURE" ELMCOLVEC(L, U, I, A, B, X); "CODE" 34022;

        NORM:= 0;
        "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" W:= 0;
            "FOR" I:= 1 "STEP" 1 "UNTIL" J "DO" W:= ABS(A[I,J]) + W;
            "FOR" I:= J + 1 "STEP" 1 "UNTIL" N "DO" W:= ABS(A[J,I]) +
            W; "IF" W > NORM "THEN" NORM:= W
        "END";
        MACHTOL:= EM[0] * NORM; EM[1]:= NORM; R:= N;
        "FOR" R1:= N - 1 "STEP" -1 "UNTIL" 1 "DO"
        "BEGIN" D[R]:= A[R,R]; X:= TAMMAT(1, R - 2, R, R, A, A);
            A1:= A[R1,R]; "IF" SQRT(X) <= MACHTOL "THEN"
            "BEGIN" BO:= B[R1]:= A1; BB[R1]:= BO * BO;A[R,R]:= 1 "END"
            "ELSE"
            "BEGIN" BBO:= BB[R1]:= A1 * A1 + X;
                BO:= "IF" A1 > 0 "THEN" -SQRT(BBO) "ELSE" SQRT(BBO);
                A1:= A[R1,R]:= A1 - BO; W:= A[R,R]:= 1 / (A1 * BO);
                "FOR" J:= 1 "STEP" 1 "UNTIL" R1 "DO" B[J]:= (TAMMAT(1,
                J, J, R, A, A) + MATMAT(J + 1, R1, J, R, A, A)) * W;
                ELMVECCOL(1, R1, R, B, A, TAMVEC(1, R1, R, A, B) *
                W * .5); "FOR" J:= 1 "STEP" 1 "UNTIL" R1 "DO"
                "BEGIN" ELMCOL(1, J, J, R, A, A, B[J]);
                    ELMCOLVEC(1, J, J, A, B, A[J,R])
                "END"; B[R1]:= BO
            "END"; R:= R1
        "END";
        D[1]:= A[1,1]; A[1,1]:= 1; B[N]:= BB[N]:= 0
    "END" TFMSYMTRI2;
        "EOP"
```

```
"CODE" 34141;
    "COMMENT" MCA 2301;
    "PROCEDURE" BAKSYMTRI2(A, N, N1, N2, VEC); "VALUE" N, N1, N2;
    "INTEGER" N, N1, N2; "ARRAY" A, VEC;
    "BEGIN" "INTEGER" I, J, K; "REAL" W;

        "REAL" "PROCEDURE" TAMMAT(L, U, I, J, A, B); "CODE" 34014;
        "PROCEDURE" ELMCOL(L, U, I, J, A, B, X); "CODE" 34023;

        "FOR" J:= 2 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" W:= A[J,J]; "IF" W < 0 "THEN"
            "FOR" K:= N1 "STEP" 1 "UNTIL" N2 "DO"
            ELMCOL(1, J - 1, K, J, VEC, A,
            TAMMAT(1, J - 1, J, K, A, VEC) * W)
        "END"
    "END" BAKSYMTRI2;
        "EOP"


"CODE" 34142;
    "COMMENT" MCA 2302;
    "PROCEDURE" TEMPREVEC(A, N); "VALUE" N; "INTEGER" N; "ARRAY" A;
    "BEGIN" "INTEGER" I, J, J1, K; "REAL" AB;

        "REAL" "PROCEDURE" TAMMAT(L, U, I, J, A, B); "CODE" 34014;
        "PROCEDURE" ELMCOL(L, U, I, J, A, B, X); "CODE" 34023;

        J1:= 1;
        "FOR" J:= 2 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" "FOR" I:= 1 "STEP" 1 "UNTIL" J1 - 1 ,
            J "STEP" 1 "UNTIL" N "DO" A[I,J1]:= 0;
            A[J1,J1]:= 1; AB:= A[J,J];
            "IF" AB < 0 "THEN"
            "FOR" K:= 1 "STEP" 1 "UNTIL" J1 "DO"
            ELMCOL(1, J1, K, J, A, A,
            TAMMAT(1, J1, J, K, A, A) * AB); J1:= J
        "END";
        "FOR" I:= N - 1 "STEP" -1 "UNTIL" 1 "DO"
        A[I,N]:= 0; A[N,N]:= 1
    "END" TEMPREVEC;
        "EOP"
```

```
"CODE" 34143;
    "COMMENT" MCA 2305;
    "PROCEDURE" TFMSYMTRI1(A, N, D, B, BB, EM); "VALUE" N;"INTEGER" N;
     "ARRAY" A, B, BB, D, EM;
   "BEGIN" "INTEGER" I, J, R, R1, P, Q, TI, TJ;
        "REAL" S, W, X, A1, B0, BB0, D0, NORM, MACHTOL;

        "REAL" "PROCEDURE" VECVEC(L, U, SHIFT, A, B); "CODE" 34010;
        "REAL" "PROCEDURE" SEQVEC(L, U, IL, SHIFT, A, B);"CODE" 34016;
        "PROCEDURE" ELMVEC(L, U, SHIFT, A, B, X); "CODE" 34020;

        NORM:= 0; TJ:= 0;
        "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" W:= 0;
            "FOR" I:= 1 "STEP" 1 "UNTIL" J "DO" W:= ABS(A[I + TJ]) +W;
            TJ:= TJ + J; TI:= TJ + J;
            "FOR" I:= J + 1 "STEP" 1 "UNTIL" N "DO"
            "BEGIN" W:= ABS(A[TI]) + W; TI:= TI + I "END";
            "IF" W > NORM "THEN" NORM:= W
        "END";
        MACHTOL:= EM[0] * NORM; EM[1]:= NORM; Q:= (N + 1) * N // 2;
        R:= N; "FOR" R1:= N - 1 "STEP" -1 "UNTIL" 1 "DO"
        "BEGIN" P:= Q - R; D[R]:= A[Q];
            X:= VECVEC(P + 1, Q - 2, 0, A, A);
            A1:= A[Q - 1]; "IF" SQRT(X) <= MACHTOL "THEN"
            "BEGIN" B0:= B[R1]:= A1; BB[R1]:= B0 * B0; A[Q]:= 1 "END"
            "ELSE"
            "BEGIN" BB0:= BB[R1]:= A1 * A1 + X;
                B0:= "IF" A1 > 0 "THEN" -SQRT(BB0) "ELSE" SQRT(BB0);
                A1:= A[Q - 1]:= A1 - B0; W:= A[Q]:= 1 / (A1 * B0);
                TJ:= 0; "FOR" J:= 1 "STEP" 1 "UNTIL" R1 "DO"
                "BEGIN" TI:= TJ + J; S:= VECVEC(TJ + 1, TI, P - TJ,
                    A, A); TJ:= TI + J;
                    B[J]:= (SEQVEC(J + 1, R1, TJ, P, A, A) + S) * W;
                    TJ:= TI
                "END";
                ELMVEC(1, R1, P, B, A, VECVEC(1,R1,P,B,A)* W *.5);
                TJ:= 0; "FOR" J:= 1 "STEP" 1 "UNTIL" R1 "DO"
                "BEGIN" TI:= TJ + J; ELMVEC(TJ + 1, TI, P - TJ, A, A,
                    B[J]);ELMVEC(TJ + 1, TI, -TJ, A, B, A[J + P]);
                    TJ:= TI
                "END"; B[R1]:= B0
            "END";
            Q:= P; R:= R1
        "END";
        D[1]:= A[1]; A[1]:= 1; B[N]:= BB[N]:= 0
    "END" TFMSYMTRI1;
        "EOP"
```

```
"CODE" 34144;
    "COMMENT" MCA 2306;
    "PROCEDURE" BAKSYMTRI1(A, N, N1, N2, VEC); "VALUE" N, N1, N2;
    "INTEGER" N, N1, N2; "ARRAY" A, VEC;
    "BEGIN" "INTEGER" J, J1, K, TI, TJ;
        "REAL" W; "ARRAY" AUXVEC[1:N];

        "REAL" "PROCEDURE" VECVEC(L, U, SHIFT, A, B); "CODE" 34019;
        "PROCEDURE" ELMVEC(L, U, SHIFT, A, B, X); "CODE" 34020;

        "FOR" K:= N1 "STEP" 1 "UNTIL" N2 "DO"
        "BEGIN" "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO"
            AUXVEC[J]:= VEC[J,K]; TJ:= J1:= 1;
            "FOR" J:= 2 "STEP" 1 "UNTIL" N "DO"
            "BEGIN" TI:= TJ + J; W:= A[TI];
                "IF" W < 0 "THEN" ELMVEC(1, J1, TJ, AUXVEC,A,VECVEC(1,
                J1, TJ, AUXVEC, A) * W); J1:= J; TJ:= TI
            "END";
            "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO" VEC[J,K]:= AUXVEC[J]
        "END"
    "END" BAKSYMTRI1;
        "EOP"
```

AUTHORS: T.J.DEKKER AND W.HOFFMANN.

CONTRIBUTORS: W.HOFFMANN, J.G.VERWER.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 731112.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THREE PROCEDURES.
A) TFMREAHES TRANSFORMS A MATRIX INTO A SIMILAR UPPER-HESSENBERG MATRIX BY MEANS OF WILKINSON'S TRANSFORMATION,
B) BAKREAHES1 PERFORMS THE CORRESPONDING BACK TRANSFORMATION ON A VECTOR AND SHOULD BE CALLED AFTER TFMREAHES,
C) BAKREAHES2 PERFORMS THE CORRESPONDING BACK TRANSFORMATION ON THE COLUMNS OF A MATRIX AND SHOULD BE CALLED AFTER TFMREAHES.

KEYWORDS:

SIMILARITY TRANSFORMATION,
UPPER-HESSENBERG MATRIX.

SUBSECTION: TFMREAHES.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" TFMREAHES(A, N, EM, INT); "VALUE" N;
"INTEGER" N; "ARRAY" A, EM; "INTEGER" "ARRAY" INT;

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

A: <ARRAY IDENTIFIER>;
"ARRAY"A[1:N,1:N];
ENTRY: THE MATRIX TO BE TRANSFORMED;
EXIT: THE UPPER-HESSENBERG MATRIX IS DELIVERED IN THE UPPER TRIANGLE AND THE FIRST SUBDIAGONAL OF A, THE (NONTRIVIAL ELEMENTS OF THE) TRANSFORMING MATRIX, L, IN THE REMAINING PART OF A, I.E. A[I,J] = L[I,J + 1], FOR I = 3,...,N AND J = 1,...,I - 2;

EM: <ARRAY IDENTIFIER>;
"ARRAY"EM[0:1];
ENTRY: EM[0], THE MACHINE PRECISION;
EXIT: EM[1], THE INFINITY NORM OF THE ORIGINAL MATRIX;

INT: <ARRAY IDENTIFIER>;
"INTEGER""ARRAY" INT[1:N];
EXIT: THE PIVOTAL INDICES DEFINING THE STABILIZING ROW AND COLUMN INTERCHANGES;

PROCEDURES USED:

MATVEC	=	CP34011,
MATMAT	=	CP34013,
ICHCOL	=	CP34031,
ICHROW	=	CP34032.

REQUIRED CENTRAL MEMORY:
EXECUTION FIELD LENGTH:
A ONE-DIMENSIONAL REAL ARRAY OF LENGTH N IS DECLARED.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
WILKINSON'S TRANSFORMATION IS A TRIANGULAR SIMILARITY TRANSFORMATION WITH STABILIZING ROW AND COLUMN INTERCHANGES TRANSFORMING A MATRIX, M, INTO AN UPPER-HESSENBERG MATRIX, H. THE TRANSFORMING MATRIX IS THE PRODUCT OF A PERMUTATION MATRIX, P, AND A UNIT LOWER-TRIANGULAR MATRIX, L. THE NONDIAGONAL ELEMENTS IN THE FIRST COLUMN OF L ARE 0, AND THE ROW AND COLUMN INTERCHANGES ARE CHOSEN IN SUCH A WAY THAT THE ABSOLUTE VALUE OF EACH ELEMENT OF L IS AT MOST 1.
BECAUSE OF THE SPECIAL FORM OF L, THE MATRICES H AND L CAN BE STORED TOGETHER IN THE ARRAY USED FOR THE MATRIX M (SEE CALLING SEQUENCE). FOR FURTHER DETAILS SEE REFERENCE [1] AND [2].

SUBSECTION: BAKREAHES1.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" BAKREAHES1(A, N, INT, V); "VALUE" N;
"INTEGER" N; "ARRAY" A, V; "INTEGER" "ARRAY" INT;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE LENGTH OF THE VECTOR TO BE TRANSFORMED;
A: <ARRAY IDENTIFIER>;
"ARRAY"A[1:N,1:N];
ENTRY: THE (NONTRIVIAL ELEMENTS OF THE) TRANSFORMING MATRIX, L, AS PRODUCED BY TFMREAHES MUST BE GIVEN IN THE PART BELOW THE FIRST SUBDIAGONAL OF A, I.E. A[I,J] = L[I,J + 1], FOR I = 3,...,N AND J = 1,...,I - 2;
INT: <ARRAY IDENTIFIER>;
"INTEGER""ARRAY" INT[1:N];
ENTRY: PIVOTAL INDICES DEFINING THE STABILIZING ROW AND COLUMN INTERCHANGES AS PRODUCED BY TFMREAHES;
V: <ARRAY IDENTIFIER>;
"ARRAY"V[1:N];
ENTRY: THE VECTOR TO BE TRANSFORMED;
EXIT: THE TRANSFORMED VECTOR.

PROCEDURES USED:
MATVEC = CP34011.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N SQUARED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE BACK TRANSFORMATION WHICH CORRESPONDS TO WILKINSON'S TRANSFORMATION AS PERFORMED BY TFMREAHES TRANSFORMS A VECTOR, X, INTO THE VECTOR PLX, WHERE PL IS THE TRANSFORMING MATRIX.

SUBSECTION: BAKREAHES2.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" BAKREAHES2(A, N, N1, N2, INT, VEC); "VALUE" N, N1, N2;
"INTEGER" N, N1, N2; "ARRAY" A, VEC; "INTEGER" "ARRAY" INT;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE LENGTH OF THE VECTORS TO BE TRANSFORMED;
N1, N2: <ARITHMETIC EXPRESSION>;
THE COLUMN NUMBERS OF THE FIRST AND LAST VECTOR TO BE TRANSFORMED;
A: <ARRAY IDENTIFIER>;
"ARRAY"A[1:N,1:N];
ENTRY: THE (NONTRIVIAL ELEMENTS OF THE) TRANSFORMING MATRIX, L, AS PRODUCED BY TFMREAHES MUST BE GIVEN IN THE PART BELOW THE FIRST SUBDIAGONAL OF A, I.E. A[I,J] = L[I,J + 1], FOR I = 3,...,N AND J = 1,...,I - 2;
INT: <ARRAY IDENTIFIER>;
"INTEGER""ARRAY" INT[1:N];
ENTRY: PIVOTAL INDICES DEFINING THE STABILIZING ROW AND COLUMN INTERCHANGES AS PRODUCED BY TFMREAHES;
VEC: <ARRAY IDENTIFIER>;
"ARRAY"VEC[1:N,N1:N2];
ENTRY: THE N2 - N1 + 1 VECTORS OF LENGTH N TO BE TRANSFORMED;
EXIT: THE N2 - N1 + 1 VECTORS OF LENGTH N RESULTING FROM THE BACK TRANSFORMATION;

PROCEDURES USED:
TAMVEC = CP34012,
ICHROW = CP34032.

REQUIRED CENTRAL MEMORY:
EXECUTION FIELD LENGTH:
IONAL REAL ARRAY OF LENGTH N IS DECLARED.

RUNNING TIME: ROUGHLY PROPORTIONAL TO (N2 - N1 + 1) * N * N.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
SEE SUBSECTION BAKREAHES1.

REFERENCES:
[1] DEKKER, T. J. AND HOFFMANN, W,
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2,
MATHEMATICAL CENTRE TRACTS 23,
MATHEMATISCH CENTRUM, AMSTERDAM, 1968;

[2] J. H. WILKINSON, THE ALGEBRAIC EIGENVALUE PROBLEM,
CLARENDON PRESS, OXFORD, 1965.

EXAMPLES OF USE:

EXAMPLES OF USE OF TEMREAHES, BAKREAHES1 AND BAKREAHES2 CAN BE FOUND IN THE PROCEDURES FOR CALCULATING EIGENVALUES AND EIGENVECTORS AS DESCRIBED IN SECTION 3.3.1.2.2.

SOURCE TEXT(S) :

```
"CODE" 34170;
    "COMMENT" MCA 2400;
    "PROCEDURE" TEMREAHES(A, N, EM, INT); "VALUE" N; "INTEGER" N;
     "ARRAY" A, EM; "INTEGER" "ARRAY" INT;
    "BEGIN" "INTEGER" I, J, J1, K, L;
        "REAL" S, T, MACHTOL, MACHEPS, NORM;
         "ARRAY" B[0:N - 1];

        "REAL" "PROCEDURE" MATVEC(L, U, I, A, B); "CODE" 34011;
        "REAL" "PROCEDURE" MATMAT(L, U, I, J, A, B); "CODE" 34013;
        "PROCEDURE" ICHCOL(L, U, I, J, A); "CODE" 34031;
        "PROCEDURE" ICHROW(L, U, I, J, A); "CODE" 34032;

        MACHEPS:= EM[0]; NORM:= 0;
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" S:= 0;
            "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO" S:= S + ABS(A[I,J]);
            "IF" S > NORM "THEN" NORM:= S
        "END";
        EM[1]:= NORM; MACHTOL:= NORM * MACHEPS; INT[1]:= 0;
        "FOR" J:= 2 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" J1:= J - 1; L:= 0; S:= MACHTOL;
            "FOR" K:= J + 1 "STEP" 1 "UNTIL" N "DO"
            "BEGIN" T:= ABS(A[K,J1]); "IF" T > S "THEN"
                "BEGIN" L:= K; S:= T "END"
            "END";
            "IF" L ^= 0 "THEN"
            "BEGIN" "IF" ABS(A[J,J1]) < S "THEN"
                "BEGIN" ICHROW(1, N, J, L, A);
                    ICHCOL(1, N, J, L, A)
                "END"
                "ELSE" L:= J; T:= A[J,J1];
                "FOR" K:= J + 1 "STEP" 1 "UNTIL" N "DO"
                A[K,J1]:= A[K,J1] / T
            "END"
            "ELSE"
            "FOR" K:= J + 1 "STEP" 1 "UNTIL" N "DO" A[K,J1]:= 0;
            "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
            B[I - 1]:= A[I,J]:= A[I,J] +
            ("IF" L = 0 "THEN" 0 "ELSE" MATMAT(J + 1, N, I, J1, A, A))-
            MATVEC(1, "IF" J1 < I - 2 "THEN" J1 "ELSE" I - 2, I, A, B);
            INT[J]:= L
        "END"
    "END" TEMREAHES;
        "EOP"
```

```
"CODE" 34171;
    "COMMENT" MCA 2401;
    "PROCEDURE" BAKREAHES1(A, N, INT, V); "VALUE" N; "INTEGER" N;
     "ARRAY" A, V; "INTEGER" "ARRAY" INT;
    "BEGIN" "INTEGER" I, L;
        "REAL" W; "ARRAY" X[1:N];

        "REAL" "PROCEDURE" MATVEC(L, U, I, A, B); "CODE" 34011;

        "FOR" I:= 2 "STEP" 1 "UNTIL" N "DO" X[I - 1]:= V[I];
        "FOR" I:= N "STEP" -1 "UNTIL" 2 "DO"
        "BEGIN" V[I]:= V[I] + MATVEC(1, I - 2, I, A, X);
            L:= INT[I]; "IF" L > I "THEN"
            "BEGIN" W:= V[I]; V[I]:= V[L]; V[L]:= W "END"
        "END"
    "END" BAKREAHES1;
        "EOP"


"CODE" 34172;
    "COMMENT" MCA 2402;
    "PROCEDURE" BAKREAHES2(A, N, N1, N2, INT, VEC); "VALUE" N, N1, N2;
    "INTEGER" N, N1, N2; "ARRAY" A, VEC; "INTEGER" "ARRAY" INT;
    "BEGIN" "INTEGER" I, L, K; "ARRAY" U[1:N];

        "REAL" "PROCEDURE" TAMVEC(L, U, I, A, B); "CODE" 34012;
        "PROCEDURE" ICHROW(L, U, I, J, A); "CODE" 34032;

        "FOR" I:= N "STEP" -1 "UNTIL" 2 "DO"
        "BEGIN" "FOR" K:= I - 2 "STEP" -1 "UNTIL" 1 "DO"
            U[K + 1]:= A[I,K];
            "FOR" K:= N1 "STEP" 1 "UNTIL" N2 "DO"
            VEC[I,K]:= VEC[I,K] + TAMVEC(2 , I - 1, K, VEC, U);
            L:= INT[I]; "IF" L > I "THEN" ICHROW(N1, N2, I, L, VEC)
        "END"
    "END" BAKREAHES2;
        "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

CONTRIBUTORS : H.FIOLET, C.G. VAN DER LAAN.

INSTITUTE : MATHEMATICAL CENTRE.

RECEIVED : 730903.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS THREE PROCEDURES:
A) HSHHRMTRI TRANSFORMS THE HERMITIAN MATRIX M INTO A SIMILAR REAL SYMMETRIC TRIDIAGONAL MATRIX S;
B) BAKHRMTRI PERFORMS A BACK TRANSFORMATION CORRESPONDING TO HSHHRMTRI;
C) HSHHRMTRIVAL DELIVERS THE MAIN DIAGONAL ELEMENTS AND THE SQUARES OF THE CODIAGONAL ELEMENTS OF A HERMITIAN TRIDIAGONAL MATRIX WHICH IS UNITARY SIMILAR WITH A GIVEN HERMITIAN MATRIX.

KEYWORDS :

HERMITIAN MATRIX ,
TRIDIAGONALIZATION ,
COMPLEX HOUSEHOLDER,S TRANSFORMATION .

SUBSECTION : HSHHRMTRI.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :
"PROCEDURE" HSHHRMTRI(A, N, D, B, BB, EM, TR, TI): "VALUE" N;
"INTEGER" N; "ARRAY" A, D, B, BB, EM, TR, TI;

THE MEANING OF THE FORMAL PARAMETERS IS :
A,TR,TI: <ARRAY IDENTIFIER>;
"ARRAY"A[1:N,1:N];
"ARRAY" TR,TI[1:N-1];
ENTRY: THE REAL PART OF THE UPPER TRIANGLE OF THE HERMITIAN MATRIX MUST BE GIVEN IN THE UPPER TRIANGULAR PART OF A (THE ELEMENTS A[I,J], I<=J); THE IMAGINARY PART OF THE STRICT LOWER TRIANGLE OF THE HERMITIAN MATRIX MUST BE GIVEN IN THE STRICT LOWER PART OF A (THE ELEMENTS A[I,J],I>J);
EXIT: DATA FOR THE BACKTRANSFORMATION;

```
N:          <ARITHMETIC EXPRESSION>;
            THE ORDER OF THE MATRIX;
D:          <ARRAY IDENTIFIER>;
            "ARRAY"D[1:N];
            EXIT : THE MAIN DIAGONAL OF THE RESULTING SYMMETRIC
                   TRIDIAGONAL MATRIX;
B:          <ARRAY IDENTIFIER>;
            "ARRAY"B[1:N-1];
            EXIT: THE CODIAGONAL ELEMENTS OF THE RESULTING SYMMETRIC
                  TRIDIAGONAL MATRIX;
BB:         <APRAY IDENTIFIER>;
            "ARRAY"BB[1:N-1];
            EXIT : THE SQUARES OF THE MODULI OF THE CODIAGONAL
                   ELEMENTS OF THE RESULTING SYMMETRIC TRIDIAGONAL
                   MATRIX;
EM:         <ARRAY IDENTIFIER>;
            "ARRAY"EM[0:1];
            ENTRY: EM[0], THE MACHINE PRECISION;
            EXIT:  EM[1], AN ESTIMATE FOR A NORM OF THE ORIGINAL
                          MATRIX.
```

PROCEDURES USED :

```
MATVEC     = CP34011 ,
TAMVEC     = CP34012 ,
MATMAT     = CP34013 ,
TAMMAT     = CP34014 ,
MATTAM     = CP34015 ,
ELMVECCOL  = CP34021 ,
ELMCOLVEC  = CP34022 ,
ELMCOL     = CP34023 ,
ELMROW     = CP34024 ,
ELMVECROW  = CP34026 ,
ELMROWVEC  = CP34027 ,
ELMROWCOL  = CP34028 ,
ELMCOLROW  = CP34029 ,
CARPOL     = CP34344 .
```

RUNNING TIME : PROPORTIONAL TO N CUBED .

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE HSHHRMTRIVAL (THIS SECTION).

SUBSECTION : BAKHRMTRI.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :
"PROCEDURE" BAKHRMTRI(A, N, N1, N2, VECR, VECI, TR, TI):
"VALUE" N, N1, N2; "INTEGER" N, N1, N2;

THE MEANING OF THE FORMAL PARAMETERS IS :

A,TR,TI: <ARRAY IDENTIFIER>;
"ARRAY"A[1:N,1:N];
"ARRAY" TR,TI[1:N-1];
ENTRY: THE DATA FOR THE BACKTRANSFORMATION AS PRODUCED
BY HSHHRMTRI;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE MATRIX OF WHICH THE EIGENVECTORS ARE
CALCULATED;

N1,N2: <ARITHMETIC EXPRESSION>;
THE EIGENVECTORS CORRESPONDING TO THE EIGENVALUES WITH
INDICES N1,...,N2 ARE TO BE TRANSFORMED;

VECR,VECI: <ARRAY IDENTIFIER>;
"ARRAY" VECR,VECI[1:N,N1:N2];
ENTRY:
THE BACK TRANSFORMATION IS PERFORMED ON THE REAL
EIGENVECTORS GIVEN IN THE COLUMNS OF ARRAY VECR;
EXIT:
VECR : REAL PART OF THE TRANSFORMED EIGENVECTORS;
VECI : IMAGINARY PART OF THE TRANSFORMED EIGENVECTORS.

PROCEDURES USED :

MATMAT = CP34013 ,
TAMMAT = CP34014 ,
ELMCOL = CP34023 ,
ELMCOLROW = CP34029 ,
COMMUL = CP34341 ,
COMROWCST = CP34353 .

RUNNING TIME: PROPORTIONAL TO (N2-N1+1)*N**2 .

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE HSHHRMTRIVAL (THIS SECTION).

SUBSECTION : HSHHRMTRIVAL.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :
"PROCEDURE" HSHHRMTRIVAL(A, N, D, BB, EM); "VALUE" N; "INTEGER" N;
"ARRAY" A, D, BB, EM;

THE MEANING OF THE FORMAL PARAMETERS IS :

A: <ARRAY IDENTIFIER>;
"ARRAY"A[1:N,1:N];
ENTRY: THE REAL PART OF THE UPPER TRIANGLE OF THE HERMITIAN MATRIX MUST BE GIVEN IN THE UPPER TRIANGULAR PART OF A (THE ELEMENTS A[I,J], I<=J); THE IMAGINARY PART OF THE STRICT LOWER TRIANGLE OF THE HERMITIAN MATRIX MUST BE GIVEN IN THE STRICT LOWER PART OF A (THE ELEMENTS A[I,J],I>J); THE ELEMENTS OF A ARE ALTERED;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

D: <ARRAY IDENTIFIER>;
"ARRAY"D[1:N];
EXIT: THE MAIN DIAGONAL OF THE RESULTING HERMITIAN TRIDIAGONAL MATRIX;

BB: <ARRAY IDENTIFIER>;
"ARRAY"BB[1:N-1];
EXIT: THE SQUARES OF THE MODULI OF THE CODIAGONAL ELEMENTS OF THE RESULTING HERMITIAN TRIDIAGONAL MATRIX;

EM: <ARRAY IDENTIFIER>;
"ARRAY"EM[0:1];
ENTRY: EM[0],THE MACHINE PRECISION;
EXIT: EM[1], AN ESTIMATE FOR A NORM OF THE ORIGINAL MATRIX;

PROCEDURES USED :

MATVEC = CP34011 ,
TAMVEC = CP34012 ,
MATMAT = CP34013 ,
TAMMAT = CP34014 ,
MATTAM = CP34015 ,
ELMVECCOL = CP34021 ,
ELMCOLVEC = CP34022 ,
ELMCOL = CP34023 ,
ELMROW = CP34024 ,
ELMVECROW = CP34026 ,
ELMROWVEC = CP34027 ,
ELMROWCOL = CP34028 ,
ELMCOLROW = CP34029 .

RUNNING TIME : PROPORTIONAL TO N CUBED .

LANGUAGE : ALGOL 60.

THE FOLLOWING HOLDS FOR THE THREE PROCEDURES :

METHOD AND PERFORMANCE :

HSHHRMTRIVAL TRANSFORMS A HERMITIAN MATRIX INTO A SIMILAR HERMITIAN TRIDIAGONAL MATRIX BY MEANS OF HOUSEHOLDER'S TRANSFORMATION.
HSHHRMTRI TRANSFORMS A HERMITIAN MATRIX INTO A SIMILAR REAL TRIDIAGONAL MATRIX BY MEANS OF HOUSEHOLDER'S TRANSFORMATION FOLLOWED BY A COMPLEX DIAGONAL UNITARY SIMILARITY TRANSFORMATION IN ORDER TO MAKE THE RESULTING TRIDIAGONAL MATRIX REAL SYMMETRIC;
HOUSEHOLDER'S TRANSFORMATION FOR COMPLEX HERMITIAN MATRICES IS A UNITARY SIMILARITY TRANSFORMATION, TRANSFORMING A HERMITIAN MATRIX INTO A SIMILAR COMPLEX TRIDIAGONAL ONE (SEE WILKINSON, 1965, P. 342-343). LET M BE A GIVEN HERMITIAN MATRIX OF ORDER N, WITH REAL PART MR AND IMAGINARY PART MI, P THE TRANSFORMING MATRIX AND T THE RESULTING HERMITIAN TRIDIAGONAL MATRIX. SINCE P IS UNITARY, WE HAVE T = P"MP, WHERE " STANDS FOR CONJUGATING AND TRANSPOSING. THE MATRIX P IS THE PRODUCT OF N-2 HOUSEHOLDER MATRICES, THESE BEING UNITARY HERMITIAN MATRICES OF THE FORM I-U"U/T, WHERE T IS A SCALAR (>0), AND U A COMPLEX VECTOR. THE K-TH HOUSEHOLDER MATRIX, K=1,...,N-2, IS CHOSEN IN SUCH A WAY THAT THE LAST K ELEMENTS OF U VANISH, AND THE DESIRED ZEROS ARE INTRODUCED IN THE (N-K+1)-TH COLUMN AND ROW OF THE MATRIX M. HOWEVER, IF THE EUCLIDIAN NORM OF THE FIRST N-K-1 ELEMENTS OF COLUMN N-K+1 OF THE MATRIX M IS SMALLER THAN THE MACHINE PRECISION TIMES THE INFINITY NORM OF THE MATRIX (NORM(MR) + NORM(MI)), THEN THE K-TH TRANSFORMATION IS SKIPPED (I.E. THE K-TH HOUSEHOLDER MATRIX IS REPLACED BY I).

THE COMPLEX DIAGONAL SIMILARITY TRANSFORMATION D TRANSFORMS THE HERMITIAN TRIDIAGONAL MATRIX T INTO A REAL SYMMETRIC TRIDIAGONAL MATRIX S (MUELLER, 1966). THE DIAGONAL OF D IS CHOSEN IN SUCH A WAY THAT THE CODIAGONAL ELEMENTS OF T ARE TRANSFORMED INTO THEIR ABSOLUTE VALUES.
BAKHRMTRI PERFORMS THE BACK TRANSFORMATION TO REPLACE THE EIGENVECTORS OF THE TRIDIAGONAL SYMMETRIC MATRIX S BY THE EIGENVECTORS OF THE ORIGINAL HERMITIAN MATRIX M. IF X IS AN EIGENVECTOR OF S THEN PDX IS THE CORRESPONDING EIGENVECTOR OF M. STARTING FROM THE VECTOR V=DX, THE VECTOR PDX IS OBTAINED BY SUCCESSIVELY REPLACING V BY THE K-TH HOUSEHOLDER MATRIX TIMES V, FOR K=N-2,...,1. THE RESULTING VECTOR V THEN EQUALS PDX.

REFERENCES :

MUELLER, D.J. (1966),
HOUSEHOLDER,S METHOD FOR COMPLEX MATRICES AND EIGENSYSTEMS OF HERMITIAN MATRICES,
NUMER.MATH., 8, P.72-92;

WILKINSON, J.H. (1965),
THE ALGEBRAIC EIGENVALUE PROBLEM,
CLARENDON PRESS, OXFORD.

EXAMPLE OF USE :

THE PROCEDURES HSHHRMTRIVAL AND BAKHRMTRI ARE USED IN SECTION 3.3.2.1. :
EIGVALHRM AND ORIVALHRM USE HSHHRMTRIVAL,
EIGHRM AND ORIHRM USE BAKHRMTRI .
AS A FORMAL TEST OF THE PROCEDURE HSHHRMTRI, THE FOLLOWING MATRIX WAS USED (SEE GREGORY AND KARNEY, CHAPTER 6, EXAMPLE 6.6) :

3	1	0	+2I
1	3	-2I	0
0	+2I	1	1
-2I	0	1	1

```
"BEGIN"
"COMMENT" GREGORY AND KARNEY,CHAPTER 6, EXAMPLE 6.6;
"PROCEDURE" HSHHRMTRI(A,N,D,B,BB,EM,TR,TI);"CODE" 34363;
"PROCEDURE" INIMAT(LR,UR,LC,UC,A,X);"CODE" 31011;
"REAL" "ARRAY" A[1:4,1:4],D,B,BB[1:4],TR,TI[1:3],EM[0:1];
"INTEGER" I,J;
"PROCEDURE" OUT(S,A,N);
"VALUE" N;"INTEGER" N;"ARRAY" A;"STRING" S;
"BEGIN" "INTEGER" I,J;
    OUTPUT(61,"("10S")",S);
    "FOR" I:=1 "STEP" 1 "UNTIL" N "DO"
    OUTPUT(61,"("+D.3DBB")",A[I]);
    OUTPUT(61,"("/")")
"END" OUT;

INIMAT(1,4,1,4,A,0);
A[1,1]:=A[2,2]:=3;
A[1,2]:=A[3,3]:=A[3,4]:=A[4,4]:=1;
A[3,2]:=2;A[4,1]:=-2;
EM[0]:="-14;
OUTPUT(61,"("""("INITIAL MATRIX GIVEN IN ARRAY A[1:4,1:4]:")",/")");
"FOR" I:=1 "STEP" 1 "UNTIL" 4 "DO"
"BEGIN" "FOR" J:=1 "STEP" 1 "UNTIL" 4 "DO"
    OUTPUT(61,"("-DBBB")",A[I,J]);
    OUTPUT(61,"("/")")
"END";
OUTPUT(61,"("/,"("HSHHRMTRI DELIVERS:")",//")");
HSHHRMTRI(A,4,D,B,BB,EM,TR,TI);
OUT("("D[1:4]: ")",D,4);
OUT("("B[1:3]: ")",B,3);
OUT("("BB[1:3]: ")",BB,3);
OUT("("EM[1]: ")",EM,1);
"END"
```

```
OUTPUT :
INITIAL MATRIX GIVEN IN ARRAY A[1:4,1:4]:
 3    1    0    0
 0    3    0    0
 0    2    1    1
-2    0    0    1

HSHHRMTRI DELIVERS:

D[1:4]:   +3.000  +1.400  +2.600  +1.000
B[1:3]:   +2.236  +0.800  +2.236
BB[1:3]:  +5.000  +0.640  +5.000
EM[1]:    +6.000
```

SOURCE TEXT(S) :

```
"CODE" 34363;
    "PROCEDURE" HSHHRMTRI(A, N, D, B, BB, EM, TR, TI); "VALUE" N;
    "INTEGER" N; "ARRAY" A, D, B, BB, EM, TR, TI;
    "BEGIN" "INTEGER" I, J, J1, JM1, R, RM1;
        "REAL" NRM, W, TOL2, X, AR, AI, MOD, C, S, H, K, T, Q,
        AJR, ARJ, BJ, BBJ;
        "REAL" "PROCEDURE" MATVEC(L,U,I,A,B);"CODE" 34011;
        "REAL" "PROCEDURE" TAMVEC(L,U,I,A,B);"CODE" 34012;
        "REAL" "PROCEDURE" MATMAT(L,U,I,J,A,B);"CODE" 34013;
        "REAL" "PROCEDURE" TAMMAT(L,U,I,J,A,B);"CODE" 34014;
        "REAL" "PROCEDURE" MATTAM(L,U,I,J,A,B);"CODE" 34015;
        "PROCEDURE" ELMVECCOL(L,U,I,A,B,X);"CODE" 34021;
        "PROCEDURE" ELMCOLVEC(L,U,I,A,B,X);"CODE" 34022;
        "PROCEDURE" ELMCOL(L,U,I,J,A,B,X);"CODE" 34023;
        "PROCEDURE" ELMROW(L,U,I,J,A,B,X);"CODE" 34024;
        "PROCEDURE" ELMVECROW(L,U,I,A,B,X);"CODE" 34026;
        "PROCEDURE" ELMROWVEC(L,U,I,A,B,X);"CODE" 34027;
        "PROCEDURE" ELMROWCOL(L,U,I,J,A,B,X);"CODE" 34028;
        "PROCEDURE" ELMCOLROW(L,U,I,J,A,B,X);"CODE" 34029;
        "PROCEDURE" CARPOL(AR,AI,R,C,S);"CODE" 34344;
        NRM:= 0;
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" W:= ABS(A[I,I]);
            "FOR" J:= I - 1 "STEP" - 1 "UNTIL" 1, I + 1 "STEP" 1
            "UNTIL" N "DO" W:= W + ABS(A[I,J]) + ABS(A[J,I]);
            "IF" W > NRM "THEN" NRM:= W
        "END" I;
        TOL2:= (EM[0] * NRM) ** 2; EM[1]:= NRM; R:= N;
        "FOR" RM1:= N - 1 "STEP" - 1 "UNTIL" 1 "DO"
        "BEGIN" X:= TAMMAT(1, R - 2, R, R, A, A) + MATTAM(1, R -
            2, R, R, A, A); AR:= A[RM1,R]; AI:= - A[R,RM1];
            D[R]:= A[R,R]; CARPOL(AR, AI, MOD, C, S);
            "IF" X < TOL2 "THEN"
            "BEGIN" A[R,R]:= - 1; B[RM1]:= MOD;
                BB[RM1]:= MOD * MOD
            "END"
```

```
        "ELSE"
        "BEGIN" H:= MOD * MOD + X; K:= SQRT(H);
            T:= A[R,R]:= H + MOD * K;
            "IF" AR = 0 "AND" AI = 0 "THEN" A[RM1,R]:= K "ELSE"
            "BEGIN" A[RM1,R]:= AR + C * K;
                A[R,RM1]:= - AI - S * K; S:= - S
            "END";
            C:= - C; J:= 1; JM1:= 0;
            "FOR" J1:= 2 "STEP" 1 "UNTIL" R "DO"
            "BEGIN" B[J]:= (TAMMAT(1, J, J, R, A, A) +
                MATMAT(J1, RM1, J, R, A, A) + MATTAM(1,
                JM1, J, R, A, A) - MATMAT(J1, RM1, R, J,
                A, A)) / T;
                BB[J]:= (MATMAT(1, JM1, J, R, A, A) -
                TAMMAT(J1, RM1, J, R, A, A) - MATMAT(1, J,
                R, J, A, A) - MATTAM(J1, RM1, J, R, A, A))
                / T; JM1:= J; J:= J1
            "END" J1;
            Q:= (TAMVEC(1, RM1, R, A, B) - MATVEC(1, RM1,
            R, A, BB)) / T / 2;
            ELMVECCOL(1, RM1, R, B, A, - Q);
            ELMVECROW(1, RM1, R, BB, A, Q); J:= 1;
            "FOR" J1:= 2 "STEP" 1 "UNTIL" R "DO"
            "BEGIN" AJR:= A[J,R]; ARJ:= A[R,J]; BJ:= B[J];
                BBJ:= BB[J];
                ELMROWVEC(J, RM1, J, A, B, - AJR);
                ELMROWVEC(J, RM1, J, A, BB, ARJ);
                ELMROWCOL(J, RM1, J, R, A, A, - BJ);
                ELMROW(J, RM1, J, R, A, A, BBJ);
                ELMCOLVEC(J1, RM1, J, A, B, - ARJ);
                ELMCOLVEC(J1, RM1, J, A, BB, - AJR);
                ELMCOL(J1, RM1, J, R, A, A, BBJ);
                ELMCOLROW(J1, RM1, J, R, A, A, BJ); J:= J1;
            "END" J1;
            BB[RM1]:= H; B[RM1]:= K;
        "END";
        TR[RM1]:= C; TI[RM1]:= S; R:= RM1;
    "END" RM1;
    D[1]:= A[1,1];
"END" HSHHRMTRI;
    "EOP"
```

```
"CODE" 34365;
    "PROCEDURE" BAKHRMTRI(A, N, N1, N2, VECR, VECI, TR, TI);
    "VALUE" N, N1, N2; "INTEGER" N, N1, N2;
     "ARRAY" A, VECR, VECI, TR, TI;
    "BEGIN" "INTEGER" I, J, R, RM1;
        "REAL" C, S, T, QR, QI;
        "REAL" "PROCEDURE" MATMAT(L,U,I,J,A,B);"CODE" 34013;
        "REAL" "PROCEDURE" TAMMAT(L,U,I,J,A,B);"CODE" 34014;
        "PROCEDURE" ELMCOL(L,U,I,J,A,B,X);"CODE" 34023;
        "PROCEDURE" ELMCOLROW(L,U,I,J,A,B,X);"CODE" 34029;
        "PROCEDURE" COMMUL(AR,AI,BR,BI,RR,RI);"CODE" 34341;
        "PROCEDURE" COMROWCST(L,U,I,AR,AI,XR,XI);"CODE" 34353;
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        "FOR" J:= N1 "STEP" 1 "UNTIL" N2 "DO" VECI[I,J]:= 0; C:= 1;
        S:= 0;
        "FOR" J:= N - 1 "STEP" - 1 "UNTIL" 1 "DO"
        "BEGIN" COMMUL(C, S, TR[J], TI[J], C, S);
            COMROWCST(N1, N2, J, VECR, VECI, C, S)
        "END" J;
        RM1:= 2;
        "FOR" R:= 3 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" T:= A[R,R]; "IF" T > 0 "THEN"
            "FOR" J:= N1 "STEP" 1 "UNTIL" N2 "DO"
            "BEGIN" QR:= (TAMMAT(1, RM1, R, J, A, VECR) -
                MATMAT(1, RM1, R, J, A, VECI)) / T;
                QI:= (TAMMAT(1, RM1, R, J, A, VECI) +
                MATMAT(1, RM1, R, J, A, VECR)) / T;
                ELMCOL(1, RM1, J, R, VECR, A, - QR);
                ELMCOLROW(1, RM1, J, R, VECR, A, - QI);
                ELMCOLROW(1, RM1, J, R, VECI, A, QR);
                ELMCOL(1, RM1, J, R, VECI, A, - QI)
            "END";
            RM1:= R;
        "END" R
    "END" BAKHRMTRI;
        "EOP"
```

```
"CODE" 34364;
    "PROCEDURE" HSHHRMTRIVAL(A, N, D, BB, EM); "VALUE" N; "INTEGER" N;
     "ARRAY" A, D, BB, EM;
    "BEGIN" "INTEGER" I, J, J1, JM1, R, RM1;
        "REAL" NRM, W, TOL2, X, AR, AI, H, T, Q, AJR, ARJ, DJ,
        BBJ, MOD2;
        "REAL" "PROCEDURE" MATVEC(L,U,I,A,B);"CODE" 34011;
        "REAL" "PROCEDURE" TAMVEC(L,U,I,A,B);"CODE" 34012;
        "REAL" "PROCEDURE" MATMAT(L,U,I,J,A,B);"CODE" 34013;
        "REAL" "PROCEDURE" TAMMAT(L,U,I,J,A,B);"CODE" 34014;
        "REAL" "PROCEDURE" MATTAM(L,U,I,J,A,B);"CODE" 34015;
        "PROCEDURE" ELMVECCOL(L,U,I,A,B,X);"CODE" 34021;
        "PROCEDURE" ELMCOLVEC(L,U,I,A,B,X);"CODE" 34022;
        "PROCEDURE" ELMCOL(L,U,I,J,A,B,X);"CODE" 34023;
        "PROCEDURE" ELMROW(L,U,I,J,A,B,X);"CODE" 34024;
        "PROCEDURE" ELMVECROW(L,U,I,A,B,X);"CODE" 34026;
        "PROCEDURE" ELMROWVEC(L,U,I,A,B,X);"CODE" 34027;
        "PROCEDURE" ELMROWCOL(L,U,I,J,A,B,X);"CODE" 34028;
        "PROCEDURE" ELMCOLROW(L,U,I,J,A,B,X);"CODE" 34029;
        NRM:= 0;
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" W:= ABS(A[I,I]);
            "FOR" J:= I - 1 "STEP" - 1 "UNTIL" 1, I + 1 "STEP" 1
            "UNTIL" N "DO" W:= W + ABS(A[I,J]) + ABS(A[J,I]);
            "IF" W > NRM "THEN" NRM:= W
        "END" I;
        TOL2:= (EM[0] * NRM) ** 2; EM[1]:= NRM; R:= N;
        "FOR" RM1:= N - 1 "STEP" - 1 "UNTIL" 1 "DO"
        "BEGIN" X:= TAMMAT(1, R - 2, R, R, A, A) + MATTAM(1, R -
            2, R, R, A, A); AR:= A[RM1,R]; AI:= - A[R,RM1];
            D[R]:= A[R,R];
            "IF" X < TOL2 "THEN" BB[RM1]:= AR * AR + AI * AI "ELSE"
            "BEGIN" MOD2:= AR * AR + AI * AI; "IF" MOD2 = 0 "THEN"
                "BEGIN" A[RM1,R]:= SQRT(X); T:= X "END"
                "ELSE"
                "BEGIN" X:= X + MOD2; H:= SQRT(MOD2 * X);
                    T:= X + H; H:= 1 + X / H;
                    A[R,RM1]:= - AI * H; A[RM1,R]:= AR * H;
                "END";
                                                        "COMMENT"
```

```
            J:= 1; JM1:= 0;
            "FOR" J1:= 2 "STEP" 1 "UNTIL" R "DO"
            "BEGIN" D[J]:= (TAMMAT(1, J, J, R, A, A) +
                MATMAT(J1, RM1, J, R, A, A) + MATTAM(1,
                JM1, J, R, A, A) - MATMAT(J1, RM1, R, J,
                A, A)) / T;
                BB[J]:= (MATMAT(1, JM1, J, R, A, A) -
                TAMMAT(J1, RM1, J, R, A, A) - MATMAT(1, J,
                R, J, A, A) - MATTAM(J1, RM1, J, R, A, A))
                / T; JM1:= J; J:= J1
            "END" J1;
            Q:= (TAMVEC(1, RM1, R, A, D) - MATVEC(1, RM1,
            R, A, BB)) / T / 2;
            ELMVECCOL(1, RM1, R, D, A, - Q);
            ELMVECROW(1, RM1, R, BB, A, Q); J:= 1;
            "FOR" J1:= 2 "STEP" 1 "UNTIL" R "DO"

            "BEGIN" AJR:= A[J,R]; ARJ:= A[R,J]; DJ:= D[J];
                BBJ:= BB[J];
                ELMROWVEC(J, RM1, J, A, D, - AJR);
                ELMROWVEC(J, RM1, J, A, BB, ARJ);
                ELMROWCOL(J, RM1, J, R, A, A, - DJ);
                ELMROW(J, RM1, J, R, A, A, BBJ);
                ELMCOLVEC(J1, RM1, J, A, D, - ARJ);
                ELMCOLVEC(J1, RM1, J, A, BB, - AJR);
                ELMCOL(J1, RM1, J, R, A, A, BBJ);
                ELMCOLROW(J1, RM1, J, R, A, A, DJ); J:= J1;
            "END" J1;
            BB[RM1]:= X;
        "END";
        R:= RM1;
    "END" RM1;
    D[1]:= A[1,1];
"END" HSHHRMTRIVAL;
    "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

CONTRIBUTORS : H.FIOLET, C.G. VAN DER LAAN.

INSTITUTE : MATHEMATICAL CENTRE.

RECEIVED: 731016.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS THE PROCEDURES HSHCOMHES AND BAKCOMHES.
HSHCOMHES TRANSFORMS A COMPLEX MATRIX BY MEANS OF HOUSEHOLDER'S TRANSFORMATION FOLLOWED BY A COMPLEX DIAGONAL TRANSFORMATION INTO A SIMILAR UNITARY UPPER-HESSENBERG MATRIX WITH A REAL NONNEGATIVE SUBDIAGONAL.
BAKCOMHES PERFORMS THE CORRESPONDING BACK TRANSFORMATION.

KEYWORDS:

COMPLEX EIGENPROBLEM,
REDUCTION HESSENBERG FORM,
HOUSEHOLDER'S TRANSFORMATION.

SUBSECTION: HSHCOMHES.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" HSHCOMHES(AR, AI, N, EM, B, TR, TI, DEL); "VALUE" N;
"INTEGER" N; "ARRAY" AR, AI, EM, B, TR, TI, DEL;

THE MEANING OF THE FORMAL PARAMETERS IS:
AR,AI: <ARRAY IDENTIFIER>;
"ARRAY" AR,AI[1:N,1:N];
ENTRY:
THE REAL PART AND THE IMAGINARY PART OF THE MATRIX TO BE TRANSFORMED MUST BE GIVEN IN THE ARRAYS AR AND AI, RESPECTIVELY;
EXIT:
THE REAL PART AND THE IMAGINARY PART OF THE UPPER TRIANGLE OF THE RESULTING UPPER-HESSENBERG MATRIX ARE DELIVERED IN THE CORRESPONDING PARTS OF THE ARRAYS AR AND AI, RESPECTIVELY; DATA FOR THE HOUSEHOLDER BACK-TRANSFORMATION ARE DELIVERED IN THE STRICT LOWER TRIANGLES OF THE ARRAYS AR AND AI;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

EM: <ARRAY IDENTIFIER>;
"ARRAY"EM[0:1];
ENTRY:
EM[0]: THE MACHINE PRECISION;
EM[1]: AN ESTIMATE OF THE NORM OF THE COMPLEX MATRIX;
(OR, E.G. THE SUM OF THE INFINITY NORMS OF THE REAL
(PART AND IMAGINARY PART OF THE MATRIX);

B: <ARRAY IDENTIFIER>;
"ARRAY"B[1:N-1];
EXIT:
THE REAL NONNEGATIVE SUBDIAGONAL OF THE RESULTING
UPPER-HESSENBERG MATRIX;

TR,TI: <ARRAY IDENTIFIER>;
"ARRAY" TR,TI[1:N];
EXIT:
THE REAL PART AND THE IMAGINARY PART OF THE DIAGONAL
ELEMENTS OF A DIAGONAL SIMILARITY TRANSFORMATION ARE
DELIVERED IN THE ARRAYS TR AND TI, RESPECTIVELY: BY THIS
INFORMATION THE COMPLEX UPPER-HESSENBERG MATRIX IS
TRANSFORMED INTO A UPPER-HESSENBERG MATRIX WITH A REAL
SUBDIAGONAL;

DEL: <ARRAY IDENTIFIER>;
"ARRAY"DEL[1:N-2];
EXIT:
INFORMATION CONCERNING THE SEQUENCE OF HOUSEHOLDER
MATRICES.

PROCEDURES USED:

HSHCOMCOL = CP34355,
MATMAT = CP34013,
ELMROWCOL = CP34028,
HSHCOMPRD = CP34356,
CARPOL = CP34344,
COMMUL = CP34341.
COMCOLCST = CP34352,
COMROWCST = CP34353.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE BAKCOMHES (THIS SECTION).

SUBSECTION: BAKCOMHES.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" BAKCOMHES(AR, AI, TR, TI, DEL, VR, VI, N, N1, N2);
"VALUE" N, N1, N2; "INTEGER" N, N1, N2;
"ARRAY" AR, AI, TR, TI, DEL, VR, VI;

THE MEANING OF THE FORMAL PARAMETERS IS:

AR,AI,TR,TI,DEL:
<ARRAY IDENTIFIER>;
"ARRAY" AR,AI[1:N,1:N];
"ARRAY" TR,TI[1:N];
"ARRAY"DEL[1:N-2];
ENTRY: THE DATA FOR THE BACKTRANSFORMATION AS PRODUCED BY HSHCOMHES;

VR,VI:
<ARRAY IDENTIFIER>;
"ARRAY" VR,VI[1:N,N1:N2];
ENTRY:
THE BACK TRANSFORMATION IS PERFORMED ON THE EIGENVECTORS WITH THE REAL PARTS GIVEN IN ARRAY VR AND THE IMAGINARY PARTS GIVEN IN ARRAY VI;
EXIT:
THE REAL PARTS AND IMAGINARY PARTS OF THE RESULTING EIGENVECTORS ARE DELIVERED IN THE COLUMNS OF THE ARRAYS VR AND VI, RESPECTIVELY;

N:
<ARITHMETIC EXPRESSION>;
THE ORDER OF THE MATRIX OF WHICH THE EIGENVECTORS ARE CALCULATED;

N1,N2:
<ARITHMETIC EXPRESSION>;
THE EIGENVECTORS CORRESPONDING TO THE EIGENVALUES WITH INDICES N1,...,N2 ARE TO BE TRANSFORMED;

PROCEDURES USED:

COMROWCST = CP34353,
HSHCOMPRD = CP34356.

RUNNING TIME: PROPORTIONAL TO (N2-N1) * N**2.

LANGUAGE : ALGOL 60.

THE FOLLOWING HOLDS FOR BOTH PROCEDURES:

METHOD AND PERFORMANCE:

HSHCOMHES:
HOUSEHOLDER'S TRANSFORMATION (FOR COMPLEX MATRICES) IS A UNITARY SIMILARITY TRANSFORMATION, WHICH TRANSFORMS A COMPLEX MATRIX INTO A SIMILAR UPPER-HESSENBERG MATRIX (SEE WILKINSON, 1965, P. 347-349). LET M BE A GIVEN COMPLEX MATRIX OF ORDER N, P THE TRANSFORMING MATRIX AND H THE RESULTING UPPER-HESSENBERG MATRIX. SINCE P IS UNITARY, WE THEN HAVE H = P''MP, WHERE '' STANDS FOR CONJUGATING AND TRANSPOSING. THE MATRIX P IS THE PRODUCT OF N-2 HOUSEHOLDER MATRICES. THESE BEING UNITARY HERMITEAN MATRICES OF THE FORM I - UU''/T, WHERE T IS A SCALAR (>0), AND U A COMPLEX VECTOR. THE R-TH HOUSEHOLDER MATRIX, R=1,...,N-2, IS CHOSEN IN SUCH A WAY THAT THE FIRST R ELEMENTS OF U VANISH, AND THE DESIRED ZEROS ARE INTRODUCED IN THE LAST N-R-1 ELEMENTS OF THE R-TH COLUMN OF THE MATRIX M. HOWEVER, IF THE EUCLIDEAN NORM OF THE LAST N-R-1 ELEMENTS OF COLUMN R OF THE MATRIX M IS SMALLER THAN THE MACHINE PRECISION TIMES A NORM OF THE MATRIX THEN THE R-TH TRANSFORMATION IS SKIPPED (I.E. THE R-TH HOUSEHOLDER MATRIX IS REPLACED BY I). THE COMPLEX DIAGONAL SIMILARITY TRANSFORMATION D TRANSFORMS THE UPPER-HESSENBERG MATRIX H INTO AN UPPER-HESSENBERG MATRIX HR, WITH REAL NONNEGATIVE ELEMENTS. THE DIAGONAL OF D IS CHOSEN IN SUCH A WAY THAT SUBDIAGONAL ELEMENTS OF H ARE TRANSFORMED INTO THEIR ABSOLUTE VALUES (SEE MUELLER, 1966).
BAKCOMHES:
THE BACK TRANSFORMATION TRANSFORMS A COMPLEX VECTOR X INTO THE COMPLEX VECTOR PDX. IF X IS AN EIGENVECTOR OF H THEN PDX IS THE CORRESPONDING EIGENVECTOR OF M. STARTING FROM THE VECTOR V=DX, THE VECTOR PDX IS OBTAINED BY SUCCESSIVELY REPLACING V BY THE R-TH HOUSEHOLDER MATRIX TIMES V, FOR R=N-2,...,1. THE RESULTING VECTOR THEN EQUALS PDX.

REFERENCES:

MUELLER, D.J. (1966),
HOUSEHOLDER'S METHOD FOR COMPLEX MATRICES AND EIGENSYSTEMS OF HERMITIAN MATRICES,
NUMER.MATH., 8, P.72-92;

WILKINSON, J.H. (1965),
THE ALGEBRAIC EIGENVALUE PROBLEM,
CLARENDON PRESS, OXFORD;

EXAMPLE OF USE:

HSHCOMHES IS USED IN THE PROCEDURES EIGVALCOM AND EIGCOM.
BAKCOMHES IS USED IN THE PROCEDURE EIGCOM.
(SEE SECTION 3.3.2.2.2.).

SOURCE TEXT(S) :

```
"CODE" 34366;
    "PROCEDURE" HSHCOMHES(AR, AI, N, EM, B, TR, TI, DEL); "VALUE" N;
    "INTEGER" N; "ARRAY" AR, AI, EM, B, TR, TI, DEL;
    "BEGIN" "INTEGER" R, RM1, I, J, NM1;
        "REAL" TOL, T, XR, XI;
        "REAL" "PROCEDURE" MATMAT(L,U,I,J,A,B);"CODE" 34013;
        "PROCEDURE" ELMROWCOL(L,U,I,J,A,B,X);"CODE" 34028;
        "PROCEDURE" HSHCOMPRD(I,II,L,U,J,AR,AI,BR,BI,T);"CODE" 34356;
        "PROCEDURE" COMCOLCST(L,U,J,AR,AI,XR,XI);"CODE" 34352;
        "PROCEDURE" COMROWCST(L,U,I,AR,AI,XR,XI);"CODE" 34353;
        "PROCEDURE" CARPOL(AR,AI,R,C,S);"CODE" 34344;
        "PROCEDURE" COMMUL(AR,AI,BR,BI,RR,RI);"CODE" 34341;
        "BOOLEAN" "PROCEDURE" HSHCOMCOL(L,U,J,AR,AI,TOL,K,C,S,T);
        "CODE" 34355;
        NM1:= N - 1; TOL:= (EM[0] * EM[1]) ** 2; RM1:= 1;
        "FOR" R:= 2 "STEP" 1 "UNTIL" NM1 "DO"
        "BEGIN" "IF" HSHCOMCOL(R, N, RM1, AR, AI, TOL, B[RM1],
            TR[R], TI[R], T) "THEN"
            "BEGIN" "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
                "BEGIN" XR:= (MATMAT(R, N, I, RM1, AI, AI) -
                    MATMAT(R, N, I, RM1, AR, AR)) / T;
                    XI:= ( - MATMAT(R, N, I, RM1, AR, AI) -
                    MATMAT(R, N, I, RM1, AI, AR)) / T;
                    ELMROWCOL(R, N, I, RM1, AR, AR, XR);
                    ELMROWCOL(R, N, I, RM1, AR, AI, XI);
                    ELMROWCOL(R, N, I, RM1, AI, AR, XI);
                    ELMROWCOL(R, N, I, RM1, AI, AI, - XR)
                "END";
                HSHCOMPRD(R, N, R, N, RM1, AR, AI, AR, AI, T);
            "END";
            DEL[RM1]:= T; RM1:= R
        "END" FORR;
        "IF" N > 1 "THEN" CARPOL(AR[N,NM1], AI[N,NM1], B[NM1],
        TR[N], TI[N]); RM1:= 1; TR[1]:= 1; TI[1]:= 0;
        "FOR" R:= 2 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" COMMUL(TR[RM1], TI[RM1], TR[R], TI[R], TR[R],
            TI[R]); COMCOLCST(1, RM1, R, AR, AI, TR[R], TI[R]);
            COMROWCST(R + 1, N, R, AR, AI, TR[R], - TI[R]);
            RM1:= R
        "END";
    "END" HSHCOMHES;
        "EOP"
```

```
"CODE" 34367;
    "PROCEDURE" BAKCOMHES(AR, AI, TR, TI, DEL, VR, VI, N, N1, N2);
    "VALUE" N, N1, N2; "INTEGER" N, N1, N2;
     "ARRAY" AR, AI, TR, TI, DEL, VR, VI;
    "BEGIN" "INTEGER" I, R, RM1;
        "REAL" H;
        "PROCEDURE" HSHCOMPRD(I,II,L,U,J,AR,AI,BR,BI,T);"CODE" 34356;
        "PROCEDURE" COMROWCST(L,U,I,AR,AI,XR,XI);"CODE" 34353;
        "FOR" I:= 2 "STEP" 1 "UNTIL" N "DO" COMROWCST(N1, N2, I, VR,
        VI, TR[I], TI[I]); R:= N - 1;
        "FOR" RM1:= N - 2 "STEP" - 1 "UNTIL" 1 "DO"
        "BEGIN" H:= DEL[RM1];
            "IF" H > 0 "THEN" HSHCOMPRD(R, N, N1, N2, RM1, VR, VI,
            AR, AI, H); R:= RM1
        "END"
    "END" BAKCOMHES;
        "EOP"
```

AUTHOR : D.T.WINTER

INSTITUTE : MATHEMATICAL CENTRE

RECEIVED : 731217

BRIEF DESCRIPTION :

THIS SECTION CONTAINS THREE PROCEDURES :

1. HSHREABID.
   THIS PROCEDURE TRANSFORMS A GIVEN MATRIX TO BIDIAGONAL FORM, BY PREMULTIPLYING AND POSTMULTIPLYING THE GIVEN MATRIX WITH ORTHOGONAL MATRICES.
2. PSTTFMMAT.
   THIS PROCEDURE CALCULATES THE POSTMULTIPLYING MATRIX FROM THE DATA GENERATED BY HSHREABID.
3. PRETFMMAT.
   THIS PROCEDURE CALCULATES THE PREMULTIPLYING MATRIX FROM THE DATA GENERATED BY HSHREABID.

KEYWORDS :
HOUSEHOLDER'S TRANSFORMATION
BIDIAGONALISATION

SUBSECTION : HSHREABID

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE IS :

```
"PROCEDURE" HSHREABID(A, M, N, D, B, EM);
"VALUE" M, N; "INTEGER" M, N; "ARRAY" A, D, B, EM;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:M,1:N];
ENTRY: THE GIVEN MATRIX;
EXIT: DATA CONCERNING THE PREMULTIPLYING AND POSTMULTIPLYING MATRICES;

M: <ARITHMETIC EXPRESSION>;
THE NUMBER OF ROWS OF THE GIVEN MATRIX;

N: <ARITHMETIC EXPRESSION>;
THE NUMBER OF COLUMNS OF THE GIVEN MATRIX,
N SHOULD SATISFY N <= M;

D: <ARRAY IDENTIFIER>;
"ARRAY"D[1:N];
EXIT: THE DIAGONAL OF THE BIDIAGONAL MATRIX;

B: <ARRAY IDENTIFIER>;
"ARRAY"B[1:N];
EXIT: THE SUPERDIAGONAL OF THE BIDIAGONAL MATRIX IS DELIVERED IN B[1:N-1];

EM: <ARRAY IDENTIFIER>;
"ARRAY"EM[0:1];
ENTRY: EM[0]: THE MACHINE-PRECISION;
EXIT: EM[1]: THE INFINITY NORM OF THE GIVEN MATRIX.

PROCEDURES USED :
TAMMAT = CP34014
MATTAM = CP34015
ELMCOL = CP34023
ELMROW = CP34024

RUNNING TIME :
RUNNING TIME IS ROUGHLY PROPORTIONAL TO (M + N) * N * N

METHOD AND PERFORMANCE :
LET US ASSUME A GIVEN MATRIX A[1:M , 1:N], WITH M >= N. FIRSTLY WE PREMULTIPLY A WITH A HOUSEHOLDER MATRIX, CHOSEN IN SUCH A WAY THAT THE FIRST COLUMN OF THE RESULTING MATRIX A' IS ZERO WITH THE EXCEPTION OF THE FIRST ELEMENT. SECONDLY WE POSTMULTIPLY A' WITH A HOUSEHOLDER MATRIX SO THAT THE FIRST ROW OF THE RESULTING MATRIX IS ZERO WITH THE EXCEPTION OF THE FIRST TWO ELEMENTS. NOW WE REMOVE THE FIRST ROW AND COLUMN, AND REPEAT THIS PROCESS UNTIL THE MATRIX IS TOTALLY TRANSFORMED TO BIDIAGONAL FORM. THIS PROCEDURE IS A REWRITING OF A PART OF THE PROCEDURE SVD PUBLISHED BY G.H.GOLUB AND C.REINSCH[1]. HOWEVER IN CONTRAST TO THEIR PROCEDURE, HERE WE SKIP A TRANSFORMATION IF THE COLUMN OR ROW ON WHICH OUR ATTENTION IS FOCUSSED IS ALREADY (NEARLY) IN THE DESIRED FORM, I.E. IF THE SUM OF THE SQUARES OF THE ELEMENTS THAT OUGHT TO BE ZERO IS SMALLER THAN A CERTAIN CONSTANT, IN SVD THE TRANSFORMATION IS SKIPPED ONLY IF THE NORM OF THE FULL ROW OR COLUMN IS SMALL ENOUGH. OUR WAY SEEMS TO GIVE BETTER RESULTS, AS SOME ILL-DEFINED TRANSFORMATIONS ARE SKIPPED. MOREOVER, IF A TRANSFORMATION IS SKIPPED, WE DO NOT STORE A ZERO IN THE DIAGONAL OR SUPERDIAGONAL, BUT WE STORE THE VALUE THAT WOULD HAVE BEEN FOUND IF THE COLUMN OR ROW WAS IN THE DESIRED FORM ALREADY.

LANGUAGE : ALGOL-60

SUBSECTION : PSTTFMMAT

CALLING SEQUENCE :
THE HEADING OF THE PROCEDURE IS :
"PROCEDURE" PSTTFMMAT(A, N, V, B);
"VALUE" N; "INTEGER" N; "ARRAY" A, V, B;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARRAY IDENTIFIER>;
"ARRAY"A[1:N,1:N];
THE DATA CONCERNING THE POSTMULTIPLYING MATRIX, AS GENERATED BY HSHREABID;
N: <ARITHMETIC EXPRESSION>;
THE NUMBER OF COLUMNS AND ROWS OF A;
V: <ARRAY IDENTIFIER>;
"ARRAY"V[1:N,1:N];
EXIT: THE POSTMULTIPLYING MATRIX;
B: <ARRAY IDENTIFIER>;
"ARRAY"B[1:N];
THE SUPERDIAGONAL AS GENERATED BY HSHREABID.

PROCEDURES USED :
MATMAT = CP34013
ELMCOL = CP34023

RUNNING TIME :
THE RUNNING TIME IS ABOUT PROPORTIONAL TO N ** 3

LANGUAGE : ALGOL 60

SUBSECTION : PRETFMMAT

CALLING SEQUENCE :
THE HEADING OF THE PROCEDURE IS :
"PROCEDURE" PRETFMMAT(A, M, N, D);
"VALUE" M, N; "INTEGER" M, N; "ARRAY" A, D;

THE MEANING OF THE FORMAL PARAMETERS IS :
A: <ARRAY IDENTIFIER>;
"ARRAY"A[1:M,1:N];
ENTRY: THE DATA CONCERNING THE PREMULTIPLYING MATRIX AS GENERATED BY HSHREABID
EXIT : THE PREMULTIPLYING MATRIX.
M: <ARITHMETIC EXPRESSION>;
THE NUMBER OF ROWS OF A.
N: <ARITHMETIC EXPRESSION>;
THE NUMBER OF COLUMNS OF A, N SHOULD SATISFY N <= M.
D: <ARRAY IDENTIFIER>;
"ARRAY"D[1:N];
THE DIAGONAL AS GENERATED BY HSHREABID.

PROCEDURES USED :
TAMMAT = CP34014
ELMCOL = CP34023

RUNNING TIME :
THE RUNNING TIME IS ABOUT PROPORTIONAL TO M * N * N

LANGUAGE : ALGOL-60

REFERENCES :
[1] WILKINSON, J.H. AND C.REINSCH
HANDBOOK FOR AUTOMATIC COMPUTATION, VOL. 2
LINEAR ALGEBRA
HEIDELBERG (1971)

EXAMPLE OF USE :

FOR AN EXAMPLE OF USE ONE IS REFERRED TO SECTION 3.5.1.2

SOURCE TEXT(S) :

```
"CODE" 34260;
"PROCEDURE" HSHREABID(A, M, N, D, B, EM);
"VALUE" M, N; "INTEGER" M, N; "ARRAY" A, D, B, EM;
"BEGIN" "INTEGER" I, J, I1;
    "REAL" NORM, MACHTOL, W, S, F, G, H;

    "REAL" "PROCEDURE" TAMMAT(L, U, I, J, A, B);
    "VALUE" L, U, I, J; "INTEGER" L, U, I, J; "ARRAY" A, B;
    "CODE" 34014;
    "REAL" "PROCEDURE" MATTAM(L, U, I, J, A, B);
    "VALUE" L, U, I, J; "ARRAY" A, B;
    "CODE" 34015;

    "PROCEDURE" ELMCOL(L, U, I, J, A, B, X);
    "VALUE" L, U, I, J, X; "INTEGER" L, U, I, J; "REAL" X;
     "ARRAY" A, B;
    "CODE" 34023;
    "PROCEDURE" ELMROW(L, U, I, J, A, B, X);
    "VALUE" L, U, I, J, X; "INTEGER" L, U, I, J; "REAL" X;
     "ARRAY" A, B;
    "CODE" 34024;

    NORM:= 0;
    "FOR" I:= 1 "STEP" 1 "UNTIL" M "DO"
    "BEGIN" W:= 0;
        "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO" W:= ABS(A[I,J]) + W;
        "IF" W > NORM "THEN" NORM:= W
    "END";
    MACHTOL:= EM[0] * NORM; EM[1]:= NORM;
    "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
    "BEGIN" I1:= I + 1; S:= TAMMAT(I1, M, I, I, A, A);
        "IF" S < MACHTOL "THEN" D[I]:= A[I,I] "ELSE"
        "BEGIN" F:= A[I,I]; S:= F * F + S;
            D[I]:= G:= "IF" F < 0 "THEN" SQRT(S) "ELSE" - SQRT(S);
            H:= F * G - S; A[I,I]:= F - G;
            "FOR" J:= I1 "STEP" 1 "UNTIL" N "DO"
            ELMCOL(I, M, J, I, A, A, TAMMAT(I, M, I, J, A, A) / H)
        "END";
        "IF" I < N "THEN"
        "BEGIN" S:= MATTAM(I1 + 1, N, I, I, A, A);
            "IF" S < MACHTOL "THEN" B[I]:= A[I,I1] "ELSE"
            "BEGIN" F:= A[I,I1]; S:= F * F + S;
                B[I]:= G:= "IF" F < 0 "THEN" SQRT(S) "ELSE" - SQRT(S);
                H:= F * G - S; A[I,I1]:= F - G;
                "FOR" J:= I1 "STEP" 1 "UNTIL" M "DO"
                ELMROW(I1, N, J, I, A, A, MATTAM(I1, N, I, J, A, A) /
                H)
            "END"
        "END"
    "END"
"END" HSHREABID;
        "EOP"
```

```
"CODE" 34261;
"PROCEDURE" PSTTFMMAT(A, N, V, B);
"VALUE" N; "INTEGER" N; "ARRAY" A, V, B;
"BEGIN" "INTEGER" I, I1, J;
    "REAL" H;
    "REAL" "PROCEDURE" MATMAT(L, U, I, J, A, B);
    "VALUE" L, U, I, J; "INTEGER" L, U, I, J; "ARRAY" A, B;
    "CODE" 34013;
    "PROCEDURE" ELMCOL(L, U, I, J, A, B, X);
    "VALUE" L, U, I, J, X; "INTEGER" L, U, I, J; "REAL" X;
     "ARRAY" A, B;
    "CODE" 34023;

    I1:= N; V[N,N]:= 1;
    "FOR" I:= N - 1 "STEP" - 1 "UNTIL" 1 "DO"
    "BEGIN" H:= B[I] * A[I,I1]; "IF" H < 0 "THEN"
        "BEGIN" "FOR" J:= I1 "STEP" 1 "UNTIL" N "DO" V[J,I]:= A[I,J] /
            H;
            "FOR" J:= I1 "STEP" 1 "UNTIL" N "DO"
            ELMCOL(I1, N, J, I, V, V, MATMAT(I1, N, I, J, A, V))
        "END";
        "FOR" J:= I1 "STEP" 1 "UNTIL" N "DO" V[I,J]:= V[J,I]:= 0;
        V[I,I]:= 1; I1:= I
    "END"
"END" PSTTFMMAT;
        "EOP"

"CODE" 34262;
"PROCEDURE" PRETFMMAT(A, M, N, D);
"VALUE" M, N; "INTEGER" M, N; "ARRAY" A, D;
"BEGIN" "INTEGER" I, I1, J;
    "REAL" G, H;
    "REAL" "PROCEDURE" TAMMAT(L, U, I, J, A, B);
    "VALUE" L, U, I, J; "INTEGER" L, U, I, J; "ARRAY" A, B;
    "CODE" 34014;
    "PROCEDURE" ELMCOL(L, U, I, J, A, B, X);
    "VALUE" L, U, I, J, X; "INTEGER" L, U, I, J; "REAL" X;
     "ARRAY" A, B;
    "CODE" 34023;

    "FOR" I:= N "STEP" - 1 "UNTIL" 1 "DO"
    "BEGIN" I1:= I + 1; G:= D[I]; H:= G * A[I,I];
        "FOR" J:= I1 "STEP" 1 "UNTIL" N "DO" A[I,J]:= 0;
        "IF" H < 0 "THEN"
        "BEGIN" "FOR" J:= I1 "STEP" 1 "UNTIL" N "DO"
            ELMCOL(I, M, J, I, A, A, TAMMAT(I1, M, I, J, A, A) / H);
            "FOR" J:= I "STEP" 1 "UNTIL" M "DO" A[J,I]:= A[J,I] / G
        "END"
        "ELSE"
        "FOR" J:= I "STEP" 1 "UNTIL" M "DO" A[J,I]:= 0;
        A[I,I]:= A[I,I] + 1
    "END"
"END" PRETFMMAT;
        "EOP"
```

AUTHORS: T.J.DEKKER AND W.HOFFMANN.

CONTRIBUTORS: W.HOFFMANN, J.G.VERWER.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730716.

BRIEF DESCRIPTION:
THIS SECTION CONTAINS FOUR PROCEDURES FOR CALCULATING EIGENVALUES OR EIGENVECTORS OF A SYMMETRIC TRIDIAGONAL MATRIX.
VALSYMTRI CALCULATES ALL, OR SOME CONSECUTIVE, EIGENVALUES OF A SYMMETRIC TRIDIAGONAL MATRIX BY MEANS OF LINEAR INTERPOLATION USING A STURM SEQUENCE;
VECSYMTRI CALCULATES THE CORRESPONDING EIGENVECTORS BY MEANS OF INVERSE ITERATION.
QRIVALSYMTRI CALCULATES ALL EIGENVALUES OF A SYMMETRIC TRIDIAGONAL MATRIX BY MEANS OF QR ITERATION;
QRISYMTRI CALCULATES THE EIGENVECTORS AS WELL.
WHEN ALL EIGENVALUES HAVE TO BE CALCULATED, QRIVALSYMTRI IS PREFERABLE WITH RESPECT TO THE RUNNING TIME; WHEN THE EIGENVECTORS ALSO HAVE TO BE CALCULATED, INVERSE ITERATION IS PREFERABLE.

KEYWORDS:
EIGENVALUES,
EIGENVECTORS,
TRIDIAGONAL MATRIX,
STURM-SEQUENCE,
INVERSE ITERATION,
QR ITERATION.

SUBSECTION: VALSYMTRI.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" VALSYMTRI(D, BB, N, N1, N2, VAL, EM);
"VALUE" N, N1, N2; "INTEGER" N, N1, N2;
"ARRAY" D, BB, VAL, EM;

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

D: <ARRAY IDENTIFIER>;
"ARRAY" D[1:N];
ENTRY: THE MAIN DIAGONAL OF THE SYMMETRIC TRIDIAGONAL MATRIX;

BB: <ARRAY IDENTIFIER>;
"ARRAY" BB[1:N-1];
ENTRY: THE SQUARES OF THE CODIAGONAL ELEMENTS OF THE SYMMETRIC TRIDIAGONAL MATRIX;

N1,N2: <ARITHMETIC EXPRESSION>;
THE SERIAL NUMBER OF THE FIRST AND LAST EIGENVALUE TO BE CALCULATED, RESPECTIVELY;

VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[N1:N2];
EXIT: THE N2-N1+1 CALCULATED CONSECUTIVE EIGENVALUES IN NONINCREASING ORDER;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:3];
ENTRY: EM[0], THE MACHINE PRECISION,
EM[1], AN UPPERBOUND FOR THE MODULI OF THE EIGENVALUES OF THE GIVEN MATRIX,
EM[2], A RELATIVE TOLERANCE FOR THE EIGENVALUES;
EXIT: EM[3], THE TOTAL NUMBER OF ITERATIONS USED FOR CALCULATING THE EIGENVALUES.

PROCEDURES USED:
ZERO1N = CP34150.

RUNNING TIME:
DEPENDS STRONGLY ON THE DISTANCE OF SUCCESSIVE EIGENVALUES.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
LET T DENOTE THE GIVEN SYMMETRIC TRIDIAGONAL MATRIX OF ORDER N AND I THE IDENTITY MATRIX. THE EIGENVALUES OF T ARE THE ZEROES OF THE N-TH DEGREE POLYNOMIAL P(N,X) = DET(T - X*I). INSTEAD OF SEARCHING FOR THE ZEROES OF P(N,X) WE LOOK FOR THE ZEROES OF THE FUNCTION F(N,X) = P(N,X) / P(N-1,X). MAINTAINING A LOWER BOUND FOR ABS(P(N-1,X)) WE DO AVOID OVERFLOW OF THE REAL NUMBER CAPACITY IN THE COMPUTATION OF F(N,X). THIS FUNCTION CAN BE CALCULATED AS FOLLOWS:

```
        F(1,X) = D[1] - X,
        F(K,X) = D[K] - X - BB[K-1] /
        ("IF" ABS(F(K-1,X)) > MACHTOL "THEN" F(K-1,X)
        "ELSE" "IF" F(K-1,X) <= 0 "THEN" -MACHTOL
        "ELSE" MACHTOL),  K = 2, . . . ,N,
```

WHERE MACHTOL EQUALS EM[0] * EM[1].
USING THE STURM SEQUENCE PROPERTY OF (F(K,X)), K=1,2,...,N, WE CAN LOCATE THE DESIRED EIGENVALUES BY MEANS OF THE PROCEDURE ZERDIN (SECTION 5.1.1.1). FOR FURTHER DETAILS SEE REF[1], REF[2].

SUBSECTION: VECSYMTRI.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:

```
"PROCEDURE" VECSYMTRI(D, B, N, N1, N2, VAL, VEC, EM);
"VALUE" N, N1, N2; "INTEGER" N, N1, N2;
"ARRAY" D, B, VAL, VEC, EM;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

D: <ARRAY IDENTIFIER>;
"ARRAY" D[1:N],
ENTRY: THE MAIN DIAGONAL OF THE SYMMETRIC TRIDIAGONAL MATRIX;

B: <ARRAY IDENTIFIER>;
"ARRAY" B[1:N];
ENTRY: THE CODIAGONAL OF THE SYMMETRIC TRIDIAGONAL MATRIX FOLLOWED BY AN ADDITIONAL ELEMENT 0;

N1, N2: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE ARRAY VAL (SEE ALSO METHOD AND PERFORMANCE);

VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[N1:N2];
ENTRY: A ROW OF NONINCREASING EIGENVALUES AS DELIVERED BY VALSYMTRI;

VEC: <ARRAY IDENTIFIER>;
"ARRAY" VEC[1:N,N1:N2];
EXIT: THE EIGENVECTORS CORRESPONDING WITH THE GIVEN EIGENVALUES (SEE ALSO METHOD AND PERFORMANCE);

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:9];
ENTRY: EM[0], THE MACHINE PRECISION,
EM[1], A NORM OF THE GIVEN MATRIX,
EM[4], THE ORTHOGONALISATION PARAMETER (SEE ALSO METHOD AND PERFORMANCE),
EM[6], THE RELATIVE TOLERANCE FOR THE EIGENVECTORS,
EM[8], THE MAXIMUM NUMBER OF ITERATIONS ALLOWED FOR THE CALCULATION OF EACH EIGENVECTOR;

EXIT: EM[5], THE NUMBER OF EIGENVECTORS INVOLVED IN THE LAST GRAM-SCHMIDT ORTHOGONALISATION (SEE METHOD AND PERFORMANCE),
EM[7], THE MAXIMUM EUCLIDEAN NORM OF THE RESIDUES,
EM[9], THE LARGEST NUMBER OF ITERATIONS PERFORMED FOR THE CALCULATION OF SOME EIGENVECTOR (SEE METHOD AND PERFORMANCE).

PROCEDURES USED:
VECVEC = CP34010,
TAMVEC = CP34012,
ELMVECCOL= CP34021.

REQUIRED CENTRAL MEMORY:
EXECUTION FIELD LENGTH: FIVE AUXILIARY ONE-DIMENSIONAL REAL ARRAYS AND ONE BOOLEAN ARRAY, ALL OF LENGTH N, ARE USED.

RUNNING TIME: THE PROCESS IS OF ORDER N FOR EACH EIGENVECTOR.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
AN EIGENVECTOR OF A SYMMETRIC TRIDIAGONAL MATRIX T, CORRESPONDING TO AN EIGENVALUE LAMBDA, IS CALCULATED BY MEANS OF INVERSE ITERATION: I.E. STARTING FROM SOME INITIAL VECTOR X, THE LINEAR SYSTEM (T - LAMBDA * I)Y = X IS SOLVED ITERATIVELY, THE SOLUTION Y, DIVIDED BY ITS EUCLIDEAN NORM REPLACING X EACH TIME.
IF THE DISTANCE BETWEEN SOME APPROXIMATE EIGENVALUES IS SMALLER THAN MACHTOL (=EM[0] * EM[1]), THEN THEY ARE SLIGHTLY MODIFIED SUCH THAT THE DISTANCE BETWEEN THEM EQUALS MACHTOL. IF THE DISTANCE BETWEEN SOME EIGENVALUES IS SMALLER THAN THE ORTHOGONALISATION PARAMETER (=EM[4]) TIMES EM[1], THEN IN EACH ITERATION STEP GRAM-SCHMIDT ORTHOGONALISATION IS CARRIED OUT, SO THAT THE EIGENVECTORS OBTAINED ARE ORTHOGONAL WITHIN WORKING PRECISION. THE ITERATION ENDS AS SOON AS EITHER THE EUCLIDEAN NORM OF THE RESIDUE IS SMALLER THAN EM[1] * EM[6], OR THE MAXIMUM ALLOWED NUMBER OF ITERATIONS (=EM[8]) HAS BEEN PERFORMED. IN THE LATTER CASE EM[9]:= EM[8] + 1.
IF N1 > 1, THEN VECSYMTRI SHOULD BE PRECEDED BY ONE OR MORE CALLS OF VECSYMTRI PRODUCING A NUMBER OF EIGENVECTORS CORRESPONDING TO THE PRECEDING EIGENVALUES. MOREOVER ONE MUST GIVE EM[5], AS PRODUCED BY THE LAST CALL OF VECSYMTRI: THE K-TH TO N2-TH EIGENVALUES, WHERE K = N1 - EM[5], MUST BE GIVEN IN ARRAY VAL[K:N2] IN MONOTONICALLY NONINCREASING ORDER (THE K-TH TO (N1-1)-TH EIGENVALUES BEING NEEDED FOR THE MODIFYING MENTIONED ABOVE), AND THE CORRESPONDING EIGENVECTORS UP TO THE (N1-1)-TH (WHICH ARE NEEDED FOR THE GRAM-SCHMIDT ORTHOGONALISATION) IN THE CORRESPONDING COLUMNS OF ARRAY VEC[1:N,K:N2].
THE TOLERANCES SHOULD SATISFY: EM[0] (< EM[2]) < EM[6] AND EM[4] >= EM[0] / EM[6]. FOR FURTHER DETAILS SEE REF[1].

SUBSECTION: ORIVALSYMTRI.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" ORIVALSYMTRI(D, BB, N, EM);
"VALUE" N; "INTEGER" N; "ARRAY" D, BB, EM;

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

D: <ARRAY IDENTIFIER>;
"ARRAY" D[1:N];
ENTRY: THE MAIN DIAGONAL OF THE SYMMETRIC TRIDIAGONAL MATRIX;
EXIT: THE EIGENVALUES OF THE MATRIX IN SOME ARBITRARY ORDER;

BB: <ARRAY IDENTIFIER>;
"ARRAY" BB[1:N];
ENTRY: THE SQUARES OF THE CODIAGONAL ELEMENTS OF THE SYMMETRIC TRIDIAGONAL MATRIX FOLLOWED BY AN ADDITONAL ELEMENT 0;
EXIT: THE SQUARES OF THE CODIAGONAL ELEMENTS OF THE SYMMETRIC TRIDIAGONAL MATRIX RESULTING FROM THE QR ITERATION;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY: EM[0], THE MACHINE PRECISION;
EM[1], A NORM OF THE GIVEN MATRIX;
EM[2], A RELATIVE TOLERANCE FOR THE EIGENVALUES;
EM[4], THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
EXIT: EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE CODIAGONAL ELEMENTS NEGLECTED;
EM[5], THE NUMBER OF ITERATIONS PERFORMED.

MOREOVER:
ORIVALSYMTRI:= THE NUMBER OF EIGENVALUES NOT CALCULATED.

PROCEDURES USED: NONE.

RUNNING TIME: THE PROCESS IS OF ORDER N SQUARED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
IN ORIVALSYMTRI THE EIGENVALUES OF A SYMMETRIC TRIDIAGONAL MATRIX ARE CALCULATED BY MEANS OF QR-ITERATION. FOR THIS PROCEDURE WE USED ESSENTIALLY THE SQUARE-ROOT-FREE VERSION OF THE QR ALGORITHM DUE TO REINSCH[3].
IN ADDITION TO THE RELATIVE ERROR, WHICH IS SUPPOSED TO BE BOUNDED BY EM[1] * EM[2] (I.E. MATRIX NORM TIMES RELATIVE TOLERANCE), THE CALCULATED EIGENVALUES HAVE AN ABSOLUTE ERROR WHICH IS BOUNDED BY BY EM[0] * EM[1] (I.E. MACHINE PRECISION TIMES MATRIX NORM).
IN PARTICULAR, WHEN SOME EIGENVALUES ARE VERY SMALL COMPARED TO THE MATRIX NORM, THE ACCURACY OF THE CALCULATED EIGENVALUES CAN BE INCREASED BY GIVING EM[0] A (POSITIVE) VALUE WHICH IS LESS THAN THE MACHINE PRECISION.
A PARTICULAR CHOICE OF EM[0] IS HARMLESS FOR THE PROCEDURE PROVIDED THAT FOR EACH I THE CALCULATION OF BB[I] / EM[0] ** 2 CAUSES NO OVERFLOW AND THE CALCULATION OF (EM[0] * EM[1]) ** 2 CAUSES NO UNDERFLOW.
ONE SHOULD NOTICE THAT THE NUMBER OF QR ITERATIONS INCREASES BY A SMALLER CHOICE OF EM[0].
FOR FURTHER DETAILS SEE [2], [3].

SUBSECTION: QRISYMTRI.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" QRISYMTRI(A, N, D, B, BB, EM);
"VALUE" N; "INTEGER" N; "ARRAY" A, D, B, BB, EM;

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

D: <ARRAY IDENTIFIER>;
"ARRAY" D[1:N];
ENTRY: THE MAIN DIAGONAL OF THE SYMMETRIC TRIDIAGONAL MATRIX;
EXIT: THE EIGENVALUES OF THE MATRIX IN SOME ARBITRARY ORDER;

B: <ARRAY IDENTIFIER>;
"ARRAY" B[1:N];
ENTRY: THE CODIAGONAL OF THE SYMMETRIC TRIDIAGONAL MATRIX FOLLOWED BY AN ADDITIONAL ELEMENT 0;
EXIT: THE CODIAGONAL OF THE SYMMETRIC TRIDIAGONAL MATRIX RESULTING FROM THE QR ITERATION, FOLLOWED BY AN ADDITIONAL ELEMENT 0;

BB: <ARRAY IDENTIFIER>;
"ARRAY" BB[1:N];
ENTRY: THE SQUARED CODIAGONAL ELEMENTS OF THE SYMMETRIC TRIDIAGONAL MATRIX, FOLLOWED BY AN ADDITIONAL ELEMENT 0;
EXIT: THE SQUARED CODIAGONAL ELEMENTS OF THE SYMMETRIC TRIDIAGONAL MATRIX RESULTING FROM THE QR ITERATION;

A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: SOME MATRIX S, SAY, (POSSIBLY THE IDENTITY MATRIX):
EXIT: THE EIGENVECTORS OF THE ORIGINAL SYMMETRIC TRIDIAGONAL MATRIX, PREMULTIPLIED BY S (SEE METHOD AND PERFORMANCE):

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY: EM[0], THE MACHINE PRECISION;
EM[1], A NORM OF THE GIVEN MATRIX;
EM[2], A RELATIVE TOLERANCE FOR THE QR ITERATION;
EM[4], THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
EXIT: EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE CODIAGONAL ELEMENTS NEGLECTED;
EM[5], THE NUMBER OF ITERATIONS PERFORMED.

MOREOVER:
QRISYMTRI:= THE NUMBER OF EIGENVALUES AND -VECTORS NOT CALCULATED.

PROCEDURES USED:
ROTCOL = CP34040.

RUNNING TIME: THE PROCESS IS OF ORDER N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
IN QRISYMTRI THE EIGENVALUES AND EIGENVECTORS OF A SYMMETRIC TRIDIAGONAL MATRIX ARE COMPUTED SIMULTANEOUSLY.
IN MOST APPLICATIONS QRISYMTRI IS USED IN THE COMPUTATION OF EIGENVALUES AND -VECTORS OF A GENERAL SYMMETRIC MATRIX (SEE QRISYM SECTION 3.3.1.1.2); IN THAT CASE ARRAY A IS INITIALLY GIVEN THE VALUE OF THE TRANSFORMING MATRIX (TFMPREVEC SECTION 3.2.1.2.1.1).
FOR THE COMPUTATION OF EIGENVALUES AND EIGENVECTORS OF A SYMMETRIC TRIDIAGONAL MATRIX, ARRAY A HAS TO BE INITIALIZED TO THE IDENTITY MATRIX. THE AVERAGE NUMBER OF ITERATIONS IS ABOUT 3N. WHEN THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS, THEN QRISYMTRI:= 0; OTHERWISE QRISYMTRI:= THE NUMBER, K, OF EIGENVALUES NOT CALCULATED, EM[5]:= EM[4] + 1 AND ONLY THE LAST N - K ELEMENTS OF D AND THE LAST N - K COLUMNS OF A ARE APPROXIMATE EIGENVALUES AND -VECTORS RESPECTIVELY, OF THE ORIGINAL MATRIX.
FOR FURTHER DETAILS SEE REF[1], REF[2].

REFERENCES:

[1] DEKKER, T.J. AND HOFFMANN, W.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2,
MATHEMATICAL CENTRE TRACTS 23,
MATHEMATISCH CENTRUM, AMSTERDAM. 1968;

[2] WILKINSON, J.H.
THE ALGEBRAIC EIGENVALUE PROBLEM,
CLARENDON PRESS, OXFORD 1965.

[3] REINSCH, CHR.H.
A STABLE, RATIONAL QR ALGORITHM FOR THE COMPUTATION OF THE
EIGENVALUES OF AN HERMITIAN, TRIDIAGONAL MATRIX.
MATH. OF COMP. VOL 25(1971) PP. 591-597.

EXAMPLE OF USE:

THE FIRST AND SECOND EIGENVALUE IN MONOTONICALLY NON-INCREASING ORDER AND THE CORRESPONDING EIGENVECTORS OF T, WITH N = 4 AND T[I,J] = "IF" I = J "THEN" 2 "ELSE" "IF" ABS(I - J) = 1 "THEN" - 1 "ELSE" 0, MAY BE OBTAINED BY THE FOLLOWING PROGRAM:

```
"BEGIN"
    "INTEGER" J;
    "ARRAY" B, D[1:4], BB[1:3], VAL[1:2], EM[0:9], VEC[1:4,1:2];
    "PROCEDURE" VALSYMTRI(D, BB, N, N1, N2, VAL, EM); "CODE" 34151;
    "PROCEDURE" VECSYMTRI(D,B,N,N1,N2, VAL, VEC, EM); "CODE" 34152;

    EM[0]:= "-14; EM[1]:= 4; EM[2]:= "-12;
    EM[4]:= "-3; EM[6]:= "-10; EM[8]:= 5;
    "FOR" J:= 1, 2, 3, 4 "DO" D[J]:= 2; B[4]:= 0;
    "FOR" J:= 1, 2, 3 "DO"
    "BEGIN" BB[J]:= 1; B[J]:= -1 "END";
    VALSYMTRI(D, BB, 4, 1, 2, VAL, EM);
    VECSYMTRI(D, B, 4, 1, 2, VAL, VEC, EM);
    OUTPUT(61, "("2(+.13D"+2D, 2B), 2/")", VAL[1], VAL[2]);
    "FOR" J:= 1, 2, 3, 4 "DO"
    OUTPUT(61, "("2(+.13D"+2D, 2B), /")", VEC[J,1], VEC[J,2]);
    OUTPUT(61, "("/, .2D"+2D, /, 3(2ZD, /)")",
    EM[7], EM[3], EM[5], EM[9])
"END"
```

THE PROGRAM DELIVERS:

```
THE EIGENVALUES:  +.3618033988751"+01  +.2618033988750"+01

THE EIGENVECTORS: +.3717480344602"+00  +.6015009550075"+00
                  +.6015009550075"+00  +.3717480344602"+00
                  +.6015009550075"+00  -.3717480344602"+00
                  +.3717480344602"+00  -.6015009550075"+00

EM[7] = .15"-11
EM[3] = 24
EM[5] = 1
EM[9] = 1  .
```

SOURCE TEXT(S):

```
"CODE" 34151;
    "COMMENT" MCA 2311;
    "PROCEDURE" VALSYMTRI(D, BB, N, N1, N2, VAL, EM);
    "VALUE" N, N1, N2;
    "INTEGER" N, N1, N2; "ARRAY" D, BB, VAL, EM;
        "BEGIN" "INTEGER" K, COUNT;
        "REAL" MAX, X, Y, MACHEPS, NORM, RE, MACHTOL, UB, LB, LAMBDA;

        "REAL" "PROCEDURE" STURM;
        "BEGIN" "INTEGER" P, I; "REAL" F;
            COUNT:= COUNT + 1;
            P:= K; F:= D[1] - X;
            "FOR" I:= 2 "STEP" 1 "UNTIL" N "DO"
            "BEGIN" "IF" F <= 0 "THEN"
                "BEGIN" P:= P + 1;
                    "IF" P > N "THEN" "GOTO" OUT
                "END"
                "ELSE" "IF" P < I - 1 "THEN"
                "BEGIN" LB:= X; "GOTO" OUT "END";
                "IF" ABS(F) < MACHTOL "THEN"
                F:= "IF" F <= 0 "THEN" - MACHTOL "ELSE" MACHTOL;
                F:= D[I] - X - BB[I - 1] / F
            "END";
            "IF" P = N "OR" F <= 0 "THEN"
            "BEGIN" "IF" X < UB "THEN" UB:= X "END" "ELSE" LB:= X;
       OUT: STURM:= "IF" P = N "THEN" F "ELSE" (N - P) * MAX
        "END" STURM;

        "BOOLEAN" "PROCEDURE" ZEROIN(X, Y, FX, TOLX); "CODE"34150;

        MACHEPS:= EM[0]; NORM:= EM[1]; RE:= EM[2];
        MACHTOL:= NORM * MACHEPS; MAX:= NORM / MACHEPS; COUNT:= 0;
        UB:= 1.1 * NORM; LB:= - UB; LAMBDA:= UB;
        "FOR" K:= N1 "STEP" 1 "UNTIL" N2 "DO"
        "BEGIN" X:= LB; Y:= UB; LB:= -1.1  * NORM;
            ZEROIN(X, Y, STURM, ABS(X) * RE + MACHTOL);
            VAL[K]:= LAMBDA:= "IF" X > LAMBDA "THEN" LAMBDA "ELSE" X;
            "IF" UB > X "THEN" UB:= "IF" X > Y "THEN" X "ELSE" Y
        "END";
        EM[3]:= COUNT
    "END" VALSYMTRI;
        "EOP"
```

```
"CODE" 34152;
    "COMMENT" MCA 23.2;
    "PROCEDURE" VECSYMTRI(D, B, N, N1, N2, VAL, VEC, EM);
    "VALUE" N, N1, N2;
    "INTEGER" N, N1, N2; "ARRAY" D, B, VAL, VEC, EM;
    "BEGIN" "INTEGER" I, J, K, COUNT, MAXCOUNT, COUNTLIM, ORTH, IND;
        "REAL" BI, BI1, U, W, Y, MI1, LAMBDA, OLDLAMBDA, ORTHEPS,
        VALSPREAD, SPR, RES, MAXRES, OLDRES, NORM, NEWNORM, OLDNORM,
        MACHTOL, VECTOL;
        "ARRAY" M, P, Q, R, X[1:N];
        "BOOLEAN" "ARRAY" INT[1:N];

        "REAL" "PROCEDURE" VECVEC(L, U, SHIFT, A, B); "CODE" 34010;
        "PROCEDURE" ELMVECCOL(L, U, I, A, B, X); "CODE" 34021;
        "REAL" "PROCEDURE" TAMVEC(L, U, I, A, B); "CODE" 34012;

        NORM:= EM[1]; MACHTOL:= EM[0] * NORM; VALSPREAD:= EM[4] * NORM;
        VECTOL:= EM[6] * NORM; COUNTLIM:= EM[8]; ORTHEPS:= SQRT(EM[0]);
        MAXCOUNT:= IND:= 0; MAXRES:= 0;
        "IF" N1 > 1 "THEN"
        "BEGIN" ORTH:= EM[5]; OLDLAMBDA:= VAL[N1 - ORTH];
            "FOR" K:= N1 - ORTH + 1 "STEP" 1 "UNTIL" N1 - 1  "DO"
            "BEGIN" LAMBDA:= VAL[K]; SPR:= OLDLAMBDA - LAMBDA;
                "IF" SPR < MACHTOL "THEN" LAMBDA:= OLDLAMBDA - MACHTOL;
                OLDLAMBDA:= LAMBDA
            "END"
        "END" "ELSE" ORTH:= 1;
        "FOR" K:= N1 "STEP" 1 "UNTIL" N2 "DO"
        "BEGIN" LAMBDA:= VAL[K]; "IF" K > 1 "THEN"
            "BEGIN" SPR:= OLDLAMBDA - LAMBDA;
                "IF" SPR < VALSPREAD "THEN"
                "BEGIN" "IF" SPR < MACHTOL "THEN"
                    LAMBDA:= OLDLAMBDA - MACHTOL;
                    ORTH:= ORTH +1
                "END" "ELSE" ORTH:= 1
            "END";
            COUNT:= 0; U:= D[1] - LAMBDA; BI:= W:= B[1];
            "IF" ABS(BI) < MACHTOL "THEN" BI:= MACHTOL;
            "FOR" I:= 1 "STEP" 1 "UNTIL" N - 1  "DO"
            "BEGIN" BI1:= B[I + 1];
                "IF" ABS(BI1) < MACHTOL "THEN" BI1:= MACHTOL;
                "IF" ABS(BI) >= ABS(U) "THEN"
                "BEGIN" MI1:= M[I + 1]:= U / BI; P[I]:= BI;
                    Y:= Q[I]:= D[I + 1] - LAMBDA; R[I]:= BI1;
                    U:= W - MI1 * Y; W:= - MI1 * BI1; INT[I]:= "TRUE"
                "END"
                "ELSE"
                "BEGIN" MI1:= M[I + 1]:= BI / U; P[I]:= U; Q[I]:= W;
                    R[I]:= 0; U:= D[I + 1] - LAMBDA - MI1 * W;W:= BI1;
                    INT[I]:= "FALSE"
                "END";
                X[I]:= 1; BI:= BI1
            "END" TRANSFORM
```

```
        P[N]:= "IF" ABS(U) < MACHTOL "THEN" MACHTOL "ELSE" U;
        Q[N]:= R[N]:= 0; X[N]:= 1; "GOTO" ENTRY;
    ITERATE: W:= X[1];
        "FOR" I:= 2 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" "IF" INT[I - 1] "THEN"
            "BEGIN" U:= W; W:= X[I - 1]:= X[I] "END"
            "ELSE" U:= X[I]; W:= X[I]:= U - M[I] * W
        "END" ALTERNATE;
    ENTRY: U:= W:= 0;
        "FOR" I:= N "STEP" -1 "UNTIL" 1 "DO"
        "BEGIN" Y:= U; U:= X[I]:= (X[I] - Q[I] * U - R[I] * W) /
            P[I]; W:= Y
        "END" NEXT ITERATION;
        NEWNORM:= SQRT(VECVEC(1, N, 0, X, X)); "IF" ORTH > 1"THEN"
        "BEGIN" OLDNORM:= NEWNORM;
            "FOR" J:= K - ORTH + 1 "STEP" 1 "UNTIL" K - 1 "DO"
            ELMVECCOL(1, N, J, X, VEC, -TAMVEC(1, N, J, VEC, X));
            NEWNORM:= SQRT(VECVEC(1, N, 0, X, X));
            "IF" NEWNORM < ORTHEPS * OLDNORM "THEN"
            "BEGIN" IND:= IND + 1; COUNT:= 1;
                "FOR" I:= 1 "STEP" 1 "UNTIL" IND - 1,
                IND + 1 "STEP" 1 "UNTIL" N "DO" X[I]:= 0;
                X[IND]:= 1; "IF" IND = N "THEN" IND:= 0;
                "GOTO" ITERATE
            "END" NEW START
        "END" ORTHOGONALISATION;
        RES:= 1 / NEWNORM; "IF" RES > VECTOL "OR" COUNT = 0 "THEN"
        "BEGIN" COUNT:= COUNT + 1; "IF" COUNT <= COUNTLIM "THEN"
            "BEGIN" "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
                X[I]:= X[I] * RES; "GOTO" ITERATE
            "END"
        "END";
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO" VEC[I,K]:= X[I] * RES;
        "IF" COUNT > MAXCOUNT "THEN" MAXCOUNT:= COUNT;
        "IF" RES > MAXRES "THEN" MAXRES:= RES; OLDLAMBDA:= LAMBDA
    "END";
    EM[5]:= ORTH; EM[7]:= MAXRES; EM[9]:= MAXCOUNT
"END" VECSYMTRI;
    "EOP"
```

```
"CODE" 34160;
"INTEGER" "PROCEDURE" ORIVALSYMTRI(D, BB, N, EM); "VALUE" N;
"INTEGER" N; "ARRAY" D, BB, EM;
"BEGIN" "INTEGER" I, I1, LOW, OLDLOW, N1, COUNT, MAX;
    "REAL" BBTOL, BBMAX, BBI, BBN1, MACHTOL, DN, DELTA, F, NUM,
    SHIFT, G, H, T, P, R, S, C, OLDG;
    BBTOL:= (EM[2] * EM[1]) ** 2; MACHTOL:= EM[0] * EM[1];
    MAX:= EM[4]; BBMAX:= 0; COUNT:= 0; OLDLOW:= N;
    "FOR" N1:= N - 1 "WHILE" N > 0 "DO"
    "BEGIN"
        "FOR" I:= N, I - 1 "WHILE" ("IF" I >= 1 "THEN"
        BB[I] > BBTOL "ELSE" "FALSE") "DO" LOW:= I;
        "IF" LOW > 1 "THEN" "BEGIN" "IF" BB[LOW-1] > BBMAX "THEN"
        BBMAX:= BB[LOW-1] "END";
        "IF" LOW = N "THEN" N:= N1 "ELSE"
        "BEGIN" DN:= D[N]; DELTA:= D[N1] - DN;
            BBN1:= BB[N1];
            "IF" ABS(DELTA) < MACHTOL "THEN" R:= SQRT(BBN1) "ELSE"
            "BEGIN"
                F:= 2 / DELTA; NUM:= BBN1 * F;
                R:= -NUM / (SQRT(NUM * F + 1) + 1)
            "END";
            "IF" LOW = N1 "THEN"
            "BEGIN" D[N]:= DN + R; D[N1]:= D[N1] - R; N:= N - 2
            "END"
            "ELSE"
            "BEGIN" COUNT:= COUNT + 1;
                "IF" COUNT > MAX "THEN" "GOTO" END;
                "IF" LOW < OLDLOW "THEN"
                "BEGIN" SHIFT:= 0; OLDLOW:= LOW "END"
                "ELSE"  SHIFT:= DN + R;
                H:= D[LOW] - SHIFT;
                "IF" ABS(H) < MACHTOL "THEN" H:= "IF" H <= 0 "THEN"
                -MACHTOL "ELSE" MACHTOL;
                G:= H; T:= G * H;
                BBI:= BB[LOW]; P:= T + BBI; I1:= LOW;
                "FOR" I:= LOW + 1 "STEP" 1 "UNTIL" N "DO"
                "BEGIN" S:= BBI / P; C:= T / P;
                    H:= D[I] - SHIFT - BBI / H;
                    "IF" ABS(H) < MACHTOL "THEN" H:= "IF" H <= 0
                    "THEN" -MACHTOL "ELSE" MACHTOL;
                    OLDG:= G; G:= H * C; T:= G * H;
                    D[I1]:= OLDG - G + D[I];
                    BBI:= "IF" I = N "THEN" 0 "ELSE" BB[I];
                    P:= T + BBI; BB[I1]:= S * P; I1:= I
                "END";
                D[N]:= G + SHIFT
            "END" ORSTEP
        "END"
    "END";
 END: EM[3]:= SQRT(BBMAX); EM[5]:= COUNT; ORIVALSYMTRI:= N
"END" ORIVALSYMTRI;
     "EOP"
```

```
"CODE" 34161;
    "COMMENT" MCA 2321;
    "INTEGER" "PROCEDURE" QRISYMTRI(A, N, D, B, BB, EM); "VALUE" N;
    "INTEGER" N; "ARRAY" A, D, B, BB, EM;
    "BEGIN" "INTEGER" I, J, J1, K, M, M1, COUNT, MAX;
        "REAL" BBMAX, R, S, SIN, T, C, COS, OLDCOS, G, P, W, TOL, TOL2,
        LAMBDA, DK1, A0, A1;

        "PROCEDURE" ROTCOL(L, U, I, J, A, C, S); "CODE" 34040;

        TOL:= EM[2] * EM[1]; TOL2:= TOL * TOL; COUNT:= 0; BBMAX:= 0;
        MAX:= EM[4]; M:= N;
     IN: K:= M; M1:= M - 1;
     NEXT: K:= K - 1; "IF" K > 0 "THEN"
        "BEGIN" "IF" BB[K] >= TOL2 "THEN" "GOTO" NEXT;
            "IF" BB[K] > BBMAX "THEN" BBMAX:= BB[K]
        "END";
        "IF" K = M1 "THEN" M:= M1 "ELSE"
        "BEGIN"
            T:= D[M] - D[M1]; R:= BB[M1];
            "IF" ABS(T) < TOL "THEN" S:= SQRT(R) "ELSE"
            "BEGIN" W:= 2 / T; S:= W * R / (SQRT(W * W * R + 1) + 1)
            "END"; "IF" K = M - 2 "THEN"
            "BEGIN" D[M]:= D[M] + S; D[M1]:= D[M1] - S;
                T:= - S / B[M1]; R:= SQRT(T * T + 1); COS:= 1 / R;
                SIN:= T / R; ROTCOL(1,N,M1,M,A,COS,SIN); M:= M - 2
            "END"
            "ELSE"
            "BEGIN" COUNT:= COUNT + 1;
                "IF" COUNT > MAX "THEN" "GOTO" END;
                LAMBDA:= D[M] + S; "IF" ABS(T) < TOL "THEN"
                "BEGIN" W:= D[M1] - S;
                    "IF" ABS(W) < ABS(LAMBDA) "THEN" LAMBDA:= W
                "END";
                K:= K + 1; T:= D[K] - LAMBDA; COS:= 1; W:= B[K];
                P:= SQRT(T * T + W * W); J1:= K;
                "FOR" J:= K + 1 "STEP" 1 "UNTIL" M "DO"
                "BEGIN" OLDCOS:= COS; COS:= T / P; SIN:= W / P;
                    DK1:= D[J] - LAMBDA; T:= OLDCOS * T;
                    D[J1]:= (T + DK1) * SIN * SIN + LAMBDA + T;
                    T:= COS * DK1 - SIN * W * OLDCOS; W:= B[J];
                    P:= SQRT(T * T + W * W); G:= B[J1]:= SIN * P;
                    BB[J1]:= G * G; ROTCOL(1, N, J1, J, A, COS, SIN);
                    J1:= J
                "END";
                D[M]:= COS * T + LAMBDA; "IF" T < 0 "THEN" B[M1]:= - G
            "END" QRSTEP
        "END";
        "IF" M > 0 "THEN" "GOTO" IN;
     END: EM[3]:= SQRT(BBMAX); EM[5]:= COUNT; QRISYMTRI:= M
    "END" QRISYMTRI;
        "EOP"
```

AUTHORS: T.J.DEKKER AND W.HOFFMANN.

CONTRIBUTORS: W.HOFFMANN, J.G.VERWER.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730924.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS SEVEN PROCEDURES.
A) EIGVALSYM1 AND EIGVALSYM2 CALCULATE ALL EIGENVALUES, OR SOME CONSECUTIVE EIGENVALUES INCLUDING THE LARGEST, OF A SYMMETRIC MATRIX USING LINEAR INTERPOLATION ON A FUNCTION DERIVED FROM A STURM SEQUENCE.
B) EIGSYM1 AND EIGSYM2 CALCULATE THE CORRESPONDING EIGENVECTORS AS WELL, BY MEANS OF INVERSE ITERATION.
C) QRIVALSYM1 AND QRIVALSYM2 CALCULATE ALL EIGENVALUES OF A SYMMETRIC MATRIX BY MEANS OF QR ITERATION.
D) QRISYM CALCULATES ALL EIGENVECTORS AS WELL IN THE SAME ITERATION PROCESS.
EIGVALSYM1, EIGSYM1 AND QRIVALSYM1 USE IONAL ARRAY FOR THE GIVEN SYMMETRIC MATRIX; THE OTHER PROCEDURES EXPECT THE MATRIX TO BE STORED IN "ARRAY".
QRISYM DELIVERS THE EIGENVECTORS IN THE ARRAY THAT WAS USED FOR THE ORIGINAL MATRIX IN CONTRAST WITH EIGSYM1 AND EIGSYM2 WHICH DELIVER THE EIGENVECTORS IN AN EXTRA ARRAY.
WHEN ALL EIGENVALUES HAVE TO BE CALCULATED, THE PROCEDURES USING QR ITERATION ARE PREFERABLE WITH RESPECT TO THEIR RUNNING TIME. WHEN ALSO THE EIGENVECTORS HAVE TO BE CALCULATED THE PROCEDURES USING INVERSE ITERATION ARE FASTER; HOWEVER, THE ONE USING QR ITERATION USES LESS MEMORY SPACE.

KEYWORDS:
EIGENVALUES,
EIGENVECTORS,
SYMMETRIC MATRIX,
STURM-SEQUENCE,
INVERSE ITERATION,
QR ITERATION.

SUBSECTION: EIGVALSYM2.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" EIGVALSYM2(A, N, NUMVAL, VAL, EM);
"VALUE" N, NUMVAL; "INTEGER" N, NUMVAL; "ARRAY" A, VAL, EM;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;
NUMVAL: <ARITHMETIC EXPRESSION>;
THE SERIAL NUMBER OF THE LAST EIGENVALUE TO BE CALCULATED;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE UPPER TRIANGLE OF THE SYMMETRIC MATRIX MUST BE GIVEN IN THE UPPER TRIANGULAR PART OF A (THE ELEMENTS A[I,J], I<= J);
EXIT: THE DATA FOR HOUSEHOLDER'S BACK TRANSFORMATION (WHICH ISN'T USED BY THIS PROCEDURE) IS DELIVERED IN THE UPPER TRIANGULAR PART OF A;
THE ELEMENTS A[I,J] FOR I > J ARE NEITHER USED NOR CHANGED;
VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:NUMVAL];
EXIT: THE NUMVAL LARGEST EIGENVALUES IN MONOTONICALLY NON-INCREASING ORDER;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:3];
ENTRY: EM[0], THE MACHINE PRECISION,
EM[2], THE RELATIVE TOLERANCE FOR THE EIGENVALUES;
EXIT: EM[1], THE INFINITY NORM OF THE ORIGINAL MATRIX,
EM[3], THE NUMBER OF ITERATIONS USED FOR CALCULATING THE NUMVAL EIGENVALUES.

PROCEDURES USED:
TFMSYMTRI2 = CP34140,
VALSYMTRI = CP34151.

REQUIRED CENTRAL MEMORY:
EXECUTION FIELD LENGTH:
THREE ONE-DIMENSIONAL REAL ARRAYS OF LENGTH N ARE USED.

RUNNING TIME:
ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE BODY OF EIGVALSYM2 CONSISTS OF TWO PROCEDURE STATEMENTS; THE FIRST IS A CALL OF TFMSYMTRI2 TO TRANSFORM THE SYMMETRIC MATRIX INTO A SIMILAR TRIDIAGONAL MATRIX BY MEANS OF HOUSEHOLDER'S TRANSFORMATION; THE SECOND IS A CALL OF VALSYMTRI TO CALCULATE THE DESIRED EIGENVALUES. OPERATION DETAILS OF BOTH PROCEDURES ARE GIVEN IN THEIR DESCRIPTION.

SUBSECTION: EIGVALSYM1.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" EIGVALSYM1(A, N, NUMVAL, VAL, EM);
"VALUE" N, NUMVAL; "INTEGER" N, NUMVAL; "ARRAY" A, VAL, EM;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;
NUMVAL: <ARITHMETIC EXPRESSION>;
THE SERIAL NUMBER OF THE LAST EIGENVALUE TO BE CALCULATED;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:(N+1)*N//2];
ENTRY: THE UPPER TRIANGLE OF THE SYMMETRIC MATRIX MUST BE GIVEN IN SUCH A WAY THAT THE (I,J)-TH ELEMENT OF THE MATRIX IS A[(J-1)*J//2+I], 1 <= I <= J <= N;
EXIT: THE DATA FOR HOUSEHOLDER'S BACK TRANSFORMATION (WHICH ISN'T USED BY THIS PROCEDURE).
VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:NUMVAL];
EXIT: THE NUMVAL LARGEST EIGENVALUES IN MONOTONICALLY NON-INCREASING ORDER;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:3];
ENTRY: EM[0], THE MACHINE PRECISION,
EM[2], THE RELATIVE TOLERANCE FOR THE EIGENVALUES;
EXIT: EM[1], THE INFINITY NORM OF THE ORIGINAL MATRIX,
EM[3], THE NUMBER OF ITERATIONS USED FOR CALCULATING THE NUMVAL EIGENVALUES.

PROCEDURES USED:
TFMSYMTRI1 = CP34143,
VALSYMTRI = CP34151.

REQUIRED CENTRAL MEMORY:
EXECUTION FIELD LENGTH:
THREE ONE-DIMENSIONAL REAL ARRAYS OF LENGTH N ARE USED.

RUNNING TIME:
ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE BODY OF EIGVALSYM1 CONSISTS OF TWO PROCEDURE STATEMENTS; THE FIRST IS A CALL OF TFMSYMTRI1 TO TRANSFORM THE SYMMETRIC MATRIX INTO A SIMILAR TRIDIAGONAL MATRIX BY MEANS OF HOUSEHOLDER'S TRANSFORMATION; THE SECOND IS A CALL OF VALSYMTRI TO CALCULATE THE DESIRED EIGENVALUES. OPERATION DETAILS OF BOTH PROCEDURES ARE GIVEN IN THEIR DESCRIPTION.

SUBSECTION: EIGSYM2.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" EIGSYM2(A, N, NUMVAL, VAL, VEC, EM);
"VALUE" N, NUMVAL; "INTEGER" N, NUMVAL; "ARRAY" A, VAL, VEC, EM;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;
NUMVAL: <ARITHMETIC EXPRESSION>;
THE SERIAL NUMBER OF THE LAST EIGENVALUE TO BE CALCULATED;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE UPPER TRIANGLE OF THE SYMMETRIC MATRIX MUST BE GIVEN IN THE UPPER TRIANGULAR PART OF A (THE ELEMENTS A[I,J], I<= J);
EXIT: THE DATA FOR HOUSEHOLDER'S BACK TRANSFORMATION IS DELIVERED IN THE UPPER TRIANGULAR PART OF A;
THE ELEMENTS A[I,J] FOR I > J ARE NEITHER USED NOR CHANGED;
VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:NUMVAL];
EXIT: THE NUMVAL LARGEST EIGENVALUES IN MONOTONICALLY NON-INCREASING ORDER;
VEC: <ARRAY IDENTIFIER>;
"ARRAY" VEC[1:N,1:NUMVAL];
EXIT: THE NUMVAL CALCULATED EIGENVECTORS, STORED COLUMN-WISE, CORRESPONDING TO THE CALCULATED EIGENVALUES;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:9];
ENTRY: EM[0], THE MACHINE PRECISION,
EM[2], THE RELATIVE TOLERANCE FOR THE EIGENVALUES,
EM[4], THE ORTHOGONALISATION PARAMETER (SEE METHOD AND PERFORMANCE),
EM[6], THE TOLERANCE FOR THE EIGENVECTORS,
EM[8], THE MAXIMUM NUMBER OF INVERSE ITERATIONS ALLOWED FOR THE CALCULATION OF EACH EIGEN-VECTOR;
EXIT: EM[1], THE INFINITY NORM OF THE MATRIX,
EM[3], THE NUMBER OF ITERATIONS USED FOR CALCULATING THE NUMVAL EIGENVALUES,
EM[5], THE NUMBER OF EIGENVECTORS INVOLVED IN THE LAST GRAM-SCHMIDT ORTHOGONALISATION,
EM[7], THE MAXIMUM EUCLIDEAN NORM OF THE RESIDUES OF THE CALCULATED EIGENVECTORS,
EM[9], THE LARGEST NUMBER OF INVERSE ITERATIONS PERFORMED FOR THE CALCULATION OF SOME EIGEN-VECTOR.

PROCEDURES USED:

TFMSYMTRI2	=	CP34140,
VALSYMTRI	=	CP34151,
VECSYMTRI	=	CP34152,
BAKSYMTRI2	=	CP34141.

REQUIRED CENTRAL MEMORY:

EXECUTION FIELD LENGTH:

THREE ONE-DIMENSIONAL REAL ARRAYS OF LENGTH N ARE DECLARED; MOREOVER, VECSYMTRI USES FIVE ONE-DIMENSIONAL REAL ARRAYS OF LENGTH N AND ONE BOOLEAN ARRAY OF LENGTH N.

RUNNING TIME:

ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

THE BODY OF EIGSYM2 CONSISTS OF FOUR PROCEDURE STATEMENTS: THE FIRST IS A CALL OF TFMSYMTRI2 TO TRANSFORM THE SYMMETRIC MATRIX INTO A SIMILAR TRIDIAGONAL MATRIX BY MEANS OF HOUSEHOLDERS TRANSFORMATION,
THE SECOND IS A CALL OF VALSYMTRI TO CALCULATE THE DESIRED EIGENVALUES,
THE THIRD IS A CALL OF VECSYMTRI TO CALCULATE THE CORRESPONDING EIGENVECTORS AND
THE FOURTH IS A CALL OF BAKSYMTRI2 TO PERFORM THE BACK TRANSFORMATION.
THE PARAMETERS EM[5], EM[7] AND EM[9] ARE GIVEN ITS VALUE IN THE PROCEDURE VECSYMTRI. FOR A POSSIBLY SUBSEQUENT CALL OF VECSYMTRI THE VALUE OF EM[5] IS NEEDED. WHEN CONSECUTIVE EIGENVALUES ARE TOO CLOSE TOGETHER, THE CORRESPONDING EIGENVECTORS ARE NOT NECESSARILY DELIVERED ORTHOGONAL BY INVERSE ITERATION (THE METHOD WHICH IS USED IN VECSYMTRI). THEREFORE GRAM-SCHMIDT ORTHOGONALISATION IS APPLIED ON THE EIGENVECTORS WHEN THE DISTANCE BETWEEN TWO CONSECUTIVE EIGENVALUES IS SMALLER THAN EM[4].
FOR FURTHER DETAILS ONE IS REFERRED TO THE SPECIFIC PROCEDURE DESCRIPTIONS.

SUBSECTION: EIGSYM1.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" EIGSYM1(A, N, NUMVAL, VAL, VEC, EM);
"VALUE" N, NUMVAL; "INTEGER" N, NUMVAL; "ARRAY" A, VAL, VEC, EM;

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

NUMVAL: <ARITHMETIC EXPRESSION>;
THE SERIAL NUMBER OF THE LAST EIGENVALUE TO BE CALCULATED;

A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:(N+1)*N//2];
ENTRY: THE UPPER TRIANGLE OF THE SYMMETRIC MATRIX MUST BE GIVEN IN SUCH A WAY THAT THE (I,J)-TH ELEMENT OF THE MATRIX IS A[(J-1)*J//2+I], 1 <= I <= J <= N;
EXIT: THE DATA FOR HOUSEHOLDER'S BACK TRANSFORMATION;

VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:NUMVAL];
EXIT: THE NUMVAL LARGEST EIGENVALUES IN MONOTONICALLY NON-INCREASING ORDER;

VEC: <ARRAY IDENTIFIER>;
"ARRAY" VEC[1:N,1:NUMVAL];
EXIT: THE NUMVAL CALCULATED EIGENVECTORS, STORED COLUMN-WISE, CORRESPONDING TO THE CALCULATED EIGENVALUES;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:9];
ENTRY: EM[0], THE MACHINE PRECISION,
EM[2], THE RELATIVE TOLERANCE FOR THE EIGENVALUES,
EM[4], THE ORTHOGONALISATION PARAMETER (SEE METHOD AND PERFORMANCE),
EM[6], THE TOLERANCE FOR THE EIGENVECTORS,
EM[8], THE MAXIMUM NUMBER OF INVERSE ITERATIONS ALLOWED FOR THE CALCULATION OF EACH EIGEN-VECTOR;
EXIT: EM[1], THE INFINITY NORM OF THE MATRIX,
EM[3], THE NUMBER OF ITERATIONS USED FOR CALCULATING THE NUMVAL EIGENVALUES,
EM[5], THE NUMBER OF EIGENVECTORS INVOLVED IN THE LAST GRAM-SCHMIDT ORTHOGONALISATION,
EM[7], THE MAXIMUM EUCLIDEAN NORM OF THE RESIDUES OF THE CALCULATED EIGENVECTORS,
EM[9], THE LARGEST NUMBER OF INVERSE ITERATIONS PERFORMED FOR THE CALCULATION OF SOME EIGEN-VECTOR.

PROCEDURES USED:

TFMSYMTRI1	=	CP34143,
VALSYMTRI	=	CP34151,
VECSYMTRI	=	CP34152,
BAKSYMTRI1	=	CP34144.

REQUIRED CENTRAL MEMORY:

EXECUTION FIELD LENGTH:

THREE ONE-DIMENSIONAL REAL ARRAYS OF LENGTH N ARE DECLARED; MOREOVER, VECSYMTRI AND BAKSYMTRI1 USE A TOTAL AMOUNT OF SIX ONE-DIMENSIONAL REAL ARRAYS OF LENGTH N AND ONE BOOLEAN ARRAY OF LENGTH N.

RUNNING TIME:

ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

THE BODY OF EIGSYM1 CONSISTS OF FOUR PROCEDURE STATEMENTS; THE FIRST IS A CALL OF TFMSYMTRI1 TO TRANSFORM THE SYMMETRIC MATRIX INTO A SIMILAR TRIDIAGONAL MATRIX BY MEANS OF HOUSEHOLDERS TRANSFORMATION,
THE SECOND IS A CALL OF VALSYMTRI TO CALCULATE THE DESIRED EIGENVALUES,
THE THIRD IS A CALL OF VECSYMTRI TO CALCULATE THE CORRESPONDING EIGENVECTORS AND
THE FOURTH IS A CALL OF BAKSYMTRI1 TO PERFORM THE BACK TRANSFORMATION.
FOR DETAILS ONE IS REFERRED TO EIGSYM2 OR TO THE DESCRIPTIONS OF THE FOUR PROCEDURES USED.

SUBSECTION: ORIVALSYM2.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" ORIVALSYM2(A, N, VAL, EM);
"VALUE" N; "INTEGER" N; "ARRAY" A, VAL, EM;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE UPPER TRIANGLE OF THE SYMMETRIC MATRIX MUST BE GIVEN IN THE UPPER TRIANGULAR PART OF A (THE ELEMENTS A[I,J], I<= J);
EXIT: THE DATA FOR HOUSEHOLDER'S BACK TRANSFORMATION (WHICH ISN'T USED BY THIS PROCEDURE) IS DELIVERED IN THE UPPER TRIANGULAR PART OF A;
THE ELEMENTS A[I,J] FOR I > J ARE NEITHER USED NOR CHANGED;
VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:N];
EXIT: THE EIGENVALUES OF THE MATRIX IN SOME ARBITRARY ORDER;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY: EM[0], THE MACHINE PRECISION,
EM[2], THE RELATIVE TOLERANCE FOR THE EIGENVALUES,
EM[4], THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
EXIT: EM[1], THE INFINITY NORM OF THE MATRIX,
EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE CODIAGONAL ELEMENTS NEGLECTED,
EM[5], THE NUMBER OF ITERATIONS PERFORMED;
MOREOVER:
ORIVALSYM2:= THE NUMBER OF EIGENVALUES NOT CALCULATED.

PROCEDURES USED:
TFMSYMTRI2 = CP34140,
ORIVALSYMTRI = CP34160.

REQUIRED CENTRAL MEMORY:
EXECUTION FIELD LENGTH:
IONAL REAL ARRAYS OF LENGTH N ARE USED.

RUNNING TIME:
ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE BODY OF ORIVALSYM2 CONSISTS OF TWO PROCEDURE STATEMENTS: THE FIRST IS A CALL OF TFMSYMTRI2 TO TRANSFORM THE SYMMETRIC MATRIX INTO A SIMILAR TRIDIAGONAL MATRIX BY MEANS OF HOUSEHOLDER'S TRANSFORMATION: THE SECOND IS A CALL OF ORIVALSYMTRI TO CALCULATE THE EIGENVALUES. WHEN THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS THEN ORIVALSYM2:= 0; OTHERWISE ORIVALSYM2:= THE NUMBER, K, OF EIGENVALUES NOT CALCULATED, EM[5]:= EM[4] + 1 AND ONLY THE LAST N - K ELEMENTS OF VAL ARE APPROXIMATE EIGENVALUES OF THE GIVEN MATRIX. OPERATION DETAILS OF BOTH PROCEDURES USED ARE GIVEN IN THEIR DESCRIPTION.

SUBSECTION: ORIVALSYM1.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" ORIVALSYM1(A, N, VAL, EM);
"VALUE" N; "INTEGER" N; "ARRAY" A, VAL, EM;

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:(N+1)*N//2];
ENTRY: THE UPPER TRIANGLE OF THE SYMMETRIC MATRIX MUST BE GIVEN IN SUCH A WAY THAT THE (I,J)-TH ELEMENT OF THE MATRIX IS A[(J-1)*J//2+I], 1 <= I <= J <= N;
EXIT: THE DATA FOR HOUSEHOLDER'S BACK TRANSFORMATION (WHICH ISN'T USED BY THIS PROCEDURE).

VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:N];
EXIT: THE EIGENVALUES OF THE MATRIX IN SOME ARBITRARY ORDER;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY: EM[0], THE MACHINE PRECISION,
EM[2], THE RELATIVE TOLERANCE FOR THE EIGENVALUES,
EM[4], THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
EXIT: EM[1], THE INFINITY NORM OF THE MATRIX,
EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE CODIAGONAL ELEMENTS NEGLECTED,
EM[5], THE NUMBER OF ITERATIONS PERFORMED;

MOREOVER:
ORIVALSYM1:= THE NUMBER OF EIGENVALUES NOT CALCULATED.

PROCEDURES USED:
TFMSYMTRI1 = CP34143.
ORIVALSYMTRI = CP34160.

REQUIRED CENTRAL MEMORY:
EXECUTION FIELD LENGTH:
TWO ONE-DIMENSIONAL REAL ARRAYS OF LENGTH N ARE USED.

RUNNING TIME:
ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE BODY OF QRIVALSYM1 CONSISTS OF TWO PROCEDURE STATEMENTS: THE FIRST IS A CALL OF TFMSYMTRI1 TO TRANSFORM THE SYMMETRIC MATRIX INTO A SIMILAR TRIDIAGONAL MATRIX BY MEANS OF HOUSEHOLDER'S TRANSFORMATION; THE SECOND IS A CALL OF QRIVALSYMTRI TO CALCULATE THE EIGENVALUES. WHEN THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS THEN QRIVALSYM1:= 0; OTHERWISE QRIVALSYM1:= THE NUMBER, K, OF EIGENVALUES NOT CALCULATED, EM[5]:= EM[4] + 1 AND ONLY THE LAST N - K ELEMENTS OF VAL ARE APPROXIMATE EIGENVALUES OF THE GIVEN MATRIX. OPERATION DETAILS OF BOTH PROCEDURES USED ARE GIVEN IN THEIR DESCRIPTION.

SUBSECTION: QRISYM.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" QRISYM(A, N, VAL, EM);
"VALUE" N; "INTEGER" N; "ARRAY" A, VAL, EM;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE UPPER TRIANGLE OF THE SYMMETRIC MATRIX MUST BE GIVEN IN THE UPPER TRIANGULAR PART OF A (THE ELEMENTS A[I,J], I<= J);
EXIT: THE EIGENVECTORS OF THE SYMMETRIC MATRIX, STORED COLUMNWISE;
VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:N];
EXIT: THE EIGENVALUES OF THE MATRIX CORRESPONDING TO THE CALCULATED EIGENVECTORS;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY: EM[0], THE MACHINE PRECISION,
EM[2], THE RELATIVE TOLERANCE FOR THE QR ITERATION,
EM[4], THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
EXIT: EM[1], THE INFINITY NORM OF THE MATRIX,
EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE CODIAGONAL ELEMENTS NEGLECTED,
EM[5], THE NUMBER OF ITERATIONS PERFORMED;
MOREOVER:
QRISYM := THE NUMBER OF EIGENVALUES AND -VECTORS NOT CALCULATED.

PROCEDURES USED:

TFMSYMTRI2	=	CP34140,
TFMPREVEC	=	CP34142,
QRISYMTRI	=	CP34161.

REQUIRED CENTRAL MEMORY:

EXECUTION FIELD LENGTH:
TWO ONE-DIMENSIONAL REAL ARRAYS OF LENGTH N ARE USED.

RUNNING TIME:
PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE BODY OF QRISYM CONSISTS OF THREE PROCEDURE STATEMENTS; THE FIRST IS A CALL OF TFMSYMTRI2 TO TRANSFORM THE SYMMETRIC MATRIX INTO A SIMILAR TRIDIAGONAL MATRIX BY MEANS OF HOUSEHOLDER'S TRANSFORMATION, THE SECOND IS A CALL OF TFMPREVEC TO PERFORM THE DESIRED BACK TRANSFORMATION ON THE EIGENVECTORS IN ADVANCE AND THE THIRD IS A CALL OF QRISYMTRI TO CALCULATE THE EIGENVALUES AND THE EIGENVECTORS. WHEN THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS THEN QRISYM:= 0; OTHERWISE QRISYM:= THE NUMBER, K, OF EIGENVALUES AND -VECTORS NOT CALCULATED, EM[5]:= EM[4] + 1 AND ONLY THE LAST N - K ELEMENTS OF VAL AND THE LAST N - K COLUMNS OF A ARE APPROXIMATE EIGENVALUES AND EIGENVECTORS RESPECTIVELY OF THE GIVEN MATRIX. OPERATION DETAILS OF THE PROCEDURES USED ARE GIVEN IN THEIR DESCRIPTION.

EXAMPLES OF USE:

THE TWO LARGEST EIGENVALUES IN MONOTONICALLY NON INCREASING ORDER AND THE CORRESPONDING EIGENVECTORS OF M, WITH N = 4 AND M[I,J] = 1 / (I + J - 1), MAY BE OBTAINED BY THE FOLLOWING PROGRAM:

```
"BEGIN"
    "INTEGER" I, J;
    "ARRAY" A[1:10], VAL[1:2], EM[0:9], VEC[1:4,1:2];
    "PROCEDURE" EIGSYM1(A, N, NUMVAL, VAL, VEC, EM); "CODE" 34156;

    EM[0]:= "-14; EM[2]:= "-12; EM[4]:= "-3;
    EM[6]:= 10"-10; EM[8]:= 5;
    "FOR" I:= 1 "STEP" 1 "UNTIL" 4 "DO"
    "FOR" J:= I "STEP" 1 "UNTIL" 4 "DO"
    A[(J * J - J) / 2 + I]:= 1 / (I + J - 1);
    EIGSYM1(A, 4, 2, VAL, VEC, EM);
    OUTPUT(61, "("2(+.13D"+2D, 2B), 2/")", VAL[1], VAL[2]);
    "FOR" I:= 1, 2, 3, 4 "DO"
    OUTPUT(61, "("2(+.13D"+2D, 2B), /")", VEC[I,1], VEC[I,2]);
    OUTPUT(61, "("2(.2D"+2D, /), 3(2ZD, /)")",
    EM[1], EM[7], EM[3], EM[5], EM[9])
"END"
```

```
    THE PROGRAM DELIVERS(THE RESULTS ARE CORRECT UP TO TWELVE DIGITS):

    THE EIGENVALUES:  +.1500214280059"+01   +.1691412202214"+00

    THE EIGENVECTORS: -.7926082911638"+00   +.5820756994972"+00
                      -.4519231209016"+00   -.3705021850671"+00
                      -.3224163985818"+00   -.5095786345018"+00
                      -.2521611696882"+00   -.5140482722222"+00

    EM[1] = .21"+01
    EM[7] = .92"-14
    EM[3] = 32
    EM[5] = 1
    EM[9] = 1 .


    THE TWO LARGEST EIGENVALUES OF M, WITH N = 4 AND M[I,J] =
    1 / (I + J - 1), MAY BE OBTAINED IN MONOTONICALLY NON INCREASING
    ORDER BY THE FOLLOWING PROGRAM:

    "BEGIN"
        "INTEGER" I, J;
        "ARRAY" A[1:4,1:4], VAL[1:2], EM[0:3];
        "PROCEDURE" EIGVALSYM2(A, N, NUMVAL, VAL, EM): "CODE" 34153;

        EM[0]:= "-14; EM[2]:= "-12;
        "FOR" I:= 1 "STEP" 1 "UNTIL" 4 "DO"
        "FOR" J:= I "STEP" 1 "UNTIL" 4 "DO" A[I,J]:= 1 / (I + J -1);
        EIGVALSYM2(A, 4, 2, VAL, EM);
        OUTPUT(61, "("2(+.13D"+2D, 2B)")", VAL[1], VAL[2]);
        OUTPUT(61, "("2/, .2D"+2D, /, 2ZD")", EM[1], EM[3])
    "END"

    THE PROGRAM DELIVERS(THE RESULTS ARE CORRECT UP TO TWELVE DIGITS):

    THE EIGENVALUES: +.1500214280059"+01  +.1691412202214"+00
    EM[3] = .21"+01
    EM[1] = 32  .
```

SOURCE TEXT(S) :

```
"CODE" 34153;
"COMMENT" MCA 2313;
"PROCEDURE" EIGVALSYM2(A, N, NUMVAL, VAL, EM); "VALUE" N, NUMVAL;
"INTEGER" N, NUMVAL; "ARRAY" A, VAL, EM;
"BEGIN" "ARRAY" B, BB, D[1:N];

        "PROCEDURE" TFMSYMTRI2(A, N, D, B, BB, EM); "CODE" 34140;
        "PROCEDURE" VALSYMTRI(D, BB, N, N1, N2, VAL, EM); "CODE" 34151;

        TFMSYMTRI2(A, N, D, B, BB, EM);
        VALSYMTRI(D, BB, N, 1, NUMVAL, VAL, EM)
    "END" EIGVALSYM2;
        "EOP"

"CODE" 34154;
"COMMENT" MCA 2314;
"PROCEDURE" EIGSYM2(A, N, NUMVAL, VAL, VEC, EM); "VALUE" N, NUMVAL;
"INTEGER" N, NUMVAL; "ARRAY" A, VAL, VEC, EM;
"BEGIN" "ARRAY" B, BB, D[1:N];

        "PROCEDURE" TFMSYMTRI2(A, N, D, B, BB, EM); "CODE" 34140;
        "PROCEDURE" VALSYMTRI(D, BB, N, N1, N2, VAL, EM); "CODE" 34151;
        "PROCEDURE" VECSYMTRI(D, B, N, N1, N2, VAL, VEC, EM);
        "CODE" 34152;
        "PROCEDURE" BAKSYMTRI2(A, N, N1, N2, VEC); "CODE" 34141;

        TFMSYMTRI2(A, N, D, B, BB, EM);
        VALSYMTRI(D, BB, N, 1, NUMVAL, VAL, EM);
        VECSYMTRI(D, B, N, 1, NUMVAL, VAL, VEC, EM);
        BAKSYMTRI2(A, N, 1, NUMVAL, VEC)
    "END" EIGSYM2;
        "EOP"

"CODE" 34155;
"COMMENT" MCA 2318;
"PROCEDURE" EIGVALSYM1(A, N, NUMVAL, VAL, EM); "VALUE" N, NUMVAL;
"INTEGER" N, NUMVAL; "ARRAY" A, VAL, EM;
"BEGIN" "ARRAY" B, BB, D[1:N];

        "PROCEDURE" TFMSYMTRI1(A, N, D, B, BB, EM); "CODE" 34143;
        "PROCEDURE" VALSYMTRI(D, BB, N, N1, N2, VAL, EM); "CODE" 34151;

        TFMSYMTRI1(A, N, D, B, BB, EM);
        VALSYMTRI(D, BB, N, 1, NUMVAL, VAL, EM)
    "END" EIGVALSYM1;
        "EOP"
```

```
"CODE" 34156;
"COMMENT" MCA 2319;
"PROCEDURE" EIGSYM1(A, N, NUMVAL, VAL, VEC, EM); "VALUE" N, NUMVAL;
"INTEGER" N, NUMVAL; "ARRAY" A, VAL, VEC, EM;
"BEGIN" "ARRAY" B, BB, D[1:N];

        "PROCEDURE" TFMSYMTRI1(A, N, D, B, BB, EM); "CODE" 34143;
        "PROCEDURE" VALSYMTRI(D, BB, N, N1, N2, VAL, EM); "CODE" 34151;
        "PROCEDURE" VECSYMTRI(D, B, N, N1, N2, VAL, VEC, EM);
        "CODE" 34152;
        "PROCEDURE" BAKSYMTRI1(A, N, N1, N2, VEC); "CODE" 34144;

        TFMSYMTRI1(A, N, D, B, BB, EM);
        VALSYMTRI(D, BB, N, 1, NUMVAL, VAL, EM);
        VECSYMTRI(D, B, N, 1, NUMVAL, VAL, VEC, EM);
        BAKSYMTRI1(A, N, 1, NUMVAL, VEC)
    "END" EIGSYM1;
        "EOP"

"CODE" 34162;
"COMMENT" MCA 2322;
"INTEGER" "PROCEDURE" QRIVALSYM2(A, N, VAL, EM); "VALUE" N;
"INTEGER" N; "ARRAY" A, VAL, EM;
"BEGIN" "ARRAY" B, BB[1:N];

        "PROCEDURE" TFMSYMTRI2(A, N, D, B, BB, EM); "CODE" 34140;
        "INTEGER" "PROCEDURE" QRIVALSYMTRI(D, BB, N, EM);
        "CODE" 34160;

        TFMSYMTRI2(A, N, VAL, B, BB, EM);
        QRIVALSYM2:= QRIVALSYMTRI(VAL, BB, N, EM)
    "END" QRIVALSYM2;
        "EOP"
```

```
"CODE" 34163;
"COMMENT" MCA 2323;
"INTEGER" "PROCEDURE" QRISYM(A, N, VAL, EM); "VALUE" N;
"INTEGER" N; "ARRAY" A, VAL, EM;
"BEGIN" "ARRAY" B, BB[1:N];

        "PROCEDURE" TFMSYMTRI2(A, N, D, B, BB, EM); "CODE" 34140;
        "PROCEDURE" TFMPREVEC(A, N); "CODE" 34142;
        "INTEGER" "PROCEDURE" QRISYMTRI(A, N, D, B, BB, EM);
        "CODE" 34161;

        TFMSYMTRI2(A, N, VAL, B, BB, EM); TFMPREVEC(A, N);
        QRISYM:= QRISYMTRI(A, N, VAL, B, BB, EM)
    "END" QRISYM;
        "EOP"

"CODE" 34164;
"COMMENT" MCA 2327;
"INTEGER" "PROCEDURE" QRIVALSYM1(A, N, VAL, EM); "VALUE" N;
"INTEGER" N; "ARRAY" A, VAL, EM;
"BEGIN" "ARRAY" B, BB[1 : N];

        "PROCEDURE" TFMSYMTRI1(A, N, D, B, BB, EM); "CODE" 34143;
        "INTEGER" "PROCEDURE" QRIVALSYMTRI(D, BB, N, EM);
        "CODE" 34160;

        TFMSYMTRI1(A, N, VAL, B, BB, EM);
        QRIVALSYM1:= QRIVALSYMTRI(VAL, BB, N, EM)
    "END" QRIVALSYM1;
        "EOP"
```

AUTHORS: J.J.G. ADMIRAAL, A.C. YSSELSTEIN.

CONTRIBUTOR: J.J.G. ADMIRAAL.

INSTITUTE: UNIVERSITY OF AMSTERDAM.

RECEIVED: 761101.

BRIEF DESCRIPTION:
THIS SECTION CONTAINS THREE PROCEDURES FOR SORTING THE ELEMENTS OF A VECTOR AND CORRESPONDINGLY PERMUTING THE ELEMENTS OF A VECTOR OR A MATRIX ROW.
A) MERGESORT DELIVERS A PERMUTATION OF INDICES CORRESPONDING TO SORTING THE ELEMENTS OF A GIVEN VECTOR INTO NON-DECREASING ORDER.
B) VECPERM PERMUTES THE ELEMENTS OF A GIVEN VECTOR CORRESPONDING TO A GIVEN PERMUTATION OF INDICES.
C) ROWPERM PERMUTES THE ELEMENTS OF A GIVEN ROW OF A MATRIX CORRESPONDING TO A GIVEN PERMUTATION OF INDICES.

KEYWORDS:
SORTING.
PERMUTING.

SUBSECTION: MERGESORT.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" MERGESORT(VEC1,VEC2,LOW,UPP);
"VALUE" LOW,UPP;"INTEGER" LOW,UPP;"ARRAY" VEC1,VEC2;
"CODE" 36405;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
VEC1   :<ARRAY IDENTIFIER>;
        "ARRAY" VEC1[LOW:UPP];
        ENTRY: THE VECTOR TO BE SORTED INTO
               NONDECREASING ORDER;
        EXIT: THE CONTENTS OF VEC1 ARE LEFT INVARIANT;
VEC2   :<ARRAY IDENTIFIER>;
        "INTEGER""ARRAY" VEC2[LOW:UPP];
        EXIT: THE PERMUTATION OF INDICES CORRESPONDING TO
              SORTING THE ELEMENTS OF VEC1 INTO
              NON-DECREASING ORDER;
LOW    : <ARITHMETIC EXPRESSION>;
        THE LOWER INDEX OF THE ARRAYS VEC1 AND VEC2;
UPP    : <ARITHMETIC EXPRESSION>;
        THE UPPER INDEX OF THE ARRAYS VEC1 AND VEC2;
```

PROCEDURES USED: NONE.

REQUIRED CENTRAL MEMORY: ONE LOCAL INTEGER ARRAY OF
LENGTH N,WHERE N = UPP - LOW + 1.

RUNNING TIME: AVERAGE PROPORTIONAL TO N * LN(N).

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SORTING BY MERGING. ([1],[2])

EXAMPLE OF USE: THE PROCEDURE MERGESORT IS USED IN SYMEIGIMP
(SECTION 3.3.1.1.3.3).

SUBSECTION: VECPERM.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" VECPERM(PERM,LOW,UPP,VECTOR);
"VALUE" LOW,UPP; "INTEGER" LOW,UPP;
"INTEGER" "ARRAY" PERM;"ARRAY" VECTOR;
"CODE" 36404;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
PERM   :<ARRAY IDENTIFIER>;
        "INTEGER" "ARRAY" PERM[LOW:UPP];
        ENTRY: A GIVEN PERMUTATION (E.G.AS PRODUCED BY
        MERGESORT) OF THE NUMBERS IN THE ARRAY VECTOR;
LOW    :<ARITHMETIC EXPRESSION>;
        THE LOWER INDEX OF THE ARRAYS PERM AND VECTOR;
UPP    :<ARITHMETIC EXPRESSION>;
        THE UPPER INDEX OF THE ARRAYS PERM AND VECTOR;
VECTOR :<ARRAY IDENTIFIER>;
        "ARRAY" VECTOR[LOW:UPP];
        ENTRY: THE REAL VECTOR TO BE PERMUTED;
        EXIT: THE PERMUTED VECTOR ELEMENTS.
```

PROCEDURE USED: NONE.

REQUIRED CENTRAL MEMORY :
ONE LOCAL BOOLEAN ARRAY OF LENGTH N IS DECLARED.

RUNNING TIME: PROPORTIONAL TO N.

LANGUAGE: ALGOL 60.

EXAMPLE OF USE: THE PROCEDURE VECPERM IS USED IN SYMEIGIMP;
(SECTION 3.3.1.1.3.3).

SUBSECTION: ROWPERM.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" ROWPERM(PERM,LOW,UPP,I,MATRIX);
"VALUE" LOW,UPP,I;"INTEGER" LOW,UPP,I;
"INTEGER" "ARRAY" PERM;"ARRAY" MATRIX;
"CODE" 36403;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
PERM   :<ARRAY IDENTIFIER>;
        "INTEGER" "ARRAY" PERM[LOW:UPP];
        ENTRY: A GIVEN PERMUTATION (E.G. AS PRODUCED BY
        MERGESORT) OF THE NUMBERS IN THE ARRAY VECTOR;
LOW    :<ARITHMETIC EXPRESSION>;
        THE LOWER INDEX OF THE ARRAY PERM;
UPP    :<ARITHMETIC EXPRESSION>;
        THE UPPER INDEX OF THE ARRAY PERM;
I      :<ARITHMETIC EXPRESSION>;
        THE ROW INDEX OF THE MATRIX ELEMENTS;
MATRIX :<ARRAY IDENTIFIER>;
        "ARRAY" MATRIX [I :I, LOW : UP];
        ENTRY:MATRIX [I, LOW : UP] SHOULD CONTAIN THE
        ELEMENTS TO BE PERMUTED;
        EXIT: MATRIX[I, LOW : UP] CONTAINS THE ROW OF
        PERMUTED ELEMENTS;
```

PROCEDURES USED: NONE.

REQUIRED CENTRAL MEMORY:
ONE LOCAL BOOLEAN ARRAY OF LENGTH N IS DECLARED.

RUNNING TIME: PROPORTIONAL TO N, WHERE N = UPP - LOW + 1.

LANGUAGE: ALGOL 60.

EXAMPLE OF USE: THE PROCEDURE ROWPERM IS USED IN SYMEIGIMP.
(SECTION 3.3.1.1.3.3).

REFERENCES:

[1] D.E. KNUTH , THE ART OF COMPUTER PROGRAMMING,
VOL. 3/ SORTING AND SEARCHING,ADDISON-WESLEY 1973.
(SECTION 5.2.4 P.159-173).

[2] A.V. AHO, J.E. HOPCROFT & J.D. ULLMAN,
THE DESIGN AND ANALYSIS OF COMPUTER ALGORITHMS,
ADDISON-WESLEY 1974.
(SECTION I.M, P65-67).

SOURCE TEXTS:

```
CODE" 36405;
"PROCEDURE" MERGESORT(A,P,LOW,UP);"VALUE" LOW,UP;
"INTEGER" LOW,UP;"ARRAY" A;"INTEGER" "ARRAY" P;
"BEGIN" "INTEGER" I,L,R,PL,PR,LO,STEP,STAP,UMLP1,UMSP1,REST,RESTV;
    "BOOLEAN" ROUT,LOUT; "INTEGER" "ARRAY" HP[LOW:UP];
    "PROCEDURE" MERGE(LO,LS,RS);"VALUE" LO,LS,RS;"INTEGER" LO,LS,RS;
    "BEGIN" L:=LO; R:=LO+LS; LOUT:=ROUT:="FALSE";
        "FOR" I:=LO,I+1 "WHILE" ^(LOUT "OR" ROUT) "DO"
        "BEGIN" PL:=P[L];PR:=P[R];"IF" A[PL]>A[PR] "THEN"
            "BEGIN" HP[I]:=PR;R:=R+1;ROUT:=R=LO+LS+RS "END" "ELSE"
            "BEGIN" HP[I]:=PL;L:=L+1;LOUT:=L=LO+LS     "END"
        "END" FOR I;
        "IF" ROUT "THEN"
        "BEGIN" "FOR" I:=LO+LS-1 "STEP" -1 "UNTIL" L "DO"
            P[I+RS]:=P[I];R:=L+RS
        "END";
        "FOR" I:=R-1 "STEP" -1 "UNTIL" LO "DO" P[I]:=HP[I];
    "END" MERGE;
    "FOR" I:=LOW "STEP" 1 "UNTIL" UP "DO" P[I]:=I;RESTV:=0;
    UMLP1:=UP-LOW+1;
    "FOR" STEP:=1, STEP*2 "WHILE" STEP < UMLP1 "DO"
    "BEGIN" STAP:=2*STEP;UMSP1:=UP-STAP+1;
        "FOR" LO:=LOW "STEP" STAP "UNTIL" UMSP1 "DO"
        MERGE(LO,STEP,STEP); REST:=UP-LO+1;
        "IF" REST>RESTV & RESTV>0 "THEN" MERGE(LO,REST-RESTV,RESTV);
        RESTV:=REST
    "END" FOR STEP
"END" MERGESORT;
```

```
"CODE" 36404;
"PROCEDURE" VECPERM(PERM,LOW,UPP,VECTOR);"VALUE" LOW,UPP;
"INTEGER" LOW,UPP;"INTEGER" "ARRAY" PERM;"REAL" "ARRAY" VECTOR;
"BEGIN" "INTEGER" T,J,K;"REAL" A;"BOOLEAN" "ARRAY" TODO[LOW:UPP];
    "FOR" T:=LOW "STEP" 1 "UNTIL" UPP "DO" TODO[T]:="TRUE";
    "FOR" T:=LOW "STEP" 1 "UNTIL" UPP "DO"
    "BEGIN" "IF" TODO[T] "THEN"
        "BEGIN" K:=T;A:=VECTOR[K];
            "FOR" J:=PERM[K] "WHILE" J^=T "DO"
            "BEGIN" VECTOR[K]:=VECTOR[J];TODO[K]:="FALSE";K:=J
            "END";VECTOR[K]:=A;TODO[K]:="FALSE"
        "END" CYCLE;
    "END" FOR T;
"END" VECPERM;

"CODE" 36403;
"PROCEDURE" ROWPERM(PERM,LOW,UPP,I,MAT);"VALUE" LOW,UPP,I;
"INTEGER" LOW,UPP,I;"INTEGER" "ARRAY" PERM;"REAL" "ARRAY" MAT;
"BEGIN" "INTEGER" T,J,K;"REAL" A;"BOOLEAN" "ARRAY" TODO[LOW:UPP];
    "FOR" T:=LOW "STEP" 1 "UNTIL" UPP "DO" TODO[T]:="TRUE";
    "FOR" T:=LOW "STEP" 1 "UNTIL" UPP "DO"
    "BEGIN" "IF" TODO[T] "THEN"
        "BEGIN" K:=T;A:=MAT[I,K];
            "FOR" J:=PERM[K] "WHILE" J^=T "DO"
            "BEGIN" MAT[I,K]:=MAT[I,J];TODO[K]:="FALSE";K:=J
            "END";MAT[I,K]:=A;TODO[K]:="FALSE"
        "END" CYCLE;
    "END" FOR T;
"END" ROWPERM;
```

AUTHOR/CONTRIBUTOR: J.J.G. ADMIRAAL.

INSTITUTE: UNIVERSITY OF AMSTERDAM.

RECEIVED: 761101.

BRIEF DESCRIPTION:
THE PROCEDURE ORTHOG ORTHOGONALIZES SOME ADJACENT MATRIX COLUMNS
ACCORDING TO THE MODIFIED GRAM SCHMIDT METHOD (SEE [1]).

KEYWORDS:
MATRIX COLUMNS,
MODIFIED GRAM SCHMIDT ORTHOGONALIZATION.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" ORTHOG(N,LC,UC,X);
"VALUE" N,LC,UC; "INTEGER" N,LC,UC;"ARRAY" X;
"CODE" 36402;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N     :<ARITHMETIC EXPRESSION>;
       THE ORDER OF THE MATRIX X;
LC    :<ARITHMETIC EXPRESSION>;
       THE LOWER COLUMN INDEX OF THE MATRIX COLUMNS;
UC    :<ARITHMETIC EXPRESSION>;
       THE UPPER COLUMN INDEX OF THE MATRIX COLUMNS;
X     :<ARRAY IDENTIFIER>;
       "ARRAY" X[1:N,LC:UC];
       ENTRY: THE MATRIX COLUMNS,TO BE
              ORTHOGONALIZED;
       EXIT: THE ORTHOGONALIZED MATRIX COLUMNS.
```

PROCEDURES USED:
TAMMAT = CP34014,
ELMCOL = CP34023.

REQUIRED CENTRAL MEMORY: NO LOCAL ARRAYS ARE DECLARED.

RUNNING TIME: PROPORTIONAL TO N**3.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE MODIFIED GRAM SCHMIDT METHOD (SEE [1],CHAPTER 4.54).

EXAMPLE OF USE: THE PROCEDURE ORTHOG IS USED IN SYMEIGIMP.
(SECTION 3.3.1.1.3.3).

REFERENCES:
[1] J.H. WILKINSON.
THE ALGEBRAIC EIGENVALUE PROBLEM.
CLARENDON PRESS,OXFORD,1965.

SOURCE TEXT:

```
"CODE" 36402;
"PROCEDURE" ORTHOG(N,LC,UC,X);"VALUE" N,LC,UC;
"INTEGER" N,LC,UC;"ARRAY" X;
"BEGIN" "INTEGER" I,J,K; "REAL" NORMX;
    "REAL" "PROCEDURE" TAMMAT(L,U,I,J,A,B); "CODE" 34014;
    "PROCEDURE" ELMCOL(L,U,I,J,A,B,X); "CODE" 34023;
    "FOR" J:=LC "STEP" 1 "UNTIL" UC "DO"
    "BEGIN" NORMX:=SQRT(TAMMAT(1,N,J,J,X,X));
        "FOR" I:=1 "STEP" 1 "UNTIL" N "DO" X[I,J]:=X[I,J]/NORMX;
        "FOR" K:=J+1 "STEP" 1 "UNTIL" UC "DO"
        ELMCOL(1,N,K,J,X,X,-TAMMAT(1,N,K,J,X,X))
    "END"
"END" ORTHOG;
```

AUTHOR/CONTRIBUTOR: J.J.G. ADMIRAAL.

INSTITUTE: UNIVERSITY OF AMSTERDAM.

RECEIVED: 761101.

BRIEF DESCRIPTION:
THE PROCEDURE SYMEIGIMP IMPROVES A GIVEN APPROXIMATION OF A REAL SYMMETRIC EIGENSYSTEM AND CALCULATES ERROR BOUNDS FOR THE EIGENVALUES.

KEYWORDS:
EIGENVALUES.
EIGENVECTORS.
SYMMETRIC MATRIX.
RAYLEIGH QUOTIENTS.
ERROR BOUNDS.
IMPROVED EIGENSYSTEM.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS :

```
"PROCEDURE" SYMEIGIMP(N,A,VEC,VAL1,VAL2,LBOUND,UBOUND,AUX);
"VALUE" N;"INTEGER" N;
"ARRAY" A,VEC,VAL1,VAL2,LBOUND,UBOUND,AUX;
"CODE" 36401;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N:      <ARITHMETIC EXPRESSION>;
        THE ORDER OF MATRIX A;
A:      <ARRAY IDENTIFIER>;
        "ARRAY" A[1:N,1:N] CONTAINS A REAL SYMMETRIC MATRIX
        WHOSE EIGENSYSTEM HAS TO BE IMPROVED;
VEC:    <ARRAY IDENTIFIER>;
        "ARRAY" VEC[1:N,1:N] CONTAINS A MATRIX WHOSE COLUMNS ARE
        A SYSTEM OF APPROXIMATE EIGENVECTORS OF MATRIX A;
        ENTRY: INITIAL APPROXIMATIONS;
        EXIT: IMPROVED APPROXIMATIONS;
VAL1:   <ARRAY IDENTIFIER>;
        "ARRAY" VAL1[1:N];
        ENTRY: INITIAL APPROXIMATIONS OF THE EIGENVALUES OF A;
        EXIT: THE HEAD PARTS OF THE DOUBLE PRECISION IMPROVED
        APPROXIMATIONS OF THE EIGENVALUES OF A;
VAL2:   <ARRAY IDENTIFIER>;
        "ARRAY" VAL2[1:N];
        EXIT: THE TAIL PARTS OF THE DOUBLE PRECISION
              IMPROVED EIGENVALUES OF A;
```

```
    LBOUND,
    UBOUND: <ARRAY IDENTIFIER>;
            EXIT: "ARRAY" LBOUND, UBOUND [1:N] CONTAIN THE LOWER
                  AND UPPER ERRORBOUNDS RESPECTIVELY FOR THE EIGENVALUE
                  APPROXIMATIONS IN VAL1,VAL2[1:N] SUCH THAT THE
                  I-TH EXACT EIGENVALUE LIES BETWEEN VAL1[I]+VAL2[I]
                  -LBOUND[I] AND VAL1[I]+VAL2[I]+UBOUND[I];
    AUX:    <ARRAY IDENTIFIER>;
            "ARRAY" AUX[0:5];
            ENTRY: AUX[0]= THE RELATIVE PRECISION OF THE ELEMENTS OF A;
                   AUX[2]= THE RELATIVE TOLERANCE FOR THE RESIDUAL MATRIX;
                           THE ITERATION ENDS WHEN THE MAXIMUM ABSOLUTE
                           VALUE OF THE RESIDU ELEMENTS IS SMALLER THAN
                           AUX[2]*AUX[1].
                   AUX[4]= THE MAXIMUM NUMBER OF ITERATIONS ALLOWED;
            EXIT:  AUX[1]= INFINITY NORM OF THE MATRIX A;
                   AUX[3]= MAXIMUM ABSOLUTE ELEMENT OF THE RESIDUAL MATRIX;
                   AUX[5]= NUMBER OF ITERATIONS;

PROCEDURES USED:
    LNGMATVEC = CP34411,
    LNGMATMAT = CP34413,
    LNGTAMMAT = CP34414,
    VECVEC    = CP34010,
    MATMAT    = CP34013,
    TAMMAT    = CP34014,
    MERGESORT = CP36405,
    VECPERM   = CP36404,
    ROWPERM   = CP36403,
    ORTHOG    = CP36402,
    QRISYM    = CP34163,
    INFNRMMAT = CP31064.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

REQUIRED CENTRAL MEMORY:
    AUXILIARY ARRAYS ARE DECLARED TO A TOTAL OF 3*N*N + 6*N REALS
    AND N INTEGERS; MOREOVER, N INTEGERS OR N BOOLEANS ARE USED
    BY MERGESORT, VECPERM AND ROWPERM.

METHOD AND PERFORMANCE: SEE[1].
```

REFERENCES:

[1]. J.J.G. ADMIRAAL.
ITERATIEF VERBETEREN VAN REEEL SYMMETRISCH EIGENSYSTEEM
EN BEREKENEN VAN FOUTGRENZEN VOOR DE VERKREGEN EIGENWAARDEN.
DOCTORAL SCRIPTION,MARCH 1976,
UNIVERSITEIT VAN AMSTERDAM.

[2]. R.T. GREGORY AND D.L. KARNEY.
A COLLECTION OF MATRICES FOR TESTING COMPUTATIONAL
ALGORITHMS,
WILEY-INTERSCIENCE, 1969.

EXAMPLE OF USE.

```
"BEGIN" "INTEGER" I,J;"REAL" S;
   "ARRAY" A,X[1:4,1:4],VAL1,VAL2,LBOUND,UBOUND[1:4],EM,AUX[0:5];
   "PROCEDURE" SYMEIGIMP(N,A,X,VAL1,VAL2,LBOUND,UBOUND,AUX);
   "CODE" 36401;
   "INTEGER" "PROCEDURE" ORISYM(A,N,VAL,EM);"CODE" 34163;
   A[1,1]:=A[2,2]:=A[3,3]:=A[4,4]:=6;
   A[1,2]:=A[2,1]:=A[3,1]:=A[1,3]:=4;
   A[4,2]:=A[2,4]:=A[3,4]:=A[4,3]:=4;
   A[1,4]:=A[4,1]:=A[3,2]:=A[2,3]:=1;
   "FOR" I:=1 "STEP" 1 "UNTIL" 4 "DO"
   "FOR" J:=I "STEP" 1 "UNTIL" 4 "DO" X[I,J]:=X[J,I]:=A[I,J];
   OUTPUT(61,"(""("A")",/,4(4(+DB),/)")",A);
   EM[0]:="-14;EM[4]:=100;EM[2]:="-5;
   ORISYM(X,4,VAL1,EM);
   AUX[0]:=0;AUX[4]:=10;AUX[2]:="-14;
   SYMEIGIMP(4,A,X,VAL1,VAL2,LBOUND,UBOUND,AUX);
   OUTPUT(61,"("/,"("THE EXACT EIGENVALUES ARE: -1 , +5 , +5 , +15")",
   //,
   "("THE DIFFERENCES BETWEEN THE CALCULATED AND THE EXACT EIGENVALUES"
   )",//,4(N,/)")",(VAL1[1]+1)+VAL2[1],(VAL1[2]-5)+VAL2[2],(VAL1[3]-
   5)+VAL2[3],(VAL1[4]-15)+VAL2[4]);
   OUTPUT(61,"("/,"("LOWERBOUNDS  UPPERBOUNDS")",//")");
   "FOR" I:=1 "STEP" 1 "UNTIL" 4 "DO"
   OUTPUT(61,"("2(+D.D"+DD5B),/")",LBOUND[I],UBOUND[I]);
   OUTPUT(61,"("/,"("NUMBER OF ITERATIONS = ")",ZD//,
   "("INFINITY NORM OF A = ")",ZD//,
   "("MAXIMUM ABSOLUTE ELEMENT OF RESIDU = ")",D.D"+DD")",
   AUX[5],AUX[1],AUX[3])
"END"  EXAMPLE OF USE
```

DELIVERS:

A
+6 +4 +4 +1
+4 +6 +1 +4
+4 +1 +6 +4
+1 +4 +4 +6

THE EXACT EIGENVALUES ARE: -1 , +5 , +5 , +15

THE DIFFERENCES BETWEEN THE CALCULATED AND THE EXACT EIGENVALUES

-6.3423147029256"-022
+5.5934784498910"-018
+4.0389678347316"-028
-5.5947317864427"-018

LOWERBOUNDS	UPPERBOUNDS
+1.2"-23	+1.2"-23
+7.5"-09	+7.5"-09
+1.0"-13	+1.0"-13
+5.6"-18	+5.6"-18

NUMBER OF ITERATIONS = 2

INFINITY NORM OF A = 15

MAXIMUM ABSOLUTE ELEMENT OF RESIDU = 2.8"-14

```
SOURCETEXT:

"CODE" 36401;
"PROCEDURE" SYMEIGIMP(N,A,VEC,VAL1,VAL2,LBOUND,UBOUND,AUX);
"VALUE" N;"INTEGER" N;"ARRAY" A,VEC,VAL1,VAL2,LBOUND,UBOUND,AUX;
"BEGIN"
    "PROCEDURE" ORTHOG(N,LC,UC,X);"CODE" 36402;
    "PROCEDURE" MERGESORT(VEC1,VEC2,LOW,UPP);"CODE" 36405;
    "PROCEDURE" VECPERM(PERM,LOW,UPP,VECTOR);"CODE" 36404;
    "PROCEDURE" ROWPERM(PERM,LOW,UPP,MATRIX);"CODE" 36403;
    "INTEGER" K,I,J,IO,I1,ITER,MAXITP1;"REAL" S,HEAD,TAIL,MAX,TOL,
    MATEPS,RELERRA,RELTOLR,NORMA;"INTEGER" "ARRAY" PERM[1:N];
    "ARRAY" R,P,Y[1:N,1:N],RQ,RQT,EPS,Z,VAL3,ETA[1:N];
    "PROCEDURE" BOUNDS(IO,I1,N,LBOUND,UBOUND);"VALUE" IO,I1,N;
    "INTEGER" IO,I1,N;"ARRAY" LBOUND,UBOUND;
    "BEGIN" "INTEGER" K,I,J,IO1;"REAL" EPS2,DL,DR;
        "FOR" I:=IO,IO1 "WHILE" I<=I1 "DO"
        "BEGIN" J:=IO1:=I;
            "FOR" J:=J+1 "WHILE" "IF" J>I1 "THEN" "FALSE" "ELSE"
            RQ[J]-RQ[J-1]<=EPS[J]+EPS[J-1] "DO" IO1:=J;
            "IF" I = IO1 "THEN"
            "BEGIN"
                "IF" I<N "THEN"
                "BEGIN"
                    "IF" I=1 "THEN" DL:=DR:=RQ[I+1]-RQ[I]-EPS[I+1]
                    "ELSE" "BEGIN" DL:=RQ[I]-RQ[I-1]-EPS[I-1];
                                   DR:=RQ[I+1]-RQ[I]-EPS[I+1]
                           "END"
                "END" "ELSE" DL:=DR:=RQ[I]-RQ[I-1]-EPS[I-1];
                EPS2:=EPS[I]*EPS[I];LBOUND[I]:=EPS2/DR+MATEPS;
                UBOUND[I]:=EPS2/DL+MATEPS
            "END" "ELSE"
            "BEGIN" "FOR" K:=I "STEP" 1 "UNTIL" IO1 "DO"
                LBOUND[K]:=UBOUND[K]:=EPS[K]+MATEPS
            "END";IO1:=IO1+1
        "END"
    "END" BOUNDS;
    "PROCEDURE" LNGMATVEC(L,U,I,A,B,C,CC,D,DD);"CODE" 34411;
    "PROCEDURE" LNGTAMMAT(L,U,I,J,A,B,C,CC,D,DD);"CODE" 34414;
    "PROCEDURE" LNGMATMAT(L,U,I,J,A,B,C,CC,D,DD);"CODE" 34413;
    "INTEGER" "PROCEDURE" QRISYM(A,N,VAL,EM);"CODE" 34163;
    "REAL""PROCEDURE" VECVEC(L,U,SHIFT,A,B);"CODE" 34010;
    "REAL""PROCEDURE" MATMAT(L,U,I,J,A,B);"CODE" 34013;
    "REAL""PROCEDURE" TAMMAT(L,U,I,J,A,B);"CODE" 34014;
    "REAL" "PROCEDURE" INFNRMMAT(LC,UC,LR,UR,K,A);"CODE" 31064;
    "BOOLEAN" STOP;STOP:="FALSE";NORMA:=INFNRMMAT(1,N,1,N,S,A);
    RELERRA:=AUX[0];RELTOLR:=AUX[2];MAXITP1:=AUX[4]+1;
    MATEPS:=RELERRA*NORMA;TOL:=RELTOLR*NORMA;
    "FOR" ITER:=1 "STEP" 1 "UNTIL" MAXITP1 "DO"
    "BEGIN" STOP:="TRUE";MAX:=0;
                                                        "COMMENT"
```

```
    "FOR" J:=1 "STEP" 1 "UNTIL" N "DO"
    "FOR" I:=1 "STEP" 1 "UNTIL" N "DO"
    "BEGIN"
        LNGMATVEC(J,J,I,VEC,VAL1,0,0,HEAD,TAIL);
        LNGMATMAT(1,N,I,J,A,VEC,-HEAD,-TAIL,R[I,J],TAIL);
        "IF"ABS(R[I,J])>MAX "THEN" MAX:=ABS(R[I,J])
    "END";"IF" MAX > TOL "THEN" STOP:="FALSE";
    "IF" "NOT" STOP "AND" ITER<MAXITP1 "THEN"
    "BEGIN"
        "FOR" I:=1 "STEP" 1 "UNTIL" N "DO"
        LNGTAMMAT(1,N,I,I,VEC,R,VAL1[I],0,RQ[I],RQT[I]);
        "FOR" J:=1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" "FOR" I:=1 "STEP" 1 "UNTIL" N "DO"
            ETA[I]:=R[I,J]-(RQ[J]-VAL1[J])*VEC[I,J];
            Z[J]:=SQRT(VECVEC(1,N,0,ETA,ETA))
        "END";
        MERGESORT(RQ,PERM,1,N);VECPERM(PERM,1,N,RQ);
        "FOR" I:=1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" EPS[I]:=Z[PERM[I]];VAL3[I]:=VAL1[PERM[I]];
            ROWPERM(PERM,1,N,I,VEC);ROWPERM(PERM,1,N,I,R)
        "END";
        "FOR" I:=1 "STEP" 1 "UNTIL" N "DO"
        "FOR" J:=I "STEP" 1 "UNTIL" N "DO"
        P[I,J]:=P[J,I]:=TAMMAT(1,N,I,J,VEC,R);
    "END";
    "FOR"I0:=1,I1+1 "WHILE" I0<=N "DO"
    "BEGIN" J:=I1:=I0;
        "FOR" J:=J+1 "WHILE" "IF" J>N "THEN" "FALSE" "ELSE"
        RQ[J]-RQ[J-1]<=SQRT((EPS[J]+EPS[J-1])*NORMA) "DO" I1:=J;
        "IF" STOP "OR" ITER=MAXITP1 "THEN"
        BOUNDS(I0,I1,N,LBOUND,UBOUND) "ELSE"
        "BEGIN"
            "IF" I0=I1 "THEN"
            "BEGIN" "FOR" K:=1 "STEP" 1 "UNTIL" N "DO"
                "IF" K=I0 "THEN" Y[K,I0]:=1 "ELSE"
                R[K,I0]:=P[K,I0];
                VAL1[I0]:=RQ[I0];VAL2[I0]:=RQT[PERM[I0]]
            "END" "ELSE"
            "BEGIN""INTEGER" N1,I0M1,I1P1;"REAL" M1;"ARRAY"EM[0:5];
                N1:=I1-I0+1;EM[0]:=EM[2]:="-14;EM[4]:=10*N1;
                "BEGIN" "ARRAY" PP[1:N1,1:N1],VAL4[1:N1];M1:=0;
                    "FOR" K:=I0 "STEP" 1 "UNTIL" I1 "DO"
                    M1:=M1+VAL3[K];M1:=M1/N1;
                                                    "COMMENT"
```

```
                        "FOR" I:=1 "STEP" 1 "UNTIL" N1 "DO"
                        "FOR" J:=1 "STEP" 1 "UNTIL" N1 "DO"
                        "BEGIN" PP[I,J]:=P[I+IO-1,J+IO-1];
                            "IF" I=J "THEN"
                            PP[I,J]:=PP[I,J]+VAL3[J+IO-1]-M1
                        "END";"FOR" I:=IO "STEP" 1 "UNTIL" I1 "DO"
                        "BEGIN" VAL3[I]:=M1;VAL1[I]:=RQ[I];
                            VAL2[I]:=RQT[PERM[I]]
                        "END";
                        ORISYM(PP,N1,VAL4,EM);
                        MERGESORT(VAL4,PERM,1,N1);
                        "FOR" I:=1 "STEP" 1 "UNTIL" N1 "DO"
                        "FOR" J:=1 "STEP" 1 "UNTIL" N1 "DO"
                        P[I+IO-1,J+IO-1]:=PP[I,PERM[J]];
                        IOM1:=IO-1;I1P1:=I1+1;
                        "FOR" J:=IO "STEP" 1 "UNTIL" I1 "DO"
                        "BEGIN" "FOR" I:=1 "STEP" 1 "UNTIL" IOM1,
                            I1P1 "STEP" 1 "UNTIL" N "DO"
                            "BEGIN" S:=0;
                                "FOR" K:=IO "STEP" 1 "UNTIL" I1 "DO"
                                S:=S+P[I,K]*P[K,J];
                                R[I,J]:=S
                            "END";"FOR" I:=IO "STEP" 1 "UNTIL" I1 "DO"
                            Y[I,J]:=P[I,J]
                        "END" FOR J
                    "END" INNERBLOCK
                "END" I1>IO
            "END" NOT STOP
        "END" FOR IO;
        "IF" "NOT" STOP "AND" ITER<MAXITP1 "THEN"
        "BEGIN"
            "FOR" J:=1 "STEP" 1 "UNTIL" N "DO"
            "FOR" I:=1 "STEP" 1 "UNTIL" N "DO"
            "IF" VAL3[I]^=VAL3[J] "THEN"
            Y[I,J]:=R[I,J]/(VAL3[J]-VAL3[I]);
            "FOR" I:=1 "STEP" 1 "UNTIL" N "DO"
            "BEGIN" "FOR" J:=1 "STEP" 1 "UNTIL" N "DO"
                Z[J]:=MATMAT(1,N,I,J,VEC,Y);
                "FOR" J:=1 "STEP" 1 "UNTIL" N "DO" VEC[I,J]:=Z[J]
            "END";ORTHOG(N,1,N,VEC)
        "END" "ELSE"
        "BEGIN" AUX[5]:=ITER-1;"GOTO" EXIT "END"
    "END" FOR ITER;
    EXIT: AUX[1]:=NORMA;AUX[3]:=MAX
"END" SYMEIGIMP;
        "EOP"
```

AUTHORS: T.J. DEKKER, W. HOFFMANN.

CONTRIBUTORS: W. HOFFMANN, S.P.N. VAN KAMPEN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 731115.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS FIVE PROCEDURES:
A) REAVALORI CALCULATES THE EIGENVALUES OF A REAL UPPER-HESSENBERG MATRIX, PROVIDED THAT ALL EIGENVALUES ARE REAL, BY MEANS OF SINGLE QR ITERATION;
B) REAVECHES CALCULATES THE EIGENVECTOR CORRESPONDING TO A GIVEN REAL EIGENVALUE OF A REAL UPPER-HESSENBERG MATRIX BY MEANS OF INVERSE ITERATION;
C) REAQRI CALCULATES THE EIGENVALUES AND EIGENVECTORS OF A REAL UPPER-HESSENBERG MATRIX, PROVIDED THAT ALL EIGENVALUES ARE REAL, BY MEANS OF SINGLE QR ITERATION;
D) COMVALQRI CALCULATES THE REAL AND COMPLEX EIGENVALUES OF A REAL UPPER-HESSENBERG MATRIX BY MEANS OF DOUBLE QR ITERATION;
E) COMVECHES CALCULATES THE EIGENVECTOR CORRESPONDING TO A GIVEN COMPLEX EIGENVALUE OF A REAL UPPER-HESSENBERG MATRIX BY MEANS OF INVERSE ITERATION.

KEYWORDS:

EIGENVALUE,
EIGENVECTOR,
UPPER-HESSENBERG MATRIX,
QR ITERATION,
INVERSE ITERATION.

SUBSECTION: REAVALQRI.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" REAVALQRI(A, N, EM, VAL); "VALUE" N;
"INTEGER" N; "ARRAY" A, EM, VAL;

THE MEANING OF THE FORMAL PARAMETERS IS:

A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE ELEMENTS OF THE REAL UPPER-HESSENBERG MATRIX MUST BE GIVEN IN THE UPPER TRIANGLE AND THE FIRST SUBDIAGONAL OF ARRAY A;
EXIT: THE HESSENBERG PART OF ARRAY A IS ALTERED;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY: EM[0], THE MACHINE PRECISION;
EM[1], A NORM OF THE GIVEN MATRIX;
EM[2], THE RELATIVE TOLERANCE USED FOR THE QR ITERATION;
IF THE ABSOLUTE VALUE OF SOME SUBDIAGONAL ELEMENT IS SMALLER THAN EM[1] * EM[2], THEN THIS ELEMENT IS NEGLECTED AND THE MATRIX IS PARTITIONED;
EM[4], THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
FOR THE CD CYBER 73-28 SUITABLE VALUES OF THE DATA TO BE GIVEN IN EM ARE:
EM[0] = "-14,
EM[2] > EM[0] (E.G. EM[2] = "-13),
EM[4] = 10 * N;
EXIT: EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE SUBDIAGONAL ELEMENTS NEGLECTED;
EM[5], THE NUMBER OF QR ITERATIONS PERFORMED;
IF THE ITERATION PROCESS IS NOT COMPLETED WITHIN EM[4] ITERATIONS, THE VALUE EM[4] + 1 IS DELIVERED AND IN THIS CASE ONLY THE LAST N - K ELEMENTS OF VAL ARE APPROXIMATE EIGENVALUES OF THE GIVEN MATRIX, WHERE K IS DELIVERED IN REAVALQRI;

VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:N];
THE EIGENVALUES OF THE GIVEN MATRIX ARE DELIVERED IN VAL.

MOREOVER:
REAVALQRI DELIVERS 0, PROVIDED THAT THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE REAVALQRI DELIVERS THE NUMBER OF EIGENVALUES NOT CALCULATED.

PROCEDURES USED:

ROTCOL = CP34040,
ROTROW = CP34041.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

THE METHOD USED IN THE PROCEDURE REAVALQRI IS THE SINGLE QR ITERATION OF FRANCIS (SEE REF[1], P. 54, REF[2] P. 515 - 543 AND REF[3]). THE EIGENVALUES OF A REAL UPPER-HESSENBERG MATRIX ARE CALCULATED, PROVIDED THAT THE MATRIX HAS REAL EIGENVALUES ONLY.

REFERENCES:

[1]. T.J. DEKKER AND W. HOFFMANN.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2.
MC TRACT 23, 1968, MATH. CENTR., AMSTERDAM.

[2]. J.H. WILKINSON.
THE ALGEBRAIC EIGENVALUE PROBLEM.
CLARENDON PRESS, OXFORD, 1965.

[3]. J.G. FRANCIS.
THE QR TRANSFORMATION, PARTS 1 AND 2.
COMP. J. 4 (1961), 265 - 271 AND 332 - 345.

EXAMPLE OF USE:

THE PROCEDURE REAVALQRI IS USED IN REAEIGVAL, SECTION 3.3.1.2.2.

SUBSECTION: REAVECHES.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" REAVECHES(A, N, LAMBDA, EM, V); "VALUE" N, LAMBDA;
"INTEGER" N; "REAL" LAMBDA; "ARRAY" A, EM, V;

THE MEANING OF THE FORMAL PARAMETERS IS:

A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE ELEMENTS OF THE REAL UPPER-HESSENBERG MATRIX MUST BE GIVEN IN THE UPPER TRIANGLE AND THE FIRST SUBDIAGONAL OF ARRAY A;
EXIT: THE HESSENBERG PART OF ARRAY A IS ALTERED;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

LAMBDA: <ARITHMETIC EXPRESSION>;
THE GIVEN REAL EIGENVALUE OF THE UPPER-HESSENBERG MATRIX;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:9];
ENTRY: EM[0], THE MACHINE PRECISION;
EM[1], A NORM OF THE GIVEN MATRIX;
EM[6], THE TOLERANCE USED FOR THE EIGENVECTOR; THE INVERSE ITERATION ENDS IF THE EUCLIDIAN NORM OF THE RESIDUE VECTOR IS SMALLER THAN EM[1] * EM[6];
EM[8], THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
FOR THE CD CYBER 73-28 SUITABLE VALUES OF THE DATA TO BE GIVEN IN EM ARE:
EM[0] = "-14,
EM[6] = "-10,
EM[8] = 5;
EXIT: EM[7], THE EUCLIDIAN NORM OF THE RESIDUE VECTOR OF THE CALCULATED EIGENVECTOR;
EM[9], THE NUMBER OF INVERSE ITERATIONS PERFORMED; IF EM[7] REMAINS LARGER THAN EM[1] * EM[6] DURING EM[8] ITERATIONS, THE VALUE EM[8] + 1 IS DELIVERED;

V: <ARRAY IDENTIFIER>;
"ARRAY" V[1:N];
THE CALCULATED EIGENVECTOR IS DELIVERED IN V.

PROCEDURES USED:

VECVEC = CP34010,
MATVEC = CP34011.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N SQUARED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

THE PROCEDURE REAVECHES CALCULATES AN EIGENVECTOR CORRESPONDING TO A GIVEN APPROXIMATE REAL EIGENVALUE OF A REAL UPPER-HESSENBERG MATRIX, BY MEANS OF INVERSE ITERATION (SEE REF[1], P. 55, REF[2], P. 619 - 629 AND REF[3]).

REFERENCES:

[1]. T.J. DEKKER AND W. HOFFMANN.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2.
MC TRACT 23, 1968, MATH. CENTR., AMSTERDAM.

[2]. J.H. WILKINSON.
THE ALGEBRAIC EIGENVALUE PROBLEM.
CLARENDON PRESS, OXFORD, 1965.

[3]. J.M. VARAH.
EIGENVECTORS OF A REAL MATRIX BY INVERSE ITERATION.
STANFORD UNIVERSITY, TECH. REP. NO. CS 34, 1966.

EXAMPLE OF USE:

THE PROCEDURE REAVECHES IS USED IN REAEIG1, SECTION 3.3.1.2.2.

SUBSECTION: REAQRI.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" REAQRI(A, N, EM, VAL, VEC); "VALUE" N;
"INTEGER" N; "ARRAY" A, EM, VAL, VEC;

THE MEANING OF THE FORMAL PARAMETERS IS:

A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE ELEMENTS OF THE REAL UPPER-HESSENBERG MATRIX MUST BE GIVEN IN THE UPPER TRIANGLE AND THE FIRST SUBDIAGONAL OF ARRAY A;
EXIT: THE HESSENBERG PART OF ARRAY A IS ALTERED;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY: EM[0], THE MACHINE PRECISION;
EM[1], A NORM OF THE GIVEN MATRIX;
EM[2], THE RELATIVE TOLERANCE USED FOR THE QR ITERATION;
IF THE ABSOLUTE VALUE OF SOME SUBDIAGONAL ELEMENT IS SMALLER THAN EM[1] * EM[2], THEN THIS ELEMENT IS NEGLECTED AND THE MATRIX IS PARTITIONED;
EM[4], THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
FOR THE CD CYBER 73-28 SUITABLE VALUES OF THE DATA TO BE GIVEN IN EM ARE:
EM[0] = "-14,
EM[2] > EM[0] (E.G. EM[2] = "-13),
EM[4] = 10 * N;
EXIT: EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE SUBDIAGONAL ELEMENTS NEGLECTED;
EM[5], THE NUMBER OF QR ITERATIONS PERFORMED;
IF THE ITERATION PROCESS IS NOT COMPLETED WITHIN EM[4] ITERATIONS, THE VALUE EM[4] + 1 IS DELIVERED; IN THIS CASE ONLY THE LAST N - K ELEMENTS OF VAL AND THE LAST N - K COLUMNS OF VEC ARE APPROXIMATED EIGENVALUES AND EIGENVECTORS OF THE GIVEN MATRIX, WHERE K IS DELIVERED IN REAQRI;

VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:N];
THE EIGENVALUES OF THE GIVEN MATRIX ARE DELIVERED IN VAL;

VEC: <ARRAY IDENTIFIER>;
"ARRAY" VEC[1:N,1:N];
THE CALCULATED EIGENVECTORS, CORRESPONDING TO THE EIGENVALUES IN ARRAY VAL[1:N], ARE DELIVERED IN THE COLUMNS OF ARRAY VEC.

MOREOVER:
REAQRI DELIVERS 0, PROVIDED THAT THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE REAQRI DELIVERS THE NUMBER OF EIGENVALUES AND EIGENVECTORS NOT CALCULATED.

PROCEDURES USED:

MATVEC = CP34011,
ROTCOL = CP34040,
ROTROW = CP34041.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

THE PROCEDURE REAQRI CALCULATES THE EIGENVALUES OF AN UPPER-HESSENBERG MATRIX BY MEANS OF SINGLE QR ITERATION (SEE METHOD AND PERFORMANCE OF REAVALQRI, THIS SECTION).
THE EIGENVECTORS ARE CALCULATED BY A DIRECT METHOD (SEE REF[1], P. 55-56), IN CONTRAST WITH REAVECHES WHICH USES INVERSE ITERATION.
IF THE HESSENBERG MATRIX IS NOT TOO ILL-CONDITIONED WITH RESPECT TO ITS EIGENVALUE PROBLEM, THEN THIS METHOD YIELDS NUMERICALLY INDEPENDENT EIGENVECTORS AND IS COMPETITIVE WITH INVERSE ITERATION AS TO ACCURACY AND COMPUTATION TIME.
IF THE QR ITERATION PROCESS IS NOT COMPLETED WITHIN THE GIVEN NUMBER OF ITERATIONS, NOT ALL EIGENVALUES AND EIGENVECTORS ARE DELIVERED.
THE PROCEDURE REAQRI SHOULD BE USED ONLY IF ALL EIGENVALUES ARE REAL.

REFERENCES:
[1]. T.J. DEKKER AND W. HOFFMANN.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2.
MC TRACT 23, 1968, MATH. CENTR., AMSTERDAM.

EXAMPLE OF USE:

THE PROCEDURE REAQRI IS USED IN REAEIG3, SECTION 3.3.1.2.2.

SUBSECTION: COMVALQRI.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" COMVALQRI(A, N, EM, RE, IM); "VALUE" N;
"INTEGER" N; "ARRAY" A, EM, RE, IM;

THE MEANING OF THE FORMAL PARAMETERS IS:

A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE ELEMENTS OF THE REAL UPPER-HESSENBERG MATRIX MUST BE GIVEN IN THE UPPER TRIANGLE AND THE FIRST SUBDIAGONAL OF ARRAY A;
EXIT: THE HESSENBERG PART OF ARRAY A IS ALTERED;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY: EM[0], THE MACHINE PRECISION;
EM[1], A NORM OF THE GIVEN MATRIX;
EM[2], THE RELATIVE TOLERANCE USED FOR THE QR ITERATION;
IF THE ABSOLUTE VALUE OF SOME SUBDIAGONAL ELEMENT IS SMALLER THAN EM[1] * EM[2], THEN THIS ELEMENT IS NEGLECTED AND THE MATRIX IS PARTITIONED;
EM[4], THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
FOR THE CD CYBER 73-28 SUITABLE VALUES OF THE DATA TO BE GIVEN IN EM ARE:
EM[0] = "-14,
EM[2] > EM[0] (E.G. EM[2] = "-13),
EM[4] = 10 * N;
EXIT: EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE SUBDIAGONAL ELEMENTS NEGLECTED;
EM[5], THE NUMBER OF QR ITERATIONS PERFORMED;
IF THE ITERATION PROCESS IS NOT COMPLETED WITHIN EM[4] ITERATIONS, THE VALUE EM[4] + 1 IS DELIVERED AND IN THIS CASE ONLY THE LAST N - K ELEMENTS OF RE AND IM ARE APPROXIMATE EIGENVALUES OF THE GIVEN MATRIX, WHERE K IS DELIVERED IN COMVALQRI;

RE,IM: <ARRAY IDENTIFIER>;
"ARRAY" RE, IM[1:N];
THE REAL AND IMAGINARY PARTS OF THE CALCULATED EIGENVALUES OF THE GIVEN MATRIX ARE DELIVERED IN ARRAY RE, IM[1:N], THE MEMBERS OF EACH NONREAL COMPLEX CONJUGATE PAIR BEING CONSECUTIVE.

MOREOVER:
COMVALQRI DELIVERS 0, PROVIDED THAT THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE COMVALQRI DELIVERS THE NUMBER OF EIGENVALUES NOT CALCULATED.

PROCEDURES USED: NONE.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

THE METHOD USED IN THE PROCEDURE COMVALQRI FOR CALCULATING THE REAL AND COMPLEX EIGENVALUES OF A REAL UPPER-HESSENBERG MATRIX IS THE DOUBLE QR ITERATION OF FRANCIS (SEE REF[1], P. 74, REF[2] P. 528 - 537 AND REF[3]).

REFERENCES:

[1]. T.J. DEKKER AND W. HOFFMANN.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2.
MC TRACT 23, 1968, MATH. CENTR., AMSTERDAM.

[2]. J.H. WILKINSON.
THE ALGEBRAIC EIGENVALUE PROBLEM.
CLARENDON PRESS, OXFORD, 1965.

[3]. J.G. FRANCIS.
THE QR TRANSFORMATION, PARTS 1 AND 2.
COMP. J. 4 (1961), 265 - 271 AND 332 - 345.

EXAMPLE OF USE:

THE COMPLEX EIGENVALUES AND -VECTORS OF H, WITH N = 4 AND H[I,J] = "IF" I = 1 "THEN" -1 "ELSE" "IF" I - J = 1 "THEN" 1 "ELSE" 0, MAY BE OBTAINED BY THE FOLLOWING PROGRAM:

```
"BEGIN" "INTEGER" I, J, M;
    "ARRAY" A[1:4,1:4], RE, IM[1:4], EM[0:9];
    "INTEGER" "PROCEDURE" COMVALQRI(A, N, EM, RE, IM);
    "CODE" 34190;
    "PROCEDURE" COMVECHES(A, N, LAMBDA, MU, EM, U, V);
    "CODE" 34191;

    EM[0]:= "-14; EM[2]:= "-13; EM[1]:= 4; EM[4]:= 40;
    EM[6]:= "-10; EM[8]:= 5;
    "FOR" I:= 1, 2, 3, 4 "DO" "FOR" J:= 1, 2, 3, 4 "DO" A[I,J]:=
    "IF" I = 1 "THEN" -1 "ELSE" "IF" I - J = 1 "THEN" 1 "ELSE" 0;
    M:= COMVALQRI(A, 4, EM, RE, IM); OUTPUT(61, "("D, /")", M);
```

```
        "FOR" J:= M + 1 "STEP" 1 "UNTIL" 4 "DO"
        "BEGIN" "INTEGER" K; "ARRAY" U, V[1:4];
            "FOR" I:= 1, 2, 3, 4 "DO" "FOR" K:= 1, 2, 3, 4 "DO"
            A[I,K]:= "IF" I = 1 "THEN" -1 "ELSE"
            "IF" I = K = 1 "THEN" 1 "ELSE" 0;
            COMVECHES(A, 4, RE[J], IM[J], EM, U, V);
            OUTPUT(61, "("/, 2(+.13D"+2D, 2B), 2/")", RE[J], IM[J]);
            "FOR" I:= 1, 2, 3, 4 "DO"
            OUTPUT(61, "("21B, 2(+.13D"+2D, 2B), /")", U[I], V[I])
        "END";
        OUTPUT(61, "("/, 2(.2D"+2D, /), 2(ZD, /)")",
        EM[3], EM[7], EM[5], EM[9])
    "END"
```

THE PROGRAM DELIVERS (THE RESULTS ARE CORRECT UP TO TWELVE DIGITS):

THE NUMBER OF NOT CALCULATED EIGENVALUES: 0

THE EIGENVALUES AND -VECTORS:

```
+.3090169943750"+00  +.9510565162952"+00

                     -.2527643931136"+00  -.4314048696688"+00
                     -.4883989055049"+00  +.1070817869743"+00
                     -.4908273055667"-01  +.4975850535950"+00
                     +.4580641097602"+00  +.2004426884413"+00

+.3090169943750"+00  -.9510565162952"+00

                     -.2527643931136"+00  +.4314048696688"+00
                     -.4883989055049"+00  -.1070817869743"+00
                     -.4908273055667"-01  -.4975850535950"+00
                     +.4580641097602"+00  -.2004426884413"+00

-.8090169943749"+00  +.5877852522924"+00

                     +.1095191711534"+00  -.4878581260468"+00
                     -.3753586823743"+00  +.3303117611685"+00
                     +.4978239349006"+00  -.4659753040772"-01
                     -.4301373647081"+00  -.2549153731770"+00

-.8090169943749"+00  -.5877852522924"+00

                     +.1095191711534"+00  +.4878581260468"+00
                     -.3753586823743"+00  -.3303117611685"+00
                     +.4978239349006"+00  +.4659753040772"-01
                     -.4301373647081"+00  +.2549153731770"+00
```

THE ARRAY EM: EM[3] = .67"-22
EM[7] = .17"-13
EM[5] = 9
EM[9] = 1 .

SUBSECTION: COMVECHES.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" COMVECHES(A, N, LAMBDA, MU, EM, U, V);
"VALUE" N, LAMBDA, MU;
"INTEGER" N; "REAL" LAMBDA, MU; "ARRAY" A, EM, U, V;

THE MEANING OF THE FORMAL PARAMETERS IS:

A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE ELEMENTS OF THE REAL UPPER-HESSENBERG MATRIX MUST BE GIVEN IN THE UPPER TRIANGLE AND THE FIRST SUBDIAGONAL OF ARRAY A;
EXIT: THE HESSENBERG PART OF ARRAY A IS ALTERED;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

LAMBDA, MU:
<ARITHMETIC EXPRESSION>;
THE REAL AND IMAGINARY PART OF THE GIVEN EIGENVALUE;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:9];
ENTRY: EM[0], THE MACHINE PRECISION;
EM[1], A NORM OF THE GIVEN MATRIX;
EM[6], THE TOLERANCE USED FOR THE EIGENVECTOR; THE INVERSE ITERATION ENDS IF THE EUCLIDIAN NORM OF THE RESIDUE VECTOR IS SMALLER THAN EM[1] * EM[6];
EM[8], THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
FOR THE CD CYBER 73-28 SUITABLE VALUES OF THE DATA TO BE GIVEN IN EM ARE:
EM[0] = "-14,
EM[6] = "-10,
EM[8] = 5;
EXIT: EM[7], THE EUCLIDIAN NORM OF THE RESIDUE VECTOR OF THE CALCULATED EIGENVECTOR;
EM[9], THE NUMBER OF INVERSE ITERATIONS PERFORMED; IF EM[7] REMAINS LARGER THAN EM[1] * EM[6] DURING EM[8] ITERATIONS, THE VALUE EM[8] + 1 IS DELIVERED;

U, V: <ARRAY IDENTIFIER>;
"ARRAY" U, V[1:N];
THE REAL AND IMAGINARY PARTS OF THE CALCULATED EIGENVECTOR ARE DELIVERED IN THE ARRAYS U, V[1:N].

PROCEDURES USED:

VECVEC = CP34010,
MATVEC = CP34011,
TAMVEC = CP34012.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N SQUARED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

THE PROCEDURE COMVECHES CALCULATES AN EIGENVECTOR CORRESPONDING TO A GIVEN APPROXIMATE EIGENVALUE OF A REAL UPPER-HESSENBERG MATRIX, BY MEANS OF INVERSE ITERATION (SEE REF[1], P. 75, REF[2], P. 629 - 633 AND REF[3]).

REFERENCES:

[1]. T.J. DEKKER AND W. HOFFMANN.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2.
MC TRACT 23, 1968, MATH. CENTR., AMSTERDAM.

[2]. J.H. WILKINSON.
THE ALGEBRAIC EIGENVALUE PROBLEM.
CLARENDON PRESS, OXFORD, 1965.

[3]. J.M. VARAH.
EIGENVECTORS OF A REAL MATRIX BY INVERSE ITERATION.
STANFORD UNIVERSITY, TECH. REP. NO. CS 34, 1966.

EXAMPLE OF USE:

SEE EXAMPLE OF USE OF COMVALQRI, THIS SECTION.

SOURCE TEXT(S) :

```
"CODE" 34180;
    "COMMENT" MCA 2410;
    "INTEGER" "PROCEDURE" REAVALQRI(A, N, EM, VAL); "VALUE" N;
    "INTEGER" N; "ARRAY" A, EM, VAL;
    "BEGIN" "INTEGER" N1, I, I1, J, Q, MAX, COUNT;
        "REAL" DET, W, SHIFT, KAPPA, NU, MU, R, TOL, DELTA, MACHTOL, S;

        "PROCEDURE" ROTCOL(L, U, I, J, A, C, S); "CODE" 34040;
        "PROCEDURE" ROTROW(L, U, I, J, A, C, S); "CODE" 34041;
        "COMMENT"
```

```
    MACHTOL:= EM[0] * EM[1]; TOL:= EM[1] * EM[2]; MAX:= EM[4];
    COUNT:= 0; R:= 0;
IN: N1:= N - 1;
    "FOR" I:= N, I - 1 "WHILE" ("IF" I >= 1 "THEN"
    ABS(A[I + 1,I]) > TOL "ELSE" "FALSE") "DO" Q:= I;
    "IF" Q > 1 "THEN"
    "BEGIN" "IF" ABS(A[Q,Q - 1]) > R "THEN"
        R:= ABS(A[Q,Q - 1])
    "END";
    "IF" Q = N "THEN"
    "BEGIN" VAL[N]:= A[N,N]; N:= N1 "END"
    "ELSE"
    "BEGIN" DELTA:= A[N,N] - A[N1,N1]; DET:= A[N,N1] * A[N1,N];
        "IF" ABS(DELTA) < MACHTOL "THEN" S:= SQRT(DET) "ELSE"
        "BEGIN" W:= 2 / DELTA; S:= W * W * DET + 1;
            S:= "IF" S <= 0 "THEN" -DELTA * .5 "ELSE"
            W * DET / (SQRT(S) + 1)
        "END";
        "IF" Q = N1 "THEN"
        "BEGIN" VAL[N]:= A[N,N] + S;
            VAL[N1]:= A[N1,N1] - S; N:= N - 2
        "END"
        "ELSE"
        "BEGIN" COUNT:= COUNT + 1;
            "IF" COUNT > MAX "THEN" "GOTO" OUT;
            SHIFT:= A[N,N] + S; "IF" ABS(DELTA) < TOL "THEN"
            "BEGIN" W:= A[N1,N1] - S;
                "IF" ABS(W) < ABS(SHIFT) "THEN" SHIFT:= W
            "END";
            A[Q,Q]:= A[Q,Q] - SHIFT;
            "FOR" I:= Q "STEP" 1 "UNTIL" N - 1 "DO"
            "BEGIN" I1:= I + 1; A[I1,I1]:= A[I1,I1] - SHIFT;
                KAPPA:= SQRT(A[I,I] ** 2 + A[I1,I] ** 2);
                "IF" I > Q "THEN"
                "BEGIN" A[I,I - 1]:= KAPPA * NU;
                    W:= KAPPA * MU
                "END"
                "ELSE" W:= KAPPA; MU:= A[I,I] / KAPPA;
                NU:= A[I1,I] / KAPPA; A[I,I]:= W;
                ROTROW(I1, N, I, I1, A, MU, NU);
                ROTCOL(Q, I, I, I1, A, MU, NU);
                A[I,I]:= A[I,I] + SHIFT
            "END";
            A[N,N - 1]:= A[N,N] * NU; A[N,N]:= A[N,N] * MU + SHIFT
        "END"
    "END";
    "IF" N > 0 "THEN" "GOTO" IN;
OUT: EM[3]:= R; EM[5]:= COUNT; REAVALQRI:= N
"END" REAVALQRI;
    "EOP"
```

```
"CODE" 34181;
    "COMMENT" MCA 2411;
    "PROCEDURE" REAVECHES(A, N, LAMBDA, EM, V); "VALUE" N, LAMBDA;
    "INTEGER" N; "REAL" LAMBDA; "ARRAY" A, EM, V;
    "BEGIN" "INTEGER" I, I1, J, COUNT, MAX;
        "REAL" M, R, NORM, MACHTOL, TOL;
        "BOOLEAN" "ARRAY" P[1:N];

        "REAL" "PROCEDURE" VECVEC(L, U, SHIFT, A, B); "CODE" 34010;
        "REAL" "PROCEDURE" MATVEC(L, U, I, A, B); "CODE" 34011;

        NORM:= EM[1]; MACHTOL:= EM[0] * NORM; TOL:= EM[6] * NORM;
        MAX:= EM[8]; A[1,1]:= A[1,1] - LAMBDA;
     GAUSS: "FOR" I:= 1 "STEP" 1 "UNTIL" N - 1 "DO"
        "BEGIN" I1:= I + 1; R:= A[I,I]; M:= A[I1,I];
            "IF" ABS(M) < MACHTOL "THEN" M:= MACHTOL;
            P[I]:= ABS(M) <= ABS(R);
            "IF" P[I] "THEN"
            "BEGIN" A[I1,I]:= M:= M / R;
                "FOR" J:= I1 "STEP" 1 "UNTIL" N "DO"
                A[I1,J]:= ("IF" J > I1 "THEN" A[I1,J]
                "ELSE" A[I1,J] - LAMBDA) - M * A[I,J]
            "END"
            "ELSE"
            "BEGIN" A[I,I]:= M; A[I1,I]:= M:= R / M;
                "FOR" J:= I1 "STEP" 1 "UNTIL" N "DO"
                "BEGIN" R:= ("IF" J > I1 "THEN" A[I1,J] "ELSE"
                    A[I1,J] - LAMBDA);
                    A[I1,J]:= A[I,J] - M * R; A[I,J]:= R
                "END"
            "END"
        "END" GAUSS;
        "IF" ABS(A[N,N]) < MACHTOL "THEN" A[N,N]:= MACHTOL;
        "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO" V[J]:= 1; COUNT:= 0;
     FORWARD: COUNT:= COUNT + 1; "IF" COUNT > MAX "THEN" "GOTO" OUT;
        "FOR" I:= 1 "STEP" 1 "UNTIL" N - 1 "DO"
        "BEGIN" I1:= I + 1;
            "IF" P[I] "THEN" V[I1]:= V[I1] - A[I1,I] * V[I] "ELSE"
            "BEGIN" R:= V[I1]; V[I1]:= V[I] - A[I1,I] * R;
                V[I]:=R
            "END"
        "END" FORWARD;
     BACKWARD: "FOR" I:= N "STEP" -1 "UNTIL" 1 "DO"
        V[I]:= (V[I] - MATVEC(I + 1, N, I, A, V)) / A[I,I];
        R:= 1 / SQRT(VECVEC(1, N, 0, V, V));
        "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO" V[J]:= V[J] * R;
        "IF" R > TOL "THEN" "GOTO" FORWARD;
     OUT: EM[7]:= R; EM[9]:= COUNT
    "END" REAVECHES;
        "EOP"
```

```
"CODE" 34186;
    "COMMENT" MCA 2416;
    "INTEGER" "PROCEDURE" REAQRI(A, N, EM, VAL, VEC); "VALUE" N;
    "INTEGER" N; "ARRAY" A, EM, VAL, VEC;
    "BEGIN" "INTEGER" M1, I, I1, M, J, Q, MAX, COUNT;
        "REAL" W, SHIFT, KAPPA, NU, MU, R, TOL, S, MACHTOL,
        ELMAX, T, DELTA, DET;
        "ARRAY" TF[1:N];

        "REAL" "PROCEDURE" MATVEC(L, U, I, A, B); "CODE" 34011;
        "PROCEDURE" ROTCOL(L, U, I, J, A, C, S); "CODE" 34040;
        "PROCEDURE" ROTROW(L, U, I, J, A, C, S); "CODE" 34041;

        MACHTOL:= EM[0] * EM[1]; TOL:= EM[1] * EM[2]; MAX:= EM[4];
        COUNT:= 0; ELMAX:= 0; M:= N;
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" VEC[I,I]:= 1;
            "FOR" J:= I + 1 "STEP" 1 "UNTIL" N "DO"
            VEC[I,J]:= VEC[J,I]:= 0
        "END";
     IN: M1:= M - 1;
        "FOR" I:= M, I - 1 "WHILE" ("IF" I >= 1 "THEN"
        ABS(A[I + 1,I]) > TOL "ELSE" "FALSE") "DO" Q:= I;
        "IF" Q > 1 "THEN"
        "BEGIN" "IF" ABS(A[Q,Q - 1]) > ELMAX "THEN"
            ELMAX:= ABS(A[Q, Q - 1])
        "END";
        "IF" Q = M "THEN"
        "BEGIN" VAL[M]:= A[M,M]; M:= M1 "END"
        "ELSE"
        "BEGIN" DELTA:= A[M,M] - A[M1,M1]; DET:= A[M,M1] * A[M1,M];
            "IF" ABS(DELTA) < MACHTOL "THEN" S:= SQRT(DET) "ELSE"
            "BEGIN" W:= 2 / DELTA; S:= W * W * DET + 1;
                S:= "IF" S <= 0 "THEN" -DELTA * .5 "ELSE"
                W * DET / (SQRT(S) + 1)
            "END";
            "IF" Q = M1 "THEN"
            "BEGIN" A[M,M]:= VAL[M]:= A[M,M] + S;
                A[Q,Q]:= VAL[Q]:= A[Q,Q] - S;
                T:= "IF" ABS(S) < MACHTOL "THEN"
                (S + DELTA) / A[M,Q] "ELSE" A[Q,M] / S;
                R:= SQRT(T * T + 1); NU:= 1 / R;
                MU:= -T * NU; A[Q,M]:= A[Q,M] - A[M,Q];
                ROTROW(Q + 2, N, Q, M, A, MU, NU);
                ROTCOL(1, Q - 1, Q, M, A, MU, NU);
                ROTCOL(1, N, Q, M, VEC, MU, NU); M:= M - 2
            "END"
```

```
            "ELSE"
            "BEGIN" COUNT:= COUNT + 1;
                "IF" COUNT > MAX "THEN" "GOTO" END;
                SHIFT:= A[M,M] + S; "IF" ABS(DELTA) < TOL "THEN"
                "BEGIN" W:= A[M1,M1] - S;
                    "IF" ABS(W) < ABS(SHIFT) "THEN" SHIFT:= W
                "END";
                A[Q,Q]:= A[Q,Q] - SHIFT;
                "FOR" I:= Q "STEP" 1 "UNTIL" M1 "DO"
                "BEGIN" I1:= I + 1; A[I1,I1]:= A[I1,I1] - SHIFT;
                    KAPPA:= SQRT(A[I,I] ** 2 + A[I1,I] ** 2);
                    "IF" I > Q "THEN"
                    "BEGIN" A[I,I - 1]:= KAPPA * NU;
                        W:= KAPPA * MU
                    "END"
                    "ELSE" W:= KAPPA; MU:= A[I,I] / KAPPA;
                    NU:= A[I1,I] / KAPPA; A[I,I]:= W;
                    ROTROW(I1, N, I, I1, A, MU, NU);
                    ROTCOL(1, I, I, I1, A, MU, NU);
                    A[I,I]:= A[I,I] + SHIFT;
                    ROTCOL(1, N, I, I1, VEC, MU, NU)
                "END";
                A[M,M1]:= A[M,M] * NU; A[M,M]:= A[M,M] * MU + SHIFT
            "END"
        "END";
        "IF" M > 0 "THEN" "GOTO" IN;
        "FOR" J:= N "STEP" -1 "UNTIL" 2 "DO"
        "BEGIN" TF[J]:= 1; T:= A[J,J];
            "FOR" I:= J - 1 "STEP" -1 "UNTIL" 1 "DO"
            "BEGIN" DELTA:= T - A[I,I];
                TF[I]:= MATVEC(I + 1, J, I, A, TF) /
                ("IF" ABS(DELTA) < MACHTOL "THEN" MACHTOL "ELSE" DELTA)
            "END";
            "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
            VEC[I,J]:= MATVEC(1, J, I, VEC, TF)
        "END";
    END: EM[3]:= ELMAX; EM[5]:= COUNT; REAQRI:= M
"END" REAQRI;
    "EOP"
```

```
"CODE" 34190;
    "COMMENT" MCA 2420;
    "INTEGER" "PROCEDURE" COMVALQRI(A, N, EM, RE, IM); "VALUE" N;
    "INTEGER" N; "ARRAY" A, EM, RE, IM;
    "BEGIN" "INTEGER" I, J, P, Q, MAX, COUNT, N1, P1, P2, IMIN1,
        I1, I2, I3;
        "REAL" DISC, SIGMA, RHO, G1, G2, G3, PSI1, PSI2, AA, E, K,
        S, NORM, MACHTOL2, TOL, W;
        "BOOLEAN" B;

        NORM:= EM[1]; MACHTOL2:= (EM[0] * NORM) ** 2;
        TOL:= EM[2] * NORM; MAX:= EM[4]; COUNT:= 0; W:= 0;
     IN: "FOR" I:= N, I - 1 "WHILE"
        ("IF" I >= 1 "THEN" ABS(A[I + 1,I]) > TOL "ELSE" "FALSE")
        "DO" Q:= I; "IF" Q > 1 "THEN"
        "BEGIN" "IF" ABS(A[Q,Q - 1]) > W "THEN" W:= ABS(A[Q,Q - 1])
        "END";
        "IF" Q >= N - 1 "THEN"
        "BEGIN" N1:= N - 1; "IF" Q = N "THEN"
            "BEGIN" RE[N]:= A[N,N]; IM[N]:= 0; N:= N1 "END"
            "ELSE"
            "BEGIN" SIGMA:= A[N,N] - A[N1,N1];
                RHO:= -A[N,N1] * A[N1,N];
                DISC:= SIGMA ** 2 - 4 * RHO; "IF" DISC > 0 "THEN"
                "BEGIN" DISC:= SQRT(DISC);
                    S:= -2 * RHO / (SIGMA + ("IF" SIGMA >= 0
                    "THEN" DISC "ELSE" -DISC));
                    RE[N]:= A[N,N] + S;
                    RE[N1]:= A[N1,N1] - S; IM[N]:= IM[N1]:= 0
                "END"
                "ELSE"
                "BEGIN" RE[N]:= RE[N1]:= (A[N1,N1] + A[N,N]) / 2;
                    IM[N1]:= SQRT( -DISC) / 2; IM[N]:= -IM[N1]
                "END";
                N:= N - 2
            "END"
        "END"
        "ELSE"
        "BEGIN" COUNT:= COUNT + 1; "IF" COUNT > MAX "THEN"
            "GOTO" OUT; N1:= N - 1;
            SIGMA:= A[N,N] + A[N1,N1] + SQRT(ABS(A[N1,N - 2] * A[N,N1])
            * EM[0]); RHO:= A[N,N] * A[N1,N1] - A[N,N1] * A[N1,N];
            "FOR" I:= N - 1, I - 1 "WHILE"
            ("IF" I - 1 >= Q "THEN" ABS(A[I,I - 1] *
            A[I1,I] * (ABS(A[I,I] + A[I1,I1] - SIGMA) +
            ABS(A[I + 2,I1]))) > ABS(A[I,I] * ((A[I,I] - SIGMA) +
            A[I,I1] * A[I1,I] + RHO)) * TOL
            "ELSE" "FALSE") "DO" P1:= I1:= I; P:= P1 - 1;
            P2:= P + 2;
                                                    "COMMENT"
```

```
        "FOR" I:= P "STEP" 1 "UNTIL" N - 1 "DO"
        "BEGIN" IMIN1:= I - 1; I1:= I + 1; I2:= I + 2;
            "IF" I = P "THEN"
            "BEGIN" G1:= A[P,P] * (A[P,P] - SIGMA) + A[P,P1] *
                A[P1,P] + RHO;
                G2:= A[P1,P] * (A[P,P] + A[P1,P1] - SIGMA);
                "IF" P1 <= N1 "THEN"
                "BEGIN" G3:= A[P1,P] * A[P2,P1]; A[P2,P]:= 0 "END"
                "ELSE" G3:= 0
            "END"
            "ELSE"
            "BEGIN" G1:= A[I,IMIN1]; G2:= A[I1,IMIN1];
                G3:= "IF" I2 <= N "THEN" A[I2,IMIN1] "ELSE" 0
            "END";
            K:= "IF" G1 >= 0 "THEN"
            SQRT(G1 ** 2 + G2 ** 2 + G3 ** 2) "ELSE"
            -SQRT(G1 ** 2 + G2 ** 2 + G3 ** 2);
            B:= ABS(K) > MACHTOL2;
            AA:= "IF" B "THEN" G1 / K + 1 "ELSE" 2;
            PSI1:= "IF" B "THEN" G2 / (G1 + K) "ELSE" 0;
            PSI2:= "IF" B "THEN" G3 / (G1 + K) "ELSE" 0;
            "IF" I ^= Q "THEN" A[I,IMIN1]:= "IF" I = P "THEN"
            -A[I,IMIN1] "ELSE" -K;
            "FOR" J:= I "STEP" 1 "UNTIL" N "DO"
            "BEGIN" E:= AA * (A[I,J] + PSI1 * A[I1,J] +
                ("IF" I2 <= N "THEN" PSI2 * A[I2,J] "ELSE" 0));
                A[I,J]:= A[I,J] - E; A[I1,J]:= A[I1,J] - PSI1 * E;
                "IF" I2 <= N "THEN" A[I2,J]:= A[I2,J] - PSI2 * E
            "END";
            "FOR" J:= Q "STEP" 1 "UNTIL"
            ("IF" I2 <= N "THEN" I2 "ELSE" N) "DO"
            "BEGIN" E:= AA * (A[J,I] + PSI1 * A[J,I1] +
                ("IF" I2 <= N "THEN" PSI2 * A[J,I2] "ELSE" 0));
                A[J,I]:= A[J,I] - E; A[J,I1]:= A[J,I1] - PSI1 * E;
                "IF" I2 <= N "THEN" A[J,I2]:= A[J,I2] - PSI2 * E
            "END";
            "IF" I2 <= N1 "THEN"
            "BEGIN" I3:= I + 3; E:= AA * PSI2 * A[I3,I2];
                A[I3,I]:= -E;
                A[I3,I1]:= -PSI1 * E;
                A[I3,I2]:= A[I3,I2] - PSI2 * E
            "END"
        "END"
    "END";
    "IF" N > 0 "THEN" "GOTO" IN;
OUT: EM[3]:= W; EM[5]:= COUNT; COMVALQRI:= N
"END" COMVALQRI;
    "EOP"
```

```
"CODE" 34191;
    "COMMENT" MCA 2421;
    "PROCEDURE" COMVECHES(A, N, LAMBDA, MU, EM, U, V);
    "VALUE" N, LAMBDA, MU;
    "INTEGER" N; "REAL" LAMBDA, MU; "ARRAY" A, EM, U, V;
    "BEGIN" "INTEGER" I, I1, J, COUNT, MAX;
        "REAL" AA, BB, D, M, R, S, W, X, Y, NORM, MACHTOL, TOL;
        "ARRAY" G, F[1:N];
        "BOOLEAN" "ARRAY" P[1:N];

        "REAL" "PROCEDURE" VECVEC(L, U, SHIFT, A, B); "CODE" 34010;
        "REAL" "PROCEDURE" MATVEC(L, U, I, A, B); "CODE" 34011;
        "REAL" "PROCEDURE" TAMVEC(L, U, I, A, B); "CODE" 34012;

        NORM:= EM[1]; MACHTOL:= EM[0] * NORM; TOL:= EM[6] * NORM;
        MAX:= EM[8];
        "FOR" I:= 2 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" F[I - 1]:= A[I,I - 1]; A[I,1]:= 0 "END";
        AA:= A[1,1] - LAMBDA; BB:= -MU;
        "FOR" I:= 1 "STEP" 1 "UNTIL" N - 1 "DO"
        "BEGIN" I1:= I + 1; M:= F[I];
            "IF" ABS(M) < MACHTOL "THEN" M:= MACHTOL;
            A[I,I]:= M; D:= AA ** 2 + BB ** 2; P[I]:= ABS(M) < SQRT(D);
            "IF" P[I] "THEN"
            "BEGIN" "COMMENT" A[I,J] * FACTOR AND A[I1,J] - A[I,J];
                F[I]:= R:= M * AA / D; G[I]:= S:= -M * BB / D;
                W:= A[I1,I]; X:= A[I,I1]; A[I1,I]:= Y:= X * S + W * R;
                A[I,I1]:= X:= X * R - W * S;
                AA:= A[I1,I1] - LAMBDA - X; BB:= -(MU + Y);
                "FOR" J:= I + 2 "STEP" 1 "UNTIL" N "DO"
                "BEGIN" W:= A[J,I]; X:= A[I,J];
                    A[J,I]:= Y:= X * S + W * R;
                    A[I,J]:= X:= X * R - W * S; A[J,I1]:= -Y;
                    A[I1,J]:= A[I1,J] - X
                "END"
            "END"
            "ELSE"
            "BEGIN" "COMMENT" INTERCHANGE A[I1,J] AND
                A[I,J] - A[I1,J] * FACTOR;
                F[I]:= R:= AA / M; G[I]:= S:= BB / M;
                W:= A[I1,I1] - LAMBDA; AA:= A[I,I1] - R * W - S * MU;
                A[I,I1]:= W; BB:= A[I1,I] - S * W + R * MU;
                A[I1,I]:= -MU;
                "FOR" J:= I + 2 "STEP" 1 "UNTIL" N "DO"
                "BEGIN" W:= A[I1,J]; A[I1,J]:= A[I,J] - R * W;
                    A[I,J]:= W;
                    A[J,I1]:= A[J,I] - S * W; A[J,I]:= 0
                "END"
            "END"
        "END"
```

```
      P[N]:= "TRUE"; D:= AA ** 2 + BB ** 2; "IF" D < MACHTOL ** 2
      "THEN" "BEGIN" AA:= MACHTOL; BB:= 0; D:= MACHTOL ** 2 "END";
      A[N,N]:= D; F[N]:= AA; G[N]:= -BB;
      "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
      "BEGIN" U[I]:= 1; V[I]:= 0 "END";
      COUNT:= 0;
   FORWARD: "IF" COUNT > MAX "THEN" "GOTO" OUTM;
      "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
      "BEGIN" "IF" P[I] "THEN"
           "BEGIN" W:= V[I]; V[I]:= G[I] * U[I] + F[I] * W;
              U[I]:= F[I] * U[I] - G[I] * W; "IF" I < N "THEN"
              "BEGIN" V[I + 1]:= V[I + 1] - V[I];
                  U[I + 1]:= U[I + 1] - U[I]
              "END"
           "END"
           "ELSE"
           "BEGIN" AA:= U[I + 1]; BB:= V[I + 1];
               U[I + 1]:= U[I] - (F[I] * AA - G[I] * BB); U[I]:= AA;
               V[I + 1]:= V[I] - (G[I] * AA + F[I] * BB); V[I]:= BB
           "END"
      "END" FORWARD;
   BACKWARD: "FOR" I:= N "STEP" -1 "UNTIL" 1 "DO"
      "BEGIN" I1:= I + 1;
          U[I]:= (U[I] - MATVEC(I1, N, I, A, U) + ("IF" P[I] "THEN"
          TAMVEC(I1, N, I, A, V) "ELSE" A[I1,I] * V[I1])) / A[I,I];
          V[I]:= (V[I] - MATVEC(I1, N, I, A, V) - ("IF" P[I] "THEN"
          TAMVEC(I1, N, I, A, U) "ELSE" A[I1,I] * U[I1])) / A[I,I]
      "END" BACKWARD;
   NORMALISE: W:= 1 / SQRT(VECVEC(1, N, 0, U, U) +
      VECVEC(1, N, 0, V, V));
      "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO"
      "BEGIN" U[J]:= U[J] * W; V[J]:= V[J] * W "END";
      COUNT:= COUNT + 1; "IF" W > TOL "THEN" "GOTO" FORWARD;
   OUTM: EM[7]:= W; EM[9]:= COUNT
  "END" COMVECHES;
      "EOP"
```

AUTHORS : T.J. DEKKER, W. HOFFMANN.

CONTRIBUTORS: W. HOFFMANN, S.P.N. VAN KAMPEN, J.G. VERWER.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 731205.

BRIEF DESCRIPTION:
THIS SECTION CONTAINS FIVE PROCEDURES FOR CALCULATING EIGENVALUES AND / OR EIGENVECTORS OF REAL MATRICES:
A) REAEIGVAL CALCULATES THE EIGENVALUES OF A MATRIX, PROVIDED THAT ALL EIGENVALUES ARE REAL,
B) REAEIG1 CALCULATES THE EIGENVALUES, PROVIDED THAT THEY ARE ALL REAL, AND THE EIGENVECTORS OF A MATRIX,
C) REAEIG3 CALCULATES THE EIGENVALUES, PROVIDED THAT THEY ARE ALL REAL, AND THE EIGENVECTORS OF A MATRIX,
D) COMEIGVAL CALCULATES THE EIGENVALUES OF A MATRIX,
E) COMEIG1 CALCULATES THE EIGENVALUES AND EIGENVECTORS OF A MATRIX.

KEYWORDS:
EIGENVALUES,
EIGENVECTORS.

SUBSECTION: REAEIGVAL.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" REAEIGVAL(A, N, EM, VAL); "VALUE" N;
"INTEGER" N; "ARRAY" A, EM, VAL;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE MATRIX WHOSE EIGENVALUES ARE TO BE CALCULATED;
EXIT: THE ARRAY ELEMENTS ARE ALTERED;
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY: EM[0], THE MACHINE PRECISION;
EM[2], THE RELATIVE TOLERANCE USED FOR THE QR ITERATION;
EM[4], THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
FOR THE CD CYBER 73-28 SUITABLE VALUES OF THE DATA TO BE GIVEN IN EM ARE:
EM[0] = "-14,
EM[2] > EM[0] (E.G. EM[2] = "-13),
EM[4] = 10 * N;
EXIT: EM[1], THE INFINITY NORM OF THE EQUILIBRATED MATRIX;
EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE SUBDIAGONAL ELEMENTS NEGLECTED;
EM[5], THE NUMBER OF QR ITERATIONS PERFORMED;
IF THE ITERATION PROCESS IS NOT COMPLETED WITHIN EM[4] ITERATIONS, THE VALUE EM[4] + 1 IS DELIVERED AND IN THIS CASE ONLY THE LAST N - K ELEMENTS OF VAL ARE APPROXIMATE EIGENVALUES OF THE GIVEN MATRIX, WHERE K IS DELIVERED IN REAEIGVAL;
VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:N];
EXIT: THE EIGENVALUES OF THE GIVEN MATRIX ARE DELIVERED IN MONOTONICALLY NONINCREASING ORDER;

MOREOVER:
REAEIGVAL DELIVERS 0, PROVIDED THAT THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE REAEIGVAL DELIVERS K, THE NUMBER OF EIGENVALUES NOT CALCULATED.

PROCEDURES USED:
EQILBR = CP34173,
TFMREAHES = CP34170,
REAVALQRI = CP34180.

REQUIRED CENTRAL MEMORY:
EXECUTION FIELD LENGTH: 3N.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE GIVEN MATRIX IS EQUILIBRATED BY CALLING EQILBR (SEE SECTION 3.2.1.1.1) AND TRANSFORMED TO A SIMILAR UPPER-HESSENBERG MATRIX BY CALLING TFMREAHES (SEE SECTION 3.2.1.2.1.2). THE EIGENVALUES ARE THEN CALCULATED BY CALLING REAVALQRI, WHICH USES SINGLE QR ITERATION (SEE SECTION 3.3.1.2.1).
THE PROCEDURE REAEIGVAL SHOULD BE USED ONLY IF ALL EIGENVALUES ARE REAL.
FOR FURTHER DETAILS SEE REFERENCES [1], [2] AND [3].

SUBSECTION: REAEIG1.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" REAEIG1(A, N, EM, VAL, VEC): "VALUE" N;
"INTEGER" N; "ARRAY" A, EM, VAL, VEC;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE MATRIX WHOSE EIGENVALUES AND EIGENVECTORS ARE TO BE CALCULATED;
EXIT: THE ARRAY ELEMENTS ARE ALTERED;
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

```
EM:     <ARRAY IDENTIFIER>;
        "ARRAY" EM[0:9];
        ENTRY: EM[0], THE MACHINE PRECISION;
               EM[2], THE RELATIVE TOLERANCE USED FOR THE QR
                      ITERATION;
               EM[4], THE MAXIMUM ALLOWED NUMBER OF QR
                      ITERATIONS;
               EM[6], THE TOLERANCE USED FOR THE EIGENVECTORS;
                      FOR EACH EIGENVECTOR THE INVERSE ITERATION
                      ENDS IF THE EUCLIDEAN NORM OF THE RESIDUE
                      VECTOR IS SMALLER THAN EM[1] * EM[6];
               EM[8], THE MAXIMUM ALLOWED NUMBER OF INVERSE
                      ITERATIONS FOR THE CALCULATION OF EACH
                      EIGENVECTOR;
               FOR THE CD CYBER 73-28 SUITABLE VALUES OF THE
               DATA TO BE GIVEN IN EM ARE:
               EM[0] = "-14,
               EM[2] > EM[0] (E.G. EM[2] = "-13),
               EM[4] = 10 * N,
               EM[6] > EM[2] (E.G. EM[6] = "-10),
               EM[8] = 5;
        EXIT:  EM[1], THE INFINITY NORM OF THE EQUILIBRATED MATRIX;
               EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE SUBDIAGONAL
                      ELEMENTS NEGLECTED;
               EM[5], THE NUMBER OF QR ITERATIONS PERFORMED;
                      IF THE ITERATION PROCESS IS NOT COMPLETED
                      WITHIN EM[4] ITERATIONS, THE VALUE EM[4] + 1
                      IS DELIVERED AND IN THIS CASE ONLY THE LAST
                      N - K ELEMENTS OF VAL AND COLUMNS OF VEC ARE
                      APPROXIMATE EIGENVALUES AND EIGENVECTORS OF
                      THE GIVEN MATRIX, WHERE K IS DELIVERED IN
                      REAEIG1;
               EM[7], THE MAXIMUM EUCLIDIAN NORM OF THE RESIDUES
                      OF THE CALCULATED EIGENVECTORS (OF THE TRANS-
                      FORMED MATRIX);
               EM[9], THE LARGEST NUMBER OF INVERSE ITERATIONS
                      PERFORMED FOR THE CALCULATION OF SOME EIGEN-
                      VECTOR; IF, FOR SOME EIGENVECTOR THE
                      EUCLIDEAN NORM OF THE RESIDUE REMAINS
                      LARGER THAN EM[1] * EM[6], THE VALUE
                      EM[8] + 1 IS DELIVERED; NEVERTHELESS THE
                      EIGENVECTORS MAY THEN VERY WELL BE USEFUL,
                      THIS SHOULD BE JUDGED FROM THE VALUE
                      DELIVERED IN EM[7] OR FROM SOME OTHER TEST;
VAL:    <ARRAY IDENTIFIER>;
        "ARRAY" VAL[1:N];
        EXIT: THE EIGENVALUES OF THE GIVEN MATRIX ARE DELIVERED
              IN MONOTONICALLY DECREASING ORDER;
VEC:    <ARRAY IDENTIFIER>;
        "ARRAY" VEC[1:N,1:N];
        EXIT: THE CALCULATED EIGENVECTORS, CORRESPONDING TO THE
              EIGENVALUES IN ARRAY VAL[1:N], ARE DELIVERED IN THE
              COLUMNS OF ARRAY VEC;
```

MOREOVER:
REAEIG1 DELIVERS 0, PROVIDED THAT THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE REAEIG1 DELIVERS K, THE NUMBER OF EIGENVALUES AND EIGENVECTORS NOT CALCULATED.

PROCEDURES USED:
EQILBR = CP34173,
TFMREAHES = CP34170,
BAKREAHES2 = CP34172,
BAKLBR = CP34174,
REAVALQRI = CP34180,
REAVECHES = CP34181,
REASCL = CP34183.

REQUIRED CENTRAL MEMORY:
EXECUTION FIELD LENGTH: N * N + 5N.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

OPTIONS: F.

METHOD AND PERFORMANCE:
THE GIVEN MATRIX IS EQUILIBRATED BY CALLING EQILBR (SEE SECTION 3.2.1.1.1) AND TRANSFORMED TO A SIMILAR UPPER-HESSENBERG MATRIX BY CALLING TFMREAHES (SEE SECTION 3.2.1.2.1.2). THE EIGENVALUES ARE THEN CALCULATED BY CALLING REAVALQRI, WHICH USES SINGLE QR ITERATION (SEE SECTION 3.3.1.2.1).
FURTHERMORE, TO FIND THE EIGENVECTORS WILKINSON'S DEVICE IS FIRST APPLIED [2, P.328 AND 628]. SUBSEQUENTLY THE EIGENVECTORS OF THE UPPER-HESSENBERG MATRIX ARE CALCULATED BY CALLING REAVECHES, WHICH USES INVERSE ITERATION (SEE SECTION 3.3.1.2.1). THE CALCULATED VECTORS ARE THEN BACK-TRANSFORMED TO THE CORRESPONDING EIGENVECTORS OF THE GIVEN MATRIX BY CALLING BAKREAHES2 AND BAKLBR (SEE SECTIONS 3.2.1.2.1.2 AND 3.2.1.1.1). FINALLY THE APPROXIMATE EIGENVECTORS ARE NORMALIZED BY CALLING REASCL (SEE SECTION 1.1.9) SUCH THAT, IN EACH EIGENVECTOR, AN ELEMENT OF MAXIMUM ABSOLUTE VALUE EQUALS 1.
THE PROCEDURE REAEIG1 SHOULD BE USED ONLY IF ALL EIGENVALUES ARE REAL.
FOR FURTHER DETAILS SEE THE GIVEN REFERENCES.

SUBSECTION: REAEIG3.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" REAEIG3(A, N, EM, VAL, VEC); "VALUE" N;
"INTEGER" N; "ARRAY" A, EM, VAL, VEC;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE MATRIX WHOSE EIGENVALUES AND EIGENVECTORS ARE TO BE CALCULATED;
EXIT: THE ARRAY ELEMENTS ARE ALTERED;
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY: EM[0], THE MACHINE PRECISION;
EM[2], THE RELATIVE TOLERANCE USED FOR THE QR ITERATION;
EM[4], THE MAXIMUM ALLOWED NUMBER OF QR ITERATIONS;
FOR THE CD CYBER 73-28 SUITABLE VALUES OF THE DATA TO BE GIVEN IN EM ARE:
EM[0] = "-14,
EM[2] > EM[0] (E.G. EM[2] = "-13),
EM[4] = 10 * N;
EXIT: EM[1], THE INFINITY NORM OF THE EQUILIBRATED MATRIX;
EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE SUBDIAGONAL ELEMENTS NEGLECTED;
EM[5], THE NUMBER OF QR ITERATIONS PERFORMED;
IF THE ITERATION PROCESS IS NOT COMPLETED WITHIN EM[4] ITERATIONS, THE VALUE EM[4] + 1 IS DELIVERED. IN THIS CASE ONLY THE LAST N - K ELEMENTS OF VAL ARE APPROXIMATE EIGENVALUES OF THE GIVEN MATRIX AND NO USEFUL EIGENVECTORS ARE DELIVERED. THE VALUE K IS DELIVERED IN REAEIG3;
VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:N];
EXIT: THE EIGENVALUES OF THE GIVEN MATRIX ARE DELIVERED;

VEC: <ARRAY IDENTIFIER>;
"ARRAY" VEC[1:N,1:N];
EXIT: THE CALCULATED EIGENVECTORS, CORRESPONDING TO THE EIGENVALUES IN ARRAY VAL[1:N], ARE DELIVERED IN THE COLUMNS OF ARRAY VEC;

MOREOVER:
REAEIG3 DELIVERS 0, PROVIDED THAT THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE REAEIG3 DELIVERS K, THE NUMBER OF EIGENVALUES NOT CALCULATED.

PROCEDURES USED:
EQILBR = CP34173,
TEMREAHES = CP34170,
BAKREAHES2 = CP34172,
BAKLBR = CP34174,
REAQRI = CP34186,
REASCL = CP34183.

REQUIRED CENTRAL MEMORY:
EXECUTION FIELD LENGTH: 4N.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE GIVEN MATRIX IS EQUILIBRATED BY CALLING EQILBR (SEE SECTION 3.2.1.1.1) AND TRANSFORMED TO A SIMILAR UPPER-HESSENBERG MATRIX BY CALLING TEMREAHES (SEE SECTION 3.2.1.2.1.2). THE EIGENVALUES AND EIGENVECTORS OF THE UPPER-HESSENBERG MATRIX ARE THEN CALCULATED BY CALLING REAQRI, WHICH USES SINGLE QR ITERATION FOR THE EIGENVALUES AND A DIRECT METHOD FOR THE EIGENVECTORS (SEE SECTION 3.3.1.2.1). FINALLY THE EIGENVECTORS OF THE UPPER-HESSENBERG MATRIX ARE BACK-TRANSFORMED TO THE CORRESPONDING EIGENVECTORS OF THE GIVEN MATRIX BY CALLING BAKREAHES2 (SEE SECTION 3.1.2.1.2.1) AND NORMALIZED BY CALLING REASCL (SEE SECTION 1.1.9) SUCH THAT, IN EACH EIGENVECTOR, AN ELEMENT OF MAXIMUM ABSOLUTE VALUE EQUALS 1.
THE PROCEDURE REAEIG3 SHOULD BE USED ONLY IF ALL EIGENVALUES ARE REAL.
FOR FURTHER DETAILS SEE THE GIVEN REFERENCES.

SUBSECTION: COMEIGVAL.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" COMEIGVAL(A, N, EM, RE, IM); "VALUE" N;
"INTEGER" N; "ARRAY" A, EM, RE, IM;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE MATRIX WHOSE EIGENVALUES ARE TO BE CALCULATED;
EXIT: THE ARRAY ELEMENTS ARE ALTERED;
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY: EM[0], THE MACHINE PRECISION;
EM[2], THE RELATIVE TOLERANCE USED FOR THE QR ITERATION;
EM[4], THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
FOR THE CD CYBER 73-28 SUITABLE VALUES OF THE DATA TO BE GIVEN IN EM ARE:
EM[0] = "-14,
EM[2] > EM[0] (E.G. EM[2] = "-13),
EM[4] = 10 * N;
EXIT: EM[1], THE INFINITY NORM OF THE EQUILIBRATED MATRIX;
EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE SUBDIAGONAL ELEMENTS NEGLECTED;
EM[5], THE NUMBER OF QR ITERATIONS PERFORMED;
IF THE ITERATION PROCESS IS NOT COMPLETED WITHIN EM[4] ITERATIONS, THE VALUE EM[4] + 1 IS DELIVERED AND IN THIS CASE ONLY THE LAST N - K ELEMENTS OF RE AND IM ARE APPROXIMATE EIGENVALUES OF THE GIVEN MATRIX, WHERE K IS DELIVERED IN COMEIGVAL;
RE,IM: <ARRAY IDENTIFIER>;
"ARRAY" RE, IM[1:N];
EXIT: THE REAL AND IMAGINARY PARTS OF THE CALCULATED EIGENVALUES OF THE GIVEN MATRIX ARE DELIVERED IN ARRAY RE, IM[1:N], THE MEMBERS OF EACH NONREAL COMPLEX CONJUGATE PAIR BEING CONSECUTIVE;

MOREOVER:
COMEIGVAL DELIVERS 0, PROVIDED THAT THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE COMEIGVAL DELIVERS K, THE NUMBER OF EIGENVALUES NOT CALCULATED.

PROCEDURES USED:
EQILBR = CP34173,
TFMREAHES = CP34170,
COMVALQRI = CP34190.

REQUIRED CENTRAL MEMORY:
EXECUTION FIELD LENGTH: 3N.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE GIVEN MATRIX IS EQUILIBRATED BY CALLING EQILBR (SEE SECTION 3.2.1.1.1) AND TRANSFORMED TO A SIMILAR UPPER-HESSENBERG MATRIX BY CALLING TEMREAHES (SEE SECTION 3.2.1.2.1.2). THE EIGENVALUES ARE THEN CALCULATED BY CALLING COMVALQRI, WHICH USES DOUBLE QR ITERATION (SEE SECTION 3.3.1.2.1).
FOR FURTHER DETAILS SEE REFERENCES [1], [2] AND [3].

SUBSECTION: COMEIG1.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"INTEGER" "PROCEDURE" COMEIG1(A, N, EM, RE, IM, VEC); "VALUE" N;
"INTEGER" N; "ARRAY" A, EM, RE, IM, VEC;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE MATRIX WHOSE EIGENVALUES AND EIGENVECTORS ARE TO BE CALCULATED;
EXIT: THE ARRAY ELEMENTS ARE ALTERED;
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:9];
ENTRY: EM[0], THE MACHINE PRECISION;
EM[2], THE RELATIVE TOLERANCE USED FOR THE QR ITERATION;
EM[4], THE MAXIMUM ALLOWED NUMBER OF QR ITERATIONS;
EM[6], THE TOLERANCE USED FOR THE EIGENVECTORS; FOR EACH EIGENVECTOR THE INVERSE ITERATION ENDS IF THE EUCLIDEAN NORM OF THE RESIDUE VECTOR IS SMALLER THAN EM[1] * EM[6];
EM[8], THE MAXIMUM ALLOWED NUMBER OF INVERSE ITERATIONS FOR THE CALCULATION OF EACH EIGENVECTOR;
FOR THE CD CYBER 73-28 SUITABLE VALUES OF THE DATA TO BE GIVEN IN EM ARE:
EM[0] = "-14,
EM[2] > EM[0] (E.G. EM[2] = "-13),
EM[4] = 10 * N,
EM[6] > EM[2] (E.G. EM[6] = "-10),
EM[8] = 5;

EXIT: EM[1], THE INFINITY NORM OF THE EQUILIBRATED MATRIX;
EM[3], THE MAXIMUM ABSOLUTE VALUE OF THE SUBDIAGONAL ELEMENTS NEGLECTED;
EM[5], THE NUMBER OF QR ITERATIONS PERFORMED;
IF THE ITERATION PROCESS IS NOT COMPLETED WITHIN EM[4] ITERATIONS, THE VALUE EM[4] + 1 IS DELIVERED AND IN THIS CASE ONLY THE LAST N - K ELEMENTS OF RE, IM AND COLUMNS OF VEC ARE APPROXIMATE EIGENVALUES AND EIGENVECTORS OF THE GIVEN MATRIX, WHERE K IS DELIVERED IN COMEIG1;
EM[7], THE MAXIMUM EUCLIDIAN NORM OF THE RESIDUES OF THE CALCULATED EIGENVECTORS (OF THE TRANSFORMED MATRIX);
EM[9], THE LARGEST NUMBER OF INVERSE ITERATIONS PERFORMED FOR THE CALCULATION OF SOME EIGENVECTOR; IF THE EUCLIDIAN NORM OF THE RESIDUE FOR ONE OR MORE EIGENVECTORS REMAINS LARGER THAN EM[1] * EM[6], THE VALUE EM[8]+1 IS DELIVERED; NEVERTHELESS THE EIGENVECTORS MAY THEN VERY WELL BE USEFUL, THIS SHOULD BE JUDGED FROM THE VALUE DELIVERED IN EM[7] OR FROM SOME OTHER TEST;

RE,IM: <ARRAY IDENTIFIER>;
"ARRAY" RE, IM[1:N];
EXIT: THE REAL AND IMAGINARY PARTS OF THE CALCULATED EIGENVALUES OF THE GIVEN MATRIX ARE DELIVERED IN ARRAY RE, IM[1:N], THE MEMBERS OF EACH NONREAL COMPLEX CONJUGATE PAIR BEING CONSECUTIVE;

VEC: <ARRAY IDENTFIER>;
"ARRAY" VEC[1:N,1:N];
EXIT: THE CALCULATED EIGENVECTORS ARE DELIVERED IN THE COLUMNS OF ARRAY VEC;
AN EIGENVECTOR, CORRESPONDING TO A REAL EIGENVALUE GIVEN IN ARRAY RE, IS DELIVERED IN THE CORRESPONDING COLUMN OF ARRAY VEC;
THE REAL AND IMAGINARY PART OF AN EIGENVECTOR, CORRESPONDING TO THE FIRST MEMBER OF A NONREAL COMPLEX CONJUGATE PAIR OF EIGENVALUES GIVEN IN THE ARRAYS RE, IM, ARE DELIVERED IN THE TWO CONSECUTIVE COLUMNS OF ARRAY VEC CORRESPONDING TO THIS PAIR (THE EIGENVECTORS CORRESPONDING TO THE SECOND MEMBERS OF NONREAL COMPLEX CONJUGATE PAIRS ARE NOT DELIVERED, SINCE THEY ARE SIMPLY THE COMPLEX CONJUGATE OF THOSE CORRESPONDING TO THE FIRST MEMBER OF SUCH PAIRS);

MOREOVER:
COMEIG1 DELIVERS 0, PROVIDED THAT THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE COMEIG1 DELIVERS K, THE NUMBER OF EIGENVALUES AND EIGENVECTORS NOT CALCULATED.

PROCEDURES USED:
EQILBR = CP34173,
TFMREAHES = CP34170,
BAKREAHES2 = CP34172,
BAKLBR = CP34174,
REAVECHES = CP34181,
COMVALQRI = CP34190,
COMVECHES = CP34191,
COMSCL = CP34193.

REQUIRED CENTRAL MEMORY:
EXECUTION FIELD LENGTH: N * N + 5N.

RUNNING TIME: ROUGHLY PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE GIVEN MATRIX IS EQUILIBRATED BY CALLING EQILBR (SEE SECTION 3.2.1.1.1) AND TRANSFORMED TO A SIMILAR UPPER-HESSENBERG MATRIX BY CALLING TFMREAHES (SEE SECTION 3.2.1.2.1.2). THE EIGENVALUES ARE THEN CALCULATED BY CALLING COMVALQRI, WHICH USES DOUBLE QR ITERATION (SEE SECTION 3.3.1.2.1).
FURTHERMORE, TO FIND THE EIGENVECTORS WILKINSON'S DEVICE IS FIRST APPLIED [2, P.328 AND 628]. SUBSEQUENTLY THE EIGENVECTORS OF THE UPPER-HESSENBERG MATRIX ARE COMPUTED BY CALLING REAVECHES FOR THE REAL EIGENVALUES AND COMVECHES FOR THE OTHERS (SECTION 3.3.1.2.1.). THE COMPUTED VECTORS ARE THEN BACK-TRANSFORMED TO THE CORRESPONDING EIGENVECTORS OF THE GIVEN MATRIX BY CALLING BAKREAHES2 AND BAKLBR (SEE SECTIONS 3.2.1.2.1.2 AND 3.2.1.1.1). FINALLY THE APPROXIMATE EIGENVECTORS ARE NORMALIZED BY CALLING COMSCL (SEE SECTION 1.1.9) SUCH THAT, IN EACH EIGENVECTOR, AN ELEMENT OF MAXIMUM MODULUS EQUALS 1.
FOR FURTHER DETAILS SEE THE GIVEN REFERENCES.

REFERENCES:
[1]. T.J. DEKKER AND W. HOFFMANN.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2.
MC TRACT 23, 1968, MATH. CENTR., AMSTERDAM.
[2]. J.H. WILKINSON.
THE ALGEBRAIC EIGENVALUE PROBLEM.
CLARENDON PRESS, OXFORD, 1965.
[3]. J.G. FRANCIS.
THE QR TRANSFORMATION, PARTS 1 AND 2.
COMP. J. 4 (1961), 265 - 271 AND 332 - 345.
[4]. J.M. VARAH.
EIGENVECTORS OF A REAL MATRIX BY INVERSE ITERATION.
STANFORD UNIVERSITY, TECH. REP. NO. CS 34, 1966.

EXAMPLE OF USE:

IN THIS SECTION WE ONLY GIVE AN EXAMPLE OF USE OF THE PROCEDURES REAEIG3 AND COMEIGVAL, BECAUSE A CALL OF THE OTHER PROCEDURES IS ALMOST SIMILAR.

THE EIGENVALUES AND CORRESPONDING EIGENVECTORS OF A MATRIX, STORED IN ARRAY A, WITH A[I,J]:= "IF" I=1 "THEN" 1 "ELSE" 1 / (I + J - 1), MAY BE OBTAINED BY THE PROCEDURE REAEIG3 IN THE FOLLOWING PROGRAM:

```
"BEGIN" "INTEGER" I, J, M;
    "ARRAY" A, VEC[1:4,1:4], EM[0:5], VAL[1:4];
    "INTEGER" "PROCEDURE" REAEIG3(A, N, EM, VAL, VEC);
    "CODE" 34187;

    "FOR" I:= 1, 2, 3, 4 "DO" "FOR" J:= 1, 2, 3, 4 "DO"
    A[I,J]:= "IF" I = 1 "THEN" 1 "ELSE" 1 / ( I + J - 1);
    EM[0]:= "-14; EM[2]:= "-13; EM[4]:= 40;
    M:= REAEIG3(A, 4, EM, VAL, VEC);
    OUTPUT(61, "("D, /")", M);
    "FOR" I:= M + 1 "STEP" 1 "UNTIL" 4 "DO"
    OUTPUT(61, "("/, 2(+.13D"+2D, 2B), /, 3(21B, +.13D"+2D, /)")",
    VAL[I], VEC[1,I], VEC[2,I], VEC[3,I], VEC[4,I]);
    OUTPUT(61, "("/, 2(.2D"+2D, /), ZD")", EM[1], EM[3], EM[5])
"END"
```

THE PROGRAM DELIVERS(THE RESULTS ARE CORRECT UP TO TWELVE DIGITS):

THE NUMBER OF NOT CALCULATED EIGENVALUES: 0

THE EIGENVALUES AND CORRESPONDING EIGENVECTORS:

```
+.1886632138548"+01  +.1000000000000"+01
                     +.3942239850770"+00
                     +.2773202862566"+00
                     +.2150878672143"+00

-.1980145931103"+00  +.1000000000000"+01
                     -.7388484093937"+00
                     -.3116238593839"+00
                     -.1475423243327"+00

-.1228293686543"-01  -.4634736456357"+00
                     +.1000000000000"+01
                     -.1542548002737"+00
                     -.3765787365625"+00

-.1441323817331"-03  +.1095712655340"+00
                     -.6208405341138"+00
                     +.1000000000000"+01
                     -.4887465241876"+00
```

```
EM[1] = .40"+01
EM[3] = .15"-14
EM[5] = 5 .
```

THE COMPLEX EIGENVALUES OF A MATRIX STORED IN ARRAY A WITH N = 3 AND THE ROWS (8, -1, -5), (-4, 4, -2) AND (18, -5, -7), MAY BE OBTAINED BY THE PROCEDURE COMEIGVAL IN THE FOLLOWING PROGRAM:

```
"BEGIN" "INTEGER" I, M;
    "ARRAY" A[1:3,1:3], EM[0:5], RE, IM[1:3];
    "INTEGER" "PROCEDURE" COMEIGVAL(A, N, EM, RE, IM);
    "CODE" 34192;

    EM[0]:= "-14; EM[2]:= "-13; EM[4]:= 30;
    A[1,1]:= 8; A[1,2]:= -1; A[1,3]:= -5;
    A[2,1]:= -4; A[2,2]:= 4; A[2,3]:= -2;
    A[3,1]:= 18; A[3,2]:= -5; A[3,3]:= -7;
    M:= COMEIGVAL(A, 3, EM, RE, IM);
    OUTPUT(61, "("D, /")", M);
    "FOR" I:= M + 1 "STEP" 1 "UNTIL" 3 "DO"
    OUTPUT(61, "("2(+.13D"+2D, 28), /")", RE[I], IM[I]);
    OUTPUT(61, "("/, 2(.2D"+2D, /), ZD")", EM[1], EM[3], EM[5])
"END"
```

THE PROGRAM DELIVERS(THE RESULTS ARE CORRECT UP TO TWELVE DIGITS):

THE NUMBER OF NOT CALCULATED EIGENVALUES: 0

```
THE EIGENVALUES: +.2000000000000"+01  +.4000000000000"+01
                 +.2000000000000"+01  -.4000000000000"+01
                 +.9999999999998"+00  +.0000000000000"+00

THE ARRAY EM: EM[1] = .30"+02
              EM[3] = .78"-17
              EM[5] = 6 .
```

SOURCE TEXTS:

```
"CODE" 34182;
    "COMMENT" MCA 2412;
    "INTEGER" "PROCEDURE" REAEIGVAL(A, N, EM, VAL); "VALUE" N;
    "INTEGER" N; "ARRAY" A, EM, VAL;
    "BEGIN" "INTEGER" I, J; "REAL" R;
        "ARRAY" D[1:N]; "INTEGER" "ARRAY" INT, INTO[1:N];

        "PROCEDURE" TFMREAHES(A, N, EM, INT); "CODE" 34170;
        "PROCEDURE" EQILBR(A, N, EM, D, INT); "CODE" 34173;
        "INTEGER" "PROCEDURE" REAVALQRI(A, N, EM, VAL); "CODE" 34180;

        EQILBR(A, N, EM, D, INTO); TFMREAHES(A, N, EM, INT);
        J:= REAEIGVAL:= REAVALQRI(A, N, EM, VAL);
        "FOR" I:= J + 1 "STEP" 1 "UNTIL" N "DO"
        "FOR" J:= I + 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" "IF" VAL[J] > VAL[I] "THEN"
            "BEGIN" R:= VAL[I]; VAL[I]:= VAL[J]; VAL[J]:= R "END"
        "END"
    "END" REAEIGVAL;
        "EOP"
```

```
"CODE" 34184;
    "COMMENT" MCA 2414;
    "INTEGER" "PROCEDURE" REAEIG1(A, N, EM, VAL, VEC); "VALUE" N;
    "INTEGER" N; "ARRAY" A, EM, VAL, VEC;
    "BEGIN" "INTEGER" I, K, MAX, J, L;
        "REAL" RESIDU, R, MACHTOL;
        "ARRAY" D, V[1:N], B[1:N,1:N];
        "INTEGER" "ARRAY" INT, INTO[1:N];

        "PROCEDURE" TFMREAHES(A, N, EM, INT); "CODE" 34170;
        "PROCEDURE" BAKREAHES2(A, N, N1, N2, INT, VEC); "CODE" 34172;
        "PROCEDURE" EQILBR(A, N, EM, D, INT); "CODE" 34173;
        "PROCEDURE" BAKLBR(N, N1, N2, D, INT, VEC); "CODE" 34174;
        "INTEGER" "PROCEDURE" REAVALQRI(A, N, EM, VAL); "CODE" 34180;
        "PROCEDURE" REAVECHES(A, N, LAMBDA, EM, V); "CODE" 34181;
        "PROCEDURE" REASCL(A, N, N1, N2); "CODE" 34183;

        RESIDU:= 0; MAX:= 0; EQILBR(A, N, EM, D, INTO);
        TFMREAHES(A, N, EM, INT);
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        "FOR" J:= ("IF" I = 1 "THEN" 1 "ELSE" I - 1)
        "STEP" 1 "UNTIL" N "DO" B[I,J]:= A[I,J];
        K:= REAEIG1:= REAVALQRI(B, N, EM, VAL);
        "FOR" I:= K + 1 "STEP" 1 "UNTIL" N "DO"
        "FOR" J:= I + 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" "IF" VAL[J] > VAL[I] "THEN"
            "BEGIN" R:= VAL[I]; VAL[I]:= VAL[J]; VAL[J]:= R "END"
        "END";
        MACHTOL:= EM[0] * EM[1];
        "FOR" L:= K + 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" "IF" L > 1 "THEN"
            "BEGIN" "IF" VAL[L - 1] - VAL[L] < MACHTOL "THEN"
                VAL[L]:= VAL[L - 1] - MACHTOL
            "END";
            "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
            "FOR" J:= ("IF" I = 1 "THEN" 1 "ELSE" I - 1)
            "STEP" 1 "UNTIL" N "DO" B[I,J]:= A[I,J];
            REAVECHES(B, N, VAL[L], EM, V);
            "IF" EM[7] > RESIDU "THEN" RESIDU:= EM[7];
            "IF" EM[9] > MAX "THEN" MAX:= EM[9];
            "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO" VEC[J,L]:= V[J]
        "END";
        EM[7]:= RESIDU; EM[9]:= MAX;
        BAKREAHES2(A, N, K + 1, N, INT, VEC);
        BAKLBR(N, K + 1, N, D, INTO, VEC);
        REASCL(VEC, N, K + 1, N)
    "END" REAEIG1;
        "EOP"
```

```
"CODE" 34187;
    "COMMENT" MCA 2417;
    "INTEGER" "PROCEDURE" REAEIG3(A, N, EM, VAL, VEC); "VALUE" N;
    "INTEGER" N; "ARRAY" A, EM, VAL, VEC;
    "BEGIN" "INTEGER" I; "REAL" S;
        "INTEGER" "ARRAY" INT, INTO[1:N]; "ARRAY" D[1:N];

        "PROCEDURE" TFMREAHES(A, N, EM, INT); "CODE" 34170;
        "PROCEDURE" BAKREAHES2(A, N, N1, N2, INT, VEC); "CODE" 34172;
        "PROCEDURE" EQILBR(A, N, EM, D, INT); "CODE" 34173;
        "PROCEDURE" BAKLBR(N, N1, N2, D, INT, VEC); "CODE" 34174;
        "PROCEDURE" REASCL(A, N, N1, N2); "CODE" 34183;
        "INTEGER" "PROCEDURE" REAQRI(A, N, EM, VAL, VEC); "CODE" 34186;

        EQILBR(A, N, EM, D, INTO); TFMREAHES(A, N, EM, INT);
        I:= REAEIG3:= REAQRI(A, N, EM, VAL, VEC);
        "IF" I = 0 "THEN"
        "BEGIN" BAKREAHES2(A, N, 1, N, INT, VEC);
            BAKLBR(N, 1, N, D, INTO, VEC); REASCL(VEC, N, 1, N)
        "END"
    "END" REAEIG3;
        "EOP"

"CODE" 34192;
    "COMMENT" MCA 2422;
    "INTEGER" "PROCEDURE" COMEIGVAL(A, N, EM, RE, IM); "VALUE" N;
    "INTEGER" N; "ARRAY" A, EM, RE, IM;
    "BEGIN" "INTEGER" "ARRAY" INT, INTO[1:N];
        "ARRAY" D[1:N];

        "PROCEDURE" EQILBR(A, N, EM, D, INT); "CODE" 34173;
        "PROCEDURE" TFMREAHES(A, N, EM, INT); "CODE" 34170;
        "INTEGER" "PROCEDURE" COMVALQRI(A, N, EM, RE, IM);
        "CODE" 34190;

        EQILBR(A, N, EM, D, INTO); TFMREAHES(A, N, EM, INT);
        COMEIGVAL:= COMVALQRI(A, N, EM, RE, IM)
    "END" COMEIGVAL;
        "EOP"
```

```
"CODE" 34194;
    "COMMENT" MCA 2424;
    "INTEGER" "PROCEDURE" COMEIG1(A, N, EM, RE, IM, VEC);
    "VALUE" N; "INTEGER" N;
    "ARRAY" A, EM, RE, IM, VEC;
    "BEGIN" "INTEGER" I, J, K, PJ, ITT;
        "REAL" X, Y, MAX, NEPS;
        "ARRAY" AB[1:N,1:N], D, U, V[1:N];
        "INTEGER" "ARRAY" INT, INTO[1:N];

        "PROCEDURE" TRANSFER;
        "BEGIN" "INTEGER" I, J;
            "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
            "FOR" J:= ("IF" I = 1 "THEN" 1 "ELSE" I - 1) "STEP" 1
            "UNTIL" N "DO" AB[I,J]:= A[I,J]
        "END" TRANSFER;

        "PROCEDURE" EQILBR(A, N, EM, D, INT); "CODE" 34173;
        "PROCEDURE" TFMREAHES(A, N, EM, INT); "CODE" 34170;
        "PROCEDURE" BAKREAHES2(A, N, N1, N2, INT, VEC); "CODE" 34172;
        "PROCEDURE" BAKLBR(N, N1, N2, D, INT, VEC); "CODE" 34174;
        "PROCEDURE" REAVECHES(A, N, LAMBDA, EM, V); "CODE" 34181;
        "PROCEDURE" COMSCL(A, N, N1, N2, IM); "CODE" 34193;
        "INTEGER" "PROCEDURE" COMVALQRI(A, N, EM, RE, IM);
        "CODE" 34190;
        "PROCEDURE" COMVECHES(A, N, LAMBDA, MU, EM, U, V);
        "CODE" 34191;

        EQILBR(A, N, EM, D, INTO); TFMREAHES(A, N, EM, INT); TRANSFER;
        K:= COMEIG1:= COMVALQRI(AB, N, EM, RE, IM);
        NEPS:= EM[0] * EM[1]; MAX:= 0; ITT:= 0;
        "FOR" I:= K + 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" X:= RE[I]; Y:= IM[I]; PJ:= 0;
         AGAIN: "FOR" J:= K + 1 "STEP" 1 "UNTIL" I - 1 "DO"
            "BEGIN" "IF" ((X - RE[J]) ** 2 +
                (Y - IM[J]) ** 2 <= NEPS ** 2) "THEN"
                "BEGIN" "IF" PJ = J "THEN" NEPS:= EM[2] * EM[1]
                    "ELSE" PJ:= J; X:= X + 2 * NEPS; "GOTO" AGAIN
                "END"
            "END";
            RE[I]:= X; TRANSFER; "IF" Y ^= 0 "THEN"
            "BEGIN" COMVECHES(AB, N, RE[I], IM[I], EM, U, V);
                "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO" VEC[J,I]:= U[J];
                I:= I + 1; RE[I]:= X
            "END"
            "ELSE" REAVECHES(AB, N, X, EM, V);
            "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO" VEC[J,I]:= V[J];
            "IF" EM[7] > MAX "THEN" MAX:= EM[7];
            ITT:= "IF" ITT > EM[9] "THEN" ITT "ELSE" EM[9]
        "END";
        EM[7]:= MAX; EM[9]:= ITT; BAKREAHES2(A, N, K + 1, N, INT, VEC);
        BAKLBR(N, K + 1, N, D, INTO, VEC); COMSCL(VEC, N, K + 1, N, IM)
    "END" COMEIG1;
        "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

CONTRIBUTORS : H.FIOLET, C.G. VAN DER LAAN.

INSTITUTE : MATHEMATICAL CENTRE.

RECEIVED: 730917.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS FOUR PROCEDURES FOR CALCULATING THE EIGENVALUES OR THE EIGENVALUES AND EIGENVECTORS OF COMPLEX HERMITIAN MATRICES.
EIGVALHRM CALCULATES THE EIGENVALUES OF A HERMITIAN MATRIX.
EIGHRM CALCULATES THE EIGENVALUES AND EIGENVECTORS OF A HERMITIAN MATRIX.
ORIVALHRM CALCULATES THE EIGENVALUES OF A HERMITIAN MATRIX.
ORIHRM CALCULATES THE EIGENVALUES AND EIGENVECTORS OF A HERMITIAN MATRIX.
WHEN A SMALL NUMBER OF EIGENVALUES OR EIGENVALUES AND EIGENVECTORS IS REQUIRED, THE USE OF EIGVALHRM OR EIGHRM IS RECOMMANDED: WHEN MORE THAN, SAY, 25 PERCENT OF THE EIGENSYSTEM IS REQUIRED. THE PROCEDURES ORIVALHRM OR ORIHRM ARE TO BE USED.

SUBSECTION: EIGVALHRM.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :
"PROCEDURE" EIGVALHRM(A, N, NUMVAL, VAL, EM); "VALUE" N, NUMVAL;
"INTEGER" N, NUMVAL; "ARRAY" A, VAL, EM;

THE MEANING OF THE FORMAL PARAMETERS IS :

A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE REAL PART OF THE UPPER TRIANGLE OF THE HERMITIAN MATRIX MUST BE GIVEN IN THE UPPER TRIANGULAR PART OF A (THE ELEMENTS A[I,J], I<=J); THE IMAGINARY PART OF THE STRICT LOWER TRIANGLE OF THE HERMITIAN MATRIX MUST BE GIVEN IN THE STRICT LOWER PART OF A (THE ELEMENTS A[I,J],I>J);
THE ELEMENTS AF A ARE ALTERED;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

NUMVAL: <ARITHMETIC EXPRESSION>;
EIGVALHRM CALCULATES THE LARGEST NUMVAL EIGENVALUES OF THE HERMITIAN MATRIX;

VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:NUMVAL];
EXIT:
IN ARRAY VAL THE LARGEST NUMVAL EIGENVALUES ARE DELIVERED IN MONOTONICALLY NONINCREASING ORDER;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:3];
ENTRY:
EM[0]: THE MACHINE PRECISION;
EM[2]: THE RELATIVE TOLERANCE FOR THE EIGENVALUES;
MORE PRECISELY: THE TOLERANCE FOR EACH EIGENVALUE LAMBDA, IS ABS(LAMBDA)*EM[2]+EM[1]*EM[0];
EXIT:
EM[1]: AN ESTIMATE OF A NORM OF THE ORIGINAL MATRIX;
EM[3]: THE NUMBER OF ITERATIONS PERFORMED.

PROCEDURES USED:

HSHHRMTRIVAL = CP34364,
VALSYMTRI = CP34151.

REQUIRED CENTRAL MEMORY:
TWO AUXILIARY ARRAYS OF ORDER N AND N - 1 RESPECTIVELY ARE DECLARED

RUNNING TIME: PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE QRIHRM (THIS SECTION).

EXAMPLE OF USE: SEE EIGHRM (THIS SECTION).

SUBSECTION: EIGHRM.

CALLING SEQUENCE :

```
    THE HEADING OF THE PROCEDURE READS :
    "PROCEDURE" EIGHRM(A, N, NUMVAL, VAL, VECR, VECI, EM);
    "VALUE" N, NUMVAL; "INTEGER" N, NUMVAL;
    "ARRAY" A, VAL, VECR, VECI, EM;

    THE MEANING OF THE FORMAL PARAMETERS IS :
    A:          <ARRAY IDENTIFIER>;
                "ARRAY" A[1:N,1:N];
                ENTRY: THE REAL PART OF THE UPPER TRIANGLE OF THE
                       HERMITIAN MATRIX MUST BE GIVEN IN THE UPPER
                       TRIANGULAR PART OF A (THE ELEMENTS A[I,J], I<=J);
                       THE IMAGINARY PART OF THE STRICT LOWER TRIANGLE
                       OF THE HERMITIAN MATRIX MUST BE GIVEN IN THE
                       STRICT LOWER PART OF A (THE ELEMENTS A[I,J],I>J);
                THE ELEMENTS AF A ARE ALTERED;
    N:          <ARITHMETIC EXPRESSION>;
                THE ORDER OF THE GIVEN MATRIX;
    NUMVAL:     <ARITHMETIC EXPRESSION>;
                EIGHRM CALCULATES THE LARGEST NUMVAL EIGENVALUES OF THE
                HERMITIAN MATRIX;
    VAL:        <ARRAY IDENTIFIER>;
                "ARRAY" VAL[1:NUMVAL];
                EXIT:
                IN ARRAY VAL THE LARGEST NUMVAL EIGENVALUES ARE
                DELIVERED IN MONOTONICALLY NONINCREASING ORDER;
    VECR,VECI:  <ARRAY IDENTIFIER>;
                "ARRAY" VECR,VECI[1:N,1:NUMVAL];
                EXIT:
                THE CALCULATED EIGENVECTORS;
                THE COMPLEX EIGENVECTOR WITH REAL PART VECR[1:N,I] AND
                IMAGINARY PART VECI[1:N,I] CORRESPONDS TO THE EIGENVALUE
                VAL[I], I=1,...,NUMVAL;
```

```
    EM:         <ARRAY IDENTIFIER>;
                "ARRAY" EM[0:9];
                ENTRY:
                EM[0]: THE MACHINE PRECISION;
                EM[2]: THE RELATIVE TOLERANCE FOR THE EIGENVALUES;
                       MORE PRECISELY: THE TOLERANCE FOR EACH EIGENVALUE
                       LAMBDA, IS ABS(LAMBDA)*EM[2]+EM[1]*EM[0];
                EM[4]: THE ORTHOGONALIZATION PARAMETER (E.G. .01);
                EM[6]: THE TOLERANCE FOR THE EIGENVECTORS;
                EM[8]: THE  MAXIMUM NUMBER OF INVERSE ITERATIONS ALLOWED
                       FOR THE CALCULATION OF EACH EIGENVECTOR;
                EXIT:
                EM[1]: AN ESTIMATE OF A NORM OF THE ORIGINAL MATRIX;
                EM[3]: THE NUMBER OF ITERATIONS PERFORMED;
                EM[5]: THE  NUMBER  OF EIGENVECTORS INVOLVED IN THE LAST
                       GRAM-SCHMIDT ORTHOGONALIZATION;
                EM[7]: THE MAXIMUM EUCLIDEAN NORM OF THE RESIDUES OF THE
                       CALCULATED EIGENVECTORS;
                EM[9]: THE   LARGEST   NUMBER   OF   INVERSE    ITERATIONS
                       PERFORMED   FOR   THE   CALCULATION   OF     SOME
                       EIGENVECTOR; IF, HOWEVER, FOR SOME  CALCULATED
                       EIGENVECTOR, THE  EUCLIDEAN NORM OF THE  RESIDUES
                       REMAINS     GREATER  THAN   EM[1]*EM[6],     THEN
                       EM[9]:=EM[8]+1.
```

PROCEDURES USED:

```
    HSHHRMTRI = CP34363,
    VALSYMTRI = CP34151,
    VECSYMTRI = CP34152,
    BAKHRMTRI = CP34365.
```

REQUIRED CENTRAL MEMORY:
THREE AUXILIARY ARRAYS OF ORDER N-1 AND TWO OF ORDER N ARE DECLARED

RUNNING TIME: PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE ORIHRM (THIS SECTION).

EXAMPLE OF USE:

LET EIGHRM CALCULATE THE LARGEST EIGENVALUE AND THE CORRESPONDING EIGENVECTOR OF THE FOLLOWING MATRIX:
(SEE GREGORY AND KARNEY, CHAPTER 6, EXAMPLE 6.6)

```
     3    1    0   +2I
     1    3   -2I   0
     0   +2I   1    1
    -2I   0    1    1
```

THE EIGENVECTORS ARE NORMALIZED BY THE PROCEDURE SCLCOM (SEE SECTION 1.2.11.).

```
"BEGIN"
"COMMENT" GREGORY AND KARNEY,CHAPTER 6, EXAMPLE 6.6;
"PROCEDURE" SCLCOM(AR,AI,N,N1,N2);"CODE" 34360;
"PROCEDURE" INIMAT(LR,UR,LC,UC,A,X);"CODE" 31011;
"PROCEDURE" EIGHRM(A,N,NUMVAL,VAL,VECR,VECI,EM);"CODE" 34369;
"REAL" "ARRAY" A[1:4,1:4],VAL[1:1],VECR,VECI[1:4,1:1],EM[0:9];
"INTEGER" I;
INIMAT(1,4,1,4,A,0);
A[1,1]:=A[2,2]:=3;
A[1,2]:=A[3,3]:=A[3,4]:=A[4,4]:=1;
A[3,2]:=2;A[4,1]:=-2;
EM[0]:=5"-14;EM[2]:="-12;
EM[4]:=.01;EM[6]:="-10;EM[8]:=5;
EIGHRM(A,4,1,VAL,VECR,VECI,EM);
SCLCOM(VECR,VECI,4,1,1);
OUTPUT(61,"("""("LARGEST EIGENVALUE: ")",N/")",VAL[1]);
OUTPUT(61,"("""("CORRESPONDING EIGENVECTOR:")",/")");
"FOR" I:=1,2,3,4 "DO"
OUTPUT(61,"("+D.D,+D.DDD,"("*I")",/")",VECR[I,1],VECI[I,1]);
"FOR" I:=1,3,5,7,9 "DO"
OUTPUT(61,"("/,"("EM[")",D,"("]: ")",+D.DDD"+DD")",I,EM[I]);
"END"
```

DELIVERS:

```
LARGEST EIGENVALUE: +4.8284271247462"000
CORRESPONDING EIGENVECTOR:
+1.0+0.000*I
+1.0+0.000*I
-0.0+0.414*I
+0.0-0.414*I

EM[1]: +6.000"+00
EM[3]: +1.800"+01
EM[5]: +1.000"+00
EM[7]: +5.303"-14
EM[9]: +1.000"+00
```

SUBSECTION: QRIVALHRM.

CALLING SEQUENCE :

```
    THE HEADING OF THE PROCEDURE READS :
    "INTEGER" "PROCEDURE" QRIVALHRM(A, N, VAL, EM); "VALUE" N;
    "INTEGER" N; "ARRAY" A, VAL, EM;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    A:          <ARRAY IDENTIFIER>;
                "APRAY" A[1:N,1:N];
                ENTRY: THE  REAL  PART  OF  THE  UPPER  TRIANGLE OF  THE
                       HERMITIAN   MATRIX  MUST  BE  GIVEN IN THE  UPPER
                       TRIANGULAR PART OF A (THE ELEMENTS A[I,J], I<=J);
                       THE  IMAGINARY  PART OF THE STRICT LOWER TRIANGLE
                       OF  THE   HERMITIAN  MATRIX  MUST BE GIVEN IN THE
                       STRICT LOWER PART OF A (THE ELEMENTS A[I,J],I>J);
                THE ELEMENTS AF A ARE ALTERED;
    N:          <ARITHMETIC EXPRESSION>;
                THE ORDER OF THE GIVEN MATRIX;
    VAL:        <ARRAY IDENTIFIER>;
                "ARRAY" VAL[1:N];
                EXIT:
                THE CALCULATED EIGENVALUES;
    EM:         <ARRAY IDENTIFIER>;
                "ARRAY" EM[0:5];
                ENTRY:
                EM[0]: THE MACHINE PRECISION;
                EM[2]: THE RELATIVE TOLERANCE FOR THE QR ITERATION;
                EM[4]: THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
                EXIT:
                EM[1]: AN ESTIMATE OF A NORM OF THE ORIGINAL MATRIX;
                EM[3]: THE  MAXIMUM   ABSOLUTE VALUE OF THE   CODIAGONAL
                       ELEMENTS NEGLECTED;
                EM[5]: THE NUMBER OF ITERATIONS PERFORMED;
                       EM[5]:= EM[4]+1 IN THE CASE QRIVALHRM^=0;

    QRIVALHRM:=0, PROVIDED  THE QR ITERATION IS COMPLETED  WITHIN EM[4]
    ITERATIONS; OTHERWISE, QRIVALHRM:=THE NUMBER OF EIGENVALUES, K, NOT
    CALCULATED AND ONLY THE LAST N-K ELEMENTS OF VAL ARE APPROXIMATE
    EIGENVALUES OF THE ORIGINAL HERMITEAN MATRIX.
```

PROCEDURES USED:

HSHHRMTRIVAL = CP34364,
QRIVALSYMTRI = CP34160.

REQUIRED CENTRAL MEMORY:
TWO AUXILIARY ARRAYS OF ORDER N ARE DECLARED.

RUNNING TIME: PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE ORIHRM (THIS SECTION).

EXAMPLE OF USE: SEE ORIHRM (THIS SECTION).

SUBSECTION: ORIHRM.

CALLING SEQUENCE :

```
    THE HEADING OF THE PROCEDURE READS :
    "INTEGER" "PROCEDURE" ORIHRM(A, N, VAL, VR, VI, EM); "VALUE" N;
    "INTEGER" N; "ARRAY" A, VAL, VR, VI, EM;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    A:          <ARRAY IDENTIFIER>;
                "ARRAY" A[1:N,1:N];
                ENTRY: THE  REAL  PART  OF  THE  UPPER  TRIANGLE OF  THE
                       HERMITIAN   MATRIX  MUST  BE  GIVEN IN THE  UPPER
                       TRIANGULAR PART OF A (THE ELEMENTS A[I,J], I<=J);
                       THE  IMAGINARY  PART OF THE STRICT LOWER TRIANGLE
                       OF  THE   HERMITIAN  MATRIX  MUST BE GIVEN IN THE
                       STRICT LOWER PART OF A (THE ELEMENTS A[I,J],I>J);
                THE ELEMENTS AF A ARE ALTERED;
    N:          <ARITHMETIC EXPRESSION>;
                THE ORDER OF THE GIVEN MATRIX;
    VAL:        <ARRAY IDENTIFIER>;
                "ARRAY" VAL[1:N];
                EXIT:
                THE CALCULATED EIGENVALUES;
    VR,VI:      <ARRAY IDENTIFIER>;
                "ARRAY" VR,VI[1:N,1:N];
                EXIT:
                THE CALCULATED EIGENVECTORS;
                THE  COMPLEX  EIGENVECTOR WITH REAL PART VR[1:N,I]   AND
                IMAGINARY  PART VI[1:N,I] CORRESPONDS TO THE  EIGENVALUE
                VAL[I], I=1,...,N;
    EM:         <ARRAY IDENTIFIER>;
                "ARRAY" EM[0:5];
                ENTRY:
                EM[0]: THE MACHINE PRECISION;
                EM[2]: THE RELATIVE TOLERANCE FOR THE QR ITERATION;
                       (E.G. THE MACHINE PRECISION);
                EM[4]: THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
                       (E.G. 10 * N);
                EXIT:
                EM[1]: AN ESTIMATE OF A NORM OF THE ORIGINAL MATRIX;
                EM[3]: THE  MAXIMUM   ABSOLUTE VALUE OF THE   CODIAGONAL
                       ELEMENTS NEGLECTED;
                EM[5]: THE NUMBER OF ITERATIONS PERFORMED;
                       EM[5]:=EM[4]+1 IN THE CASE ORIHRM^=0;
```

QRIHRM:=0, PROVIDED THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE, QRIHRM:= THE NUMBER OF EIGENVALUES, K, NOT CALCULATED AND ONLY THE LAST N-K ELEMENTS OF VAL ARE APPROXIMATE EIGENVALUES AND THE COLUMNS OF THE ARRAYS VR,VI[1:N,N-K:N] ARE APPROXIMATE EIGENVECTORS OF THE ORIGINAL HERMITEAN MATRIX .

PROCEDURES USED:

HSHHRMTRI = CP34363,
QRISYMTRI = CP34161,
BAKHRMTRI = CP34365.

REQUIRED CENTRAL MEMORY:
TWO AUXILIARY ARRAYS OF ORDER N - 1 AND TWO OF ORDER N ARE DECLARED

RUNNING TIME: PROPORTIONAL TO N CUBED.

LANGUAGE: ALGOL 60.

THE FOLLOWING HOLDS FOR THE FOUR PROCEDURES OF THIS SECTION:

METHOD AND PERFORMANCE:

FOR THE TRANSFORMATION OF THE GIVEN HERMITIAN MATRIX INTO A REAL SYMMETRIC TRIDIAGONAL MATRIX, AND FOR THE CORRESPONDING BACK TRANSFORMATION, PROCEDURES OF SECTION 3.2.1.2.2.1. ARE USED. FOR THE CALCULATION OF THE EIGENVALUES AND EIGENVECTORS OF THE RESULTING SYMMETRIC TRIDIAGONAL MATRIX, PROCEDURES OF SECTION 3.3.1.1.1. ARE USED.

EXAMPLE OF USE:

QRIHRM CALCULATES THE EIGENVALUES AND EIGENVECTORS OF THE FOLLOWING MATRIX:
(SEE GREGORY AND KARNEY, CHAPTER 6, EXAMPLE 6.6)

```
     3    1    0   +2I
     1    3   -2I   0
     0   +2I   1    1
    -2I   0    1    1
```

THE EIGENVECTORS ARE NORMALIZED BY THE PROCEDURE SCLCOM (SEE SECTION 1.2.11.).
ONLY THE EIGENVECTORS CORRESPONDING TO VAL[2] AND VAL[3] ARE PRINTED BY THE FOLLOWING PROGRAM:

```
    "BEGIN"
    "COMMENT" GREGORY AND KARNEY,CHAPTER 6, EXAMPLE 6.6;
    "INTEGER" "PROCEDURE" QRIHRM(A,N,VAL,VR,VI,EM);"CODE" 34371;
    "PROCEDURE" INIMAT(LR,UR,LC,UC,A,X);"CODE" 31011;
    "PROCEDURE" SCLCOM(AR,AI,N,N1,N2);"CODE" 34360;
    "REAL" "ARRAY" A,VR,VI[1:4,1:4],VAL[1:4],EM[0:5];"INTEGER" I;
    INIMAT(1,4,1,4,A,0);
    A[1,1]:=A[2,2]:=3;
    A[3,2]:=2;A[4,1]:=-2;
    A[1,2]:=A[3,3]:=A[3,4]:=A[4,4]:=1;
    EM[0]:=EM[2]:=5"-14;EM[4]:=20;
    OUTPUT(61,"(""(""QRIHRM: "")",D/")",QRIHRM(A,4,VAL,VR,VI,EM));
    SCLCOM(VR,VI,4,2,3);
    OUTPUT(61,"(""(""EIGENVALUES: "")"")");
    "FOR" I:=1,2,3,4 "DO" OUTPUT(61,"("/,"("VAL["")",D,"("]: "")",
                                        +D.3DBB")",I,VAL[I]);
    OUTPUT(61,"("/,"("EIGENVECTORS CORRESPONDING TO")",/,
                  "("" VAL[2] , VAL[3] "")",/")");
    "FOR" I:=1,2,3,4 "DO"
    OUTPUT(61,"("+D,+D,"("*I   ,   ")",+D,+D,"("*I")",/")",
              VR[I,2],VI[I,2],VR[I,3],VI[I,3]);
    "FOR" I:=1,3,5 "DO"
     OUTPUT(61,"("/,"("EM["")",D,"("]:   ")",+D.DDD"+DD")",I,EM[I])
    "END"

  OUTPUT:
    QRIHRM: 0
    EIGENVALUES:
    VAL[1]: +4.828
    VAL[2]: +4.000
    VAL[3]: -0.000
    VAL[4]: -0.828
    EIGENVECTORS CORRESPONDING TO
     VAL[2] , VAL[3]
    +1+0*I   ,   +0-1*I
    -1+0*I   ,   +0+1*I
    +0-1*I   ,   +1+0*I
    +0-1*I   ,   +1+0*I

    EM[1]:   +6.000"+00
    EM[3]:   +3.804"-22
    EM[5]:   +6.000"+00

SOURCE TEXT(S) :

"CODE" 34368;
    "PROCEDURE" EIGVALHRM(A, N, NUMVAL, VAL, EM); "VALUE" N, NUMVAL;
    "INTEGER" N, NUMVAL; "ARRAY" A, VAL, EM;
    "BEGIN" "ARRAY" D[1:N], BB[1:N - 1];
        "PROCEDURE" HSHHRMTRIVAL(A,N,D,BB,EM);"CODE" 34364;
        "PROCEDURE" VALSYMTRI(D,BB,N,N1,N2,VAL,EM);"CODE" 34151;
        HSHHRMTRIVAL(A, N, D, BB, EM);
        VALSYMTRI(D, BB, N, 1, NUMVAL, VAL, EM)
    "END" EIGVALHRM;
        "EOP"
```

```
"CODE" 34369;
    "PROCEDURE" EIGHRM(A, N, NUMVAL, VAL, VECR, VECI, EM);
    "VALUE" N, NUMVAL; "INTEGER" N, NUMVAL;
    "ARRAY" A, VAL, VECR, VECI, EM;
    "BEGIN" "ARRAY" BB, TR, TI[1:N - 1], D, B[1:N];
        "PROCEDURE" HSHHRMTRI(A,N,D,B,BB,EM,TR,TI);"CODE" 34363;
        "PROCEDURE" VALSYMTRI(D,BB,N,N1,N2,VAL,EM);"CODE" 34151;
        "PROCEDURE" VECSYMTRI(D,B,N,N1,N2,VAL,VEC,EM);"CODE" 34152;
        "PROCEDURE" BAKHRMTRI(A,N,N1,N2,VECR,VECI,TR,TI);"CODE" 34365;
        HSHHRMTRI(A, N, D, B, BB, EM, TR, TI);
        VALSYMTRI(D, BB, N, 1, NUMVAL, VAL, EM); B[N]:= 0;
        VECSYMTRI(D, B, N, 1, NUMVAL, VAL, VECR, EM);
        BAKHRMTRI(A, N, 1, NUMVAL, VECR, VECI, TR, TI)
    "END" EIGHRM;
        "EOP"

"CODE" 34370;
    "INTEGER" "PROCEDURE" QRIVALHRM(A, N, VAL, EM); "VALUE" N;
    "INTEGER" N; "ARRAY" A, VAL, EM;
    "BEGIN" "ARRAY" B,BB[1:N];
        "INTEGER" I;
        "PROCEDURE" HSHHRMTRIVAL(A,N,D,BB,EM);"CODE" 34364;
        "INTEGER" "PROCEDURE" QRIVALSYMTRI(D, BB, N, EM);"CODE" 34160;
        HSHHRMTRIVAL(A, N, VAL, BB, EM); B[N]:=BB[N]:= 0;
        "FOR" I:=1 "STEP" 1 "UNTIL" N-1 "DO" B[I]:=SQRT(BB[I]);
        QRIVALHRM:=QRIVALSYMTRI(VAL, BB, N, EM)
    "END" QRIVALHRM;
        "EOP"

"CODE" 34371;
    "INTEGER" "PROCEDURE" QRIHRM(A, N, VAL, VR, VI, EM); "VALUE" N;
    "INTEGER" N; "ARRAY" A, VAL, VR, VI, EM;
    "BEGIN" "INTEGER" I, J;
        "ARRAY" B, BB[1:N], TR, TI[1:N - 1];
        "PROCEDURE" HSHHRMTRI(A,N,D,B,BB,EM,TR,TI);"CODE" 34363;
        "INTEGER" "PROCEDURE" QRISYMTRI(A,N,D,B,BB,EM);"CODE" 34161;
        "PROCEDURE" BAKHRMTRI(A,N,N1,N2,VECR,VECI,TR,TI);"CODE" 34365;
        HSHHRMTRI(A, N, VAL, B, BB, EM, TR, TI);
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" VR[I,I]:= 1;
            "FOR" J:= I + 1 "STEP" 1 "UNTIL" N "DO" VR[I,J]:= VR[J,I]:=
            0
        "END";
        B[N]:= BB[N]:= 0;
        I:= QRIHRM:= QRISYMTRI(VR, N, VAL, B, BB, EM);
        BAKHRMTRI(A, N, I+1, N, VR, VI, TR, TI);
    "END" QRIHRM;
        "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

CONTRIBUTORS : H.FIOLET, C.G. VAN DER LAAN.

INSTITUTE : MATHEMATICAL CENTRE.

RECEIVED: 731016.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS THE PROCEDURES VALQRICOM AND QRICOM.
VALQRICOM CALCULATES THE EIGENVALUES OF A COMPLEX UPPER-HESSENBERG MATRIX WITH A REAL SUBDIAGONAL.
QRICOM CALCULATES THE EIGENVECTORS AS WELL.

KEYWORDS :

EIGENVALUES,
EIGENVECTORS,
COMPLEX UPPER-HESSENBERG MATRIX,
QR-ITERATION.

SUBSECTION: VALQRICOM.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"INTEGER" "PROCEDURE" VALQRICOM(A1, A2, B, N, EM, VAL1, VAL2);
"VALUE" N; "INTEGER" N; "ARRAY" A1, A2, B, EM, VAL1, VAL2;

THE MEANING OF THE FORMAL PARAMETERS IS:

A1,A2: <ARRAY IDENTIFIER>;
"ARRAY" A1,A2[1:N,1:N];
ENTRY:
THE REAL PART AND THE IMAGINARY PART OF THE UPPER TRIANGLE OF THE UPPER-HESSENBERG MATRIX MUST BE GIVEN IN THE CORRESPONDING PARTS OF THE ARRAYS A1 AND A2;
THE ELEMENTS IN THE UPPER TRIANGLE OF THE ARRAYS A1 AND A2 ARE ALTERED;

B: <ARRAY IDENTIFIER>;
"ARRAY" B[1:N-1];
ENTRY:
THE REAL SUBDIAGONAL OF THE UPPER-HESSENBERG MATRIX;
THE ELEMENTS OF THE ARRAY B ARE ALTERED;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY:
EM[0]: THE MACHINE PRECISION;
EM[1]: AN ESTIMATE OF THE NORM OF THE UPPER-HESSENBERG MATRIX (E.G. THE SUM OF THE INFINITY NORMS OF THE REAL AND IMAGINARY PARTS OF THE MATRIX);
EM[2]: THE RELATIVE TOLERANCE FOR THE QR-ITERATION; (E.G. THE MACHINE PRECISION);
EM[4]: THE MAXIMUM ALLOWED NUMBER OF ITERATIONS; (E.G. 10 * N);
EXIT:
EM[3]: THE MAXIMUM ABSOLUTE VALUE OF THE SUBDIAGONAL ELEMENTS NEGLECTED;
EM[5]: THE NUMBER OF ITERATIONS PERFORMED; EM[5]:=EM[4]+1 IN THE CASE VALQRICOM^=0;

VAL1,VAL2: <ARRAY IDENTIFIER>;
"ARRAY" VAL1,VAL2[1:N];
EXIT:
THE REAL PART AND THE IMAGINARY PART OF THE CALCULATED EIGENVALUES ARE DELIVERED IN THE ARRAYS VAL1 AND VAL2, RESPECTIVELY;

VALQRICOM:=0, PROVIDED THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE, VALQRICOM:= THE NUMBER, K, OF EIGENVALUES NOT CALCULATED AND ONLY THE LAST N-K ELEMENTS OF THE ARRAYS VAL1 AND VAL2 ARE APPROXIMATE EIGENVALUES OF THE UPPER-HESSENBERG MATRIX.

PROCEDURES USED:

```
    COMKWD     = CP34345,
    ROTCOMROW  = CP34358,
    ROTCOMCOL  = CP34357,
    COMCOLCST  = CP34352.
```

RUNNING TIME: PROPORTIONAL TO N**2 * NUMBER OF ITERATIONS.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE QRICOM (THIS SECTION).

EXAMPLE OF USE:

AS A FORMAL TEST OF THE PROCEDURE VALQRICOM THE ZEROS OF THE POLYNOMIAL X**4 + (4+2*I)* X**3 + (5+6*I)* X**2 + (2+6*I)* X + 2*I ARE OBTAINED BY MEANS OF THE CALCULATION OF THE EIGENVALUES OF THE FOLLOWING COMPANION MATRIX:
(SEE WILKINSON AND REINSCH, 1971, CONTRIBUTION II/15)

```
-4-2*I  -5-6*I  -2-6*I    -2*I
  1       0       0       0
  0       1       0       0
  0       0       1       0
```

```
    "BEGIN"
    "REAL" "ARRAY" A1,A2[1:4,1:4],B[1:3],EM[0:5],VAL1,VAL2[1:4];
    "INTEGER" I;
    "PROCEDURE" INIMAT(LR,UR,LC,UC,A,X);"CODE" 31011;
    "INTEGER" "PROCEDURE" VALQRICOM(A1,A2,B,N,EM,VAL1,VAL2);
    "CODE" 34372;
    INIMAT(1,4,1,4,A1,0);INIMAT(1,4,1,4,A2,0);
    A1[1,1]:=-4;A1[1,2]:=-5;A1[1,3]:=A2[1,1]:=A2[1,4]:=-2;
    A2[1,2]:=A2[1,3]:=-6;
    B[1]:=B[2]:=B[3]:=1;
    EM[0]:=5"-14;EM[1]:=27;EM[2]:="-12;EM[4]:=15;
    OUTPUT(61,"(""("VALQRICOM: ")",D/")",
           VALQRICOM(A1,A2,B,4,EM,VAL1,VAL2));
    OUTPUT(61,"(""("EIGENVALUES:")",/,"("REAL PART")",14B,
                  "("IMAGINARY PART")",/")");
    "FOR" I:=1,2,3,4 "DO" OUTPUT(61,"("N,N/")",VAL1[I],VAL2[I]);
    OUTPUT(61,"(""("EM[3]: ")",D.D"+DD/,"("EM[5]: ")",3D")",
                    EM[3],EM[5])
    "END"
```

```
  OUTPUT:
    VALQRICOM: 0
    EIGENVALUES:
    REAL PART                IMAGINARY PART
    -1.0000001920467"+000  -1.0000000831019"+000
    -9.9999980795324"-001  -9.9999991689805"-001
    -1.0000000047492"+000  +1.6824958523393"-007
    -9.9999999525076"-001  -1.6824956397352"-007
    EM[3]: 3.2"-14
    EM[5]: 010
```

SUBSECTION: QRICOM.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "INTEGER" "PROCEDURE" QRICOM(A1,A2,B,N,EM,VAL1,VAL2,VEC1,VEC2);
    "VALUE" N; "INTEGER" N;
    "ARRAY" A1, A2, B, EM, VAL1, VAL2, VEC1, VEC2;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    A1,A2:      <ARRAY IDENTIFIER>;
                "ARRAY" A1,A2[1:N,1:N];
                ENTRY:
                THE REAL PART AND THE IMAGINARY PART OF THE UPPER
                TRIANGLE OF THE UPPER-HESSENBERG MATRIX MUST BE GIVEN IN
                THE CORRESPONDING PARTS OF THE ARRAYS A1 AND A2;
                THE ELEMENTS IN THE UPPER TRIANGLE OF THE ARRAYS A1 AND
                A2 ARE ALTERED;
    B:          <ARRAY IDENTIFIER>;
                "ARRAY" B[1:N-1];
                ENTRY:
                THE REAL SUBDIAGONAL OF THE UPPER-HESSENBERG MATRIX;
                THE ELEMENTS OF THE ARRAY B ARE ALTERED;
    N:          <ARITHMETIC EXPRESSION>;
                THE ORDER OF THE GIVEN MATRIX;
    EM:         <ARRAY IDENTIFIER>;
                "ARRAY" EM[0:5];
                ENTRY:
                EM[0]: THE MACHINE PRECISION;
                EM[1]: AN ESTIMATE OF THE NORM OF THE UPPER-HESSENBERG
                       MATRIX (E.G. THE SUM OF THE INFINITY NORMS OF THE
                       REAL AND IMAGINARY PARTS OF THE MATRIX);
                EM[2]: THE RELATIVE TOLERANCE FOR THE QR-ITERATION;
                       (E.G. THE MACHINE PRECISION);
                EM[4]: THE MAXIMUM ALLOWED NUMBER OF ITERATIONS;
                       (E.G. 10 * N);
                EXIT:
                EM[3]: THE MAXIMUM ABSOLUTE VALUE OF THE SUBDIAGONAL
                       ELEMENTS NEGLECTED;
                EM[5] THE NUMBER OF ITERATIONS PERFORMED;
                       EM[5]:=EM[4]+1 IN THE CASE QRICOM^=0;
    VAL1,VAL2:  <ARRAY IDENTIFIER>;
                "ARRAY" VAL1,VAL2[1:N];
                EXIT:
                THE REAL PART AND THE IMAGINARY PART OF THE CALCULATED
                EIGENVALUES ARE DELIVERED IN THE ARRAYS VAL1 AND VAL2,
                RESPECTIVELY;
    VEC1,VEC2:  <ARRAY IDENTIFIER>;
                "ARRAY" VEC1,VEC2[1:N,1:N];
                EXIT:
                THE EIGENVECTORS OF THE UPPER-HESSENBERG MATRIX;
                THE EIGENVECTOR WITH REAL PART VEC1[1:N,J] AND IMAGINARY
                PART VEC2[1:N,J] CORRESPONDS TO THE EIGENVALUE
                VAL1[J] + VAL2[J] * I, J=1,...,N;
```

QRICOM:=0, PROVIDED THE PROCESS IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE, QRICOM:= THE NUMBER, K, OF EIGENVALUES NOT CALCULATED AND ONLY THE LAST N-K ELEMENTS OF THE ARRAYS VAL1 AND VAL2 ARE APPROXIMATE EIGENVALUES OF THE UPPER-HESSENBERG MATRIX, AND NO USEFUL EIGENVECTORS ARE DELIVERED.

PROCEDURES USED:

COMKWD = CP34345,
ROTCOMROW = CP34358,
ROTCOMCOL = CP34357,
COMCOLCST = CP34352,
COMROWCST = CP34353,
MATVEC = CP34011,
COMMATVEC = CP34354,
COMDIV = CP34342.

REQUIRED CENTRAL MEMORY: TWO AUXILIARY ARRAYS OF ORDER N ARE DECLARED.

RUNNING TIME: PROPORTIONAL TO N**2 * NUMBER OF ITERATIONS.

LANGUAGE: ALGOL 60.

THE FOLLOWING HOLDS FOR BOTH PROCEDURES:

METHOD AND PERFORMANCE:

THE UPPER-HESSENBERG MATRIX IS TRANSFORMED BY MEANS OF FRANCIS' QR ITERATION (FRANCIS, 1961, AND WILKINSON, 1965) INTO A COMPLEX UPPER TRIANGULAR MATRIX. THE EIGENVALUES ARE THE DIAGONAL ELEMENTS OF THE LATTER MATRIX. TO CALCULATE THE EIGENVECTORS WE FIRST SOLVE THE RESULTING TRIANGULAR SYSTEM OF LINEAR EQUATIONS AND SUBSEQUENTLY PERFORM THE CORRESPONDING BACKTRANSFORMATION.

EXAMPLE OF USE:

```
    QRICOM CALCULATES THE EIGENVALUES AND EIGENVECTORS OF THE
    FOLLOWING MATRIX:
    (SEE WILKINSON AND REINSCH, 1971, CONTRIBUTION II/15)
    -4-2*I  -5-6*I  -2-6*I    -2*I
      1       0       0       0
      0       1       0       0
      0       0       1       0

    THE EIGENVECTORS ARE NORMALIZED BY THE PROCEDURE SCLCOM.
    (SEE SECTION 1.2.11.).
    ONLY THE EIGENVECTOR CORRESPONDING TO VAL1[1] + VAL2[1] * I IS
    PRINTED BY THE FOLLOWING PROGRAM.

    "BEGIN"
    "REAL" "ARRAY" A1,A2,VEC1,VEC2[1:4,1:4],B[1:3],
                   EM[0:5],VAL1,VAL2[1:4];
    "INTEGER" I;
    "PROCEDURE" SCLCOM(AR,AI,N,N1,N2);"CODE" 34360;
    "INTEGER" "PROCEDURE" QRICOM(A1,A2,B,N,EM,VAL1,VAL2,VEC1,VEC2);
    "CODE" 34373;
    "PROCEDURE" INIMAT(LR,UR,LC,UC,A,X);"CODE" 31011;
    INIMAT(1,4,1,4,A1,0);INIMAT(1,4,1,4,A2,0);
    A1[1,1]:=-4;A1[1,2]:=-5;A1[1,3]:=A2[1,1]:=A2[1,4]:=-2;
    A2[1,2]:=A2[1,3]:=-6;
    B[1]:=B[2]:=B[3]:=1;
    EM[0]:=5"-14;EM[1]:=27;EM[2]:="-12;EM[4]:=15;
    OUTPUT(61,"(""("QRICOM: ")",D/")",
        QRICOM(A1,A2,B,4,EM,VAL1,VAL2,VEC1,VEC2));
    OUTPUT(61,"(""("EIGENVALUES:")",/,"("REAL PART")",14B,
                  "("IMAGINARY PART")",/")");
    "FOR" I:=1,2,3,4 "DO" OUTPUT(61,"("N,N/")",VAL1[I],VAL2[I]);
    SCLCOM(VEC1,VEC2,4,1,4);
    OUTPUT(61,"(""("FIRST EIGENVECTOR:")",/,"("REAL PART")",14B,
                  "("IMAGINARY PART")",/")");
    "FOR" I:=1,2,3,4 "DO" OUTPUT(61,"("N,N/")",VEC1[I,1],VEC2[I,1]);
    OUTPUT(61,"(""("EM[3]: ")",D.D"+DD/,"("EM[5]: ")",3D")",
                   EM[3],EM[5])
    "END"
```

```
    OUTPUT:

    ORICOM: 0
    EIGENVALUES:
    REAL PART                IMAGINARY PART
    -9.9999980795324"-001    -9.9999991689805"-001
    -1.0000000192046"+000    -1.0000000831019"+000
    -1.0000000047492"+000    +1.6824958523393"-007
    -9.9999999525076"-001    -1.6824956397352"-007
    FIRST EIGENVECTOR:
    REAL PART                IMAGINARY PART
    +1.0000000000000"+000    -1.7763568394003"-015
    -5.0000004155098"-001    +5.0000009602339"-001
    -5.4472417687634"-008    -5.0000013757436"-001
    +2.5000014403510"-001    +2.5000006232645"-001
    EM[3]: 3.2"-14
    EM[5]: 010
```

REFERENCES:

DEKKER, T.J. (1968),
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 1,
MATH. CENTRE TRACTS 22, MATHEMATISCH CENTRUM;

DEKKER, T.J. AND W.HOFFMANN (1968),
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2,
MATH. CENTRE TRACTS 23, MATHEMATISCH CENTRUM;

FRANCIS, J.G.F. (1961),
THE QR-TRANSFORMATION, PART 1 AND 2,
COMP.J. 4, P.265-271 AND P.332-345;

RUHE, A. (1966),
EIGENVALUES OF A COMPLEX MATRIX BY THE QR METHOD.
BIT, 6, P.350-358;

WILKINSON, J.H. (1965),
THE ALGEBRAIC EIGENVALUE PROBLEM,
CLARENDON PRESS, OXFORD;

SOURCE TEXT(S) :

```
"CODE" 34372;
    "INTEGER" "PROCEDURE" VALQRICOM(A1, A2, B, N, EM, VAL1, VAL2);
    "VALUE" N; "INTEGER" N; "ARRAY" A1, A2, B, EM, VAL1, VAL2;
    "BEGIN" "INTEGER" M, NM1, I, I1, Q, Q1, MAX, COUNT;
        "REAL" R, Z1, Z2, DD1, DD2, CC, G1, G2, K1, K2, HC, A1NN,
        A2NN, AIJ1, AIJ2, AI1I, KAPPA, NUI, MUI1, MUI2,
        MUIM11, MUIM12, NUIM1, TOL;
        "PROCEDURE" COMCOLCST(L,U,J,AR,AI,XR,XI);"CODE" 34352;
        "PROCEDURE" ROTCOMCOL(L,U,I,J,AR,AI,CR,CI,S);"CODE" 34357;
        "PROCEDURE" ROTCOMROW(L,U,I,J,AR,AI,CR,CI,S);"CODE" 34358;
        "PROCEDURE" COMKWD(PR,PI,QR,QI,GR,GI,KR,KI);"CODE" 34345;
        TOL:= EM[1] * EM[2]; MAX:= EM[4]; COUNT:= 0; R:= 0;
        M:= N; "IF" N > 1 "THEN" HC:= B[N - 1];
    IN: NM1:= N - 1;
        "FOR" I:= N, I - 1 "WHILE" ("IF" I >= 1 "THEN" ABS(B[I]) > TOL
        "ELSE" "FALSE") "DO" Q:= I; "IF" Q > 1 "THEN"
        "BEGIN" "IF" ABS(B[Q - 1]) > R "THEN" R:= ABS(B[Q - 1]) "END";
        "IF" Q = N "THEN"
        "BEGIN" VAL1[N]:= A1[N,N]; VAL2[N]:= A2[N,N]; N:= NM1;
            "IF" N > 1 "THEN" HC:= B[N - 1];
        "END"
        "ELSE"
        "BEGIN" DD1:= A1[N,N]; DD2:= A2[N,N]; CC:= B[NM1];
            COMKWD((A1[NM1,NM1] - DD1) / 2, (A2[NM1,NM1] - DD2)
            / 2, CC * A1[NM1,N], CC * A2[NM1,N], G1, G2, K1,
            K2); "IF" Q = NM1 "THEN"
            "BEGIN" VAL1[NM1]:= G1 + DD1; VAL2[NM1]:= G2 + DD2;
                VAL1[N]:= K1 + DD1; VAL2[N]:= K2 + DD2;
                N:= N - 2; "IF" N > 1 "THEN" HC:= B[N - 1];
            "END"
            "ELSE"
            "BEGIN" COUNT:= COUNT + 1;
                "IF" COUNT > MAX "THEN" "GOTO" OUT; Z1:= K1 + DD1;
                Z2:= K2 + DD2;
                "IF" ABS(CC) > ABS(HC) "THEN" Z1:= Z1 + ABS(CC);
                HC:= CC / 2; I:= Q1:= Q + 1;
                AIJ1:= A1[Q,Q] - Z1; AIJ2:= A2[Q,Q] - Z2;
                AI1I:= B[Q];
                KAPPA:= SQRT(AIJ1 ** 2 + AIJ2 ** 2 + AI1I ** 2);
                MUI1:= AIJ1 / KAPPA; MUI2:= AIJ2 / KAPPA;
                NUI:= AI1I / KAPPA; A1[Q,Q]:= KAPPA;
                A2[Q,Q]:= 0; A1[Q1,Q1]:= A1[Q1,Q1] - Z1;
                A2[Q1,Q1]:= A2[Q1,Q1] - Z2;
                ROTCOMROW(Q1, N, Q, Q1, A1, A2, MUI1, MUI2,
                NUI);
                ROTCOMCOL(Q, Q, Q, Q1, A1, A2, MUI1, - MUI2, -
                NUI); A1[Q,Q]:= A1[Q,Q] + Z1;
                A2[Q,Q]:= A2[Q,Q] + Z2;"COMMENT"
```

```
                "FOR" I1:= Q1 + 1 "STEP" 1 "UNTIL" N "DO"
                "BEGIN" AIJ1:= A1[I,I]; AIJ2:= A2[I,I];
                    AI1I:= B[I];
                    KAPPA:= SQRT(AIJ1 ** 2 + AIJ2 ** 2 + AI1I **
                    2); MUIM11:= MUI1; MUIM12:= MUI2;
                    NUIM1:= NUI; MUI1:= AIJ1 / KAPPA;
                    MUI2:= AIJ2 / KAPPA; NUI:= AI1I / KAPPA;
                    A1[I1,I1]:= A1[I1,I1] - Z1;
                    A2[I1,I1]:= A2[I1,I1] - Z2;
                    ROTCOMROW(I1, N, I, I1, A1, A2, MUI1,
                    MUI2, NUI); A1[I,I]:= MUIM11 * KAPPA;
                    A2[I,I]:= - MUIM12 * KAPPA;
                    B[I - 1]:= NUIM1 * KAPPA;

                    ROTCOMCOL(Q, I, I, I1, A1, A2, MUI1, -
                    MUI2, - NUI); A1[I,I1]:= A1[I,I] + Z1;
                    A2[I,I]:= A2[I,I] + Z2; I:= I1;
                "END";
                AIJ1:= A1[N,N]; AIJ2:= A2[N,N];
                KAPPA:= SQRT(AIJ1 ** 2 + AIJ2 ** 2);
                "IF" ("IF" KAPPA < TOL "THEN" "TRUE" "ELSE" AIJ2 ** 2
                <= EM[0] * AIJ1 ** 2) "THEN"
                "BEGIN" B[NM1]:= NUI * AIJ1;
                    A1[N,N]:= AIJ1 * MUI1 + Z1;
                    A2[N,N]:= - AIJ1 * MUI2 + Z2
                "END"
                "ELSE"
                "BEGIN" B[NM1]:= NUI * KAPPA; A1NN:= MUI1 * KAPPA;
                    A2NN:= - MUI2 * KAPPA; MUI1:= AIJ1 / KAPPA;
                    MUI2:= AIJ2 / KAPPA;
                    COMCOLCST(Q, NM1, N, A1, A2, MUI1, MUI2);
                    A1[N,N]:= MUI1 * A1NN - MUI2 * A2NN + Z1;
                    A2[N,N]:= MUI1 * A2NN + MUI2 * A1NN + Z2;
                "END";
            "END"
        "END";
        "IF" N > 0 "THEN" "GOTO" IN;
    OUT: EM[3]:= R; EM[5]:= COUNT; VALQRICOM:= N;
    "END" VALQRICOM;
        "EOP"

"CODE" 34373;
    "INTEGER" "PROCEDURE" QRICOM(A1, A2, B, N, EM, VAL1, VAL2, VEC1,
    VEC2); "VALUE" N; "INTEGER" N;
    "ARRAY" A1, A2, B, EM, VAL1, VAL2, VEC1, VEC2;
    "BEGIN" "INTEGER" M, NM1, I, I1, J, Q, Q1, MAX, COUNT;
        "REAL" R, Z1, Z2, DD1, DD2, CC, P1, P2, T1, T2, DELTA1,
        DELTA2, MV1, MV2, H, H1, H2, G1, G2, K1, K2, HC,
        AIJ12, AIJ22, A1NN, A2NN, AIJ1, AIJ2, AI1I, KAPPA,
        NUI, MUI1, MUI2, MUIM11, MUIM12, NUIM1, TOL, MACHTOL;
        "ARRAY" TF1, TF2[1:N];"COMMENT"
```

```
        "PROCEDURE" COMKWD(PR,PI,QR,QI,GR,GI,KR,KI);"CODE" 34345;
        "PROCEDURE" ROTCOMROW(L,U,I,J,AR,AI,CR,CI,S);"CODE" 34358;
        "PROCEDURE" ROTCOMCOL(L,U,I,J,AR,AI,CR,CI,S);"CODE" 34357;
        "PROCEDURE" COMCOLCST(L,U,J,AR,AI,XR,XI);"CODE" 34352;
        "PROCEDURE" COMROWCST(L,U,I,AR,AI,XR,XI);"CODE" 34353;
        "REAL" "PROCEDURE" MATVEC(L,U,I,A,B);"CODE" 34011;
        "PROCEDURE" COMMATVEC(L,U,I,AR,AI,BR,BI,RR,RI);"CODE" 34354;
        "PROCEDURE" COMDIV(XR,XI,YR,YI,ZR,ZI);"CODE" 34342;
        TOL:= EM[1] * EM[2]; MACHTOL:= EM[0] * EM[1];
        MAX:= EM[4]; COUNT:= 0; R:= 0; M:= N;
        "IF" N > 1 "THEN" HC:= B[N - 1];
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" VEC1[I,I]:= 1; VEC2[I,I]:= 0;
            "FOR" J:= I + 1 "STEP" 1 "UNTIL" N "DO" VEC1[I,J]:=
            VEC1[J,I]:= VEC2[I,J]:= VEC2[J,I]:= 0
        "END";
    IN: NM1:= N - 1;
        "FOR" I:= N, I - 1 "WHILE" ("IF" I >= 1 "THEN" ABS(B[I]) > TOL
        "ELSE" "FALSE") "DO" Q:= I; "IF" Q > 1 "THEN"
        "BEGIN" "IF" ABS(B[Q - 1]) > R "THEN" R:= ABS(B[Q - 1]) "END";
        "IF" Q = N "THEN"
        "BEGIN" VAL1[N]:= A1[N,N]; VAL2[N]:= A2[N,N]; N:= NM1;
            "IF" N > 1 "THEN" HC:= B[N - 1];
        "END"
        "ELSE"
        "BEGIN" DD1:= A1[N,N]; DD2:= A2[N,N]; CC:= B[NM1];
            P1:= (A1[NM1,NM1] - DD1) * .5;
            P2:= (A2[NM1,NM1] - DD2) * .5;
            COMKWD(P1, P2, CC * A1[NM1,N], CC * A2[NM1,N], G1,
            G2, K1, K2); "IF" Q = NM1 "THEN"
            "BEGIN" A1[N,N]:= VAL1[N]:= G1 + DD1;
                A2[N,N]:= VAL2[N]:= G2 + DD2;
                A1[Q,Q]:= VAL1[Q]:= K1 + DD1;
                A2[Q,Q]:= VAL2[Q]:= K2 + DD2;
                KAPPA:= SQRT(K1 ** 2 + K2 ** 2 + CC ** 2);
                NUI:= CC / KAPPA; MUI1:= K1 / KAPPA;
                MUI2:= K2 / KAPPA; AIJ1:= A1[Q,N];
                AIJ2:= A2[Q,N]; H1:= MUI1 ** 2 - MUI2 ** 2;
                H2:= 2 * MUI1 * MUI2; H:= - NUI * 2;
                A1[Q,N]:= H * (P1 * MUI1 + P2 * MUI2) - NUI *
                NUI * CC + AIJ1 * H1 + AIJ2 * H2;
                A2[Q,N]:= H * (P2 * MUI1 - P1 * MUI2) + AIJ2 *
                H1 - AIJ1 * H2;
                ROTCOMROW(Q + 2, M, Q, N, A1, A2, MUI1, MUI2,
                NUI);
                ROTCOMCOL(1, Q - 1, Q, N, A1, A2, MUI1, -
                MUI2, - NUI);
                ROTCOMCOL(1, M, Q, N, VEC1, VEC2, MUI1, -
                MUI2, - NUI); N:= N - 2;
                "IF" N > 1 "THEN" HC:= B[N - 1]; B[Q]:= 0
            "END"
```

```
"ELSE"
"BEGIN" COUNT:= COUNT + 1;
    "IF" COUNT > MAX "THEN" "GOTO" OUT; Z1:= K1 + DD1;
    Z2:= K2 + DD2;
    "IF" ABS(CC) > ABS(HC) "THEN" Z1:= Z1 + ABS(CC);
    HC:= CC / 2; Q1:= Q + 1; AIJ1:= A1[Q,Q] - Z1;
    AIJ2:= A2[Q,Q] - Z2; AI1I:= B[Q];
    KAPPA:= SQRT(AIJ1 ** 2 + AIJ2 ** 2 + AI1I ** 2);
    MUI1:= AIJ1 / KAPPA; MUI2:= AIJ2 / KAPPA;
    NUI:= AI1I / KAPPA; A1[Q,Q]:= KAPPA;
    A2[Q,Q]:= 0; A1[Q1,Q1]:= A1[Q1,Q1] - Z1;
    A2[Q1,Q1]:= A2[Q1,Q1] - Z2;
    ROTCOMROW(Q1, M, Q, Q1, A1, A2, MUI1, MUI2,
    NUI);
    ROTCOMCOL(1, Q, Q, Q1, A1, A2, MUI1, - MUI2, -
    NUI); A1[Q,Q]:= A1[Q,Q] + Z1;
    A2[Q,Q]:= A2[Q,Q] + Z2;
    ROTCOMCOL(1, M, Q, Q1, VEC1, VEC2, MUI1, -
    MUI2, - NUI);
    "FOR" I:= Q1 "STEP" 1 "UNTIL" NM1 "DO"
    "BEGIN" I1:= I + 1; AIJ1:= A1[I,I]; AIJ2:= A2[I,I];
        AI1I:= B[I];
        KAPPA:= SQRT(AIJ1 ** 2 + AIJ2 ** 2 + AI1I **
        2); MUIM11:= MUI1; MUIM12:= MUI2;
        NUIM1:= NUI; MUI1:= AIJ1 / KAPPA;
        MUI2:= AIJ2 / KAPPA; NUI:= AI1I / KAPPA;
        A1[I1,I1]:= A1[I1,I1] - Z1;
        A2[I1,I1]:= A2[I1,I1] - Z2;
        ROTCOMROW(I1, M, I, I1, A1, A2, MUI1,
        MUI2, NUI); A1[I,I]:= MUIM11 * KAPPA;
        A2[I,I]:= - MUIM12 * KAPPA;
        B[I - 1]:= NUIM1 * KAPPA;
        ROTCOMCOL(1, I, I, I1, A1, A2, MUI1, -
        MUI2, - NUI); A1[I,I]:= A1[I,I] + Z1;
        A2[I,I]:= A2[I,I] + Z2;
        ROTCOMCOL(1, M, I, I1, VEC1, VEC2, MUI1, -
        MUI2, - NUI);
    "END" "COMMENT"
```

```
            AIJ1:= A1[N,N]; AIJ2:= A2[N,N]; AIJ12:= AIJ1 ** 2;
            AIJ22:= AIJ2 ** 2; KAPPA:= SQRT(AIJ12 + AIJ22);
            "IF" ("IF" KAPPA < TOL "THEN" "TRUE" "ELSE" AIJ22 <=
            EM[0] * AIJ12) "THEN"
            "BEGIN" B[NM1]:= NUI * AIJ1;
                A1[N,N]:= AIJ1 * MUI1 + Z1;
                A2[N,N]:= - AIJ1 * MUI2 + Z2
            "END"
            "ELSE"
            "BEGIN" B[NM1]:= NUI * KAPPA; A1NN:= MUI1 * KAPPA;
                A2NN:= - MUI2 * KAPPA; MUI1:= AIJ1 / KAPPA;
                MUI2:= AIJ2 / KAPPA;
                COMCOLCST(1, NM1, N, A1, A2, MUI1, MUI2);
                COMCOLCST(1, NM1, N, VEC1, VEC2, MUI1,
                MUI2);
                COMROWCST(N + 1, M, N, A1, A2, MUI1, -
                MUI2);
                COMCOLCST(N, M, N, VEC1, VEC2, MUI1, MUI2);
                A1[N,N]:= MUI1 * A1NN - MUI2 * A2NN + Z1;
                A2[N,N]:= MUI1 * A2NN + MUI2 * A1NN + Z2;
            "END";
        "END";
    "END";
    "IF" N > 0 "THEN" "GOTO" IN;
    "FOR" J:= M "STEP" - 1 "UNTIL" 2 "DO"
    "BEGIN" TF1[J]:= 1; TF2[J]:= 0; T1:= A1[J,J]; T2:= A2[J,J];
        "FOR" I:= J - 1 "STEP" - 1 "UNTIL" 1 "DO"
        "BEGIN" DELTA1:= T1 - A1[I,I]; DELTA2:= T2 - A2[I,I];
            COMMATVEC(I + 1, J, I, A1, A2, TF1, TF2, MV1,
            MV2);
            "IF" ABS(DELTA1) < MACHTOL "AND" ABS(DELTA2) <
            MACHTOL "THEN"
            "BEGIN" TF1[I]:= MV1 / MACHTOL;
                TF2[I]:= MV2 / MACHTOL
            "END"
            "ELSE" COMDIV(MV1, MV2, DELTA1, DELTA2, TF1[I],
            TF2[I]);
        "END";
        "FOR" I:= 1 "STEP" 1 "UNTIL" M "DO" COMMATVEC(1, J, I,
        VEC1, VEC2, TF1, TF2, VEC1[I,J], VEC2[I,J]);
    "END";
OUT: EM[3]:= R; EM[5]:= COUNT; QRICOM:= N;
"END" QRICOM;
    "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

CONTRIBUTORS : H.FIOLET, C.G. VAN DER LAAN.

INSTITUTE : MATHEMATICAL CENTRE.

RECEIVED: 731016.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS THE PROCEDURES EIGVALCOM AND EIGCOM.
EIGVALCOM CALCULATES THE EIGENVALUES OF A COMPLEX MATRIX AND
EIGCOM CALCULATES THE EIGENVECTORS AS WELL.

KEYWORDS :

EIGENVALUES,
EIGENVECTORS,
COMPLEX MATRICES,
EQUILIBRATION,
REDUCTION HESSENBERG FORM,
HOUSEHOLDER TRANSFORMATION,
QR-ITERATION.

SUBSECTION: EIGVALCOM.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"INTEGER" "PROCEDURE" EIGVALCOM(AR, AI, N, EM, VALR, VALI);
"VALUE" N; "INTEGER" N; "ARRAY" AR, AI, EM, VALR, VALI;

THE MEANING OF THE FORMAL PARAMETERS IS:

AR,AI: <ARRAY IDENTIFIER>;
"ARRAY" AR,AI[1:N,1:N];
ENTRY:
THE REAL PART AND THE IMAGINARY PART OF THE MATRIX MUST BE GIVEN IN THE ARRAYS AR AND AI, RESPECTIVELY:
THE ELEMENTS OF THE ARRAYS AR AND AI ARE ALTERED:

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX:

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:7];
ENTRY:
EM[0]: THE MACHINE PRECISION:
EM[2]: THE RELATIVE TOLERANCE FOR THE QR-ITERATION:
(E.G. THE MACHINE PRECISION);
EM[4]: THE MAXIMUM ALLOWED NUMBER OF QR-ITERATIONS:
(E.G. 10 * N);
EM[6]: THE MAXIMUM ALLOWED NUMBER OF ITERATIONS FOR EQUILIBRATING THE ORIGINAL MATRIX (E.G. N**2/2);
EXIT:
EM[1]: THE EUCLIDEAN NORM OF THE EQUILIBRATED MATRIX;
EM[3]: THE MAXIMUM ABSOLUTE VALUE OF THE SUBDIAGONAL ELEMENTS NEGLECTED IN THE QR-ITERATION;
EM[5]: THE NUMBER OF QR-ITERATIONS PERFORMED:
EM[5]:=EM[4]+1 IN THE CASE EIGVALCOM^=0;
EM[7]: THE NUMBER OF ITERATIONS PERFORMED FOR EQUILIBRATING THE ORIGINAL MATRIX;

VALR,VALI: <ARRAY IDENTIFIER>;
"ARRAY" VALR,VALI[1:N];
EXIT:
THE REAL PART AND THE IMAGINARY PART OF THE CALCULATED EIGENVALUES ARE DELIVERED IN THE ARRAYS VALR AND VALI, RESPECTIVELY:

EIGVALCOM:=0, PROVIDED THE QR-ITERATION IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE, EIGVALCOM:= THE NUMBER, K, OF EIGENVALUES NOT CALCULATED AND ONLY THE LAST N-K ELEMENTS OF THE ARRAYS VALR AND VALI ARE APPROXIMATE EIGENVALUES OF THE ORIGINAL MATRIX.

PROCEDURES USED:

EQILBRCOM = CP34361,
COMEUCNRM = CP34359,
HSHCOMHES = CP34366,
VALQRICOM = CP34372.

REQUIRED CENTRAL MEMORY: FIVE REAL ARRAYS OF ORDER N AND ONE INTEGER ARRAY OF ORDER N ARE DECLARED.

RUNNING TIME: PROPORTIONAL TO N ** 2 * MAX(N,NUMBER OF ITERATIONS).

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE EIGCOM (THIS SECTION).

EXAMPLE OF USE: SEE EIGCOM (THIS SECTION).

SUBSECTION: EIGCOM.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"INTEGER" "PROCEDURE" EIGCOM(AR, AI, N, EM, VALR, VALI, VR, VI);
"VALUE" N; "INTEGER" N; "ARRAY" AR, AI, EM, VALR, VALI, VR, VI;

THE MEANING OF THE FORMAL PARAMETERS IS:

AR,AI: <ARRAY IDENTIFIER>;
"ARRAY" AR,AI[1:N,1:N];
ENTRY:
THE REAL PART AND THE IMAGINARY PART OF THE MATRIX MUST BE GIVEN IN THE ARRAYS AR AND AI, RESPECTIVELY;
THE ELEMENTS OF THE ARRAYS AR AND AI ARE ALTERED;

N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRIX;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:7];
ENTRY:
EM[0]: THE MACHINE PRECISION;
EM[2]: THE RELATIVE TOLERANCE FOR THE QR-ITERATION;
(E.G. THE MACHINE PRECISION);
EM[4]: THE MAXIMUM ALLOWED NUMBER OF QR-ITERATIONS;
(E.G. 10 * N);
EM[6]: THE MAXIMUM ALLOWED NUMBER OF ITERATIONS FOR EQUILIBRATING THE ORIGINAL MATRIX (E.G. N**2/2);
EXIT:
EM[1]: THE EUCLIDEAN NORM OF THE EQUILIBRATED MATRIX;
EM[3]: THE MAXIMUM ABSOLUTE VALUE OF THE SUBDIAGONAL ELEMENTS NEGLECTED IN THE QR-ITERATION;
EM[5]: THE NUMBER OF QR-ITERATIONS PERFORMED;
EM[5]:=EM[4]+1 IN THE CASE EIGCOM^=0;
EM[7]: THE NUMBER OF ITERATIONS PERFORMED FOR EQUILIBRATING THE ORIGINAL MATRIX;

```
    VALR,VALI: <ARRAY IDENTIFIER>;
               "ARRAY" VALR,VALI[1:N];
               EXIT:
               THE REAL PART AND THE IMAGINARY PART OF THE CALCULATED
               EIGENVALUES ARE DELIVERED IN THE ARRAYS VALR AND VALI,
               RESPECTIVELY;
    VR,VI:     <ARRAY IDENTIFIER>;
               "ARRAY" VR,VI[1:N,1:N];
               EXIT:
               THE EIGENVECTORS OF THE MATRIX;
               THE NORMALIZED EIGENVECTOR WITH REAL PART VR[1:N,J] AND
               IMAGINARY PART VI[1:N,J] CORRESPONDS TO THE EIGENVALUE
               VALR[J] + VALI[J] * I, J=1,...,N;
```

EIGCOM:=0, PROVIDED THE QR-ITERATION IS COMPLETED WITHIN EM[4] ITERATIONS; OTHERWISE, EIGCOM:= THE NUMBER, K, OF EIGENVALUES NOT CALCULATED AND ONLY THE LAST N-K ELEMENTS OF THE ARRAYS VALR AND VALI ARE APPROXIMATE EIGENVALUES OF THE ORIGINAL MATRIX AND NO USEFUL EIGENVECTORS ARE DELIVERED.

PROCEDURES USED:

EQILBRCOM = CP34361,
COMEUCNRM = CP34359,
HSHCOMHES = CP34366,
QRICOM = CP34373,
BAKCOMHES = CP34367,
BAKLBRCOM = CP34362,
SCLCOM = CP34360.

REQUIRED CENTRAL MEMORY: FIVE REAL ARRAYS OF ORDER N AND ONE INTEGER ARRAY OF ORDER N ARE DECLARED.

RUNNING TIME: PROPORTIONAL TO N**2 * MAX(N, NUMBER OF ITERATIONS).

LANGUAGE : ALGOL 60.

THE FOLLOWING HOLDS FOR BOTH PROCEDURES:

METHOD AND PERFORMANCE:

FOR CALCULATING THE EIGENVALUES AND EIGENVECTORS OF A COMPLEX MATRIX WE DISTINGUISH THE FOLLOWING STEPS:
1) THE MATRIX IS EQUILIBRATED (SEE ALSO SECTION 3.2.1.1.2.).
2) THE EQUILIBRATED MATRIX IS TRANSFORMED INTO HESSENBERG FORM BY MEANS OF HOUSEHOLDER MATRICES (SEE ALSO SECTION 3.2.1.2.2.2.).
3) THE HESSENBERG MATRIX IS TRANSFORMED INTO AN UPPER TRIANGULAR MATRIX BY MEANS OF QR-ITERATION WITH SHIFT OF ORIGIN AND DEFLATION (SEE ALSO SECTION 3.3.2.2.1.).

THE DIAGONAL ELEMENTS OF THE UPPER TRIANGULAR MATRIX ARE THE EIGENVALUES OF THE ORIGINAL MATRIX.
THE EIGENVECTORS OF THE ORIGINAL MATRIX ARE ORTAINED BY CALCULATING THE EIGENVECTORS OF THE UPPER TRIANGULAR MATRIX (3) FOLLOWED BY BACKTRANSFORMATIONS CORRESPONDING TO (2) AND (1).

REFERENCES:

DEKKER, T.J. (1968),
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 1,
MATH. CENTRE TRACTS 22, MATHEMATISCH CENTRUM;

DEKKER, T.J. AND W.HOFFMANN (1968),
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2,
MATH. CENTRE TRACTS 23, MATHEMATISCH CENTRUM;

FRANCIS, J.G.F. (1961),
THE QR-TRANSFORMATION, PART 1 AND 2,
COMP.J. 4, P.265-271 AND P.332-345;

MUELLER, D.J. (1966),
HOUSEHOLDER,S METHOD FOR COMPLEX MATRICES AND EIGENSYSTEMS OF HERMITIAN MATRICES,
NUMER.MATH., 8, P.72-92;

OSBORNE, E.E. (1960),
ON PRECONDITIONING OF MATRICES,
JACM., 7, P.338-354;

PARLETT, B.N. AND C.REINSCH (1969),
BALANCING A MATRIX FOR CALCULATION OF EIGENVALUES AND EIGENVECTORS,
NUM. MATH., 13, P.293-304;

WILKINSON, J.H. (1965),
THE ALGEBRAIC EIGENVALUE PROBLEM,
CLARENDON PRESS, OXFORD.

EXAMPLE OF USE:

EIGCOM CALCULATES THE EIGENVALUES AND THE EIGENVECTORS OF THE
FOLLOWING MATRIX:
(SEE WILKINSON AND REINSCH, 1971, CONTRIBUTION II/15)

```
1+3*I  2+1*I  3+2*I  1+1*I
3+4*I  1+2*I  2+1*I  4+3*I
2+3*I  1+5*I  3+1*I  5+2*I
1+2*I  3+1*I  1+4*I  5+3*I
```

ONLY THE EIGENVECTOR CORRESPONDING TO VALR[1] + VALI[1] * I IS
PRINTED BY THE FOLLOWING PROGRAM.

```
"BEGIN"
"INTEGER" "PROCEDURE" EIGCOM(AR,AI,N,EM,VALR,VALI,VR,VI);
"CODE" 34375;
"REAL" "ARRAY" AR,AI,VR,VI[1:4,1:4],EM[0:7],VALR,VALI[1:4];
"INTEGER" I;
AR[1,1]:=AR[1,4]:=AR[2,2]:=AR[3,2]:=AR[4,1]:=AR[4,3]:=
AI[1,2]:=AI[1,4]:=AI[2,3]:=AI[3,3]:=AI[4,2]:=1;
AR[1,2]:=AR[2,3]:=AR[3,1]:=AI[1,3]:=AI[2,2]:=AI[3,4]:=AI[4,1]:=2;
AR[1,3]:=AR[2,1]:=AR[3,3]:=AR[4,2]:=
AI[1,1]:=AI[2,4]:=AI[3,1]:=AI[4,4]:=3;
AR[2,4]:=AI[2,1]:=AI[4,3]:=4;
AR[3,4]:=AR[4,4]:=AI[3,2]:=5;
EM[0]:=5"-14;EM[2]:="-12;EM[4]:=10;EM[6]:=10;
OUTPUT(61,"(""(""EIGCOM: ")",D")",
          EIGCOM(AR,AI,4,EM,VALR,VALI,VR,VI));
OUTPUT(61,"("/,"("EIGENVALUES:")",/")");
"FOR" I:=1,2,3,4 "DO" OUTPUT(61,"("2(+D.4D),"(" * I")",/")",
                             VALR[I],VALI[I]);
OUTPUT(61,"(""("FIRST EIGENVECTOR:")",/")");
"FOR" I:=1,2,3,4 "DO" OUTPUT(61,"("2(+D.4D),"(" * I")",/")",
                             VR[I,1],VI[I,1]);
OUTPUT(61,"(""("EM[1]: ")",+DD.DD/,"("EM[3]: ")",+D.D"+DD/,
  "("EM[5]: ")",+ZD/,"("EM[7]: ")",+ZD")",EM[1],EM[3],EM[5],EM[7]);
"END"
```

OUTPUT:

```
EIGCOM: 0
EIGENVALUES:
-3.3710-0.7705 * I
+9.7837+9.3225 * I
+1.3657-1.4011 * I
+2.2217+1.8490 * I
FIRST EIGENVECTOR:
-0.5061+0.5835 * I
+1.0000+0.0000 * I
+0.5183-0.7147 * I
-0.5535+0.0188 * I
EM[1]: +15.30
EM[3]: +6.0"-12
EM[5]:  +7
EM[7]:  +4
```

SOURCE TEXT(S) :

```
"CODE" 34374;
    "INTEGER" "PROCEDURE" EIGVALCOM(AR, AI, N, EM, VALR, VALI);
    "VALUE" N; "INTEGER" N; "ARRAY" AR, AI, EM, VALR, VALI;
    "BEGIN" "INTEGER" "ARRAY" INT[1:N];
        "ARRAY" D, B, DEL, TR, TI[1:N];
        "PROCEDURE" HSHCOMHES(AR,AI,N,EM,B,TR,TI,DEL);"CODE" 34366;
        "REAL" "PROCEDURE" COMEUCNRM(AR,AI,LW,N);"CODE" 34359;
        "PROCEDURE" EQILBRCOM(A1,A2,N,EM,D,INT);"CODE" 34361;
        "INTEGER" "PROCEDURE" VALQRICOM(A1,A2,B,N,EM,VAL1,VAL2);
        "CODE" 34372;
        EQILBRCOM(AR, AI, N, EM, D, INT);
        EM[1]:= COMEUCNRM(AR, AI, N - 1, N);
        HSHCOMHES(AR, AI, N, EM, B, TR, TI, DEL);
        EIGVALCOM:= VALQRICOM(AR, AI, B, N, EM, VALR, VALI)
    "END" EIGVALCOM;
        "EOP"

"CODE" 34375;
    "INTEGER" "PROCEDURE" EIGCOM(AR, AI, N, EM, VALR, VALI, VR, VI);
    "VALUE" N; "INTEGER" N; "ARRAY" AR, AI, EM, VALR, VALI, VR, VI;
    "BEGIN" "INTEGER" I;
        "INTEGER" "ARRAY" INT[1:N];
        "ARRAY" D, B, DEL, TR, TI[1:N];
        "PROCEDURE" EQILBRCOM(A1,A2,N,EM,D,INT);"CODE" 34361;
        "REAL" "PROCEDURE" COMEUCNRM(AR,AI,LW,N);"CODE" 34359;
        "PROCEDURE" HSHCOMHES(AR,AI,N,EM,B,TR,TI,DEL);"CODE" 34366;
        "INTEGER" "PROCEDURE" QRICOM(A1,A2,B,N,EM,VAL1,VAL2,VEC1,VEC2);
        "CODE" 34373;
        "PROCEDURE" BAKCOMHES(AR,AI,TR,TI,DEL,VR,VI,N,N1,N2);
        "CODE" 34367;
        "PROCEDURE" BAKLBRCOM(N,N1,N2,D,INT,VR,VI);"CODE" 34362;
        "PROCEDURE" SCLCOM(AR,AI,N,N1,N2);"CODE" 34360;
        EQILBRCOM(AR, AI, N, EM, D, INT);
        EM[1]:= COMEUCNRM(AR, AI, N - 1, N);
        HSHCOMHES(AR, AI, N, EM, B, TR, TI, DEL);
        I:= EIGCOM:= QRICOM(AR, AI, B, N, EM, VALR, VALI, VR,
        VI); "IF" I = 0 "THEN"
        "BEGIN" BAKCOMHES(AR, AI, TR, TI, DEL, VR, VI, N, 1, N);
            BAKLBRCOM(N, 1, N, D, INT, VR, VI);
            SCLCOM(VR, VI, N, 1, N)
        "END"
    "END" EIGCOM;
        "EOP"
```

AUTHOR: J.J.G. ADMIRAAL.

INSTITUTE: UNIVERSITY OF AMSTERDAM.

RECEIVED: 751101.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS TWO MAIN AND NINE AUXILIARY PROCEDURES:

THE TWO MAIN PROCEDURES ARE:
A. QZIVAL FINDS N PAIRS OF SCALARS (ALFA[M],BETA[M]),WHERE BETA[M] IS REAL,SUCH THAT THE MATRIX BETA[M] * A - ALFA[M] * B IS SINGULAR.
B. QZI FINDS N PAIRS OF SCALARS (ALFA[M], BETA[M]),WHERE BETA[M] IS REAL,SUCH THAT THE MATRIX BETA[M] * A - ALFA[M] * B IS SINGULAR; MOREOVER THE GENERALIZED EIGENVECTORS (THE HOMOGENOUS SOLUTION OF (BETA[M] * A - ALFA[M] * B) * X = 0) ARE CALCULATED.

THE AUXILIARY PROCEDURES ARE:
A. HSHDECMUL:
THIS PROCEDURE CALCULATES REAL MATRICES Q AND R SUCH THAT Q.A=R WHERE A IS A GIVEN REAL SQUARE MATRIX, Q IS A PRODUCT OF HOUSEHOLDER MATRICES AND R AN UPPERTRIANGULAR MATRIX. MOREOVER Q.B IS FORMED WITH B, A GIVEN MATRIX.
B. HESTGL3:
GIVEN THE REAL SQUARE MATRICES A,B AND X, WITH B AN UPPER TRIANGULAR MATRIX,HESTGL3 CALCULATES THE MATRICES Q,Z,H,R, WHERE Q,Z ARE ORTHOGONAL, H UPPER HESSENBERG AND R AN UPPER TRIANGULAR MATRIX SUCH THAT Q.A.Z = H AND Q.B.Z = R.
FURTHER: A:= Q.A.Z ; B:= Q.B.Z AND X:= Q.X.Z.
C. HESTGL2:
SEE HESTGL3,BUT HERE THE MATRIX X HAS BEEN LEFT OUT.
D. HSH2COL:
THIS PROCEDURE CALCULATES A HOUSEHOLDER MATRIX Q SUCH THAT BY PREMULTIPLYING A GIVEN COLUMN VECTOR V BY Q A ZERO ELEMENT IS FORMED IN V.
HERE THE VECTOR V IS A COLUMN OF A MATRIX.
FURTHER: A:= Q.A AND B:= Q.B
WHERE A,B ARE TWO GIVEN REAL MATRICES.

E. HSH3COL:
THIS PROCEDURE CALCULATES A HOUSEHOLDER MATRIX
Q SUCH THAT BY PREMULTIPLYING A GIVEN COLUMN VECTOR V BY Q TWO
SUCCESSIVE ZERO ELEMENTS ARE FORMED IN V.
HERE THE VECTOR V IS A COLUMN OF A MATRIX.
FURTHER: A:= Q.A AND B:= Q.B
WHERE A AND B ARE TWO GIVEN REAL MATRICES.

F. HSH2ROW3:
THIS PROCEDURE CALCULATES A HOUSEHOLDER MATRIX
Z SUCH THAT BY POSTMULTIPLYING A GIVEN ROWVECTOR
V BY Z A ZERO ELEMENT IS FORMED IN V.
HERE THE VECTOR V IS A ROW OF A MATRIX.
FURTHER: A:= A.Z; B:= B.Z AND X:= X.Z
WHERE A,B,X ARE THREE GIVEN REAL MATRICES.

G. HSH2ROW2:
SEE HSH2ROW3, BUT HERE THE MATRIX X HAS BEEN LEFT OUT.

H. HSH3ROW3:
THIS PROCEDURE CALCULATES A HOUSEHOLDER MATRIX
Z SUCH THAT BY POSTMULTIPLYING A GIVEN ROWVECTOR
V BY Z, TWO SUCCESSIVE ZERO ELEMENTS ARE FORMED
IN V. HERE THE VECTOR V IS A ROW OF A MATRIX.
FURTHER: A:= A.Z; B:= B.Z AND X:= X.Z
WHERE A,B AND X ARE THREE GIVEN REAL MATRICES.

I: HSH3ROW2:
SEE HSH3ROW3, BUT HERE THE MATRIX X HAS BEEN LEFT OUT.

KEYWORDS:
HOUSEHOLDER'S TRANSFORMATION,
GENERALIZED EIGENVALUES,
GENERALIZED EIGENVECTORS,
UPPER HESSENBERG MATRIX,
UPPER TRIANGULAR MATRIX.

REFERENCES:

[1]. C.B. MOLER AND G.W. STEWART.
AN ALGORITHM FOR THE GENERALIZED MATRIX EIGENVALUE
PROBLEM A * X = LAMBDA * B * X.
REPORT STANFORD UNIVERSITY
STAN-CS-232-71;

SUBSECTION: QZIVAL

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE IS:
    "PROCEDURE" QZIVAL(N,A,B,ALFR,ALFI,BETA,ITER,EM);
    "VALUE" N; "INTEGER" N; "ARRAY" A,B,ALFR,ALFI,BETA,EM;
    "INTEGER" "ARRAY" ITER;
    "CODE" 34600;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N:    <ARITHMETIC EXPRESSION>;
          THE NUMBER OF ROWS AND COLUMNS OF THE MATRICES A,B
    A:    <ARRAY IDENTIFIER>;
          "ARRAY" A[1:N,1:N];
          ENTRY: THE GIVEN MATRIX;
          EXIT: A QUASI UPPER-TRIANGULAR MATRIX
                (SEE METHOD AND PERFORMANCE);
    B:    <ARRAY IDENTIFIER>;
          "ARRAY" B[1:N,1:N];
          ENTRY: THE GIVEN MATRIX;
          EXIT: AN UPPER-TRIANGULAR MATRIX;
    ALFR: <ARRAY IDENTIFIER>;
          "ARRAY" ALFR[1:N];
          EXIT : THE REAL PARTS OF ALFA[1:N]
          (SEE METHOD AND PERFORMANCE);
    ALFI: <ARRAY IDENTIFIER>;
          "ARRAY" ALFI[1:N];
          EXIT : THE IMAGINARY PARTS OF ALFA[1:N];
    BETA: <ARRAY IDENTIFIER>;
          "ARRAY" BETA[1:N];
          EXIT : THE REAL SCALARS BETA[N]
    ITER: <ARRAY IDENTIFIER>;
          "INTEGER" "ARRAY" ITER[1:N];
          TROUBLE INDICATOR AND ITERATION COUNTER;
          IF ITER[1]=0 THEN NO TROUBLE IS SIGNALIZED,
          FURTHER SEE METHOD AND PERFORMANCE;
    EM:   <ARRAY IDENTIFIER>;
          "ARRAY" EM[0:1];
           ENTRY: EM[0]: THE SMALLEST POSITIVE MACHINE NUMBER;
                  EM[1]: THE RELATIVE PRECISION OF ELEMENTS OF A AND B;
```

PROCEDURES USED:

TAMMAT	= CP 34014
ELMCOL	= CP 34023
HSHDECMUL	= CP 34602
HESTGL2	= CP 34604
HSH2COL	= CP 34605
HSH3COL	= CP 34606
HSH2ROW2	= CP 34608
HSH3ROW2	= CP 34610
CHSH2	= CP 34611
HSHVECMAT	= CP 31070
HSHVECTAM	= CP 31073

REQUIRED CENTRAL MEMORY : IN HSHDECMUL, AN ARRAY OF N REALS IS DECLARED.

RUNNING TIME: PROPORTIONAL TO N ** 3

METHOD AND PERFORMANCE:

THE PROCEDURE QZIVAL SOLVES THE GENERALIZED MATRIX
EIGENVALUE PROBLEM A * X = LAMBDA * B * X BY MEANS
OF QZ ITERATION (SEE REF[1]);
QZIVAL FINDS N PAIRS OF SCALARS (ALFA[M],BETA[M])
SUCH THAT BETA[M] * A - ALFA[M] * B IS SINGULAR.
THE EIGENVALUES OF A * X - LAMBDA * B * X CAN BE OBTAINED
BY DIVIDING ALFA[M] BY BETA[M],EXCEPT BETA[M] MIGHT BE ZERO.
IN THIS ALGORITHM ONLY UNITARY TRANSFORMATIONS ARE
APPLIED; A FORTIORI NO INVERSES ARE CALCULATED, SO
EITHER A OR B (OR BOTH) MAY BE SINGULAR.
BETA[M] IS REAL, ALFA[M] IS COMPLEX.
REAL AND IMAGINARY PARTS ARE GIVEN IN ALFR[M] AND ALFI[M].
THE OCCURRENCE OF COMPLEX PAIRS IS ALWAYS IN
SUCCESSIVE ELEMENTS, SUCH THAT ALFA[M]/BETA[M] AND
ALFA[M+1]/BETA[M+1] ARE COMPLEX CONJUGATE, BUT ALFA[M]
AND ALFA[M+1] ARE NOT NECESSARILY CONJUGATE.
ONLY REAL ARITHMETIC IS USED IN THE PROCEDURE.
IF A AND B WERE REDUCED TO TRIANGULAR FORM BY UNITARY
TRANSFORMATIONS,ALFA AND BETA WOULD BE THE DIAGONALS.
A AND B ARE ACTUALLY REDUCED TO QUASI-TRIANGULAR FORM HAVING ONLY
1-BY-1 AND 2-BY-2 BLOCKS ON THE DIAGONAL OF A.
IF ALFA[M] IS NOT REAL, THEN BETA[M] IS NOT ZERO.
ITER IS THE TROUBLE INDICATOR AND ITERATION COUNTER.
IF ITER[1]=0 THEN EVERYTHING IS O.K.
ITER[M] IS THE NUMBER OF ITERATIONS NEEDED FOR THE M-TH EIGENVALUE.
IF ITER[1] THROUGH ITER[M]= -1 THEN THE ITERATION FOR THE M-TH
EIGENVALUE DID NOT CONVERGE AND ALFA[1] THROUGH ALFA[M] AND BETA[1]
THROUGH BETA[M] ARE PROBABLY INACCURATE.

EXAMPLE OF USE:

```
"BEGIN" "ARRAY" A,B[1:4,1:4],ALFR,ALFI,BETA[1:4],EM[0:1];
    "INTEGER" "ARRAY" ITER[1:4];"INTEGER" K,L;
    "PROCEDURE" QZIVAL(N,A,B,ALFR,ALFI,BETA,ITER,EM);"CODE" 34600;
    A[1,1]:=2; A[1,2]:=3; A[1,3]:=-3; A[1,4]:=4;
```

```
    A[2,1]:=1; A[2,2]:=-1; A[2,3]:=5; A[2,4]:=1;
    A[3,1]:=0; A[3,2]:=2; A[3,3]:=6; A[3,4]:=8;
    A[4,1]:=1; A[4,2]:=1; A[4,3]:=0; A[4,4]:=4;
    B[1,1]:=1; B[1,2]:=5; B[1,3]:=9; B[1,4]:=0;
    B[2,1]:=2; B[2,2]:=6; B[2,3]:=10; B[2,4]:=2;
    B[3,1]:=3; B[3,2]:=7; B[3,3]:=11; B[3,4]:=-1;
    B[4,1]:=4; B[4,2]:=8; B[4,3]:=12; B[4,4]:=3;
    OUTPUT(61,"("""("A")",/,4(4(+ZDBB),/),/")",A);
    OUTPUT(61,"("""("B")",/,4(4(+ZDBB),/),/")",B);
    EM[0]:="-294;EM[1]:="-15;
    QZIVAL(4,A,B,ALFR,ALFI,BETA,ITER,EM);
    "FOR" K:=1 "STEP" 1 "UNTIL" 4 "DO"
    OUTPUT(61,"("""("ITER[")",D,"(""]=")",ZD,/")",K,ITER[K]);
    OUTPUT(61,"("""("ALFA(REAL PART)")"8B,"("ALFA(IMAGINARY PART)")"
    3B,"("BETA")",/")");
    "FOR" K:=1 "STEP" 1 "UNTIL" 4 "DO"
    OUTPUT(61,"("3(N),/")",ALFR[K],ALFI[K],BETA[K]);
    OUTPUT(61,"("/"("LAMBDA(REAL PART)")"6B,
    "("LAMBDA(IMAGINARY PART)")"/")");
    "FOR" K:=1 "STEP" 1 "UNTIL" 4 "DO"
    "BEGIN" "IF" BETA[K]=0 "THEN"
        OUTPUT(61,"("""("INFINITE")"15B,"("INDEFINITE")"/,")")
        "ELSE" OUTPUT(61,"("2(N),/")",ALFR[K]/BETA[K],ALFI[K]/BETA[K])
    "END"
"END"

A
 +2    +3    -3    +4
 +1    -1    +5    +1
 +0    +2    +6    +8
 +1    +1    +0    +4

B
 +1    +5    +9    +0
 +2    +6   +10    +2
 +3    +7   +11    -1
 +4    +8   +12    +3

ITER[1]= 0
ITER[2]= 0
ITER[3]= 0
ITER[4]= 5

ALFA(REAL PART)          ALFA(IMAGINARY PART)    BETA
-4.4347115652167"+000    +0.0000000000000"+000   +0.0000000000000"+000
-5.7288406521003"+000    +0.0000000000000"+000   +2.8441121744895"+000
-8.6671777386054"-001    +2.7607904944916"+000   +8.7617886336960"+000
-4.7262205157527"-001    -1.5054617625576"+000   +4.7778119295757"+000

LAMBDA(REAL PART)        LAMBDA(IMAGINARY PART)
INFINITE                 INDEFINITE
-2.0142808372628"+000    +0.0000000000000"+000
-9.8920187429234"-002    +3.1509439566644"-001
-9.8920187429236"-002    -3.1509439566645"-001
```

SUBSECTION: QZI

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE IS:
    "PROCEDURE" QZI(N,A,B,X,ALFR,ALFI,BETA,ITER,EM);
    "VALUE" N; "INTEGER" N; "ARRAY" A,B,X,ALFR,ALFI,BETA,EM;
    "INTEGER" "ARRAY" ITER;
    "CODE" 34601;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N:     <ARITHMETIC EXPRESSION>;
           THE NUMBER OF ROWS AND COLUMNS OF THE MATRICES A,B AND X;
    A:     <ARRAY IDENTIFIER>
           "ARRAY" A[1:N,1:N];
           ENTRY: THE GIVEN MATRIX A;
           EXIT: A QUASI UPPER TRIANGULAR MATRIX;
                 (SEE METHOD AND PERFORMANCE);
    B:     <ARRAY IDENTIFIER>;
           "ARRAY" B[1:N,1:N];
           ENTRY: THE GIVEN MATRIX B;
           EXIT: AN UPPER-TRIANGULAR MATRIX;
    X:     <ARRAY IDENTIFIER>;
           "ARRAY" X[1:N,1:N];
           ENTR: THE N*N UNIT MATRIX;
           EXIT: THE MATRIX OF EIGENVECTORS,
           THE EIGENVECTORS ARE STORED IN THE ARRAY X AS FOLLOWS:
           IF ALFI[M]=0 THEN X[.,M] IS THE M-TH REAL EIGENVECTOR;
           OTHERWISE, FOR EACH PAIR OF CONSECUTIVE COLUMNS
           X[.,M] AND X[.,M+1] ARE THE REAL
           AND IMAGINARY PARTS OF THE M-TH COMPLEX EIGENVECTOR.
           X[.,M] AND -X[.,M+1] ARE THE REAL AND IMAGINARY PARTS
           OF THE M+1 -ST COMPLEX EIGENVECTOR.
           THE EIGENVECTORS ARE NORMALIZED SUCH THAT THE LARGEST
           COMPONENT IS 1 OR 1 + 0 * I.
    ALFR:  <ARRAY IDENTIFIER>;
           "ARRAY" ALFR[1:N];
           EXIT: THE REAL PARTS OF ALFA[1:N];
    ALFI:  <ARRAY IDENTIFIER>;
           "ARRAY" ALFI[1:N];
           EXIT: THE IMAGINARY PARTS OF ALFA[1:N];
    BETA:  <ARRAY IDENTIFIER>;
           "ARRAY" BETA[1:N];
    ITER:  <ARRAY IDENTIFIER>;
           "INTEGER" "ARRAY" ITER[1:N];
           TROUBLE INDICATOR AND ITERATION COUNTER;
           IF ITER[1]=0 THEN NO TROUBLE IS SIGNALIZED,
           FOR FURTHER INFORMATION SEE
           METHOD AND PEFORMANCE OF PROCEDURE QZIVAL (THIS SECTION).
    EM:    <ARRAY IDENTIFIER>;
           "ARRAY" EM[0:1];
           ENTRY: EM[0]: THE SMALLEST POSITIVE MACHINE NUMBER;
                  EM[1]: THE RELATIVE PRECISION OF ELEMENTS OF A AND B;
```

PROCEDURES USED:

MATMAT	= CP 34013
TAMMAT	= CP 34014
ELMCOL	= CP 34023
HSHDECMUL	= CP 34602
HESTGL3	= CP 34603
HSH2COL	= CP 34605
HSH2ROW3	= CP 34607
HSH3ROW3	= CP 34609
HSH3COL	= CP 34606
CHSH2	= CP 34611
COMDIV	= CP 34342
HSHVECMAT	= CP 31070
HSHVECTAM	= CP 31073

RUNNING TIME: PROPORTIONAL TO N ** 3;

REQUIRED CENTRAL MEMORY : IN HSHDECMUL, AN ARRAY OF N REALS IS DECLARED.

METHOD AND PERFORMANCE:

THE PROCEDURE QZI APPLIES THE SAME METHOD AS QZIVAL.

EXAMPLE OF USE:

```
"BEGIN" "ARRAY" A,B,X[1:4,1:4],ALFR,ALFI,BETA[1:4],EM[0:1];
    "INTEGER" "ARRAY" ITER[1:4];"INTEGER" K,L;
    "PROCEDURE" QZI(N,A,B,X,ALFR,ALFI,BETA,ITER,EM);"CODE" 34601;
    A[1,1]:=2; A[1,2]:=3; A[1,3]:=-3; A[1,4]:=4;
    A[2,1]:=1; A[2,2]:=-1; A[2,3]:=5; A[2,4]:=1;
    A[3,1]:=0; A[3,2]:=2; A[3,3]:=6; A[3,4]:=8;
    A[4,1]:=1; A[4,2]:=1; A[4,3]:=0; A[4,4]:=4;
    B[1,1]:=1; B[1,2]:=5; B[1,3]:=9; B[1,4]:=0;
    B[2,1]:=2; B[2,2]:=6; B[2,3]:=10; B[2,4]:=2;
    B[3,1]:=3; B[3,2]:=7; B[3,3]:=11; B[3,4]:=-1;
    B[4,1]:=4; B[4,2]:=8; B[4,3]:=12; B[4,4]:=3;
    "FOR" K:=1,2,3,4 "DO" "FOR" L:=1,2,3,4 "DO"
    X[K,L]:="IF" K=L "THEN" 1 "ELSE" 0;
    OUTPUT(61,"(""("A")",/,4(4(+ZDBB),/),/")",A);
    OUTPUT(61,"(""("B")",/,4(4(+ZDBB),/),/")",B);
    EM[0]:="-294;EM[1]:="-15;
    QZI(4,A,B,X,ALFR,ALFI,BETA,ITER,EM);
```

```
    "FOR" K:=1 "STEP" 1 "UNTIL" 4 "DO"
    OUTPUT(61,"("""("ITER[")",D,"("]=")",ZD,/")",K,ITER[K]);
    OUTPUT(61,"("/"("EIGENVECTORS")",/,4(4(+D.8D"+2D2B),/),/")",X);
    OUTPUT(61,"("""("ALFA(REAL PART)")"8B,"("ALFA(IMAGINARY PART)")"
    9B,"("BETA")",/")");
    "FOR" K:=1 "STEP" 1 "UNTIL" 4 "DO"
    OUTPUT(61,"("3(N),/")",ALFR[K],ALFI[K],BETA[K]);
    OUTPUT(61,"("/"("LAMBDA(REAL PART)")"6B,
    "("LAMBDA(IMAGINARY PART)")"/")");
    "FOR" K:=1 "STEP" 1 "UNTIL" 4 "DO"
    "BEGIN" "IF" BETA[K]=0 "THEN"
        OUTPUT(61,"("""("INFINITE")"15B,"("INDEFINITE")"/")")
        "ELSE"
        OUTPUT(61,"("2(N),/")",ALFR[K]/BETA[K],ALFI[K]/BETA[K])
    "END"
"END"

A
 +2    +3    -3    +4
 +1    -1    +5    +1
 +0    +2    +6    +8
 +1    +1    +0    +4

B
 +1    +5    +9    +0
 +2    +6   +10    +2
 +3    +7   +11    -1
 +4    +8   +12    +3

ITER[1]= 0
ITER[2]= 0
ITER[3]= 0
ITER[4]= 5

EIGENVECTORS
-5.00000000"-01  +1.00000000"+00  -6.29204867"-01  +6.52026261"-01
+1.00000000"+00  -3.82541766"-02  +1.00000000"+00  +0.00000000"+00
-5.00000000"-01  -3.04677732"-02  +1.65896051"-01  +1.09306265"-01
-4.35116786"-15  -7.63328122"-01  -5.84845537"-01  +1.77430910"-01

ALFA(REAL PART)        ALFA(IMAGINARY PART)   BETA
-4.434711565216 7"+000  +0.000000000000 0"+000  +0.000000000000 0"+000
-5.728840652100 3"+000  +0.000000000000 0"+000  +2.844112174489 6"+000
-8.667177738605 4"-001  +2.760790494491 6"+000  +8.761788633696 0"+000
-4.726220515752 7"-001  -1.505461762557 6"+000  +4.777811929575 7"+000

LAMBDA(REAL PART)      LAMBDA(IMAGINARY PART)
INFINITE               INDEFINITE
-2.011428083726 28"+000  +0.000000000000 0"+000
-9.892018742923 4"-002  +3.150943956664 4"-001
-9.892018742923 6"-002  -3.150943956664 5"-001
```

SUBSECTION: HSHDECMUL.

CALLING SEQUENCE:

THE HEADING OF THIS PROCEDURE IS:
"PROCEDURE" HSHDECMUL(N,A,B,DWARF);
"VALUE" N,DWARF; "INTEGER"N; "REAL"DWARF; "ARRAY" A,B;
"CODE" 34602;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRICES;
A: <ARRAY IDENTIFIER>;
"REAL" "ARRAY" A[1:N,1:N];
ENTRY: THE GIVEN MATRIX A;
EXIT: THE TRANSFORMED MATRIX Q.A (SEE BRIEF DESCRIPTION);
B: <ARRAY IDENTIFIER>;
"REAL" "ARRAY" B[1:N,1:N];
ENTRY: THE GIVEN MATRIX B;
EXIT: THE UPPER TRIANGULAR MATRIX Q.B (SEE BRIEF DESCRIPTION);
DWARF: < ARITHMETIC EXPRESSION>;
THE SMALLEST POSITIVE MACHINE NUMBER.

PROCEDURES USED:
TAMMAT = CP 34014;
HSHVECMAT = CP 31070.

REQUIRED CENTRAL MEMORY : AN ARRAY OF N REALS IS DECLARED.

METHOD AND PERFORMANCE:
FOR EACH MATRIX COLUMN A[I] A HOUSEHOLDERMATRIX Q IS FORMED,
SUCH THAT Q.A[I] HAS ONLY ZERO ELEMENTS BELOW THE DIAGONAL ELEMENT.
WHEN SUCCESSIVELY FOR I = 1,2,....,N-1 THESE TRANSFORMATIONS HAVE
BEEN PERFORMED,THE MATRIX A HAS BEEN CHANGED INTO AN UPPER
TRIANGULAR MATRIX.
THE SAME TRANSFORMATIONS ARE PERFORMED ON THE MATRIX B

EXAMPLE OF USE:

THE PROCEDURE HSHDECMUL IS USED IN QZI AND QZIVAL (THIS SECTION).

SUBSECTION: HESTGL3:

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" HESTGL3(N,A,B,X);
"VALUE" N; "INTEGER" N; "ARRAY" A,B,X;
"CODE" 34603;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRICES;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE GIVEN MATRIX A;
EXIT: THE UPPER HESSENBERG MATRIX Q.A.Z (SEE BRIEF DESCRIPTION);
B: <ARRAY IDENTIFIER>;
"ARRAY" B[1:N,1:N];
ENTRY: THE GIVEN UPPER TRIANGULAR MATRIX B;
EXIT: THE UPPER TRIANGULAR MATRIX Q.B.Z (SEE BRIEF DESCRIPTION);
X: <ARRAY IDENTIFIER>;
"ARRAY" X[1:N,1:N];
ENTRY: THE GIVEN MATRIX X;
EXIT: THE TRANSFORMED MATRIX Q.X.Z (SEE BRIEF DESCRIPTION);

PROCEDURES USED:
HSH2COL = CP 34605
HSH2ROW3 = CP 34607

METHODE AND PERFORMANCE:
THE REDUCTION OF THE MATRIX A TO UPPER HESSENBERG FORM
WHILE PRESERVING THE TRIANGULARITY OF THE MATRIX B
IS THE RESULT OF A NUMBER OF STEPS, WHICH DO THE FOLLOWING
ACTIONS: INTRODUCING A ZERO ELEMENT IN A AND RESTORING
THE DISTURBED ZERO IN B, WITHOUT DISTURBING THE ZERO
INTRODUCED IN A. THIS IS DONE BY PRE-AND POSTMULTIPLICATIONS OF
HOUSEHOLDER MATRICES.
THE MATRIX X SHARES THE TRANSFORMATION.
FOR FURTHER DETAILS SEE [1]

EXAMPLE OF USE:

THE PROCEDURE HESTGL3 IS USED IN QZI (THIS SECTION).

SUBSECTION: HESTGL2:

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" HESTGL2(N,A,B); "VALUE" N; "INTEGER" N; "ARRAY" A,B;
"CODE" 34604;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE GIVEN MATRICES:
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,1:N];
ENTRY: THE GIVEN MATRIX A;
EXIT: THE UPPER HESSENBERG MATRIX Q.A.Z (SEE BRIEF DESCRIPTION);
B: <ARRAY IDENTIFIER>;
"ARRAY" B[1:N,1:N];
ENTRY: THE GIVEN UPPER TRIANGULAR MATRIX B
EXIT: THE UPPER TRIANGULAR MATRIX Q.B.Z (SEE BRIEF DESCRIPTION);

PROCEDURES USED:
HSH2COL = CP 34605
HSH2ROW2 = CP 34608

METHODE AND PERFORMANCE:
SEE HESTGL3, BUT HERE THE MATRIX X HAS BEEN LEFT OUT.

EXAMPLE OF USE:
THE PROCEDURE HESTGL2 IS USED IN QZIVAL (THIS SECTION).

SUBSECTION HSH2COL:

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" HSH2COL(LA,LB,U,I,A1,A2,A,B); "VALUE" LA,LB,U,I,A1,A2;
"INTEGER" LA,LB,U,I; "REAL" A1,A2; "ARRAY" A,B;
"CODE" 34605;

THE MEANING OF THE FORMAL PARAMETERS IS:
LA: <ARITHMETIC EXPRESSION>;
THE LOWER BOUND OF THE RUNNING COLUMN SUBSCRIPT OF A;
LB: <ARITHMETIC EXPRESSION>;
THE LOWER BOUND OF THE RUNNING COLUMN SUBSCRIPT OF B;
U: <ARITHMETIC EXPRESSION>;
THE UPPER BOUND OF THE RUNNING COLUMN SUBSCRIPTS
OF A AND B.
I: <ARITHMETIC EXPRESSION>;
THE LOWERBOUND OF THE RUNNING ROW SUBSCRIPTS OF A AND B.
I+1 IS THE UPPERBOUND.
A1: <ARITHMETIC EXPRESSION>;
THE I-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED.
A2: <ARITHMETIC EXPRESSION>;
THE (I+1)-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[I:I+1,LA:U];
ENTRY THE GIVEN MATRIX A;
EXIT: THE TRANSFORMED MATRIX Q.A (SEE BRIEF DESCRIPTION);
B: <ARRAY IDENTIFIER>;
"ARRAY" B[I:I+1,LB:U];
ENTRY: THE GIVEN MATRIX B;
EXIT: THE TRANSFORMED MATRIX Q.B (SEE BRIEF DESCRIPTION);

PROCEDURES USED:
HSHVECMAT = CP 31070

METHOD AND PERFORMANCE:
WHEN THE CALCULATED HOUSEHOLDER MATRIX Q PREMULTIPLIES
A MATRIX M. ONLY ROWS I AND I+1 ARE CHANGED.
IF THE ELEMENTS I AND I+1 IN A COLUMN OF M ARE ZERO, THEY
REMAIN ZERO IN Q.M.
THEREFORE ONLY THE SUBMATRICES A[I:I+1,LA:U] AND
B[I:I+1,LB:U] ARE CHANGED, WHERE Q.A AND Q.B ARE
OVERWRITTEN IN A RESP B.

EXAMPLE OF USE: THE PROCEDURE HSH2COL IS USED IN QZI AND QZIVAL,
(THIS SECTION).

SUBSECTION HSH3COL:

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" HSH3COL(LA,LB,U,I,A1,A2,A3,A,B);
"VALUE" LA,LB,U,I,A1,A2,A3;"INTEGER" LA,LB,I,U;"REAL" A1,A2,A3;
"ARRAY" A,B;
"CODE" 34606;

THE MEANING OF THE FORMAL PARAMETERS IS:
LA: <ARITHMETIC EXPRESSION>;
THE LOWERBOUND OF THE RUNNING COLUMN SUBSCRIPT OF A:
LB: <ARITHMETIC EXPRESSION>;
THE LOWERBOUND OF THE RUNNING COLUMN SUBSCRIPT OF B:
U: <ARITHMETIC EXPRESSION>;
THE UPPERBOUND OF THE RUNNING COLUMN SUBSCRIPT OF A AND B:
I: <ARITHMETIC EXPRESSION>;
THE LOWERBOUND OF THE RUNNING ROW SUBSCRIPT OF A AND B,
I+2 IS THE UPPERBOUND:
A1: <ARITHMETIC EXPRESSION>;
THE I-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED:
A2: <ARITHMETIC EXPRESSION>;
THE (I+1)-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED:
A3: <ARITHMETIC EXPRESSION>;
THE (I+2)-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED.
A: <ARRAY IDENTIFIER>;
"ARRAY" A[I:I+2,LA:U];
ENTRY: THE GIVEN MATRIX A;
EXIT: THE TRANSFORMED MATRIX Q.A (SEE BRIEF DESCRIPTION);
B: <ARRAY IDENTIFIER>;
"ARRAY" B[I:I+2,LB:U];
ENTRY: THE GIVEN MATRIX B;
EXIT: THE TRANSFORMED MATRIX Q.B (SEE BRIEF DESCRIPTION);

PROCEDURES USED: HSHVECMAT = CP 31070;

METHOD AND PERFORMANCE:
WHEN THE CALCULATED HOUSEHOLDER MATRIX Q PREMULTIPLIES A MATRIX
M, ONLY ROWS I, (I+1) AND (I+2) ARE CHANGED.
IF THE ELEMENTS I, I+1 AND I+2 ARE ZERO, THEN THEY REMAIN ZERO
IN Q.M.
THEREFORE ONLY THE SUBMATRICES A[I:I+2,LA:U] AND B[I:I+2,LB:U]
ARE CHANGED, WHERE Q.A AND Q.B ARE OVERWRITTEN IN A RESP B.

EXAMPLE OF USE: THE PROCEDURE HSH3COL IS USED IN QZI AND QZIVAL
(THIS SECTION).

SUBSECTION HSH2ROW3:

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE IS:
    "PROCEDURE" HSH2ROW3(L,UA,UB,UX,J,A1,A2,A,B,X);
    "VALUE" L,UA,UB,UX,J,A1,A2; "INTEGER" L,UA,UB,UX,J;
    "REAL" A1,A2;"ARRAY" A,B,X;
    "CODE" 34607;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    L:  <ARITHMETIC EXPRESSION>;
        THE LOWERBOUND OF THE RUNNING ROW SUBSCRIPT OF A,B AND X;
    UA: <ARITHMETIC EXPRESSION>;
        THE UPPERBOUND OF THE RUNNING ROW SUBSCRIPT OF A;
    UB: <ARITHMETIC EXPRESSION>;
        THE UPPERBOUND OF THE RUNNING ROW SUBSCRIPT OF B;
    UX: <ARITHMETIC EXPRESSION>;
        THE UPPERBOUND OF THE RUNNING ROW SUBSCRIPT OF X;
    J:  <ARITHMETIC EXPRESSION>;
        THE LOWERBOUND OF THE RUNNING COLUMN SUBSCRIPT OF A,B AND X:
        J+1 IS THE UPPERBOUND;
    A1: <ARITHMETIC EXPRESSION>;
        THE J-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED;
    A2: <ARITHMETIC EXPRESSION>;
        THE (J+1)-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED;
    A:  <ARRAY IDENTIFIER>;
        "ARRAY" A[L:UA,J:J+1];
        ENTRY: THE GIVEN MATRIX A;
        EXIT: THE TRANSFORMED MATRIX A.Z (SEE BRIEF DESCRIPTION);
    B:  <ARRAY IDENTIFIER>;
        "ARRAY" B[L:UB,J:J+1];
        ENTRY: THE GIVEN MATRIX B;
        EXIT: THE TRANSFORMED MATRIX B.Z (SEE BRIEF DESCRIPTION);
    X:  <ARRAY IDENTIFIER>;
        "ARRAY" X[L:UX,J:J+1];
        ENTRY: THE GIVEN MATRIX X;
        EXIT: THE TRANSFORMED MATRIX X.Z (SEE BRIEF DESCRIPTION);
```

PROCEDURES USED: HSHVECTAM = CP 31073;

METHOD AND PERFORMANCE:

WHEN THE CALCULATED HOUSEHOLDER MATRIX Z POSTMULTIPLIES A MATRIX M, ONLY COLUMNS J AND J+1 ARE CHANGED. IF THE ELEMENTS J AND J+1 IN A ROW OF M ARE ZERO, THEN THEY REMAIN ZERO IN M.Z THEREFORE ONLY THE SUBMATRICES A[L:UA,J:J+1],B[L:UB,J:J+1] AND X[L:UX,J:J+1] ARE CHANGED, WHERE A.Z, B.Z AND X.Z ARE OVERWRITTEN IN RESP. A,B AND X.

EXAMPLE OF USE: THE PROCEDURE HSH2ROW3 IS USED IN QZI (THIS SECTION).

SUBSECTION HSH2ROW2:

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"PROCEDURE HSH2ROW2(LA,LB,UA,UB,J,A1,A2,A,B); "VALUE" LA,LB,UA,
UB,J,A1,A2; "INTEGER" LA,LB,UA,UB,J; "REAL" A1,A2;"ARRAY" A,B;
"CODE" 34608;

THE MEANING OF THE FORMAL PARAMETERS IS:
LA: <ARITHMETIC EXPRESSION>;
THE LOWERBOUND OF THE RUNNING ROW SUBSCRIPT OF A;
LB: <ARITHMETIC EXPRESSION>;
THE LOWERBOUND OF THE RUNNING ROW SUBSCRIPT OF B;
UA: <ARITHMETIC EXPRESSION>;
THE UPPERBOUND OF THE RUNNING ROW SUBSCRIPT OF A;
UB: <ARITHMETIC EXPRESSION>;
THE UPPERBOUND OF THE RUNNING ROW SUBSCRIPT OF B;
J: <ARITHMETIC EXPRESSION>;
THE LOWERBOUND OF THE RUNNING COLUMN SUBSCRIPT OF A AND B;
J+1 IS THE UPPERBOUND;
A1: < ARITHMETIC EXPRESSION>;
THE J-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED;
A2: <ARITHMETIC EXPRESSION>;
THE (J+1)-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED;
A: <ARRAY IDENTIFIER>;
"REAL" "ARRAY" A[LA:UA,J:J+1];
ENTRY: THE GIVEN MATRIX A;
EXIT: THE TRANSFORMED MATRIX A.Z (SEE BRIEF DESCRIPTION);
B: <ARRAY IDENTIFIER>;
"REAL" "ARRAY" B[LB:UB,J:J+1];
ENTRY: THE GIVEN MATRIX B;
EXIT: THE TRANSFORMED MATRIX B.Z (SEE BRIEF DESCRIPTION);

PROCEDURES USED: HSHVECTAM = CP 31073;

METHOD AND PERFORMANCE:
SEE HSH2ROW3, BUT HERE THE MATRIX X HAS BEEN LEFT OUT.

EXAMPLE OF USE:
THE PROCEDURE HSH2ROW2 IS USED IN QZIVAL,
(THIS SECTION).

SUBSECTION: HSH3ROW3.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" HSH3ROW3(L,U,UX,J,A1,A2,A3,A,B,X);
"VALUE" L,U,UX,J,A1,A2,A3; "INTEGER" L,U,UX,J; "REAL" A1,A2,A3;
"ARRAY" A,B,X;
"CODE" 34609;

THE MEANING OF THE FORMAL PARAMETERS IS:
L: <ARITHMETIC EXPRESSION>;
THE LOWERBOUND OF THE RUNNING ROW SUBSCRIPT OF A AND B AND X;
U: <ARITHMETIC EXPRESSION>;
THE UPPERBOUND OF THE RUNNING ROW SUBSCRIPT OF A AND B;
UX: <ARITHMETIC EXPRESSION>;
THE UPPERBOUND OF THE RUNNING ROW SUBSCRIPT OF X;
J: <ARITHMETIC EXPRESSION>;
THE LOWERBOUND OF THE RUNNING COLUMN SUBSCRIPT OF A,B AND X;
J+2 IS THE UPPERBOUND;
A1: <ARITHMETIC EXPRESSION>;
THE J-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED;
A2: <ARITHMETIC EXPRESSION>;
THE (J+1)-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED;
A3: <ARITHMETIC EXPRESSION>;
THE (J+2)-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED;
A: <ARRAY IDENTIFIER>;
"REAL" "ARRAY" A[L:U,J:J+2];
ENTRY: THE GIVEN MATRIX A;
EXIT: THE TRANSFORMED MATRIX A.Z (SEE BRIEF DESCRIPTION);
B: <ARRAY IDENTIFIER>;
"REAL" "ARRAY" B[L:U,J:J+2];
ENTRY: THE GIVEN MATRIX B;
EXIT: THE TRANSFORMED MATRIX B.Z (SEE BRIEF DESCRIPTION);
X: <ARRAY IDENTIFIER>;
"REAL" "ARRAY" X[L:UX,J:J+2];
ENTRY: THE GIVEN MATRIX X;
EXIT: THE TRANSFORMED MATRIX X.Z (SEE BRIEF DESCRIPTION);

PROCEDURES USED: HSHVECTAM = CP 31073;

METHOD AND PERFORMANCE:
WHEN THE CALCULATED HOUSEHOLDER MATRIX Z POSTMULTIPLIES A MATRIX M,
ONLY COLUMNS J,J+1 AND J+2 ARE CHANGED.
IF THE ELEMENTS J, J+1 AND J+2 IN A ROW OF M ARE ZERO, THEN THEY
REMAIN ZERO IN M.Z.
THEREFORE ONLY THE SUBMATRICES A[L:U,J:J+2], B[L:U,J:J+2] AND
X[L:UX,J:J+2] ARE CHANGED, WHERE A.Z , B.Z AND X.Z ARE OVERWRITTEN
ON RESP. A,B AND X;

EXAMPLE OF USE: THE PROCEDURE HSH3ROW3 IS USED IN QZI (THIS SECTION).

SUBSECTION: HSH3ROW2.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE IS:
    "PROCEDURE" HSH3ROW2(LA,LB,U,J,A1,A2,A3,A,B);
    "VALUE" LA,LB,U,J,A1,A2,A3; "INTEGER" LA,LB,U,J;
    "REAL" A1,A2,A3; "ARRAY" A,B;
    "CODE" 34610;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    LA: <ARITHMETIC EXPRESSION>;
        THE LOWERBOUND OF THE RUNNING ROW SUBSCRIPT OF A;
    LB: <ARITHMETIC EXPRESSION>;
        THE LOWERBOUND OF THE RUNNING ROW SUBSCRIPT OF B;
    U:  <ARITHMETIC EXPRESSION>;
        THE UPPERBOUND OF THE RUNNING ROW SUBSCRIPT OF A AND B;
    J:  <ARITHMETIC EXPRESSION>;
        THE LOWERBOUND OF THE RUNNING COLUMN SUBSCRIPT OF A AND B,
        J+2 IS THE UPPERBOUND;
    A1: <ARITHMETIC EXPRESSION>;
        THE J-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED;
    A2: <ARITHMETIC EXPRESSION>;
        THE (J+1)-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED;
    A3: <ARITHMETIC EXPRESSION>;
        THE (J+2)-TH COMPONENT OF THE VECTOR TO BE TRANSFORMED;
    A:  <ARRAY IDENTIFIER>;
        "REAL" "ARRAY" A[LA:U,J:J+2];
        ENTRY: THE GIVEN MATRIX A;
        EXIT: THE TRANSFORMED MATRIX A.Z (SEE BRIEF DESCRIPTION);
    B:  <ARRAY IDENTIFIER>;
        "REAL" "ARRAY" B[LB:U,J:J+2];
        ENTRY: THE GIVEN MATRIX B;
        EXIT: THE TRANSFORMED MATRIX B.Z (SEE BRIEF DESCRIPTION);
```

PROCEDURES USED: HSHVECTAM = CP 31073;

METHOD AND PERFORMANCE:
SEE HSH3ROW3, BUT HERE THE MATRIX X HAS BEEN LEFT OUT.

EXAMPLE OF USE: HSH3ROW2 IS USED IN QZIVAL (THIS SECTION).

SOURCE TEXTS:

```
"CODE" 34600;
"PROCEDURE" QZIVAL(N,A,B,ALFR,ALFI,BETA,ITER,EM);
"VALUE" N;"INTEGER" N;"ARRAY" A,B,ALFR,ALFI,BETA,EM;
"INTEGER" "ARRAY" ITER;
"BEGIN" "REAL" DWARF,EPS,EPSA,EPSB;
"PROCEDURE" ELMCOL(L,U,I,J,A,B,X);"CODE" 34023;
"PROCEDURE" HSHDECMUL(N,A,B,DWARF);"CODE" 34602;
"PROCEDURE" HESTGL2(N,A,B);"CODE" 34604;
"PROCEDURE" HSH2ROW2(LA,LB,UA,UB,J,A1,A2,A,B);"CODE" 34608;
"PROCEDURE" HSH3ROW2(LA,LB,U,J,A1,A2,A3,A,B);"CODE" 34610;
"PROCEDURE" HSH2COL(LA,LB,U,I,A1,A2,A,B);"CODE" 34605;
"PROCEDURE" HSH3COL(LA,LB,U,I,A1,A2,A3,A,B);"CODE" 34606;
"PROCEDURE" CHSH2(A1R,A1I,A2R,A2I,C,SR,SI);"CODE" 34611;
"PROCEDURE" HSHVECMAT(LR,UR,LC,UC,X,U,A);"CODE" 31070;
"PROCEDURE" HSHVECTAM(LR,UR,LC,UC,X,U,A);"CODE" 31073;
"PROCEDURE" QZIT(N,A,B,EPS,EPSA,EPSB,ITER);"VALUE"N,EPS;
"REAL" EPS,EPSA,EPSB;"INTEGER" N;"INTEGER" "ARRAY" ITER;"ARRAY" A,B;
"BEGIN" "REAL" ANORM,BNORM,ANI,BNI,CONST,A10,A20,A30,B11,
    B22,B33,B44,A11,A12,A21,A22,A33,A34,A43,A44,B12,B34,OLD1,OLD2;
    "INTEGER" I,Q,M,M1,Q1,J,K,K1,K2,K3,KM1;"BOOLEAN" STATIONARY;
    ANORM:=BNORM:=0;"FOR" I:=1 "STEP" 1 "UNTIL" N "DO"
    "BEGIN" BNI:=0;ITER[I]:=0;ANI:="IF" I>1"THEN"ABS(A[I,I-1])"ELSE" 0;
        "FOR" J:=I "STEP" 1 "UNTIL" N "DO"
        "BEGIN" ANI:=ANI+ABS(A[I,J]);BNI:=BNI+ABS(B[I,J])
        "END";"IF" ANI>ANORM "THEN" ANORM:=ANI;"IF" BNI>BNORM"THEN"
        BNORM:=BNI
    "END";"IF" ANORM=0 "THEN" ANORM:=EPS;"IF"BNORM=0 "THEN" BNORM:=EPS;
    EPSA:=EPS*ANORM;EPSB:=EPS*BNORM;
    "FOR" M:=N,M "WHILE" M>=3 "DO"
  "BEGIN"
    "FOR" I:=M+1,I-1 "WHILE"("IF" I>1 "THEN"ABS(A[I,I-1])>EPSA "ELSE"
    "FALSE") "DO" Q:=I-1;
    "IF" Q>1 "THEN" A[Q,Q-1]:=0;
 L: "IF" Q>=M-1 "THEN" M:=Q-1 "ELSE"
    "BEGIN"
        "IF" ABS(B[Q,Q])<=EPSB "THEN"
        "BEGIN" B[Q,Q]:=0;Q1:=Q+1;
            HSH2COL(Q,Q,M,Q,A[Q,Q],A[Q1,Q],A,B);A[Q1,Q]:=0;
            Q:=Q1;"GOTO" L
        "END" "ELSE" M1:=M-1;Q1:=Q+1;CONST:=0.75;ITER[M]:=ITER[M]+1;
        STATIONARY:="IF" ITER[M]=1 "THEN" "TRUE" "ELSE"
        ABS(A[M,M-1])>=CONST*OLD1"AND"ABS(A[M-1,M-2])>=CONST*OLD2;
        "IF" ITER[M]>30"AND"STATIONARY "THEN"
        "BEGIN" "FOR" I:=1 "STEP" 1 "UNTIL" M "DO" ITER[I]:=-1;
            "GOTO" OUT
        "END";
        "IF" ITER[M]=10"AND"STATIONARY "THEN"
        "BEGIN" A10:=0;A20:=1;A30:=1.1605
        "END" "ELSE"
        "BEGIN" B11:=B[Q,Q];B22:="IF" ABS(B[Q1,Q1])<EPSB "THEN"EPSB
            "ELSE" B[Q1,Q1];
            B33:="IF" ABS(B[M1,M1])<EPSB "THEN" EPSB "ELSE"B[M1,M1];
                                                        "COMMENT"
```

```
            B44:="IF" ABS(B[M,M])<EPSB "THEN" EPSB "ELSE" B[M,M] ;
            A11:=A[Q,Q]/B11;A12:=A[Q,Q1]/B22;A21:=A[Q1,Q]/B11;
            A22:=A[Q1,Q1]/B22;A33:=A[M1,M1]/B33;A34:=A[M1,M]/B44;
            A43:=A[M,M1]/B33;A44:=A[M,M]/B44;B12:=B[Q,Q1]/B22;
            B34:=B[M1,M1/B44;
            A10:=((A33-A11)*(A44-A11)-A34*A43+A43*B34*A11)/A21
                 +A12-A11*B12;
            A20:=(A22-A11-A21*B12)-(A33-A11)-(A44-A11)+A43*B34;
            A30:=A[Q+2,Q1]/B22
        "END";OLD1:=ABS(A[M,M-1]);OLD2:=ABS(A[M-1,M-2]);
        "FOR" K:=Q "STEP" 1 "UNTIL" M1 "DO"
        "BEGIN" K1:=K+1;K2:=K+2;K3:="IF" K+3>M "THEN" M "ELSE" K+3;
            KM1:="IF" K=1<Q "THEN" Q "ELSE" K-1;
            "IF" K^=M1 "THEN"
            "BEGIN" "IF" K=Q "THEN"
                "BEGIN"
                 HSH3COL(KM1,KM1,M,K,A[K,KM1],A[K1,KM1],A[K2,KM1],A,B);
                  A[K1,KM1]:=A[K2,KM1]:=0
                "END";
                HSH3ROW2(Q,Q,K3,K,B[K2,K2],B[K2,K1],B[K2,K],A,B);
                B[K2,K]:=B[K2,K1]:=0 ;
            "END" "ELSE"
            "BEGIN" HSH2COL(KM1,KM1,M,K,A[K,KM1],A[K1,KM1],A,B);
                A[K1,KM1]:=0
            "END";
            HSH2ROW2(Q,Q,K3,K3,K,B[K1,K1],B[K1,K],A,B);B[K1,K]:=0
        "END"
    "END";
    OUT:
  "END"
"END" QZIT;
        "EOP"
"PROCEDURE" QZVAL(N,A,B,EPSA,EPSB,ALFR,ALFI,BETA);"VALUE" N;
"REAL" EPSA,EPSB;"INTEGER" N;"ARRAY" ALFR,ALFI,BETA,A,B;
"BEGIN" "INTEGER" M,L,J;"REAL" AN,BN,A11,A12,A21,A22,B11,B12,B22,E,C,D,
    ER,EI,A11R,A11I,A12R,A12I,A21R,A21I,A22R,A22I,CZ,SZR,SZI,
    CQ,SQR,SQI,SSR,SSI,TR,TI,BDR,BDI,R;
    "FOR" M:=N,M "WHILE" M>0 "DO"
    "IF"("IF" M>1 "THEN" A[M,M-1]=0 "ELSE" "TRUE") "THEN"
    "BEGIN" ALFR[M]:=A[M,M];BETA[M]:=B[M,M];ALFI[M]:=0;M:=M-1
    "END" "ELSE"
    "BEGIN" L:=M-1;"IF" ABS(B[L,L])<=EPSB "THEN"
        "BEGIN" B[L,L]:=0;HSH2COL(L,L,N,L,A[L,L],A[M,L],A,B);
            A[M,L]:=B[M,L]:=0;ALFR[L]:=A[L,L];ALFR[M]:=A[M,M];
            BETA[L]:=B[L,L];BETA[M]:=B[M,M];ALFI[M]:=ALFI[L]:=0;
        "END" "ELSE" "IF" ABS(B[M,M])<=EPSB "THEN"
        "BEGIN"B[M,M]:=0;HSH2ROW2(1,1,M,M,L,A[M,M],A[M,L],A,B);
            A[M,L]:=B[M,L]:=0;ALFR[L]:=A[L,L];ALFR[M]:=A[M,M];
            BETA[L]:=B[L,L];BETA[M]:=B[M,M];ALFI[M]:=ALFI[L]:=0;
        "END""ELSE"
      "BEGIN"
        AN:=ABS(A[L,L])+ABS(A[L,M])+ABS(A[M,L])+ABS(A[M,M]);
        BN:=ABS(B[L,L])+ABS(B[L,M])+ABS(B[M,M]);
        A11:=A[L,L]/AN;A12:=A[L,M]/AN;A21:=A[M,L]/AN;A22:=A[M,M]/AN;
                                                          "COMMENT"
```

```
        B11:=B[L,L]/BN;B12:=B[L,M]/BN;B22:=B[M,M]/BN;
        E:=A11/B11;
        C:=((A22-E*B22)/B22-(A21*B12)/(B11*B22))/2;
        D:=C*C+(A21*(A12-E*B12))/(B11*B22);
        "IF" D>=0 "THEN"
        "BEGIN"E:=E+("IF"C<0"THEN"C-SQRT(D)"ELSE"C+SQRT(D));
            A11:=A11-E*B11;A12:=A12-E*B12;A22:=A22-E*B22;
            "IF" ABS(A11)+ABS(A12)>=ABS(A21)+ABS(A22) "THEN"
            HSH2ROW2(1,1,M,M,L,A12,A11,A,B)"ELSE"
            HSH2ROW2(1,1,M,M,L,A22,A21,A,B);
            "IF"AN>=ABS(E)*BN"THEN"
            HSH2COL(L,L,N,L,B[L,L],B[M,L],A,B) "ELSE"
            HSH2COL(L,L,N,L,A[L,L],A[M,L],A,B);
            A[M,L]:=B[M,L]:=0;
            ALFR[L]:=A[L,L];ALFR[M]:=A[M,M];BETA[L]:=B[L,L];
            BETA[M]:=B[M,M];ALFI[M]:=ALFI[L]:=0;
        "END""ELSE"
        "BEGIN"
            ER:=E+C;EI:=SQRT(-D);A11R:=A11-ER*B11;A11I:=EI*B11;
            A12R:=A12-ER*B12;A12I:=EI*B12;A21R:=A21;A21I:=0;
            A22R:=A22-ER*B22;A22I:=EI*B22;
            "IF"ABS(A11R)+ABS(A11I)+ABS(A12R)+ABS(A12I)>=
            ABS(A21R)+ABS(A22R)+ABS(A22I)"THEN"
            CHSH2(A12R,A12I,-A11R,-A11I,CZ,SZR,SZI)"ELSE"
            CHSH2(A22R,A22I,-A21R,-A21I,CZ,SZR,SZI);
            "IF"AN>=(ABS(ER)+ABS(EI))*BN"THEN"
            CHSH2(CZ*B11+SZR*B12,SZI*B12,SZR*B22,SZI*B22,CQ,SQR,SQI)
            "ELSE"CHSH2(CZ*A11+SZR*A12,SZI*A12,CZ*A21+SZR*A22,SZI*A22,
            CQ,SQR,SQI);SSR:=SQR*SZR+SQI*SZI;SSI:=SQR*SZI-SQI*SZR;
            TR:=CQ*CZ*A11+CQ*SZR*A12+SQR*CZ*A21+SSR*A22;
            TI:=CQ*SZI*A12-SQI*CZ*A21+SSI*A22;
            BDR:=CQ*CZ*B11+CQ*SZR*B12+SSR*B22;
            BDI:=CQ*SZI*B12+SSI*B22;
            R:=SQRT(BDR*BDR+BDI*BDI);BETA[L]:=BN*R;
            ALFR[L]:=AN*(TR*BDR+TI*BDI)/R;
            ALFI[L]:=AN*(TR*BDI-TI*BDR)/R;
            TR:=SSR*A11-SQR*CZ*A12-CQ*SZR*A21+CQ*CZ*A22;
            TI:=-SSI*A11-SQI*CZ*A12+CQ*SZI*A21;
            BDR:=SSR*B11-SQR*CZ*B12+CQ*CZ*B22;
            BDI:=-SSI*B11-SQI*CZ*B12;
            R:=SQRT(BDR*BDR+BDI*BDI);BETA[M]:=BN*R;
            ALFR[M]:=AN*(TR*BDR+TI*BDI)/R;
            ALFI[M]:=AN*(TR*BDI-TI*BDR)/R;
        "END"
      "END";M:=M-2
    "END"
"END" QZVAL;
DWARF:=EM[0];EPS:=EM[1];
HSHDECMUL(N,A,B,DWARF);
HESTGL2(N,A,B);
QZIT(N,A,B,EPS,EPSA,EPSB,ITER);
QZVAL(N,A,B,EPSA,EPSB,ALFR,ALFI,BETA);
"END" QZIVAL;
        "EOP"
```

```
"CODE" 34601;
"PROCEDURE" QZI(N,A,B,X,ALFR,ALFI,BETA,ITER,EM);
"VALUE" N;"INTEGER" N;"ARRAY"A,B,X,ALFR,ALFI,BETA,EM;
"INTEGER" "ARRAY" ITER;
"BEGIN" "REAL" DWARF,EPS,EPSA,EPSB;
"REAL" "PROCEDURE" MATMAT(L,U,I,J,A,B);"CODE" 34013;
"PROCEDURE" HSHDECMUL(N,A,B,DWARF);"CODE" 34602;
"PROCEDURE" HESTGL3(N,A,B,X);"CODE" 34603;
"PROCEDURE" HSH2ROW3(L,UA,UB,UX,J,A1,A2,A,B,X);"CODE" 34607;
"PROCEDURE" HSH3ROW3(L,U,UX,J,A1,A2,A3,A,B,X);"CODE" 34609;
"PROCEDURE" HSH2COL(LA,LB,U,I,A1,A2,A,B);"CODE" 34605;
"PROCEDURE" HSH3COL(LA,LB,U,I,A1,A2,A3,A,B);"CODE" 34606;
"PROCEDURE" CHSH2(A1R,A1I,A2R,A2I,C,SR,SI);"CODE" 34611;
"PROCEDURE" COMDIV(XR,XI,YR,YI,ZR,ZI);"CODE" 34342;
"PROCEDURE" QZIT(N,A,B,X,EPS,EPSA,EPSB,ITER);"VALUE"N,EPS;
"REAL" EPS,EPSA,EPSB;"INTEGER" N;"INTEGER" "ARRAY" ITER;"ARRAY" A,B,X;
"BEGIN" "REAL" ANORM,BNORM,ANI,BNI,CONST,A10,A20,A30,B11,
    B22,B33,B44,A11,A12,A21,A22,A33,A34,A43,A44,B12,B34,OLD1,OLD2;
    "INTEGER" I,Q,M,M1,Q1,J,K,K1,K2,K3,KM1;"BOOLEAN" STATIONARY;
    ANORM:=BNORM:=0;"FOR" I:=1 "STEP" 1 "UNTIL" N "DO"
    "BEGIN" BNI:=0;ITER[I]:=0;ANI:="IF" I>1"THEN"ABS(A[I,I-1])"ELSE" 0;
        "FOR" J:=I "STEP" 1 "UNTIL" N "DO"
        "BEGIN" ANI:=ANI+ABS(A[I,J]);BNI:=BNI+ABS(B[I,J])
        "END";"IF" ANI>ANORM "THEN" ANORM:=ANI;"IF" BNI>BNORM"THEN"
        BNORM:=BNI
    "END";"IF" ANORM=0 "THEN" ANORM:=EPS;"IF"BNORM=0 "THEN" BNORM:=EPS;
    EPSA:=EPS*ANORM;EPSB:=EPS*BNORM;
    "FOR" M:=N,M "WHILE" M>=3 "DO"
  "BEGIN"
    "FOR" I:=M+1,I-1 "WHILE"("IF" I>1 "THEN"ABS(A[I,I-1])>EPSA "ELSE"
    "FALSE") "DO" Q:=I-1;
    "IF" Q>1 "THEN" A[Q,Q-1]:=0;
 L: "IF" Q>=M-1 "THEN" M:=Q-1 "ELSE"
    "BEGIN"
        "IF" ABS(B[Q,Q])<=EPSB "THEN"
        "BEGIN" B[Q,Q]:=0;Q1:=Q+1;
            HSH2COL(Q,Q,N,Q,A[Q,Q],A[Q1,Q],A,B);A[Q1,Q]:=0;
            Q:=Q1;"GOTO" L
        "END" "ELSE" M1:=M-1;Q1:=Q+1;CONST:=0.75;ITER[M]:=ITER[M]+1;
        STATIONARY:="IF" ITER[M]=1 "THEN" "TRUE" "ELSE"
        ABS(A[M,M-1])>=CONST*OLD1"AND"ABS(A[M-1,M-2])>=CONST*OLD2;
        "IF" ITER[M]>30"AND"STATIONARY "THEN"
        "BEGIN" "FOR" I:=1 "STEP" 1 "UNTIL" M "DO" ITER[I]:=-1;
            "GOTO" OUT
        "END";
        "IF" ITER[M]=10"AND"STATIONARY "THEN"
        "BEGIN" A10:=0;A20:=1;A30:=1.1605
        "END" "ELSE"
        "BEGIN" B11:=B[Q,Q];B22:="IF" ABS(B[Q1,Q1])<EPSB "THEN"EPSB
            "ELSE" B[Q1,Q1];
                                                    "COMMENT"
```

```
            B33:="IF" ABS(B[M1,M1])<EPSB "THEN" EPSB "ELSE"B[M1,M1];
            B44:="IF" ABS(B[M,M])<EPSB "THEN" EPSB "ELSE" B[M,M] ;
            A11:=A[Q,Q1/B11;A12:=A[Q,Q1]/B22;A21:=A[Q1,Q1/B11;
            A22:=A[Q1,Q1]/B22;A33:=A[M1,M1]/B33;A34:=A[M1,M]/B44;
            A43:=A[M,M1]/B33;A44:=A[M,M]/B44;B12:=B[Q,Q1]/B22;
            B34:=B[M1,M]/B44;
            A10:=((A33-A11)*(A44-A11)-A34*A43+A43*B34*A11)/A21
                 +A12-A11*B12;
            A20:=(A22-A11-A21*B12)-(A33-A11)-(A44-A11)+A43*B34;
            A30:=A[Q+2,Q1]/B22
        "END";OLD1:=ABS(A[M,M-1]);OLD2:=ABS(A[M-1,M-2]);
        "FOR" K:=Q "STEP" 1 "UNTIL" M1 "DO"
        "BEGIN" K1:=K+1;K2:=K+2;K3:="IF" K+3>M "THEN" M "ELSE" K+3;
            KM1:="IF" K-1<Q "THEN" Q "ELSE" K-1;
            "IF" K^=M1 "THEN"
            "BEGIN" "IF" K=Q "THEN"
                HSH3COL(KM1,KM1,N,K,A10,A20,A30,A,B) "ELSE"
                "BEGIN"
                 HSH3COL(KM1,KM1,N,K,A[K,KM1],A[K1,KM1],A[K2,KM1],A,B);
                  A[K1,KM1]:=A[K2,KM1]:=0
                "END";
                HSH3ROW3(1,K3,N,K,B[K2,K2],B[K2,K1],B[K2,K],A,B,X);
                B[K2,K]:=B[K2,K1]:=0 ;
            "END" "ELSE"
            "BEGIN" HSH2COL(KM1,KM1,N,K,A[K,KM1],A[K1,KM1],A,B);
                A[K1,KM1]:=0
            "END";
            HSH2ROW3(1,K3,K3,N,K,B[K1,K1],B[K1,K],A,B,X);B[K1,K]:=0
        "END"
    "END"
  "END"; OUT;
"END" QZIT;
"PROCEDURE" QZVAL(N,A,B,X,EPSA,EPSB,ALFR,ALFI,BETA);"VALUE" N;
"REAL" EPSA,EPSB;"INTEGER" N;"ARRAY" ALFR,ALFI,BETA,A,B,X;
"BEGIN" "INTEGER" M,L,J;"REAL" AN,BN,A11,A12,A21,A22,B11,B12,B22,E,C,D,
    ER,EI,A11R,A11I,A12R,A12I,A21R,A21I,A22R,A22I,CZ,SZR,SZI,
    CQ,SQR,SQI,SSR,SSI,TR,TI,BDR,BDI,R;
    "FOR" M:=N,M "WHILE" M>0 "DO"
    "IF"("IF" M>1 "THEN" A[M,M-1]=0 "ELSE" "TRUE")"THEN"
    "BEGIN" ALFR[M]:=A[M,M];BETA[M]:=B[M,M];ALFI[M]:=0;M:=M-1
    "END" "ELSE"
    "BEGIN" L:=M-1;"IF" ABS(B[L,L])<=EPSB "THEN"
        "BEGIN" B[L,L]:=0;HSH2COL(L,L,N,L,A[L,L],A[M,L],A,B);
            A[M,L]:=B[M,L]:=0;ALFR[L]:=A[L,L];ALFR[M]:=A[M,M];
            BETA[L]:=B[L,L];BETA[M]:=B[M,M];ALFI[M]:=ALFI[L]:=0;
        "END" "ELSE" "IF" ABS(B[M,M])<=EPSB "THEN"
        "BEGIN"B[M,M]:=0;HSH2ROW3(1,M,M,N,L,A[M,M],A[M,L],A,B,X);
            A[M,L]:=B[M,L]:=0;ALFR[L]:=A[L,L];ALFR[M]:=A[M,M];
            BETA[L]:=B[L,L];BETA[M]:=B[M,M];ALFI[M]:=ALFI[L]:=0;
        "END""ELSE"
      "BEGIN"
        AN:=ABS(A[L,L])+ABS(A[L,M])+ABS(A[M,L])+ABS(A[M,M]);
                                                        "COMMENT"
```

```
        BN:=ABS(B[L,L])+ABS(B[L,M])+ABS(B[M,M]);
        A11:=A[L,L]/AN;A12:=A[L,M]/AN;A21:=A[M,L]/AN;A22:=A[M,M]/AN;
        B11:=B[L,L]/BN;B12:=B[L,M]/BN;B22:=B[M,M]/BN;
        E:=A11/B11;
        C:=((A22-E*B22)/B22-(A21*B12)/(B11*B22))/2;
        D:=C*C+(A21*(A12-E*B12))/(B11*B22);
        "IF" D>=0 "THEN"
        "BEGIN"E:=E+("IF"C<0"THEN"C-SQRT(D)"ELSE"C+SQRT(D));
            A11:=A11-E*B11;A12:=A12-E*B12;A22:=A22-E*B22;
            "IF" ABS(A11)+ABS(A12)>=ABS(A21)+ABS(A22) "THEN"
            HSH2ROW3(1,M,M,N,L,A12,A11,A,B,X)"ELSE"
            HSH2ROW3(1,M,M,N,L,A22,A21,A,B,X);
            "IF"AN>=ABS(E)*BN"THEN"
            HSH2COL(L,L,N,L,B[L,L],B[M,L],A,B) "ELSE"
            HSH2COL(L,L,N,L,A[L,L],A[M,L],A,B);
            A[M,L]:=B[M,L]:=0;
            ALFR[L]:=A[L,L];ALFR[M]:=A[M,M];BETA[L]:=B[L,L];
            BETA[M]:=B[M,M];ALFI[M]:=ALFI[L]:=0;
        "END""ELSE"
        "BEGIN"
            ER:=E+C;EI:=SQRT(-D);A11R:=A11-ER*B11;A11I:=EI*B11;
            A12R:=A12-FR*B12;A12I:=EI*B12;A21R:=A21;A21I:=0;
            A22R:=A22-ER*B22;A22I:=EI*B22;
            "IF"ABS(A11R)+ABS(A11I)+ABS(A12R)+ABS(A12I)>=
            ABS(A21R)+ABS(A22R)+ABS(A22I)"THEN"
            CHSH2(A12R,A12I,-A11R,-A11I,CZ,SZR,SZI)"ELSE"
            CHSH2(A22R,A22I,-A21R,-A21I,CZ,SZR,SZI);
            "IF"AN>=(ABS(ER)+ABS(EI))*BN"THEN"
            CHSH2(CZ*B11+SZR*B12,SZI*B12,SZR*B22,SZI*B22,CQ,SQR,SQI)
            "ELSE"CHSH2(CZ*A11+SZR*A12,SZI*A12,CZ*A21+SZR*A22,SZI*A22,
            CQ,SQR,SQI);SSR:=SQR*SZR+SQI*SZI;SSI:=SQR*SZI-SQI*SZR;
            TR:=CQ*CZ*A11+CQ*SZR*A12+SQR*CZ*A21+SSR*A22;
            TI:=CQ*SZI*A12-SQI*CZ*A21+SSI*A22;
            BDR:=CQ*CZ*B11+CQ*SZR*B12+SSR*B22;
            BDI:=CQ*SZI*B12+SSI*B22;
            R:=SQRT(BDR*BDR+BDI*BDI);BETA[L]:=BN*R;
            ALFR[L]:=AN*(TR*BDR+TI*BDI)/R;
            ALFI[L]:=AN*(TR*BDI-TI*BDR)/R;
            TR:=SSR*A11-SQR*CZ*A12-CQ*SZR*A21+CQ*CZ*A22;
            TI:=-SSI*A11-SQI*CZ*A12+CQ*SZI*A21;
            BDR:=SSR*B11-SQR*CZ*B12+CQ*CZ*B22;
            BDI:=-SSI*B11-SQI*CZ*B12;
            R:=SQRT(BDR*BDR+BDI*BDI);BETA[M]:=BN*R;
            ALFR[M]:=AN*(TR*BDR+TI*BDI)/R;
            ALFI[M]:=AN*(TR*BDI-TI*BDR)/R;
        "END"
      "END";M:=M-2
    "END"
"END" QZVAL
        "EOP"
```

```
"PROCEDURE" QZVEC(N,A,B,X,EPSA,EPSB,ALFR,ALFI,BETA);"VALUE"N,EPSA,EPSB;
"INTEGER" N;"REAL" EPSA,EPSB;"ARRAY" A,B,ALFR,ALFI,BETA,X;
"BEGIN""INTEGER" M,MR,MI,L,L1,J,K;"REAL" BETM,ALFM,SL,SK,D,TKK,TKL,TLK,
    TLL,ALMI,ALMR,TR,TI,SLR,SLI,SKR,SKI,DR,DI,TKKR,TKKI,TKLR,TKLI,TLKR,
    TLKI,TLLR,TLLI,S,R;
    "FOR" M:=N "STEP" -1 "UNTIL" 1 "DO"
    "IF" ALFI[M]=0 "THEN"
    "BEGIN" "COMMENT" M-TH REAL VECTOR;
        ALFM:=ALFR[M];BETM:=BETA[M];B[M,M]:=1;L1:=M;
        "FOR" L:=M-1 "STEP" -1 "UNTIL" 1 "DO"
        "BEGIN" SL:=0;
            "FOR" J:=L1 "STEP" 1 "UNTIL" M "DO"
            SL:=SL+(BETM*A[L,J]-ALFM*B[L,J])*B[J,M];
            "IF"("IF"L^=1"THEN"BETM*A[L,L-1]=0"ELSE""TRUE")"THEN"
            "BEGIN""COMMENT" 1-1 BLOCK;
                D:=BETM*A[L,L]-ALFM*B[L,L];
                "IF" D=0 "THEN" D:=(EPSA+EPSB)/2;B[L,M]:=-SL/D
            "END""ELSE"
            "BEGIN" "COMMENT" 2-2 BLOCK;K:=L-1;SK:=0;
                "FOR"J:=L1 "STEP" 1 "UNTIL" M "DO"
                SK:=SK+(BETM*A[K,J]-ALFM*B[K,J])*B[J,M];
                TKK:=BETM*A[K,K]-ALFM*B[K,K];
                TKL:=BETM*A[K,L]-ALFM*B[K,L];
                TLK:=BETM*A[L,K];
                TLL:=BETM*A[L,L]-ALFM*B[L,L];
                D:=TKK*TLL-TKL*TLK;"IF" D=0 "THEN" D:=(EPSA+EPSB)/2;
                B[L,M]:=(TLK*SK-TKK*SL)/D;
               B[K,M]:="IF"ABS(TKK)>=ABS(TLK)"THEN"-(SK+TKL*B[L,M])/TKK
                "ELSE" -(SL+TLL*B[L,M])/TLK;L:=L-1
            "END";L1:=L
        "END"
    "END""ELSE"
    "BEGIN" "COMMENT" COMPLEX VECTOR;
        ALMR:=ALFR[M-1];ALMI:=ALFI[M-1];BETM:=BETA[M-1];MR:=M-1;MI:=M;
        B[M-1,MR]:=ALMI*B[M,M]/(BETM*A[M,M-1]);
        B[M-1,MI]:=(BETM*A[M,M]-ALMR*B[M,M])/(BETM*A[M,M-1]);
        B[M,MR]:=0;B[M,MI]:=-1;L1:=M-1;
        "FOR" L:=M-2 "STEP" -1 "UNTIL" 1 "DO"
        "BEGIN" SLR:=SLI:=0;
            "FOR" J:=L1 "STEP" 1 "UNTIL" M "DO"
            "BEGIN" TR:=BETM*A[L,J]-ALMR*B[L,J];
                TI:=-ALMI*B[L,J];
                SLR:=SLR+TR*B[J,MR]-TI*B[J,MI];
                SLI:=SLI+TR*B[J,MI]+TI*B[J,MR]
            "END";
                                                        "COMMENT"
```

```
            "IF"("IF"L^=1"THEN"BETM*A[L,L-1]=0"ELSE""TRUE")"THEN"
            "BEGIN"DR:=BETM*A[L,L]-ALMR*B[L,L];
                DI:=-ALMI*B[L,L];
                COMDIV(-SLR,-SLI,DR,DI,B[L,MR],B[L,MI]);
            "END""ELSE"
            "BEGIN" K:=L-1;SKR:=SKI:=0;
                "FOR" J:=L1 "STEP" 1 "UNTIL" M "DO"
                "BEGIN" TR:=BETM*A[K,J]-ALMR*B[K,J];
                    TI:=-ALMI*B[K,J];
                    SKR:=SKR+TR*B[J,MR]-TI*B[J,MI];
                    SKI:=SKI+TR*B[J,MI]+TI*B[J,MR]
                "END";
                TKKR:=BETM*A[K,K]-ALMR*B[K,K];
                TKKI:=-ALMI*B[K,K];
                TKLR:=BETM*A[K,L]-ALMR*B[K,L];
                TKLI:=-ALMI*B[K,L];
                TLKR:=BETM*A[L,K];TLKI:=0;
                TLLR:=BETM*A[L,L]-ALMR*B[L,L];
                TLLI:=-ALMI*B[L,L];
                DR:=TKKR*TLLR-TKKI*TLLI-TKLR*TLKR;
                DI:=TKKR*TLLI+TKKI*TLLR-TKLI*TLKR;
                "IF"DR=0"AND"DI=0"THEN"DR:=(EPSA+EPSB)/2;
                COMDIV(TLKR*SKR-TKKR*SLR+TKKI*SLI,TLKR*SKI-TKKR*SLI-
                TKKI*SLR,DR,DI,B[L,MR],B[L,MI]);
                "IF"ABS(TKKR)+ABS(TKKI)>=ABS(TLKR)"THEN"
                COMDIV(-SKR-TKLR*B[L,MR]+TKLI*B[L,MI],-SKI-TKLR*B[L,MI]
                -TKLI*B[L,MR],TKKR,TKKI,B[K,MR],B[K,MI])"ELSE"
                COMDIV(-SLR-TLLR*B[L,MR]+TLLI*B[L,MI],-SLI-TLLR*B[L,MI]
                -TLLI*B[L,MR],TLKR,TLKI,B[K,MR],B[K,MI]);L:=L-1
            "END";L1:=L
        "END";M:=M-1
    "END";
    "FOR" M:=N "STEP" -1 "UNTIL" 1 "DO"
    "FOR" K:=1 "STEP" 1 "UNTIL" N "DO"
    X[K,M]:=MATMAT(1,M,K,M,X,B);
    "FOR" M:=N "STEP" -1 "UNTIL" 1 "DO"
    "BEGIN" S:=0;"IF" ALFI[M]=0 "THEN"
        "BEGIN" "FOR" K:=1 "STEP" 1 "UNTIL" N "DO"
            "BEGIN" R:=ABS(X[K,M]);
                "IF" R>=S "THEN""BEGIN" S:=R;D:=X[K,M] "END"
            "END";"FOR" K:=1 "STEP" 1 "UNTIL" N "DO"
            X[K,M]:=X[K,M]/D
        "END""ELSE"
        "BEGIN" "FOR" K:=1 "STEP" 1 "UNTIL" N "DO"
            "BEGIN" R:=ABS(X[K,M-1])+ABS(X[K,M]);
                R:=R*SQRT((X[K,M-1]/R)**2+(X[K,M]/R)**2);
                "IF" R>=S"THEN"
                "BEGIN" S:=R;DR:=X[K,M-1];DI:=X[K,M] "END"
            "END";"FOR" K:=1 "STEP" 1 "UNTIL" N "DO"
            COMDIV(X[K,M-1],X[K,M],DR,DI,X[K,M-1],X[K,M]);M:=M-1
        "END"
    "END"
"END" QZVEC;
                                                        "COMMENT"
```

```
DWARF:=EM[0];EPS:=EM[1];
HSHDECMUL(N,A,B,DWARF);
HESTGL3(N,A,B,X);
QZIT(N,A,B,X,EPS,EPSA,EPSB,ITER);
QZVAL(N,A,B,X,EPSA,EPSB,ALFR,ALFI,BETA);
QZVEC(N,A,B,X,EPSA,EPSB,ALFR,ALFI,BETA)
"END" QZI;
          "EOP"



"CODE" 34602;
"PROCEDURE" HSHDECMUL(N,A,B,DWARF);"VALUE"N,DWARF;"INTEGER"N;
"REAL" DWARF;"ARRAY" A,B;
"BEGIN"  "ARRAY" V[1:N];"INTEGER" J,K,K1,N1;"REAL" R,T,C;
    "REAL" "PROCEDURE" TAMMAT(L,U,I,J,A,B);"CODE" 34014;
    "PROCEDURE" HSHVECMAT(LR,UR,LC,UC,X,U,A);"CODE" 31070;
    K:=1;N1:=N+1;
    "FOR" K1:=2 "STEP" 1 "UNTIL" N1 "DO"
    "BEGIN" R:=TAMMAT(K1,N,K,K,B,B);
        "IF" R>DWARF "THEN"
        "BEGIN" R:="IF" B[K,K]<0 "THEN" -SQRT(R+B[K,K]*B[K,K])
            "ELSE" SQRT(R+B[K,K]*B[K,K]);T:=B[K,K]+R;C:=-T/R;
            B[K,K]:=-R;V[K]:=1;
            "FOR" J:=K1 "STEP" 1 "UNTIL" N "DO" V[J]:=B[J,K]/T;
            HSHVECMAT(K,N,K1,N,C,V,B);HSHVECMAT(K,N,1,N,C,V,A)
        "END";K:=K1
    "END"
"END" HSHDECMUL;
          "EOP"


"CODE" 34603;
"PROCEDURE" HESTGL3(N,A,B,X);"VALUE" N;"INTEGER" N;"ARRAY" A,B,X;
"BEGIN" "INTEGER" NM1,K,L,K1,L1;
    "PROCEDURE" HSH2COL(LA,LB,U,I,A1,A2,A,B);"CODE" 34605;
    "PROCEDURE" HSH2ROW3(L,UA,UB,UX,J,A1,A2,A,B,X);"CODE" 34607;
    "IF" N>2 "THEN"
    "BEGIN" "FOR" K:=2 "STEP" 1 "UNTIL" N "DO"
        "FOR" L:=1 "STEP" 1 "UNTIL" K-1 "DO" B[K,L]:=0;
        NM1:=N-1;K:=1;
        "FOR" K1:= 2 "STEP" 1 "UNTIL" NM1 "DO"
        "BEGIN" L1:=N;
            "FOR" L:=N-1 "STEP" -1 "UNTIL" K1 "DO"
            "BEGIN"
                HSH2COL(K,L,N,L,A[L,K],A[L1,K],A,B);A[L1,K]:=0;
                HSH2ROW3(1,N,L1,N,L,B[L1,L1],B[L1,L],A,B,X);
                B[L1,L]:=0;L1:=L
            "END";K:=K1
        "END"
    "END"
"END" HESTGL3;
          "EOP"
```

```
"CODE" 34604;
"PROCEDURE" HESTGL2(N,A,B);"VALUE" N;"INTEGER" N;"ARRAY" A,B;
"BEGIN" "INTEGER" NM1,K,L,K1,L1;
    "PROCEDURE" HSH2COL(LA,LB,U,I,A1,A2,A,B);"CODE" 34605;
    "PROCEDURE" HSH2ROW2(LA,LB,UA,UB,A1,A2,A,B);"CODE" 34608;
    "IF" N>2 "THEN"
    "BEGIN" "FOR" K:=2 "STEP" 1 "UNTIL" N "DO"
        "FOR" L:=1 "STEP" 1 "UNTIL" K-1 "DO" B[K,L]:=0;
        NM1:=N-1;K:=1;
        "FOR" K1:= 2 "STEP" 1 "UNTIL" NM1 "DO"
        "BEGIN" L1:=N;
            "FOR" L:=N-1 "STEP" -1 "UNTIL" K1 "DO"
            "BEGIN"
                HSH2COL(K,L,N,L,A[L,K],A[L1,K],A,B);A[L1,K]:=0;
                HSH2ROW2(1,1,N,L1,L,B[L1,L1],B[L1,L],A,B);
                B[L1,L]:=0;L1:=L
            "END";K:=K1
        "END"
    "END"
"END" HESTGL2;
          "EOP"


"CODE" 34605;
"PROCEDURE" HSH2COL(LA,LB,U,I,A1,A2,A,B);"VALUE" LA,LB,U,I,A1,A2;
"INTEGER" LA,LB,U,I;"REAL" A1,A2;"ARRAY" A,B;
"IF"A2^=0 "THEN"
"BEGIN" "REAL" R,T,C;"ARRAY" V[I:I+1];
    "PROCEDURE" HSHVECMAT(LR,UR,LC,UC,X,U,A);"CODE" 31070;
    R:="IF" A1<0 "THEN" -SQRT(A1*A1+A2*A2) "ELSE" SQRT(A1*A1+A2*A2);
    T:=A1+R;C:=-T/R;V[I]:=1;V[I+1]:=A2/T;
    HSHVECMAT(I,I+1,LA,U,C,V,A);HSHVECMAT(I,I+1,LB,U,C,V,B)
"END" HSH2COL;
         "EOP"


"CODE" 34606;
"PROCEDURE" HSH3COL(LA,LB,U,I,A1,A2,A3,A,B);
"VALUE"LA,LB,U,I,A1,A2,A3;"INTEGER"LA,LB,I,U;"REAL"A1,A2,A3;"ARRAY"A,B;
"IF" A2^=0 "OR" A3^=0 "THEN"
"BEGIN" "REAL" R,T,C;"ARRAY" V[I:I+2];
    "PROCEDURE" HSHVECMAT(LR,UR,LC,UC,X,U,A);"CODE" 31070;
    R:="IF" A1<0 "THEN" -SQRT(A1*A1+A2*A2+A3*A3)
    "ELSE" SQRT(A1*A1+A2*A2+A3*A3);
    T:=A1+R;C:=-T/R;V[I]:=1;V[I+1]:=A2/T;V[I+2]:=A3/T;
    HSHVECMAT(I,I+2,LA,U,C,V,A);HSHVECMAT(I,I+2,LB,U,C,V,B)
"END" HSH3COL;
         "EOP"
```

```
"CODE" 34607;
"PROCEDURE" HSH2ROW3(L,UA,UB,UX,J,A1,A2,A,B,X);"VALUE" L,UA,UB,UX,
J,A1,A2;"INTEGER" L,UA,UB,UX,J;"REAL" A1,A2;"ARRAY" A,B,X;
"IF"A2^=0 "THEN"
"BEGIN""REAL" R,T,C;"INTEGER" K;"ARRAY" V[J:J+1];
    "PROCEDURE" HSHVECTAM(LR,UR,LC,UC,X,U,A);"CODE" 31073;
    R:="IF" A1<0 "THEN" -SQRT(A1*A1+A2*A2) "ELSE" SQRT(A1*A1+A2*A2);
    T:=A1+R;C:=-T/R;V[J+1]:=1;V[J]:=A2/T;
    HSHVECTAM(L,UA,J,J+1,C,V,A);HSHVECTAM(L,UB,J,J+1,C,V,B);
    HSHVECTAM(1,UX,J,J+1,C,V,X)
"END" HSH2ROW3;
        "EOP"


"CODE" 34608;
"PROCEDURE" HSH2ROW2(LA,LB,UA,UB,J,A1,A2,A,B);"VALUE"LA,LB,UA,UB,
J,A1,A2;"INTEGER" LA,LB,UA,UB,J;"REAL" A1,A2;"ARRAY" A,B;
"IF"A2^=0 "THEN"
"BEGIN" "REAL" R,T,C;"INTEGER" K;"ARRAY" V[J:J+1];
    "PROCEDURE" HSHVECTAM(LR,UR,LC,UC,X,U,A);"CODE" 31073;
    R:="IF" A1<0 "THEN" -SQRT(A1*A1+A2*A2) "ELSE" SQRT(A1*A1+A2*A2);
    T:=A1+R;C:=-T/R;V[J+1]:=1;V[J]:=A2/T;
    HSHVECTAM(LA,UA,J,J+1,C,V,A);HSHVECTAM(LB,UB,J,J+1,C,V,B)
"END" HSH2ROW2;
        "EOP"


"CODE" 34609;
"PROCEDURE" HSH3ROW3(L,U,UX,J,A1,A2,A3,A,B,X);
"VALUE"L,U,UX,J,A1,A2,A3;"INTEGER"L,J,U,UX;"REAL"A1,A2,A3;"ARRAY"A,B,X;
"IF" A2^=0 "OR" A3^=0 "THEN"
"BEGIN" "REAL" R,T,C;"ARRAY" V[J:J+2];"INTEGER" K;
    "PROCEDURE" HSHVECTAM(LR,UR,LC,UC,X,U,A);"CODE" 31073;
    R:="IF" A1<0 "THEN" -SQRT(A1*A1+A2*A2+A3*A3)
    "ELSE" SQRT(A1*A1+A2*A2+A3*A3);
    T:=A1+R;C:=-T/R;V[J+2]:=1;V[J+1]:=A2/T;V[J]:=A3/T;
    HSHVECTAM(L,U,J,J+2,C,V,A);HSHVECTAM(L,U,J,J+2,C,V,B);
    HSHVECTAM(L,UX,J,J+2,C,V,X)
"END" HSH3ROW3;
        "EOP"


"CODE" 34610;
"PROCEDURE" HSH3ROW2(LA,LB,U,J,A1,A2,A3,A,B);
"VALUE"LA,LB,U,J,A1,A2,A3;"INTEGER"LA,LB,U,J;"REAL"A1,A2,A3;"ARRAY"A,B;
"IF" A2^=0 "OR" A3^=0 "THEN"
"BEGIN" "REAL" R,T,C;"ARRAY" V[J:J+2];
    "PROCEDURE" HSHVECTAM(LR,UR,LC,UC,X,U,A);"CODE" 31073;
    R:="IF" A1<0 "THEN" -SQRT(A1*A1+A2*A2+A3*A3)
    "ELSE" SQRT(A1*A1+A2*A2+A3*A3);
    T:=A1+R;C:=-T/R;V[J+2]:=1;V[J+1]:=A2/T;V[J]:=A3/T;
    HSHVECTAM(LA,U,J,J+2,C,V,A);HSHVECTAM(LB,U,J,J+2,C,V,B)
"END" HSH3ROW2;
        "EOP"
```

AUTHORS : G.H.GOLUB AND C.REINSCH

CONTRIBUTOR : D.T.WINTER

INSTITUTE : MATHEMATICAL CENTRE

RECEIVED : 731217

BRIEF DESCRIPTION :
THIS SECTION CONTAINS TWO PROCEDURES, QRISNGVALBID AND QRISNGVALDECBID. BOTH PROCEDURES CALCULATE THE SINGULAR VALUES OF A BIDIAGONAL MATRIX. MOREOVER, THE SECOND PROCEDURE CALCULATES THE THE SINGULAR VALUES DECOMPOSITION OF A FULL MATRIX OF WHICH THE BIDIAGONAL AND THE PRE- AND POSTMULTIPLYING MATRICES, AS CALCULATED BY HSHREABID ARE GIVEN.

KEYWORDS :
SINGULAR VALUES
QR ITERATION
BIDIAGONAL MATRICES

SUBSECTION : QRISNGVALBID

CALLING SEQUENCE :
THE HEADING OF THE PROCEDURE IS :
"INTEGER" "PROCEDURE" QRISNGVALBID(D, B, N, EM);
"VALUE" N; "INTEGER" N; "ARRAY" D, B, EM;

THE MEANING OF THE FORMAL PARAMETERS IS:
D: <ARRAY IDENTIFIER>;
"ARRAY" D[1:N];
ENTRY: THE DIAGONAL OF THE BIDIAGONAL MATRIX;
EXIT: THE SINGULAR VALUES;
B: <ARRAY IDENTIFIER>;
"ARRAY" B[1:N];
ENTRY: THE SUPER DIAGONAL OF THE BIDIAGONAL MATRIX,IN B[1:N-1];
N: <ARITHMETIC EXPRESSION>;
THE LENGTH OF B AND D;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[1:7];
ENTRY: EM[1]: THE INFINITY NORM OF THE MATRIX;
EM[2]: THE RELATIVE PRECISION IN THE SINGULAR VALUES;
EM[4]: THE MAXIMAL NUMBER OF ITERATIONS TO BE PERFORMED;
EM[6]: THE MINIMAL NON-NEGLECTABLE SINGULAR VALUE;
EXIT: EM[3]: THE MAXIMAL NEGLECTED SUPERDIAGONAL ELEMENT;
EM[5]: THE NUMBER OF ITERATIONS PERFORMED;
EM[7]: THE NUMERICAL RANK OF THE MATRIX, I.E. THE NUMBER OF SINGULAR VALUES GREATER THAN OR EQUAL TO EM[6].

MOREOVER :
QRISNGVALBID:= THE NUMBER OF SINGULAR VALUES NOT FOUND, I.E. A NUMBER NOT EQUAL TO ZERO IF THE NUMBER OF ITERATIONS EXCEEDS EM[4].

PROCEDURES USED : NONE

REQUIRED CENTRAL MEMORY : NO AUXILIARY ARRAYS ARE DECLARED

RUNNING TIME :
THE RUNNING TIME DEPENDS STRONGLY UPON THE PROPERTIES OF THE MATRIX

METHOD AND PERFORMANCE :
THE METHOD IS DESCRIBED IN DETAIL IN [1]. THIS PROCEDURE IS A REWRITING OF PART OF THE PROCEDURE SVD PUBLISHED THERE BY G.H.GOLUB AND C.REINSCH.

LANGUAGE : ALGOL 60.

SUBSECTION : QRISNGVALDECBID

CALLING SEQUENCE :
THE HEADING OF THE PROCEDURE IS :
"INTEGER" "PROCEDURE" QRISNGVALDECBID(D, B, M, N, U, V, EM);
"VALUE" M, N; "INTEGER" M, N; "ARRAY" D, B, U, V, EM;

THE MEANING OF THE FORMAL PARAMETERS IS :
D: <ARRAY IDENTIFIER>;
"ARRAY" D[1:N];
ENTRY: THE DIAGONAL OF THE BIDIAGONAL MATRIX;
EXIT: THE SINGULAR VALUES;
B: <ARRAY IDENTIFIER>;
"ARRAY" B[1:N];
ENTRY: THE SUPER DIAGONAL OF THE BIDIAGONAL MATRIX,IN B[1:N-1];
M: <ARITHMETIC EXPRESSION>;
THE NUMBER OF ROWS OF THE MATRIX U;
N: <ARITHMETIC EXPRESSION>;
THE LENGTH OF B AND D, THE NUMBER OF COLUMNS OF U AND THE NUMBER OF COLUMNS AND ROWS OF V;
U: <ARRAY IDENTIFIER>;
"ARRAY" U[1:M,1:N];
ENTRY: THE PREMULTIPLYING MATRIX AS PRODUCED BY PRETFMMAT (SECTION 3.2.2.1.1);
EXIT: THE PREMULTIPLYING MATRIX U OF THE SINGULAR VALUES DECOMPOSITION U * S * V';
V: <ARRAY IDENTIFIER>;
"ARRAY" V[1:N,1:N];
ENTRY: THE TRANSPOSE OF THE POSTMULTIPLYING MATRIX AS PRODUCED BY PSTTFMMAT (SECTION 3.2.2.1.1);
EXIT: THE TRANSPOSE OF THE POSTMULTIPLYING MATRIX V OF THE SINGULAR VALUES DECOMPOSITION;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[1:7];
ENTRY: EM[1]: THE INFINITY NORM OF THE MATRIX;
EM[2]: THE RELATIVE PRECISION IN THE SINGULAR VALUES;
EM[4]: THE MAXIMAL NUMBER OF ITERATIONS TO BE PERFORMED;
EM[6]: THE MINIMAL NON-NEGLECTABLE SINGULAR VALUE;
EXIT: EM[3]: THE MAXIMAL NEGLECTED SUPERDIAGONAL ELEMENT;
EM[5]: THE NUMBER OF ITERATIONS PERFORMED;
EM[7]: THE NUMERICAL RANK OF THE MATRIX, I.E. THE NUMBER OF SINGULAR VALUES GREATER THAN OR EQUAL TO EM[6].

MOREOVER :
QRISNGVALDECBID:= THE NUMBER OF SINGULAR VALUES NOT FOUND, I.E. A NUMBER NOT EQUAL TO ZERO IF THE NUMBER OF ITERATIONS EXCEEDS EM[4].

PROCEDURES USED :

ROTCOL = CP34040

REQUIRED CENTRAL MEMORY : NO AUXILIARY ARRAYS ARE DECLARED

RUNNING TIME :
THE RUNNING TIME DEPENDS STRONGLY UPON THE PROPERTIES OF THE MATRIX

METHOD AND PERFORMANCE :
THE METHOD IS DESCRIBED IN DETAIL IN [1]. THIS PROCEDURE IS A REWRITING OF PART OF THE PROCEDURE SVD PUBLISHED THERE BY G.H.GOLUB AND C.REINSCH.

LANGUAGE : ALGOL 60

REFERENCES :
[1] WILKINSON, J.H. AND C.REINSCH
HANDBOOK OF AUTOMATIC COMPUTATION, VOL. 2
LINEAR ALGEBRA
HEIDELBERG (1971)

EXAMPLE OF USE :
FOR AN EXAMPLE OF USE ONE IS REFERRED TO SECTION 3.5.1.2

SOURCE TEXT(S):

```
"CODE" 34270;
"INTEGER" "PROCEDURE" ORISNGVALBID(D, B, N, EM);
"VALUE" N; "INTEGER" N; "ARRAY" D, B, EM;
"BEGIN" "INTEGER" N1, K, K1, I, I1, COUNT, MAX, RNK;
    "REAL" TOL, BMAX, Z, X, Y, G, H, F, C, S, MIN;
    TOL:= EM[2] * EM[1]; COUNT:= 0; BMAX:= 0; MAX:= EM[4]; MIN:= EM[6];
    RNK:= N;
IN: K:= N; N1:= N - 1;
NEXT: K:= K - 1; "IF" K > 0 "THEN"
    "BEGIN" "IF" ABS(B[K]) >= TOL "THEN"
        "BEGIN" "IF" ABS(D[K]) >= TOL "THEN" "GOTO" NEXT;
            C:= 0; S:= 1;
            "FOR" I:= K "STEP" 1 "UNTIL" N1 "DO"
            "BEGIN" F:= S * B[I]; B[I]:= C * B[I]; I1:= I + 1;
                "IF" ABS(F) < TOL "THEN" "GOTO" NEGLECT;
                G:= D[I1]; D[I1]:= H:= SQRT(F * F + G * G);
                C:= G / H; S:= - F / H
            "END"
```

```
        NEGLECT:
        "END"
        "ELSE" "IF" ABS(B[K]) > BMAX "THEN" BMAX:= ABS(B[K])
    "END":
    "IF" K = N1 "THEN"
    "BEGIN" "IF" D[N] < 0 "THEN" D[N]:= - D[N];
        "IF" D[N] <= MIN "THEN" RNK:= RNK - 1: N:= N1
    "END"
    "ELSE"
    "BEGIN" COUNT:= COUNT + 1; "IF" COUNT > MAX "THEN" "GOTO" END;
        K1:= K + 1; Z:= D[N]; X:= D[K1]; Y:= D[N1];
        G:= "IF" N1 = 1 "THEN" 0 "ELSE" B[N1 - 1]; H:= B[N1];
        F:= ((Y - Z) * (Y + Z) + (G - H) * (G + H)) / (2 * H * Y);
        G:= SQRT(F * F + 1);
        F:= ((X - Z) * (X + Z) + H * (Y / ("IF" F < 0 "THEN" F - G
        "ELSE" F + G) - H)) / X; C:= S:= 1;
        "FOR" I:= K1 + 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" I1:= I - 1; G:= B[I1]; Y:= D[I]; H:= S * G; G:= C * G;
            Z:= SQRT(F * F + H * H); C:= F / Z; S:= H / Z;
            "IF" I1 ^= K1 "THEN" B[I1 - 1]:= Z; F:= X * C + G * S;
            G:= G * C - X * S; H:= Y * S; Y:= Y * C;
            D[I1]:= Z:= SQRT(F * F + H * H); C:= F / Z; S:= H / Z;
            F:= C * G + S * Y; X:= C * Y - S * G
        "END";
        B[N1]:= F; D[N]:= X
    "END";
    "IF" N > 0 "THEN" "GOTO" IN;
END: EM[3]:= BMAX; EM[5]:= COUNT; EM[7]:= RNK; QRISNGVALBID:= N
"END" QRISNGVALBID:
        "EOP"

"CODE" 34271:
"INTEGER" "PROCEDURE" QRISNGVALDECBID(D, B, M, N, U, V, EM):
"VALUE" M, N: "INTEGER" M, N: "ARRAY" D, B, U, V, EM:
"BEGIN" "INTEGER" NO, N1, K, K1, I, I1, COUNT, MAX, RNK:
    "REAL" TOL, BMAX, Z, X, Y, G, H, F, C, S, MIN:

    "PROCEDURE" ROTCOL(L, U, I, J, A, C, S):
    "VALUE" L, U, I, J, C, S: "INTEGER" L, U, I, J;
    "REAL" C, S: "ARRAY" A;
    "CODE" 34040:"COMMENT"
```

```
    TOL:= EM[2] * EM[1]; COUNT:= 0; BMAX:= 0; MAX:= EM[4]; MIN:= EM[6];
    RNK:= NO:= N;
IN: K:= N; N1:= N - 1;
NEXT: K:= K - 1; "IF" K > 0 "THEN"
    "BEGIN" "IF" ABS(B[K]) >= TOL "THEN"
        "BEGIN" "IF" ABS(D[K]) >= TOL "THEN" "GOTO" NEXT;
            C:= 0; S:= 1;
            "FOR" I:= K "STEP" 1 "UNTIL" N1 "DO"
            "BEGIN" F:= S * B[I]; B[I]:= C * B[I]; I1:= I + 1;
                "IF" ABS(F) < TOL "THEN" "GOTO" NEGLECT;
                G:= D[I1]; D[I1]:= H:= SQRT(F * F + G * G);
                C:= G / H; S:= - F / H;
                ROTCOL(1, M, K, I1, U, C, S)
            "END";
        NEGLECT:
        "END"
        "ELSE" "IF" ABS(B[K]) > BMAX "THEN" BMAX:= ABS(B[K])
    "END";
    "IF" K = N1 "THEN"
    "BEGIN" "IF" D[N] < 0 "THEN"
        "BEGIN" D[N]:= - D[N];
            "FOR" I:= 1 "STEP" 1 "UNTIL" NO "DO" V[I,N]:= - V[I,N]
        "END";
        "IF" D[N] <= MIN "THEN" RNK:= RNK - 1; N:= N1
    "END"
    "ELSE"
    "BEGIN" COUNT:= COUNT + 1; "IF" COUNT > MAX "THEN" "GOTO" END;
        K1:= K + 1; Z:= D[N]; X:= D[K1]; Y:= D[N1];
        G:= "IF" N1 = 1 "THEN" 0 "ELSE" B[N1 - 1]; H:= B[N1];
        F:= ((Y - Z) * (Y + Z) + (G - H) * (G + H)) / (2 * H * Y);
        G:= SQRT(F * F + 1);
        F:= ((X - Z) * (X + Z) + H * (Y / ("IF" F < 0 "THEN" F - G
        "ELSE" F + G) - H)) / X; C:= S:= 1;
        "FOR" I:= K1 + 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" I1:= I - 1; G:= B[I1]; Y:= D[I]; H:= S * G; G:= C * G;
            Z:= SQRT(F * F + H * H); C:= F / Z; S:= H / Z;
            "IF" I1 ^= K1 "THEN" B[I1 - 1]:= Z; F:= X * C + G * S;
            G:= G * C - X * S; H:= Y * S; Y:= Y * C;
            ROTCOL(1, NO, I1, I, V, C, S);
            D[I1]:= Z:= SQRT(F * F + H * H); C:= F / Z; S:= H / Z;
            F:= C * G + S * Y; X:= C * Y - S * G;
            ROTCOL(1, M, I1, I, U, C, S)
        "END";
        B[N1]:= F; D[N]:= X
    "END";
    "IF" N > 0 "THEN" "GOTO" IN;
END: EM[3]:= BMAX; EM[5]:= COUNT; EM[7]:= RNK; ORISNGVALDECBID:= N
"END" ORISNGVALDECBID;
        "EOP"
```

AUTHOR : D.T.WINTER

INSTITUTE : MATHEMATICAL CENTRE

RECEIVED : 731217

BRIEF DESCRIPTION :

THIS SECTION CONTAINS TWO PROCEDURES, ORISNGVAL AND ORISNGVALDEC.
ORISNGVAL CALCULATES THE SINGULAR VALUES OF A GIVEN MATRIX.
ORISNGVALDEC CALCULATES THE SINGULAR VALUES DECOMPOSITION
U * S * V', WITH U AND V ORTHOGONAL AND S POSITIVE DIAGONAL.

KEYWORDS :

SINGULAR VALUES
QR ITERATION

SUBSECTION : QRISNGVAL

CALLING SEQUENCE :
THE HEADING OF THE PROCEDURE IS :
"INTEGER" "PROCEDURE" QRISNGVAL(A, M, N, VAL, EM);
"VALUE" M, N; "INTEGER" M, N; "ARRAY" A, VAL, EM;

THE MEANING OF THE FORMAL PARAMETERS IS :
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:M,1:N];
ENTRY: THE INPUT MATRIX;
EXIT: DATA CONCERNING THE TRANSFORMATION TO BIDIAGONAL FORM;
M: <ARITHMETIC EXPRESSION>;
THE NUMBER OF ROWS OF A;
N: <ARITHMETIC EXPRESSION>;
THE NUMBER OF COLUMNS OF A, N SHOULD SATISFY N <= M;
VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:N];
EXIT: THE SINGULAR VALUES;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:7];
ENTRY: EM[0]: THE MACHINE PRECISION;
EM[2]: THE RELATIVE PRECISION IN THE SINGULAR VALUES;
EM[4]: THE MAXIMAL NUMBER OF ITERATIONS TO BE PERFORMED;
EM[6]: THE MINIMAL NON-NEGLECTABLE SINGULAR VALUE;
EXIT: EM[1]: THE INFINITY NORM OF THE MATRIX;
EM[3]: THE MAXIMAL NEGLECTED SUPERDIAGONAL ELEMENT;
EM[5]: THE NUMBER OF ITERATIONS PERFORMED;
EM[7]: THE NUMERICAL RANK OF THE MATRIX, I.E. THE NUMBER OF
SINGULAR VALUES GREATER THAN OR EQUAL TO EM[6].

MOREOVER :
QRISNGVAL:= THE NUMBER OF SINGULAR VALUES NOT FOUND, I.E. A NUMBER
NOT EQUAL TO ZERO IF THE NUMBER OF ITERATIONS EXCEEDS EM[4].

PROCEDURES USED :
HSHREABID = CP34260
QRISNGVALBID = CP34270

REQUIRED CENTRAL MEMORY : AN AUXILIARY ARRAY OF N REALS IS DECLARED

RUNNING TIME :
THE RUNNING TIME DEPENDS UPON THE PROPERTIES OF THE MATRIX, HOWEVER
THE PROCESS OF BIDIAGONALIZATION DOMINATES, AND ITS RUNNING TIME
IS PROPORTIONAL TO (M + N) * N * N

METHOD AND PERFORMANCE :
THE MATRIX IS FIRST TRANSFORMED TO BIDIAGONAL FORM BY THE PROCEDURE
HSHREABID (SECTION 3.2.2.1.1) , AND THEN THE SINGULAR VALUES ARE
CALCULATED BY QRISNGVALBID (SECTION 3.5.1.1).

LANGUAGE : ALGOL 60

SUBSECTION : QRISNGVALDEC

CALLING SEQUENCE :
THE HEADING OF THE PROCEDURE IS :
"INTEGER" "PROCEDURE" QRISNGVALDEC(A, M, N, VAL, V, EM);
"VALUE" M, N; "INTEGER" M, N; "ARRAY" A, VAL, V, EM;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:M,1:N];
ENTRY: THE GIVEN MATRIX;
EXIT: THE MATRIX U IN THE SINGULAR VALUES DECOMPOSITION
U * S * V';
M: <ARITHMETIC EXPRESSION>;
THE NUMBER OF ROWS OF A;
N: <ARITHMETIC EXPRESSION>;
THE NUMBER OF COLUMNS OF A, N SHOULD SATISFY N <= M;
VAL: <ARRAY IDENTIFIER>;
"ARRAY" VAL[1:N];
EXIT: THE SINGULAR VALUES;
V: <ARRAY IDENTIFIER>;
"ARRAY" V[1:N,1:N];
EXIT: THE TRANSPOSE OF MATRIX V IN THE SINGULAR VALUES
DECOMPOSITION;
EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:7];
ENTRY: EM[0]: THE MACHINE PRECISION;
EM[2]: THE RELATIVE PRECISION IN THE SINGULAR VALUES;
EM[4]: THE MAXIMAL NUMBER OF ITERATIONS TO BE PERFORMED;
EM[6]: THE MINIMAL NON-NEGLECTABLE SINGULAR VALUE;
EXIT: EM[1]: THE INFINITY NORM OF THE MATRIX;
EM[3]: THE MAXIMAL NEGLECTED SUPER DIAGONAL ELEMENT;
EM[5]: THE NUMBER OF ITERATIONS PERFORMED;
EM[7]: THE NUMERICAL RANK OF THE MATRIX, I.E. THE NUMBER OF
SINGULAR VALUES GREATER THAN OR EQUAL TO EM[6].

MOREOVER :
QRISNGVALDEC:= THE NUMBER OF SINGULAR VALUES NOT FOUND, I.E. A
NUMBER NOT EQUAL TO ZERO IF THE NUMBER OF ITERATIONS EXCEEDS
EM[4].

PROCEDURES USED :
HSHREABID = CP34260
PSTTFMMAT = CP34261
PRETFMMAT = CP34262
QRISNGVALDECBID = CP34271

REQUIRED CENTRAL MEMORY : AN AUXILIARY ARRAY OF N ELEMENTS IS DECLARED

RUNNING TIME :

THE RUNNING TIME DEPENDS UPON THE PROPERTIES OF THE MATRIX, HOWEVER THE PROCESS OF BIDIAGONALIZATION DOMINATES, AND ITS RUNNING TIME IS PROPORTIONAL TO (M + N) * N * N

METHOD AND PERFORMANCE:

THE MATRIX IS FIRST TRANSFORMED TO BIDIAGONAL FORM BY THE PROCEDURE HSHREABID (SECTION 3.2.2.1.1), THE TWO TRANSFORMING MATRICES ARE CALCULATED BY THE PROCEDURES PSTTFMMAT AND PRETFMMAT (SECTIONS 3.2.2.1.2 AND 3.2.2.1.3 RESPECTIVELY), AND FINALLY THE SINGULAR VALUES DECOMPOSITION IS CALCULATED BY QRISNGVALDECBID (SECTION 3.5.1.1).

LANGUAGE : ALGOL 60

REFERENCES :

WILKINSON, J.H. AND C.REINSCH
HANDBOOK OF AUTOMATIC COMPUTATION, VOL. 2
LINEAR ALGEBRA
HEIDELBERG (1971)

EXAMPLE OF USE :

AS THE PROCEDURE QRISNGVALDEC CALCULATES THE SINGULAR VALUES OF A MATRIX IN EXACTLY THE SAME WAY AS QRISNGVAL, WE GIVE HER ONLY AN EXAMPLE OF USE OF THE PROCEDURE QRISNGVALDEC. FIRST WE GIVE A PROGRAM, AND THEN THE RESULTS OF THIS PROGRAM:

```
"BEGIN" "ARRAY" A[1:6,1:5], V[1:5,1:5], VAL[1:5], EM[0:7];
    "INTEGER" I, J;
    "INTEGER" "PROCEDURE" QRISNGVALDEC(A, M, N, VAL, V, EM);
    "VALUE" M, N; "INTEGER" M, N; "ARRAY" A, VAL, V, EM;
    "CODE" 34273;

    "FOR" I:= 1 "STEP" 1 "UNTIL" 6 "DO"
    "FOR" J:= 1 "STEP" 1 "UNTIL" 5 "DO"
    A[I,J]:= 1 / (I + J - 1);
    EM[0]:= "-14; EM[2]:= "-12; EM[4]:= 25; EM[6]:= "-10;
    I:= QRISNGVALDEC(A, 6, 5, VAL, V, EM);
    OUTPUT(61, "("3B, "("NUMBER SINGULAR VALUES NOT FOUND : ")",
    3ZD, /, 3B, "("INFINITY NORM : ")", N, /, 3B,
    "("MAX NEGLECTED SUBDIAGONAL ELEMENT : ")", N, /, 3B,
    "("NUMBER ITERATIONS : ")", 3ZD, /, 3B,
    "("NUMERICAL RANK : ")", 3ZD, /")", I, EM[1], EM[3], EM[5],
    EM[7]);
    OUTPUT(61, "("/, 3B, "("SINGULAR VALUES : ")", /")");
    "FOR" I:= 1 "STEP" 1 "UNTIL" 5 "DO"
    OUTPUT(61, "("/, 3B, N")", VAL[I]);
    OUTPUT(61, "("/, /, 3B, "("MATRIX U, FIRST 3 COLUMNS")", /")");
    "FOR" I:= 1 "STEP" 1 "UNTIL" 6 "DO"
    OUTPUT(61, "("/, 3B, 3(N)")", A[I,1], A[I,2], A[I,3]);
    OUTPUT(61, "("/, /, 13B, "("LAST 2 COLUMNS")", /")");
    "FOR" I:= 1 "STEP" 1 "UNTIL" 6 "DO"
    OUTPUT(61, "("/, 13B, 2(N)")", A[I,4], A[I,5])
"END"
```

```
NUMBER SINGULAR VALUES NOT FOUND :    0
INFINITY NORM : +2.2833333333334"+000
MAX NEGLECTED SUBDIAGONAL ELEMENT : +5.7786437871158"-014
NUMBER ITERATIONS :    5
NUMERICAL RANK :    5

SINGULAR VALUES :

+1.5921172587262"+000
+2.2449595426097"-001
+1.3610556101029"-002
+4.3245382038374"-004
+6.4001947134260"-006

MATRIX U, FIRST 3 COLUMNS

-7.5497918208386"-001  +6.1011090790645"-001  -2.3287173869184"-001
-4.3909273679284"-001  -2.2602102994174"-001  +7.0245315582712"-001
-3.1703146681544"-001  -3.7366964696148"-001  +2.1607293656979"-001
-2.4999458583084"-001  -3.9557817833576"-001  -1.4665595223684"-001
-2.0704999076883"-001  -3.8483260608872"-001  -3.6803786187007"-001
-1.7699734614538"-001  -3.6458192866515"-001  -4.9868122801331"-001

          LAST 2 COLUMNS

          +5.8625326935176"-002  -1.0184205426735"-002
          -4.8169088124009"-001  +1.7189132301455"-001
          +5.4982292571999"-001  -5.9788920283495"-001
          +4.0633053815463"-001  +4.5989617524697"-001
          -6.1755991033503"-002  +4.3029765325422"-001
          -5.4158416488948"-001  -4.6499203623570"-001
```

SOURCE TEXT(S):

```
"CODE" 34272;
"INTEGER" "PROCEDURE" QRISNGVAL(A, M, N, VAL, EM);
"VALUE" M, N; "INTEGER" M, N; "ARRAY" A, VAL, EM;
"BEGIN" "ARRAY" B[1:N];

    "PROCEDURE" HSHREABID(A, M, N, D, B, EM);
    "VALUE" M, N; "INTEGER" M, N; "ARRAY" D, B, EM;
    "CODE" 34260;

    "INTEGER" "PROCEDURE" QRISNGVALBID(D, B, N, EM);
    "VALUE" N; "INTEGER" N; "ARRAY" D, B, EM;
    "CODE" 34270;

    HSHREABID(A, M, N, VAL, B, EM);
    QRISNGVAL:= QRISNGVALBID(VAL, B, N, EM)
"END" QRISNGVAL;
        "EOP"

"CODE" 34273;
"INTEGER" "PROCEDURE" QRISNGVALDEC(A, M, N, VAL, V, EM);
"VALUE" M, N; "INTEGER" M, N; "ARRAY" A, VAL, V, EM;
"BEGIN" "ARRAY" B[1:N];

    "PROCEDURE" HSHREABID(A, M, N, D, B, EM);
    "VALUE" M, N; "INTEGER" M, N; "ARRAY" A, D, B, EM;
    "CODE" 34260;

    "PROCEDURE" PSTTFMMAT(A, N, V, B);
    "VALUE" N; "INTEGER" N; "ARRAY" A, V, B;
    "CODE" 34261;

    "PROCEDURE" PRETFMMAT(A, M, N, D);
    "VALUE" M, N; "INTEGER" M, N; "ARRAY" A, D;
    "CODE" 34262;

    "INTEGER" "PROCEDURE" QRISNGVALDECBID(D, B, M, N, U, V, EM);
    "VALUE" M, N; "INTEGER" M, N; "ARRAY" D, B, U, V, EM;
    "CODE" 34271;

    HSHREABID(A, M, N, VAL, B, EM);
    PSTTFMMAT(A, N, V, B); PRETFMMAT(A, M, N, VAL);
    QRISNGVALDEC:= QRISNGVALDECBID(VAL, B, M, N, A, V, EM)
"END" QRISNGVALDEC;
        "EOP"
```

AUTHORS: TH.J. DEKKER AND TH.H.P. REYMER

CONTRIBUTOR: TH.H.P. REYMER

INSTITUTE: UNIVERSITY OF AMSTERDAM

RECEIVED: 770427;

BRIEF DESCRIPTION:

THIS SECTION CONTAINS TWO PROCEDURES:
A) ZERPOL CALCULATES ALL ROOTS OF A POLYNOMIAL WITH REAL COEFFICIENTS BY MEANS OF LAGUERRE'S METHOD;
B) BOUNDS CALCULATES UPPERBOUNDS FOR THE ABSOLUTE ERROR IN GIVEN APPROXIMATED ZEROS OF A POLYNOMIAL WITH REAL COEFFICIENTS;

KEYWORDS:

ZEROS;
REAL POLYNOMIAL;
LAGUERRE'S METHOD;
ERROR BOUNDS;

SUBSECTION: ZERPOL

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"INTEGER" "PROCEDURE" ZERPOL(N, A, EM, RE, IM, D);
"VALUE" N; "INTEGER" N; "ARRAY" A, EM, RE, IM, D;
"CODE" 34501 ;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
ENTRY: THE DEGREE OF THE POLYNOMIAL;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[0 : N];
ENTRY: THE COEFFICIENTS OF THE POLYNOMIAL, IN SUCH A WAY THAT
P(Z) = (..(A[N]*Z + A[N-1])*Z +..+ A[1])*Z + A[0];

```
    EM:      <ARRAY IDENTIFIER>;
             "ARRAY" EM[0 : 4];
             ENTRY:  EM[0]: MACHINE PRECISION;
                     EM[1]: THE MAXIMAL NUMBER OF ITERATIONS ALLOWED FOR
                            EACH ZERO;
             EXIT:   EM[2]: FAIL INDICATION;
                            0  SUCCESSFUL CALL;
                            1  UPON ENTRY DEGREE N <= 0;
                            2  UPON ENTRY LEADING COEFFICIENT A[N] = 0;
                            3  NUMBER OF ITERATIONS EXCEEDED EM[1];
                     EM[3]: NUMBER OF NEW STARTS IN THE LAST ITERATION;
                     EM[4]: TOTAL NUMBER OF ITERATIONS PERFORMED;
             FOR THE CD CYBER 70 SYSTEM SUITABLE VALUES ARE:
             EM[0]:= "-14;
             EM[1]:= 40; IF, UPON EXIT, EM[2] = 3  AND EM[3] < 5 THEN IT
                         MAY BE  USEFUL TO  START  AGAIN  WITH  A HIGHER
                         VALUE OF EM[1];
    RE, IM:  <ARRAY IDENTIFIERS>;
             "ARRAY" RE, IM[1 : N];
             EXIT:   THE  REAL  AND  IMAGINARY PARTS OF THE ZEROS OF THE
                     POLYNOMIAL; THE  MEMBERS  OF  EACH  NONREAL COMPLEX
                     CONJUGATE PAIR ARE CONSECUTIVE;
    D:       <ARRAY IDENTIFIER>;
             "ARRAY" D[0 : N];
             EXIT:   IF THE CALL IS UNSUCCESSFUL AND ONLY N-K ZEROS HAVE
                     BEEN   FOUND,  THEN   D[0 : K]   CONTAINS   THE
                     COEFFICIENTS   OF   THE   (DEFLATED)   POLYNOMIAL;
                     MOREOVER,  THEN THE ZEROS  FOUND  ARE  DELIVERED IN
                     RE, IM[ K + 1 : N], WHEREAS THE REMAINING  PARTS
                     OF RE AND IM CONTAIN NO INFORMATION;

    ZERPOL:= THE NUMBER, K, OF ZEROS NOT FOUND;


PROCEDURES USED:

    DWARF   = CP30003;
    GIANT   = CP30004;
    COMABS  = CP34340;
    COMSQRT = CP34343;


REQUIRED CENTRAL MEMORY:

    TOTAL SIZE OF LOCAL ARRAYS IS N + 16 REAL LOCATIONS;


RUNNING TIME:

    ROUGHLY PROPORTIONAL TO N**2;
```

METHOD AND PERFORMANCE:

THE PROCEDURE USES LAGUERRE'S METHOD TO FIND ZEROS OF THE GIVEN POLYNOMIAL (SEE [2]); WHEN A ZERO HAS BEEN FOUND, A COMPOSITE DEFLATION TECHNIQUE IS USED TO OBTAIN A NEW POLYNOMIAL OF LOWER DEGREE (SEE [1], [3], [4]); IF CONVERGENCE IS NOT APPARENT, SEVERAL RESTARTS, THE NUMBER OF WHICH DEPENDS ON EM[1] BUT HAS A MAXIMUM OF 5, ARE MADE IN THE NEIGHBOURHOOD OF THE ABSOLUTE LARGEST ZERO; THE ACCURACY OF THE CALCULATED ZEROS STRONGLY DEPENDS ON THE POLYNOMIAL; A ROUGH INDICATION FOR THE ERROR IN A CALCULATED ZERO Z FOLLOWS FROM P(Z) / DP(Z), WHERE P DENOTES THE GIVEN POLYNOMIAL AND DP ITS FIRST DERIVATIVE (SEE E.G. [5]); TO FIND A TRUE UPPERBOUND FOR THESE ERRORS, ONE CAN USE PROCEDURE BOUNDS (SEE NEXT SUBSECTION); FOR A MORE DETAILED DESCRIPTION OF THE PROCEDURE AND TEST RESULTS SEE [4];

REFERENCES:

[1] D.A. ADAMS, A STOPPING CRITERION FOR POLYNOMIAL ROOT FINDING, CACM 10, NO 10, PP. 655-658, OCTOBER 1967;
[2] T.J. DEKKER, NEWTON-LAGUERRE ITERATION, MATHEMATISCH CENTRUM MR82, 1966;
[3] G. PETERS AND J.H. WILKINSON, PRACTICAL PROBLEMS ARISING IN THE SOLUTION OF POLYNOMIAL EQUATIONS, J. INST. MATHS APPLICS 1971, NO. 8, PP. 16-35;
[4] TH.H.P. REYMER, BEREKENING VAN NULPUNTEN VAN REELE POLYNOMEN EN FOUTGRENZEN VOOR DEZE NULPUNTEN, DOCTORAAL SCRIPTIE UVA, APRIL 1977;
[5] J.H. WILKINSON, ROUNDING ERRORS IN ALGEBRAIC PROCESSES, PRENTICE HALL, 1963;

EXAMPLE OF USE:

FOR AN EXAMPLE OF USE SEE PROCEDURE BOUNDS (NEXT SUBSECTION);

SUBSECTION: BOUNDS

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "PROCEDURE" BOUNDS(N,A,RE,IM,RELE,ABSE,RECENTRE,IMCENTRE,BOUND);
    "VALUE" N, RELE, ABSE; "INTEGER" N; "REAL" RELE, ABSE;
    "ARRAY" A, RE, IM, RECENTRE, IMCENTRE, BOUND;
    "CODE" 34502 ;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N:          <ARITHMETIC EXPRESSION>;
                ENTRY:  DEGREE OF THE POLYNOMIAL;
    A:          <ARRAY IDENTIFIER>;
                "ARRAY" A[0 : N];
                ENTRY:  THE COEFFICIENTS OF THE POLYNOMIAL OF WHICH
                        RE[J] + I * IM[J] ARE THE APPROXIMATED ZEROS,
                        IN SUCH A WAY THAT
                        P(Z) = (..(A[N]*Z + A[N-1])*Z +..+A[1])*Z + A[0]
    RE, IM:     <ARRAY IDENTIFIERS>;
                "ARRAY" RE, IM[1 : N];
                ENTRY:  REAL AND IMAGINARY PARTS OF APPROXIMATED ZEROS OF
                        A POLYNOMIAL, SUCH THAT THE MEMBERS OF EACH
                        NONREAL COMPLEX CONJUGATE PAIR ARE CONSECUTIVE;
                EXIT:   A PERMUTATION OF THE INPUT DATA;
    RELE:       <ARITHMETIC EXPRESSION>;
                ENTRY:  RELATIVE ERROR IN THE NON-VANISHING COEFFICIENTS
                        A[J] OF THE GIVEN POLYNOMIAL;
    ABSE:       <ARITHMETIC EXPRESSION>;
                ENTRY:  ABSOLUTE ERROR IN THE VANISHING COEFFICIENTS A[J]
                        GIVEN POLYNOMIAL; IF THERE ARE NO VANISHING
                        COEFFICIENTS, ABSE SHOULD BE ZERO;
    RECENTRE, IMCENTRE: <ARRAY IDENTIFIERS>;
                "ARRAY" RECENTRE, IMCENTRE[1 : N];
                EXIT:   REAL AND IMAGINARY PARTS OF THE CENTERS OF DISKS
                        IN WHICH SOME NUMBER OF ZEROS OF THE POLYNOMIAL
                        GIVEN BY A ARE SITUATED;
                        THE NUMBER OF IDENTICAL CENTERS DENOTES
                        THE NUMBER OF ZEROS IN THAT DISK;
    BOUND:      <ARRAY IDENTIFIER>;
                "ARRAY" BOUND[1 : N];
                EXIT:   RADIUS OF THE DISKS WHOSE CENTERS ARE GIVEN
                        CORRESPONDINGLY IN RECENTRE AND IMCENTRE;
```

PROCEDURES USED:

```
    ARREB = CP30002;
    GIANT = CP30004;
```

REQUIRED CENTRAL MEMORY:

TOTAL SIZE OF LOCAL ARRAYS IS AT MOST 7 * N REAL LOCATIONS;

RUNNING TIME:

APPROXIMATELY OF ORDER N**2;

METHOD AND PERFORMANCE:

FROM THE APPROXIMATED ZEROS A POLYNOMIAL IS RECONSTRUCTED AND COMPARED WITH THE GIVEN POLYNOMIAL; SUBSEQUENTLY, THE PROCEDURE CALCULATES DISKS SUCH THAT THE NUMBER OF GIVEN APPROXIMATED ZEROS WITHIN EACH DISK EQUALS THE NUMBER OF ZEROS OF THE GIVEN POLYNOMIAL WITHIN THAT DISK; UPON EXIT EVERY TWO NON-IDENTICAL DISKS ARE DISJOINT;
FOR A MORE DETAILED DESCRIPTION SEE [1], [2];

REFERENCES:

[1] G. PETERS AND J.H. WILKINSON, PRACTICAL PROBLEMS ARISING IN THE SOLUTION OF POLYNOMIAL EQUATIONS, J.INST.MATHS APPLICS 1971, NO 8, PP. 16-35;
[2] TH.H.P. REYMER, BEREKENING VAN NULPUNTEN VAN REELE POLYNOMEN EN FOUTGRENZEN VOOR DEZE NULPUNTEN, DOCTORAAL SCRIPTIE UVA, APRIL 1977;

EXAMPLE OF USE:

```
"BEGIN" "INTEGER" I, J;
  "ARRAY" A, D[0:7], RE, IM[1:7], EM[0:4];
  "INTEGER""PROCEDURE" ZERPOL(N,A,EM,RE,IM,D); "CODE"34501;

  A[7]:= 1; A[6]:= -3; A[5]:= -3; A[4]:= 25; A[3]:= -46;
  A[2]:= 38; A[1]:= -12; A[0]:= 0;
  EM[0]:= "-14; EM[1]:= 40;
  I:= ZERPOL(7, A, EM, RE, IM, D);
  OUTPUT(61,"("""("COEFFICIENTS OF POLYNOMIAL:")",//")");
  "FOR" J:=7 "STEP" -1 "UNTIL" 0 "DO"
  OUTPUT(61,"("-2ZD3B")",A[J]);
  OUTPUT(61,"("//,"("NUMBER NOT FOUND ZEROS ")",3ZD,/,
     "("FAIL INDICATION ")",3ZD,/,"("NUMBER NEW STARTS ")",3ZD,/,
     "("NUMBER OF ITERATIONS ")",3ZD,/")",I,EM[2],EM[3],EM[4]);
  OUTPUT(61,"("/,"("ZEROS: ")",/")");
  "FOR" J:= I+1 "STEP" 1 "UNTIL" 7 "DO" "IF" IM[J] = 0
  "THEN" OUTPUT(61,"("/,N")",RE[J])
  "ELSE" OUTPUT(61,"("/,2(N)")",RE[J],IM[J]);
```

```
  "IF" I = 0 "THEN"
  "BEGIN" "ARRAY" RECENTRE, IMCENTRE, BOUND[1:7];
    "PROCEDURE" BOUNDS(N,A,RE,IM,RELE,ABSE,RECENTRE,IMCENTRE,BOUND);
    "CODE" 34502;

    BOUNDS(7, A, RE, IM, 0, 0, RECENTRE, IMCENTRE, BOUND);
    OUTPUT(61,"("2/,"("REAL AND IMAG. PART OF CENTRE + RADIUS")",/")");
    "FOR" J:= 1 "STEP" 1 "UNTIL" 7 "DO"
    OUTPUT(61,"("/,3(N)")",RECENTRE[J],IMCENTRE[J],BOUND[J])
  "END"
"END"
```

RESULTS :

COEFFICIENTS OF POLYNOMIAL:

```
   1     -3     -3     25     -46     38     -12      0
```

```
NUMBER NOT FOUND ZEROS     0
FAIL INDICATION     0
NUMBER NEW STARTS     0
NUMBER OF ITERATIONS    11
```

ZEROS:

```
+2.0000000000000"+000
-3.0000000000000"+000
+1.0000000000000"+000  -1.0000000000000"+000
+1.0000000000000"+000  +1.0000000000000"+000
+1.0000000083024"+000
+9.9999999169752"-001
+0.0000000000000"+000
```

REAL AND IMAG. PART OF CENTRE + RADIUS

```
+2.0000000000000"+000  +0.0000000000000"+000  +1.3238111117716"-011
-3.0000000000000"+000  +0.0000000000000"+000  +3.8857510604494"-013
+1.0000000000000"+000  -1.0000000000000"+000  +4.0912729775463"-012
+1.0000000000000"+000  +1.0000000000000"+000  +4.0912729775463"-012
+9.9999999999998"-001  +0.0000000000000"+000  +2.2533888428865"-006
+9.9999999999998"-001  +0.0000000000000"+000  +2.2533888428865"-006
+0.0000000000000"+000  +0.0000000000000"+000  +0.0000000000000"+000
```

SOURCE TEXT(S) :

```
"CODE" 34501;
"INTEGER""PROCEDURE" ZERPOL(N, A, EM, RE, IM, D);
  "VALUE" N; "INTEGER" N; "ARRAY" A, EM, RE, IM, D;
"BEGIN" "INTEGER" I, TOTIT, IT, FAIL, START, UP, MAX, GIEX, ITMAX;
   "REAL" X, Y, NEWF, OLDF, MAXRAD, AE, TOL, H1, H2, LN2;
   "ARRAY" F[0 : 5], TRIES[1 : 10];

  "REAL""PROCEDURE" DWARF; "CODE" 30003;
  "REAL""PROCEDURE" GIANT; "CODE" 30004;
  "REAL""PROCEDURE" COMABS(XR, XI); "CODE" 34340;
  "PROCEDURE" COMSQRT(AR, AI, PR, PI); "CODE" 34343;

  "BOOLEAN""PROCEDURE" FUNCTION;
  "BEGIN" "INTEGER" K, M1, M2;
    "REAL" P, Q, QSQRT, F01, F02, F03, F11, F12, F13,
           F21, F22, F23, STOP;
    IT:= IT + 1;
    P:= 2 * X; Q:= -(X * X + Y * Y); QSQRT:= SQRT(-Q);
    F01:= F11:= F21:= D[0]; F02:= F12:= F22:= 0;
    M1:= N - 4; M2:= N - 2;
    STOP:= ABS(F01) * 0.8;
    "FOR" K:= 1 "STEP" 1 "UNTIL" M1 "DO"
    "BEGIN" F03:= F02; F02:= F01; F01:= D[K] + P * F02 + Q * F03;
      F13:= F12; F12:= F11; F11:= F01 + P * F12 + Q * F13;
      F23:= F22; F22:= F21; F21:= F11 + P * F22 + Q * F23;
      STOP:= QSQRT * STOP + ABS(F01)
    "END";
    "IF" M1 < 0 "THEN" M1:= 0;
    "FOR" K:= M1 + 1 "STEP" 1 "UNTIL" M2 "DO"
    "BEGIN" F03:= F02; F02:= F01; F01:= D[K] + P * F02 + Q * F03;
      F13:= F12; F12:= F11; F11:= F01 + P * F12 + Q * F13;
      STOP:= QSQRT * STOP + ABS(F01)
    "END";
    "IF" N = 3 "THEN" F21:= 0;
    F03:= F02; F02:= F01; F01:= D[N - 1] + P * F02 + Q * F03;
    F[0]:= D[N] + X * F01 + Q * F02;
    F[1]:= Y * F01;
    F[2]:= F01 - 2 * F12 * Y * Y;
    F[3]:= 2 * Y * (- X * F12 + F11);
    F[4]:= 2 * (- X * F12 + F11) - 8 * Y * Y * (- X * F22 + F21);
    F[5]:= Y * (6 * F12 - 8 * Y * Y * F22);
    STOP:= QSQRT * (QSQRT * STOP + ABS(F01)) + ABS(F[0]);
    NEWF:= F02:= COMABS(F[0], F[1]);
    FUNCTION:= F02 < (2 * ABS(X * F01) - 8 * (ABS(F[0]) + ABS(F01)
       * QSQRT) + 10 * STOP) * TOL * (1 + TOL) ** (4 * N + 3)
  "END" OF FUNCTION;                                          "COMMENT"
```

```
  "BOOLEAN""PROCEDURE" CONTROL;
  "IF" IT > ITMAX "THEN"
  "BEGIN" TOTIT:= TOTIT + IT; FAIL:= 3; "GOTO" EXIT "END"
  "ELSE" "IF" IT = 0 "THEN"
  "BEGIN" "INTEGER" I, H; "REAL" H1, SIDE;
    MAXRAD:= 0; MAX:= (GIEX - LN(ABS(D[0])) / LN2) / N;
    "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
    "BEGIN" H1:= "IF" D[I] = 0 "THEN" 0
      "ELSE" EXP(LN(ABS(D[I] / D[0])) / I);
      "IF" H1 > MAXRAD "THEN" MAXRAD:= H1
    "END";
    "FOR" I:= 1 "STEP" 1 "UNTIL" N - 1 "DO"
    "IF" D[I] ^= 0 "THEN"
    "BEGIN" H:= (GIEX - LN(ABS(D[I])) / LN2) / (N - I);
      "IF" H < MAX "THEN" MAX:= H
    "END";
    MAX:= MAX * LN2 / LN(N);
    SIDE:= - D[1] / D[0];
    SIDE:= "IF" ABS(SIDE) < TOL "THEN" 0 "ELSE" SIGN(SIDE);
    "IF" SIDE = 0 "THEN"
    "BEGIN" TRIES[7]:= TRIES[2]:= MAXRAD; TRIES[9]:= -MAXRAD;
      TRIES[4]:= TRIES[4]:= TRIES[3]:= MAXRAD / SQRT(2);
      TRIES[5]:= -TRIES[3]; TRIES[10]:= TRIES[8]:= TRIES[1]:= 0
    "END" "ELSE"
    "BEGIN" TRIES[8]:= TRIES[4]:= MAXRAD/ SQRT(2);
       TRIES[1]:= SIDE * MAXRAD; TRIES[3]:= TRIES[4] * SIDE;
       TRIES[6]:= MAXRAD; TRIES[7]:= -TRIES[3];
       TRIES[9]:= -TRIES[1]; TRIES[2]:= TRIES[5]:= TRIES[10]:= 0
    "END";
    "IF" COMABS(X, Y) > 2 * MAXRAD "THEN" X:= Y:= 0;
    CONTROL:= "FALSE"
  "END" "ELSE"
  "BEGIN" "IF" IT > 1 & NEWF >= OLDF "THEN"
    "BEGIN" UP:= UP+ 1;
      "IF" UP = 5 & START < 5 "THEN"
      "BEGIN" START:= START + 1; UP:= 0; X:= TRIES[2 * START - 1];
        Y:= TRIES[2 * START]; CONTROL:= "FALSE"
      "END" "ELSE" CONTROL:= "TRUE"
    "END" "ELSE" CONTROL:= "TRUE"
  "END" OF CONTROL;                                          "COMMENT"
```

```
;

  "PROCEDURE" DEFLATION;
  "IF" X = 0 & Y = 0 "THEN" N:= N - 1 "ELSE"
  "BEGIN" "INTEGER" I, SPLIT; "REAL" H1, H2;
    "ARRAY" B[0 : N - 1];
    "IF" Y = 0 "THEN"
    "BEGIN" N:= N - 1; B[N]:= -D[N + 1] / X;
      "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
      B[N - I]:= (B[N - I + 1] - D[N - I + 1]) / X;
      "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
      D[I]:= D[I] + D[I - 1] * X
    "END" "ELSE"
    "BEGIN" H1:= - 2 * X; H2:= X * X + Y * Y;
      N:= N - 2;
      B[N]:= D[N + 2] / H2; B[N - 1]:= (D[N + 1] - H1 * B[N]) / H2;
      "FOR" I:= 2 "STEP" 1 "UNTIL" N "DO"
      B[N - I]:=  (D[N - I + 2] - H1 * B[N - I + 1] - B[N - I + 2])/H2;
      D[1]:= D[1] - H1 * D[0];
      "FOR" I:= 2 "STEP" 1 "UNTIL" N "DO"
      D[I]:= D[I] - H1 * D[I-1] - H2 * D[I-2]
    "END";
    SPLIT:= N;
    H2:= ABS(D[N] - B[N]) / (ABS(D[N]) + ABS(B[N]));
    "FOR" I:= N - 1 "STEP" -1 "UNTIL" 0 "DO"
    "BEGIN" H1:= ABS(D[I]) + ABS(B[I]);
      "IF" H1 > TOL "THEN"
      "BEGIN" H1:= ABS(D[I] - B[I]) / H1;
        "IF" H1 < H2 "THEN" "BEGIN" H2:= H1; SPLIT:= I "END"
      "END"
    "END";
    "FOR" I := SPLIT + 1 "STEP" 1 "UNTIL" N "DO" D[I]:= B[I];
    D[SPLIT]:= (D[SPLIT] + B[SPLIT]) / 2
  "END" OF DEFLATION;

  "PROCEDURE" LAGUERRE;
  "BEGIN" "INTEGER" M;
    "REAL" S1RE, S1IM, S2RE, S2IM, DX, DY, H1, H2, H3, H4, H5, H6;
    "IF" ABS(F[0]) > ABS(F[1]) "THEN"
    "BEGIN" H1:= F[0]; H6:= F[1] / H1; H2:= F[2] + H6 * F[3];
      H3:= F[3] - H6 * F[2]; H4:= F[4] + H6 * F[5];
      H5:= F[5] - H6 * F[4]; H6:= H6 * F[1] + H1
    "END" "ELSE"
    "BEGIN" H1:= F[1]; H6:= F[0] / H1; H2:= H6 * F[2] + F[3];
      H3:= H6 * F[3] - F[2]; H4:= H6 * F[4] + F[5];
      H5:= H6 * F[5] - F[4]; H6:= H6 * F[0] + F[1]
    "END";                                                    "COMMENT"
```

```
    S1RE:= H2 / H6; S1IM:= H3 / H6;
    H2:= S1RE * S1RE - S1IM * S1IM; H3:= 2 * S1RE * S1IM;
    S2RE:= H2 - H4 / H6; S2IM:= H3 - H5 / H6;
    H1:= S2RE * S2RE + S2IM * S2IM;
    H1:= "IF" H1 ^= 0 "THEN" (S2RE * H2 + S2IM * H3) / H1 "ELSE" 1;
    M:= "IF" H1 >= N - 1 "THEN" ("IF" N > 1 "THEN" N - 1 "ELSE" 1)
        "ELSE" "IF" H1 > 1 "THEN" H1 "ELSE" 1;
    H1:= (N - M) / M;
    COMSQRT(H1 * (N * S2RE - H2), H1 * (N * S2IM - H3), H2, H3);
    "IF" S1RE * H2 + S1IM * H3 < 0 "THEN"
    "BEGIN" H2:= - H2; H3:= - H3 "END";
    H2:= S1RE + H2; H3:= S1IM + H3;
    H1:= H2 * H2 + H3 * H3;
    "IF" H1 = 0 "THEN" "BEGIN" DX:= -N; DY:= N "END" "ELSE"
    "BEGIN" DX:= - N * H2 / H1; DY:= N * H3 / H1 "END";
    H1:= ABS(X) * TOL + AE; H2:= ABS(Y) * TOL+ AE;
    "IF" ABS(DX) < H1 & ABS(DY) < H2 "THEN"
    "BEGIN" DX:= "IF" DX = 0 "THEN" H1 "ELSE" SIGN(DX) * H1;
      DY:= "IF" DY = 0 "THEN" H2 "ELSE" SIGN(DY) * H2
    "END";
    X:= X + DX; Y:= Y + DY;
    "IF" COMABS(X, Y) > 2 * MAXRAD "THEN"
    "BEGIN" H1:= "IF" ABS(X) > ABS(Y) "THEN" ABS(X) "ELSE" ABS(Y);
      H2:= LN(H1) / LN2 + 1 - MAX;
      "IF" H2 > 0 "THEN"
      "BEGIN" H2:= 2 ** H2; X:= X / H2; Y:= Y / H2 "END"
    "END"
  "END" OF LAGUERRE;

  TOTIT:= IT:= FAIL:= UP:= START:= 0; LN2:= LN(2);
  NEWF:= GIANT; AE:= DWARF; GIEX:= LN(NEWF) / LN2 - 40;
  TOL:= EM[0]; ITMAX:= EM[1];
  "FOR" I:= 0 "STEP" 1 "UNTIL" N "DO" D[I]:= A[N-I];
  "IF" N <= 0 "THEN"
  "BEGIN" FAIL:= 1; "GOTO" EXIT "END"
  "ELSE" "IF" D[0] = 0 "THEN"
  "BEGIN" FAIL:= 2; "GOTO" EXIT "END";
  "FOR" I:= 1 "WHILE" D[N] = 0 & N > 0 "DO"
  "BEGIN" RE[N]:= IM[N]:= 0; N:= N - 1 "END";
  X:= Y:= 0;                                                    "COMMENT"
```

```
  "FOR" I:= 1 "WHILE" N > 2 "DO"
  "BEGIN" "IF" CONTROL "THEN" LAGUERRE;
    OLDF:= NEWF;
    "IF" FUNCTION "THEN"
    "BEGIN" "IF" Y ^= 0 & ABS(Y) < .1 "THEN"
      "BEGIN" "REAL" H; H:= Y; Y:= 0;
         "IF" ^ FUNCTION "THEN" Y:= H
      "END";
      RE[N]:= X; IM[N]:= Y;
      "IF" Y ^= 0 "THEN" "BEGIN" RE[N - 1]:= X; IM[N - 1]:= -Y "END";
      DEFLATION; TOTIT:= TOTIT + IT; UP:= START:= IT:= 0
    "END"
  "END";
  "IF" N = 1 "THEN" "BEGIN" RE[1]:= - D[1] / D[0]; IM[1]:= 0 "END"
  "ELSE"
  "BEGIN" "REAL" H1, H2;
    H1:= - 0.5 * D[1] / D[0]; H2:= H1 * H1 - D[2] / D[0];
    "IF" H2 >= 0 "THEN"
    "BEGIN" RE[2]:= "IF" H1 < 0 "THEN" H1 - SQRT(H2)
            "ELSE" H1 + SQRT(H2);
      RE[1]:= D[2] / (D[0] * RE[2]);
      IM[2]:= IM[1]:= 0
    "END" "ELSE"
    "BEGIN" RE[2]:= RE[1]:= H1;
      IM[2]:= SQRT(-H2); IM[1]:= -IM[2]
    "END"
  "END"; N:= 0;
EXIT:  EM[2]:= FAIL; EM[3]:= START; EM[4]:= TOTIT;
  "FOR" I:= (N-1) "DIV" 2 "STEP" -1 "UNTIL" 0 "DO"
  "BEGIN" TOL := D[I]; D[I]:= D[N-I]; D[N-I]:= TOL
  "END";
  ZERPOL:= N
"END" OF ZERPOL;
        "EOP"


"CODE" 34502;
"PROCEDURE" BOUNDS(N,A,RE,IM,RELE,ABSE,RECENTRE,IMCENTRE,BOUND);
  "VALUE" N, RELE, ABSE; "INTEGER" N; "REAL" RELE, ABSE;
  "ARRAY" RE, IM, A, RECENTRE, IMCENTRE, BOUND;
"BEGIN" "INTEGER" I, J, K, L, INDEX1, INDEX2; "BOOLEAN" GOON;
  "REAL" H, MIN, RECENT, IMCENT, GIA, XK, YK, ZK, CORR;
  "ARRAY" RC, C, RCE[0:N], CLUST[1:N];

  "REAL""PROCEDURE" GIANT; "CODE" 30004;
  "REAL""PROCEDURE" ARREB; "CODE" 30002;                    "COMMENT"
```

```
"REAL""PROCEDURE" G(RAD, RECENT, IMCENT, K, M);
  "VALUE" RAD, RECENT, IMCENT, K, M; "REAL" RAD, RECENT, IMCENT;
  "INTEGER" K, M;
"BEGIN" "REAL" S, H1, H2; "INTEGER" I;
  S:= SQRT(RECENT * RECENT + IMCENT * IMCENT) + RAD;
  H1:= RC[1]; H2:= RC[0];
  "FOR" I:= 2 "STEP" 1 "UNTIL" N "DO" H1:= H1*S + RC[I];
  "FOR" I:= 1 "STEP" 1 "UNTIL" M-1, M+K "STEP" 1 "UNTIL" N "DO"
  H2:= H2 * ABS(SQRT((RE[I]-RECENT)**2 + (IM[I]-IMCENT)**2) - RAD);
  G:= "IF" H1=0 "THEN" 0 "ELSE" "IF" H2=0 "THEN" -10 "ELSE" H1 / H2
"END";

"PROCEDURE" KCLUSTER(K, M);
  "VALUE" K, M; "INTEGER" K, M;
"BEGIN" "INTEGER" I, J, STOP, L; "BOOLEAN" NONZERO;
  "REAL" RECENT, IMCENT, D, PROD, RAD, GR, R;
  "ARRAY" DIST[M: M+K-1];
  RECENT:= RE[M]; IMCENT:= IM[M]; STOP:= M+K-1;
  L:= SIGN(IMCENT); NONZERO:= L ^= 0;
  "FOR" I:= M+1 "STEP" 1 "UNTIL" STOP "DO"
  "BEGIN" RECENT:= RECENT+RE[I];
    "IF" NONZERO "THEN"
    "BEGIN" NONZERO:= L = SIGN(IM[I]); IMCENT:= IMCENT+IM[I] "END"
  "END";
  RECENT:= RECENT/K; IMCENT:= "IF" NONZERO "THEN" IMCENT/K "ELSE" 0;
  D:= 0; RAD:= 0;
  "FOR" I:= M "STEP" 1 "UNTIL" STOP "DO"
  "BEGIN" RECENTRE[I]:= RECENT; IMCENTRE[I]:= IMCENT;
    DIST[I]:= SQRT((RE[I] -RECENT)**2 + (IM[I]-IMCENT)**2);
    "IF" D < DIST[I] "THEN" D:= DIST[I]
  "END";
  GR:= ABS(G(0, RECENT, IMCENT, K, M));
  "IF" GR > 0 "THEN"
  "BEGIN" "FOR" J:= 1, 1 "WHILE" PROD <= GR "DO"
    "BEGIN" R:= RAD; RAD:= D + EXP(LN(1.1*GR)/K);
      "IF" RAD = R "THEN" RAD:= EXP(LN(1.1)/K) * RAD;
      GR:= G(RAD, RECENT, IMCENT, K, M);
      PROD:= 1;
      "FOR" I:= M "STEP" 1 "UNTIL" STOP "DO"
      PROD:= PROD*(RAD-DIST[I])
    "END"
  "END";
  "FOR" I:= M "STEP" 1 "UNTIL" STOP "DO"
  "BEGIN" BOUND[I]:= RAD; CLUST[I]:= K "END";
"END";                                                    "COMMENT"
```

```
"PROCEDURE" SHIFT(INDEX, NEW);
  "VALUE" INDEX, NEW; "INTEGER" INDEX, NEW;
"BEGIN" "INTEGER" J, PLACE, CLUSTIN;
  "REAL" BOUNDIN, IMCENT, RECENT;
  "REAL""ARRAY" WA1, WA2[1:CLUST[INDEX]];
  CLUSTIN:= CLUST[INDEX]; BOUNDIN:= BOUND[INDEX];
  IMCENT:= IMCENTRE[INDEX]; RECENT:= RECENTRE[INDEX];
  "FOR" J:= 1 "STEP" 1 "UNTIL" CLUSTIN "DO"
  "BEGIN" PLACE:=INDEX+J-1; WA1[J]:= RE[PLACE]; WA2[J]:= IM[PLACE];
  "END";
  "FOR" J:= INDEX-1 "STEP" -1 "UNTIL" NEW "DO"
  "BEGIN" PLACE:= J+CLUSTIN;
    RE[PLACE]:= RE[J]; IM[PLACE]:= IM[J]; CLUST[PLACE]:= CLUST[J];
    BOUND[PLACE]:= BOUND[J]; RECENTRE[PLACE]:= RECENTRE[J];
    IMCENTRE[PLACE]:= IMCENTRE[J]
  "END";
  "FOR" J:= NEW+CLUSTIN-1 "STEP" -1 "UNTIL" NEW "DO"
  "BEGIN" PLACE:= J+1-NEW;
    RE[J]:= WA1[PLACE]; IM[J]:= WA2[PLACE];
    BOUND[J]:= BOUNDIN; CLUST[J]:= CLUSTIN;
    RECENTRE[J]:= RECENT; IMCENTRE[J]:= IMCENT
  "END"
"END";

GIA:= GIANT;
RC[0]:= C[0]:= A[N]; RCE[0]:= ABS(C[0]); K:= 0;
"FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
"BEGIN" RC[I]:= RCE[I]:= 0 ; C[I]:= A[N-I] "END";
"FOR" I:= 0 "WHILE" K < N "DO"
"BEGIN" K:= K+1; XK:= RE[K]; YK:= IM[K]; ZK:= XK*XK+YK*YK;
  "FOR" J:= K "STEP" -1 "UNTIL" 1 "DO"
  RCE[J]:= RCE[J]+RCE[J-1]*SQRT(ZK);
  "IF" YK = 0 "THEN"
  "BEGIN" "FOR" J:= K "STEP" -1 "UNTIL" 1 "DO"
     RC[J]:= RC[J]-XK*RC[J-1]
  "END" "ELSE"
  "BEGIN" K:= K+1;
    "IF" K <= N & XK = RE[K] & YK = -IM[K] "THEN"
    "BEGIN" XK:= -2*XK;
      "FOR" J:= K "STEP" -1 "UNTIL" 1 "DO"
      RCE[J]:= RCE[J]+RCE[J-1]*SQRT(ZK);
      "FOR" J:= K "STEP" -1 "UNTIL" 2 "DO"
      RC[J]:= RC[J]+XK*RC[J-1]+ZK*RC[J-2];
      RC[1]:= RC[1]+XK*RC[0]
    "END"
  "END"
"END";
RC[0]:= RCE[0]; CORR:= 1.06*ARREB;
"FOR" I:= 1 "STEP" 1 "UNTIL" N-1 "DO"
RC[I]:= ABS(RC[I]-C[I])+RCE[I]*CORR*(N+I-2)+RELE*ABS(C[I])+ABSE;
RC[N]:= ABS(RC[N]-C[N])+RCE[N]*CORR*(N-1)+RELE*ABS(C[N])+ABSE;
"FOR" I:= 1 "STEP" 1 "UNTIL" N "DO" KCLUSTER(1, I);          "COMMENT"
```

```
  GOON:= "TRUE";
  "FOR" L:= 1 "WHILE" GOON "DO"
  "BEGIN" INDEX1:= INDEX2:= 0; MIN:= GIANT; I:= N-CLUST[N]+1;
    "FOR" I:= I "WHILE" I >= 2 "DO"
    "BEGIN" J:= I; RECENT:= RECENTRE[I]; IMCENT:= IMCENTRE[I];
      "FOR" J:= J "WHILE" J >= 2 "DO"
      "BEGIN" J:= J-CLUST[J-1];
        H:= SQRT((RECENT-RECENTRE[J])**2 + (IMCENT-IMCENTRE[J])**2);
        "IF" H < BOUND[I] + BOUND[J] & H <= MIN "THEN"
        "BEGIN" INDEX1:= J; INDEX2:= I; MIN:= H "END"
      "END"; I:= I-CLUST[I-1]
    "END";
    "IF" INDEX1 = 0 "THEN" GOON:= "FALSE" "ELSE"
    "BEGIN" "IF" IMCENTRE[INDEX1] = 0 "THEN"
      "BEGIN" "IF" IMCENTRE[INDEX2] ^= 0 "THEN"
        CLUST[INDEX2]:= 2*CLUST[INDEX2]
      "END" "ELSE" "IF" IMCENTRE[INDEX2] = 0 "THEN"
      CLUST[INDEX1]:= 2*CLUST[INDEX1];
      K:= INDEX1+CLUST[INDEX1];
      "IF" K ^= INDEX2 "THEN" SHIFT(INDEX2, K);
      K:= CLUST[INDEX1]+CLUST[K];
      KCLUSTER(K, INDEX1)
    "END"
  "END"
"END";
        "EOP"
```

AUTHORS: M. BAKKER, P.J. HARINGHUIZEN AND C.G. VAN DER LAAN.

CONTRIBUTORS: M. BAKKER, P.J. HARINGHUIZEN,
C.G. VAN DER LAAN AND M. VOORINTHOLT.

INSTITUTE: MATHEMATICAL CENTRE AND RIJKSUNIVERSITEIT GRONINGEN.

RECEIVED: 780601.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS FIVE PROCEDURES FOR CALCULATING ZEROS OF ORTHOGONAL POLYNOMIALS WHICH ARE GIVEN BY THE COEFFICIENTS OF THEIR RECURRENCE RELATION:

ALLZERORTPOL : CALCULATES ALL ZEROS,
LUPZERORTPOL : CALCULATES A NUMBER OF ADJACENT UPPER OR LOWER ZEROS,
SELZERORTPOL : CALCULATES A NUMBER OF ADJACENT ZEROS. IT IS EFFICIENT TO USE ALLZERORTPOL IF MORE THAN 50 PERCENT OF EXTREME ZEROS OR MORE THAN 25 PERCENT OF SELECTED ZEROS ARE WANTED.
ALLJACZER : CALCULATES THE ZEROS OF THE N-TH JACOBIAN POLYNOMIAL.
ALLLAGZER : CALCULATES THE ZEROS OF THE N-TH LAGUERRE POLYNOMIAL.

KEYWORDS:

ZEROS,
ORTHOGONAL POLYNOMIALS,
CHRISTOFFEL ABSCISSAS.

REFERENCES:

ABRAMOWITZ, M. AND I.A. STEGUN (1964):
HANDBOOK OF MATHEMATICAL FUNCTIONS.
DOVER PUBLICATIONS INC.

GOLUB, G.H. AND J.H. WELSCH (1969):
CALCULATION OF GAUSS QUADRATURE RULES.
MATH. COMP. VOL. 23, P.221-230.

LANCZOS, C. (1957):
APPLIED ANALYSIS.
PRENTICE HALL.

STOER, J. (1972):
EINFUEHRUNG IN DIE NUMERISCHE MATHEMATIK 1.
HEIDELBERG TASCHENBUECHER 105, SPRINGER.

WILKINSON,J AND REINSCH,C. :
HANDBOOK OF AUTOMATIC COMPUTATION. VOL. 2.
LINEAR ALGEBRA
HEIDELBERG (1971).

SUBSECTION: ALLZERORTPOL.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" ALLZERORTPOL (N, B, C, ZER, EM);
"VALUE" N; "INTEGER" N; "ARRAY" B, C, ZER, EM;
"CODE" 31362;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
ENTRY: THE DEGREE OF THE ORTHOGONAL POLYNOMIAL OF WHICH THE ZEROS ARE TO BE CALCULATED;

B, C: <ARRAY IDENTIFIER>;
"ARRAY" B, C [0:N-1];
ENTRY: THE ELEMENTS B[I] AND C[I], I = 0, 1, ... , N-1, CONTAIN THE COEFFICIENTS OF THE RECURRENCE RELATION P[I+1](X) = (X - B[I]) * P[I](X) - C[I] * P[I-1](X) I = 0, 1, ... , N-1; ASSUMED IS C[0]=0, WHILE THE CONTENTS OF THE ARRAYS ARE PRESERVED;

ZER: <ARRAY IDENTIFIER>;
"ARRAY" ZER[1:N];
EXIT: THE ZEROS OF THE N-TH DEGREE ORTHOGONAL POLYNOMIAL; (B MAY BE USED FOR ZER, BUT THEN THIS RECURRENCE COEFFICIENTS ARE OVERWRITTEN BY THE ZEROS AND THE ORIGINAL CONTENTS OF B ARE NOT PRESERVED.)

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:5];
ENTRY: EM[0]: THE MACHINE PRECISION;
EM[2]: THE RELATIVE TOLERANCE OF THE ZEROS;
EM[4]: THE MAXIMUM ALLOWED NUMBER OF ITERATIONS, E.G. 5 * N;
EXIT: EM[1]: THE MAXIMUM OF ABS(B[0])+1,C[I]+ABS(B[I])+1, (I=1,...N-2),C[N-1]+ABS(B[N-1]);
EM[3]: INFORMATION CONCERNING THE PROCESS USED: I.E. THE MAXIMUM ABSOLUTE VALUE OF THE CODIAGONAL ELEMENTS NEGLECTED, (SEE ALSO SECTION 3.3.1.1.1.);
EM[5]: THE NUMBER OF ITERATIONS PERFORMED.

PROCEDURES USED:

ORIVALSYMTRI = CP34160,
DUPVEC = CP31030.

METHOD AND PERFORMANCE : SEE SELZERORTPOL (THIS SECTION).

EXAMPLE OF USE:

AS A FORMAL TEST OF THE PROCEDURE WE CALCULATE THE ZEROS OF
THE CHEBYSHEV POLYNOMIAL (OF THE FIRST KIND) OF THE THIRD DEGREE.
THE RECURRENCE COEFFICIENTS ARE:
B[I] = 0, I = 0, 1, ...;
C[0] = 0, C[1] = .5, C[I] = .25 , I = 2, 3, ...
(IT IS RECOMMENDED TO STORE THE ELEMENTS OF THE ARRAYS B AND C IN
REVERSED ORDER IF THESE ELEMENTS ARE STRONGLY INCREASING).

```
"BEGIN" "ARRAY" B, C[0:3], ZER[1:3], EM[0:5];
    "PROCEDURE" ALLZERORTPOL(N,B,C,ZER,EM);
    "VALUE" N; "INTEGER" N; "ARRAY" B,C,ZER,EM;
    "CODE" 31362;
    EM[0]:= EM[2]:= "-14; EM[4]:=15;
    B[2]:=B[1]:=B[0]:=0;
    C[0]:= 0; C[1]:= .5; C[2]:= .25;
    ALLZERORTPOL (3, B, C, ZER, EM);
    OUTPUT(61,"("("THE THREE ZEROS:")",/,3(/ZD5B,N),2/,
              "("EM[1]:")",5BD.2D"+2D, /,
              "("EM[3]:")",5BD.2D"+2D, /,"("EM[5]:")",5ZD")",
              1,ZER[1],2,ZER[2],3,ZER[3],EM[1],EM[3],EM[5])
"END"
```

THE THREE ZEROS:

```
 1     -8.6602540378444"-001
 2     +8.6602540378444"-001
 3     -1.0000000000000"-014

EM[1]:     1.50"+00
EM[3]:     7.07"-15
EM[5]:     1
```

SUBSECTION: LUPZERORTPOL.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" LUPZERORTPOL (N, M, B, C, ZER, EM);
"VALUE" N, M; "INTEGER" N, M; "ARRAY" B, C, ZER, EM;
"CODE" 31363;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
ENTRY: THE DEGREE OF THE ORTHOGONAL POLYNOMIAL OF WHICH
THE ZEROS ARE TO BE CALCULATED;

M: <ARITHMETIC EXPRESSION>;
ENTRY: THE NUMBER OF ZEROS TO BE CALCULATED;

B, C: <ARRAY IDENTIFIER>;
"ARRAY" B, C [0:N-1];
ENTRY: THE ELEMENTS B[I] AND C[I], I = 0, 1, ... , N-1,
CONTAIN THE COEFFICIENTS OF THE RECURRENCE RELATION
P[I+1](X) = (X - B[I]) * P[I](X) - C[I] * P[I-1](X)
I = 0, 1, ... , N-1;
ASSUMED IS C[0]=0, WHILE THE CONTENTS
OF THE ARRAYS ARE NOT PRESERVED;

ZER: <ARRAY IDENTIFIER>;
"ARRAY" ZER[1:M];
EXIT: THE M LOWEST ZEROS ARE DELIVERED;
IF HOWEVER THE ARRAY B[0:N-1] CONTAINED THE OPPOSIT
VALUES OF THE CORRESPONDING RECURRENCE COEFFICIENTS
THEN THE OPPOSITE VALUES OF THE M UPPER ZEROS
ARE DELIVERED.
IN EITHER CASE, ZER[I]<ZER[I+1], I = 1,...,M-1;

EM: <ARRAY IDENTIFIER>;
"ARRAY" EM[0:6];
ENTRY: EM[0]: THE MACHINE PRECISION.
EM[2]: THE RELATIVE TOLERANCE OF THE ZEROS;
EM[4]: THE MAXIMUM ALLOWED NUMBER OF ITERATIONS,
E.G. 15 * M;
EM[6]: IF ALL ZEROS ARE KNOWN TO BE POSITIVE
THEN 1 ELSE 0;
EXIT: EM[1]: THE MAXIMUM OF ABS(B[0]) + 1 ,
C[I]+ABS(B[I])+1,(I=1,...N-2),
C[N-1]+ABS(B[N-1]);
EM[3]: INFORMATION CONCERNING THE PROCESS USED,
I.E. THE MAXIMUM ABSOLUTE VALUE OF THE
THEORETICAL ERRORS OF THE ZEROS
(SEE WILKINSON AND REINSCH, 1971, P.263);
EM[5]: THE NUMBER OF ITERATIONS PERFORMED.

PROCEDURES USED:

DUPVEC = CP31030,
INFNRMVEC = CP31061.

METHOD AND PERFORMANCE : SEE SELZERORTPOL (THIS SECTION).

EXAMPLE OF USE:

WE CALCULATE THE TWO LOWER AND THE TWO UPPER ZEROS OF THE
LAGUERRE POLYNOMIAL OF THE THIRD DEGREE.
THE RECURRENCE COEFFICIENTS ARE OBTAINED FROM [1], P.782:
B[I] = - A2I / A3I = 2I + 1;
C[I] = A4I / (A3I * A3(I-1)) * A1(I-1) = I * I, I = 0, 1, 2.
(IT IS RECOMMENDED TO STORE THE ELEMENTS OF THE ARRAYS B AND C IN
REVERSED ORDER IF THESE ELEMENTS ARE STRONGLY DECREASING).

```
"BEGIN" "ARRAY" B, C[0:3], ZER[1:2], EM[0:6];
"INTEGER" I;
"PROCEDURE"LUPZERORTPOL(N,M,B,C,ZER,EM);
"VALUE" N,M; "INTEGER" N,M; "ARRAY" B,C,ZER,EM;
"CODE" 31363;
EM[0]:= EM[2]:= "-14; EM[4]:= 45; EM[6]:= 1;
"FOR" I:= 0, 1, 2 "DO"
"BEGIN" B[I]:= 2 * I + 1; C[I]:= I * I "END";
LUPZERORTPOL (3, 2, B, C, ZER, EM);
OUTPUT(61,"("""("THE TWO LOWER ZEROS:")",/,2(/ZD5B,N),2/,
          "("EM[1]:")",5BD.2D"+2D, /,
          "("EM[3]:")",5BD.2D"+2D, /,"("EM[5]:")",5ZD")",
          1,ZER[1],2,ZER[2],EM[1],EM[3],EM[5]);
EM[6]:= 0;
"FOR" I:= 0, 1, 2 "DO"
"BEGIN" B[I]:= - 2 * I - 1; C[I]:= I * I "END";
LUPZERORTPOL (3, 2, B, C, ZER, EM);
OUTPUT(61,"("3/,"("THE TWO UPPER ZEROS:")",/,2(/ZD5B,N),2/,
          "("EM[1]:")",5BD.2D"+2D, /,
          "("EM[3]:")",5BD.2D"+2D, /,"("EM[5]:")",5ZD")",
          1,-ZER[1],2,-ZER[2],EM[1],EM[3],EM[5]);
"END"
```

THE TWO LOWER ZEROS:

```
 1      +4.1577455678348"-001
 2      +2.2942803602791"+000

EM[1]:      9.00"+00
EM[3]:      5.72"-16
EM[5]:     12
```

```
    THE TWO UPPER ZEROS:

     1      +6.2899450829375"+000
     2      +2.2942803602791"+000

    EM[1]:      9.00"+00
    EM[3]:      4.70"-20
    EM[5]:     14
```

SUBSECTION: SELZERORTPOL.

CALLING SEQUENCE:

```
    THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

    "PROCEDURE" SELZERORTPOL (N, N1, N2, B, C, ZER, EM);
    "VALUE" N, N1, N2; "INTEGER" N, N1, N2; "ARRAY" B, C, ZER, EM;
    "CODE" 31364;

    THE MEANING OF THE FORMAL PARAMETERS IS :
    N,N1,N2:<ARITHMETIC EXPRESSION>;
            ENTRY:  N IS THE DEGREE OF THE ORTHOGONAL POLYNOMIAL OF
                    WHICH THE N1-ST UP TO AND INCLUDING N2-ND ZEROS ARE
                    TO BE CALCULATED(ZER[N1]>=ZER[N2]);
    B, C:   <ARRAY IDENTIFIER>;
            "ARRAY" B,C [0 : N-1];
            ENTRY:  THE ELEMENTS B[I] AND C[I], I = 0, 1, ... , N-1,
                    CONTAIN THE COEFFICIENTS OF THE RECURRENCE RELATION
                    P[I+1](X) = (X - B[I]) * P[I](X) - C[I] * P[I-1](X)
                    I = 0, 1, ... , N-1,
                    ASSUMED IS C[0]=0, WHILE THE CONTENTS
                    OF THE ARRAYS IS PRESERVED;
    ZER:    <ARRAY IDENTIFIER>;
            "ARRAY" ZER [N1:N2];
            EXIT:   THE N2-N1+1 CALCULATED ZEROS IN DECREASING ORDER.
    EM:     <ARRAY IDENTIFIER>;
            "ARRAY" EM[0:5];
            ENTRY:  EM[0]:  THE MACHINE PRECISION.
                    EM[2]:  THE RELATIVE TOLERANCE OF THE ZEROS;
            EXIT:   EM[1]:  THE MAXIMUM OF ABS(B[0]) + 1,
                            C[I]+ABS(B[I])+1 (I=1,...N-2) AND
                            C[N-1]+ABS(B[N-1]);
                    EM[5]:  THE NUMBER OF ITERATIONS PERFORMED.
```

PROCEDURES USED:

```
   VALSYMTRI        =  CP34151.
```

METHOD AND PERFORMANCE:

THE ZEROS OF AN ORTHOGONAL POLYNOMIAL ARE THE EIGENVALUES OF A SYMMETRIC TRIDIAGONAL MATRIX (SEE GOLUB AND WELSCH (1969), LANCZOS (1957),P.375,376, STOER (1972),P.120).
THE ORTHOGONAL POLYNOMIAL IS DEFINED BY A LINEAR THREE-TERM HOMOGENEOUS RECURRENCE RELATION.

EXAMPLE OF USE :

WE CALCULATE THE THIRD ZERO OF THE LEGENDRE POLYNOMIAL OF THE FOURTH DEGREE. THE RECURRENCE COEFFICIENTS ARE OBTAINED FROM ABRAMOWITZ AND STEGUN (1964),P.782:
B[I] = 0, I = 0, 1, ...;
C[I] = A4I / (A3I * A3(I-1)) * A1(I-1) = I * I / (4 * I * I - 1),
I = 0, 1, ...
(IT IS RECOMMENDED TO STORE THE ELEMENTS OF THE ARRAYS B AND C IN REVERSED ORDER IF THESE ELEMENTS ARE STRONGLY DECREASING).

```
"BEGIN" "ARRAY" B, C[0:4], ZER[3:3], EM[0:5];
"INTEGER" I;
"PROCEDURE" SELZERORTPOL (N,N1,N2,B,C,ZER,EM);
"VALUE" N,N1,N2; "INTEGER" N,N1,N2; "ARRAY" B,C,ZER,EM;
"CODE" 31364;
EM[0]:= EM[2]:= "-14;
"FOR" I:= 0, 1, 2, 3 "DO"
"BEGIN" B[I]:= 0; C[I]:= I * I / (4 * I * I - 1) "END";
SELZERORTPOL (4, 3, 3, B, C, ZER, EM);
OUTPUT(61,"("("THE THIRD ZERO:")",2/,ZD5B,N,2/,
          "("EM[1]:")",5BD.2D"+2D, /,
          "("EM[5]:")",5ZD")",3,ZER[3],EM[1],EM[5])
"END"
```

THE THIRD ZERO:

```
 3     -3.399810435848 6"-001

EM[1]:     1.33"+00
EM[5]:    12
```

SUBSECTION: ALL JAC ZER.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" ALL JAC ZER(N, ALFA, BETA, ZER);
"VALUE" N, ALFA, BETA;
"INTEGER" N; "REAL" ALFA, BETA; "ARRAY" ZER;
"CODE" 31370;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N:      <ARITHMETIC EXPRESSION>;
        THE UPPER BOUND OF THE ARRAY ZER; N >= 1;
ALFA,BETA:
        <ARITHMETIC EXPRESSION>;
        THE PARAMETERS OF THE JACOBI POLYNOMIAL,
        SEE ABRAMOWITZ AND STEGUN (1964);
        ALFA, BETA > - 1;
ZER:    <ARRAY IDENTIFIER>;
        "ARRAY" ZER[1 : N];
        EXIT: ZER[1], ..., ZER[N] ARE THE ZEROS OF THE N-TH
              JACOBI POLYNOMIAL WITH PARAMETERS ALFA AND BETA.
```

PROCEDURES USED:

ALL ZER ORT POL = CP 31362.

REQUIRED CENTRAL MEMORY:

IF ALFA = BETA THEN TWO AUXILIARY ARRAYS OF N//2 REALS ARE USED, OTHERWISE TWO AUXILIARY ARRAYS OF N REALS ARE DECLARED.

METHOD AND PERFORMANCE:

THE JACOBI POLYNOMIALS ARE A SPECIAL CASE OF ORTHOGONAL POLYNOMIALS (SEE ABRAMOWITZ AND STEGUN (1964)); ALL JAC ZER COMPUTES THE COEFFICIENTS OF THE THREE-TERM RECURRENCE RELATION AND CALLS THE PROCEDURE ALL ZER ORT POL TO COMPUTE THE ZEROS; IF ALFA=BETA, THE POLYNOMIALS ARE ODD OR EVEN, HENCE ONLY THE POSITIVE ZEROS ARE CALCULATED; THIS IS DONE BY MEANS OF THE FORMULAS

$$P(2*M, ALFA, ALFA, X) = C(M)*P(M, ALFA, -0.5, 2*X*X - 1),$$

$$P(2*M - 1, ALFA, ALFA, X) = D(M)*P(M, ALFA, +0.5, 2*X*X - 1)*X$$

(SEE ABRAMOWITZ AND STEGUN (1964), FORMULAS 22.5.20 - 22.5.27).

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "ARRAY" X[1:3];
  "PROCEDURE" ALL JAC ZER(N, ALFA, BETA, ZER);
  "VALUE" N,ALFA,BETA; "INTEGER" N;
  "REAL" ALFA,BETA; "ARRAY" ZER;
  "CODE" 31370;
  ALL JAC ZER(3,-.5,-.5,X);
  OUTPUT(61,"("3(4B-D.13D"-ZD)")",X[1],X[2],X[3])
"END"
```

DELIVERS THE FOLOWING RESULTS:

```
-8.6602540378444"-1     0.0000000000000" 0      8.6602540378444"-1
```

SUBSECTION: ALL LAG ZER.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" ALL LAG ZER(N, ALFA, ZER);
"VALUE" N, ALFA;
"INTEGER" N; "REAL" ALFA; "ARRAY" ZER;
"CODE" 31371;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N:      <ARITHMETIC EXPRESSION>;
        THE UPPER BOUND OF THE ARRAY ZER; N >= 1;
ALFA:   <ARITHMETIC EXPRESSION>;
        THE PARAMETER OF THE LAGUERRE POLYNOMIAL,
        SEE ABRAMOWITZ AND STEGUN (1964); ALFA > -1;
ZER:    <ARRAY IDENTIFIER>;
        "ARRAY" ZER[1 : N];
        EXIT: ZER[1], ..., ZER[N] ARE THE ZEROS OF THE N-TH
              LAGUERRE POLYNOMIAL WITH PARAMETER ALFA.
```

PROCEDURES USED:

ALL ZER ORT POL = CP 31362.

REQUIRED CENTRAL MEMORY:

TWO AUXILIARY ARRAYS OF N REALS ARE USED.

METHOD AND PERFORMANCE:

THE LAGUERRE POLYNOMIALS ARE A SPECIAL CASE OF ORTHOGONAL POLYNOMIALS (SEE ABRAMOWITZ AND STEGUN (1964)); ALL LAG ZER COMPUTES THE COEFFICIENTS OF THE THREE-TERM RECURRENCE RELATION AND CALLS THE PROCEDURE ALL ZEP ORT POL TO COMPUTE THE ZEROS.

EXAMPLE OF USE:

```
"BEGIN" "ARRAY" X[1:3];
  "PROCEDURE" ALL LAG ZER(N, ALFA, ZER);
  "VALUE" N,ALFA; "INTEGER" N; "REAL" ALFA; "ARRAY" ZER;
  "CODE" 31371;
  ALL LAG ZER(3,-.5,X);
  OUTPUT(61,"("3(4B-D.13D"-ZD)")",X[1],X[2],X[3])
"END"
```

DELIVERS THE FOLOWING RESULTS:

```
5.5253437422633" 0.    1.7844927485432" 0     1.9016350919350" -1
```

SOURCE TEXT(S) :

```
"CODE"31362;
"PROCEDURE" ALLZERORTPOL (N, B, C, ZER, EM);
"VALUE" N; "INTEGER" N; "ARRAY" B, C, ZER, EM;
"BEGIN" "INTEGER"I;"REAL"NRM;"ARRAY"BB[1:N];
    "INTEGER" "PROCEDURE" QRIVALSYMTRI (D, BB, N, EM);
    "VALUE"N;"INTEGER"N;"ARRAY"D,BB,EM;
    "CODE" 34160;
    "PROCEDURE" DUPVEC (L, U, SHIFT, A, B);
    "VALUE"L,U,SHIFT;"INTEGER"L,U,SHIFT;"ARRAY"A,B;
    "CODE" 31030;
    "PROCEDURE" DUPCEV (L, U, SHIFT, A, B);
    "VALUE"L,U,SHIFT;"INTEGER"L,U,SHIFT;"ARRAY"A,B;
    "FOR" U:=U "STEP" -1 "UNTIL" L "DO" A[U]:=B[U+SHIFT];
    NRM:=ABS(B[0]);
    "FOR"I:=1"STEP"1"UNTIL"N-2"DO""IF"C[I]+ABS(B[I])>NRM"THEN"
         NRM:=C[I]+ABS(B[I]);
     "IF"N>1"THEN"NRM:="IF"NRM+1>=C[N-1]+ABS(B[N-1])"THEN"NRM+1"ELSE"
                       C[N-1]+ABS(B[N-1]);
     EM[1]:=NRM; DUPCEV(1,N,-1,ZER,B);
     DUPVEC(1,N-1,0,BB,C);BB[N]:=0;
     QRIVALSYMTRI(ZER,BB,N,EM)
"END" ALLZERORTPOL;
       "EOP"
```

```
"CODE"31363;
"PROCEDURE" LUPZERORTPOL (N, M, B, C, ZER, EM);
"VALUE" N, M; "INTEGER" N, M; "ARRAY" B, C, ZER, EM;
"BEGIN"
"PROCEDURE" RATQR(N,M,POSDEF,DLAM,EPS)TRANS:(D,B2);
  "VALUE" N,M,POSDEF,DLAM,EPS;
  "INTEGER" N,M;
  "BOOLEAN" POSDEF;
  "REAL" DLAM,EPS;
  "ARRAY" D,B2;
"COMMENT" QR ALGORITHM FOR THE COMPUTATION OF THE LOWEST EIGENVALUES
  OF A SYMMETRIC TRIDIAGONAL MATRIX. A RATIONAL VARIANT OF THE
  QR TRANSFORMATION IS USED, CONSISTING OF TWO SUCCESSIVE QD STEPS
  PER ITERATION.
  A SHIFT OF THE SPECTRUM AFTER EACH ITERATION GIVES AN ACCELERATED
  RATE OF CONVERGENCE. A NEWTON CORRECTION,DERIVED FROM THE
  CHARACTERISTIC POLYNOMIAL,IS USED AS SHIFT.
   RATQR IS IMPLEMENTED BY REINSCH AND BAUER, SEE WILKINSON AND REINSCH
   ,1971, CONTR. II-6.
  FORMATS: D,B2[1:N];
"BEGIN"
  "INTEGER" I,J,K,T; "REAL" DELTA,E,EP,ERR,P,Q,QP,R,S,TOT;
  "REAL""PROCEDURE" INFNRMVEC(L,U,K,A);
  "VALUE" L,U; "INTEGER" L,U,K; "ARRAY" A;
  "CODE" 31061;
"COMMENT" LOWER BOUND FOR EIGENVALUES FROM GERSHGORIN, INITIAL SHIFT;
  B2[1]:= ERR:= Q:= S:= 0; TOT:= D[1];
  "FOR" I:= N "STEP" -1   "UNTIL" 1 "DO"
  "BEGIN"
    P:= Q; Q:= SQRT(B2[I]); E:= D[I]-P-Q;
    "IF" E < TOT "THEN" TOT:= E
  "END" I;
  "IF" POSDEF & TOT < 0 "THEN" TOT:= 0 "ELSE"
  "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO" D[I]:= D[I]-TOT;
  T:= 0;
  "FOR" K:= 1 "STEP" 1 "UNTIL" M "DO"
  "BEGIN"
NEXT QR TRANSFORMATION:  T:= T + 1;
  TOT:= TOT + S; DELTA:= D[N]-S; I:= N;
  E:= ABS(EPS*TOT); "IF" DLAM < E "THEN" DLAM:= E;
  "IF" DELTA < = DLAM "THEN" "GOTO" CONVERGENCE;
  E:= B2[N]/DELTA; QP:= DELTA+E; P:= 1;
  "FOR" I:= N-1 "STEP" -1 "UNTIL" K "DO"
  "BEGIN"
    Q:= D[I]-S-E; R:= Q/QP; P:= P*R+1;
    EP:= E*R; D[I+1]:= QP+EP; DELTA:= Q-EP;
    "IF" DELTA < = DLAM "THEN" "GOTO" CONVERGENCE;
    E:= B2[I]/Q;QP:= DELTA+E; B2[I+1]:= QP*EP
  "END" I;
  D[K]:= QP; S:= QP/P;
                                                    "COMMENT"
```

```
  "IF" TOT+S > TOT "THEN" "GOTO" NEXT QR TRANSFORMATION;
"COMMENT" IRREGULAR END OF ITERATION,
  DEFLATE MINIMUM DIAGONAL ELEMENT;
  S:= 0; I:= K; DELTA:= QP;
  "FOR" J:= K+1 "STEP" 1 "UNTIL" N "DO"
    "IF" D[J] < DELTA "THEN"
    "BEGIN" I:= J; DELTA:= D[J] "END";
CONVERGENCE:
  "IF" I < N "THEN" B2[I+1]:= B2[I]*E/QP;
    "FOR" J:= I-1 "STEP" -1 "UNTIL" K "DO"
      "BEGIN" D[J+1]:= D[J]-S; B2[J+1]:= B2[J] "END" J;
    D[K]:= TOT; B2[K]:= ERR:= ERR+ABS(DELTA)
  "END" K;
EM[5]:=T;EM[3]:=INFNRMVEC(1,M,T,B2);
"END" RATQR;
    "PROCEDURE" DUPCEV (L, U, SHIFT, A, B);
    "VALUE"L,U,SHIFT;"INTEGER"L,U,SHIFT;"ARRAY"A,B;
    "FOR" U:=U "STEP" -1 "UNTIL" L "DO" A[U]:=B[U+SHIFT];
    "PROCEDURE" DUPVEC (L, U, SHIFT, A, B);
    "VALUE"L,U,SHIFT;"INTEGER"L,U,SHIFT;"ARRAY"A,B;
    "CODE" 31303;
    "INTEGER" I;"REAL"NRM;
    NRM:=ABS(B[0]);
    "FOR"I:=1"STEP"1"UNTIL"N-2"DO""IF"C[I]+ABS(B[I])>NRM"THEN"
        NRM:=C[I]+ABS(B[I]);
     "IF"N>1"THEN"NRM:="IF"NRM+1>=C[N-1]+ABS(B[N-1])"THEN"NRM+1"ELSE"
                        C[N-1]+ABS(B[N-1]);
     EM[1]:=NRM;
    DUPCEV(1,N,-1,B,B);
    DUPCEV(2,N,-1,C,C);
    RATQR (N, M, EM[6] = 1, EM[2], EM[0], B, C);
    DUPVEC (1, M, 0, ZER, B)
"END" LUPZERORTPOL;
        "EOP"


"CODE"31364;
"PROCEDURE" SELZERORTPOL (N, N1, N2, B, C, ZER, EM);
"VALUE" N, N1, N2; "INTEGER" N, N1, N2; "ARRAY" B, C, ZER, EM;
"BEGIN" "INTEGER"I;"REAL"NRM;"ARRAY"D[1:N];
    "PROCEDURE" DUPCEV (L, U, SHIFT, A, B);
    "VALUE"L,U,SHIFT;"INTEGER"L,U,SHIFT;"ARRAY"A,B;
    "FOR" U:=U "STEP" -1 "UNTIL" L "DO" A[U]:=B[U+SHIFT];
    "PROCEDURE" VALSYMTRI (D, BB, N, N1, N2, VAL, EM);
    "VALUE"N,N1,N2;"INTEGER"N,N1,N2;"ARRAY"D,BB,VAL,EM;
    "CODE" 34151;
    NRM:=ABS(B[0]);
    "FOR"I:=N-2"STEP"-1"UNTIL"1"DO""IF"C[I]+ABS(B[I])>NRM"THEN"
        NRM:=C[I]+ABS(B[I]);
                                                        "COMMENT"
```

```
     "IF"N>1"THEN"NRM:="IF"NRM+1>=C[N-1]+ABS(B[N-1])"THEN"NRM+1"ELSE"
                          C[N-1]+ABS(B[N-1]);
     EM[1]:=NRM;
    DUPCEV(1,N,-1,D,B);
    VALSYMTRI (D, C, N, N1, N2, ZER, EM);
    EM[5]:=EM[3]
"END" SELZERORTPOL;
        "EOP"


 "CODE" 31370;
    "PROCEDURE" ALL JAC ZER(N, ALFA, BETA, ZER);
    "VALUE" N, ALFA, BETA      ; "INTEGER" N;
    "REAL" ALFA, BETA      ; "ARRAY" ZER;
    "IF" ALFA = BETA "THEN"
    "BEGIN" "INTEGER" I, M;
        "ARRAY" A, B[0:N//2], EM[0:5];
        "REAL" MIN, GAMMA, SUM, ZERI;
        "REAL""PROCEDURE" ARREB;
        "CODE" 30002;
        "PROCEDURE" ALL ZER ORT POL(N, B, C, ZER, EM);
        "VALUE" N; "INTEGER" N; "ARRAY" B,C,ZER,EM;
        "CODE" 31362;
        M:= N//2; "IF" N ^= 2*M "THEN"
        "BEGIN" GAMMA:= + 0.5; ZER[M + 1]:= 0 "END"
        "ELSE" GAMMA:= - 0.5;
        MIN:= 0.25 - ALFA*ALFA; SUM:= ALFA + GAMMA + 2;
        A[0]:= (GAMMA - ALFA)/SUM; A[1]:= MIN/SUM/(SUM + 2);
        B[1]:= 4*(1 + ALFA)*(1 + GAMMA)/SUM/SUM/(SUM + 1);
        "FOR" I:= 2 "STEP" 1 "UNTIL" M - 1 "DO"
        "BEGIN" SUM:= I + I + ALFA + GAMMA;
            A[I]:= MIN/SUM/(SUM + 2); SUM := SUM*SUM;
            B[I]:= 4*I*(I + ALFA + GAMMA)*(I + ALFA)*(I + GAMMA)/
            SUM/(SUM - 1)
        "END";
        EM[0]:=ARREB; EM[2]:="-10; EM[4]:= 6*M;
        ALL ZER ORT POL (M, A, B, ZER, EM);
        "FOR" I:= 1 "STEP" 1 "UNTIL" M "DO"
        "BEGIN" ZER[I]:= ZERI:= - SQRT((1 + ZER[I])/2);
            ZER[N + 1 - I]:= - ZERI
        "END"
    "END" "ELSE"
    "BEGIN" "INTEGER" I; "REAL" SUM, MIN;
        "ARRAY" A, B[0:N], EM[0:5];
        "REAL""PROCEDURE" ARREB;
        "CODE" 30002;
        "PROCEDURE" ALL ZER ORT POL(N, B, C, ZER, EM);
        "VALUE" N; "INTEGER" N; "ARRAY" B,C,ZER,EM;
        "CODE" 31362;
                                                        "COMMENT"
```

```
        MIN:= (BETA - ALFA)*(BETA + ALFA);
        SUM:= ALFA + BETA + 2; B[0]:= 0;
        A[0]:= (BETA - ALFA)/SUM;
        A[1]:= MIN/SUM/(SUM + 2);
        B[1]:= 4*(1 + ALFA)*(1 + BETA)/SUM/SUM/(SUM + 1);
        "FOR" I:= 2 "STEP" 1 "UNTIL" N - 1 "DO"
        "BEGIN" SUM:= I + I + ALFA + BETA;
            A[I]:= MIN/SUM/(SUM + 2); SUM:= SUM*SUM;
            B[I]:= 4*I*(I + ALFA + BETA)*(I + ALFA)*(I + BETA)/
            (SUM - 1)/SUM
        "END";
        EM[0]:=ARREB; EM[2]:= 1.0"-8; EM[4]:= 6*N;
        ALL ZER ORT POL(N, A, B, ZER, EM)
    "END" ALL JAC ZER;
        "EOP"


"CODE" 31371;
    "PROCEDURE"  ALL LAG ZER(N, ALFA, ZER);
    "VALUE" N, ALFA      ; "INTEGER"N; "REAL" ALFA      ; "ARRAY" ZER;
    "BEGIN" "INTEGER" I; "ARRAY" A, B[0:N], EM[0:5];
        "REAL""PROCEDURE" ARREB;
        "CODE" 30002;
        "PROCEDURE" ALL ZER ORT POL(N, B, C, ZER, EM);
        "VALUE" N; "INTEGER" N; "ARRAY" B,C,ZER,EM;
        "CODE" 31362;
        B[0]:= 0; A[N - 1]:= N + N + ALFA - 1;
        "FOR" I:= 1 "STEP" 1 "UNTIL" N - 1 "DO"
        "BEGIN" A[I - 1]:= I + I + ALFA - 1;
            B[I]:= I*(I + ALFA)
        "END";
       EM[0]:=ARREB; EM[2]:= "-10;EM[4]:= 6*N;
        ALL ZER ORT POL(N, A, B, ZER, EM)
    "END" ALL LAG ZER;
        "EOP"
```

AUTHOR: C.G. VAN DER LAAN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730815.

BRIEF DESCRIPTION:

COMKWD CALCULATES THE ROOTS OF A QUADRATIC EQUATION WITH COMPLEX COEFFICIENTS.

KEYWORDS:

ZEROS,QUADRATIC EQUATION,POLYNOMIAL EQUATION,COMPLEX COEFFICIENTS.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE"COMKWD(PR,PI,QR,QI,GR,GI,KR,KI);
"VALUE"PR,PI,QR,QI;"REAL"PR,PI,QR,QI,GR,GI,KR,KI;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
PR,PI,QR,QI:<ARITHMETIC EXPRESSION>;
            ENTRY:PR,QR ARE THE REAL PARTS AND PI,QI ARE THE
                  IMAGINARY PARTS OF THE COEFFICIENTS OF THE
                  QUADRATIC EQUATION:
                  X**2-2*(PR+I*PI)*X-(QR+I*QI)=0;
GR,GI,KR,KI:<VARIABLE>;
            EXIT:THE REAL PARTS AND THE IMAGINARY PARTS OF THE
                 DINOMIAL ARE DELIVERED IN GR,KR AND GI,KI,
                 RESPECTIVELY;MOREOVER,THE MODULUS OF GR+I*GI IS
                 GREATER OR EQUAL THE MODULUS OF KR+I*KI.
```

PROCEDURES USED:

```
COMMUL=CP34341;
COMDIV=CP34342;
COMSQRT=CP34343.
```

LANGUAGE: ALGOL 60.

EXAMPLE OF USE:

```
"BEGIN""REAL"GR,GI,KR,KI;
"PROCEDURE" COMKWD(PR,PI,QR,QI,GR,GI,KR,KI);
"CODE"34345;
COMKWD(-.1,.3,.11,.02,GR,GI,KR,KI);
OUTPUT(61,"("X**2-2(-.1+.3*I)*X-( .11+.02*I) HAS ROOTS")",/,
    -D.DD,+D.DD,"("*I")",/,
    -D.DD,+D.DD,"("*I")"")",GR,GI,KR,KI)
"END"
```

```
X**2-2(-.1+.3*I)*X-( .11+.02*I) HAS ROOTS
-0.30+0.40*I
 0.10+0.20*I
```

SOURCE TEXT(S):

```
"CODE"34345;
"PROCEDURE" COMKWD(PR,PI,QR,QI,GR,GI,KR,KI);
"VALUE" PR,PI,QR,QI;"REAL" PR,PI,QR,QI,GR,GI,KR,KI;
"BEGIN"
"PROCEDURE"COMMUL (AR,AI,BR,BI,RR,RI);
"CODE"34341;
"PROCEDURE"COMDIV(XR,XI,YR,YI,ZR,ZI);
"CODE"34342;
"PROCEDURE"COMSQRT(AR,AI,PR,PI);
"CODE"34343;
"IF" QR=0 & QI = 0 "THEN"
"BEGIN" KR:=KI:=0 ;GR:=PR*2;GI:=PI*2 "END" "ELSE"
"IF" PR=0 & PI= 0 "THEN"
"BEGIN" COMSQRT(QR,QI,GR,GI);KR:=-GR;KI:=-GI "END" "ELSE"
"BEGIN" "REAL" HR,HI;
"IF" ABS(PR) > 1 "OR" ABS(PI) >1 "THEN" "BEGIN"
COMDIV(QR,QI,PR,PI,HR,HI);
COMDIV(HR,HI,PR,PI,HR,HI);
COMSQRT(1+HR,HI,HR,HI);
COMMUL(PR,PI,HR+1,HI,GR,GI);
"END" "ELSE" "BEGIN" COMSQRT(QR+(PR+PI)*(PR-PI),QI+ PR*PI*2,HR,HI);
       "IF" PR * HR + PI * HI > 0 "THEN"
       "BEGIN" GR := PR + HR;GI := PI + HI "END" "ELSE"
       "BEGIN" GR := PR - HR;GI:= PI - HI "END";
"END";
COMDIV(-QR,-QI,GR,GI,KR,KI);
"END"
"END" COMKWD;
       "EOP"
```

UITGAVEN IN DE SERIE MC SYLLABUS

Onderstaande uitgaven zijn verkrijgbaar bij het Mathematisch Centrum, 2e Boerhaavestraat 49 te Amsterdam-1005, tel. 020-947272.

MCS 1.1 F. GÖBEL & J. VAN DE LUNE, *Leergang Besliskunde, deel 1: Wiskundige basiskennis*, 1965. ISBN 90 6196 014 2.

MCS 1.2 J. HEMELRIJK & J. KRIENS, *Leergang Besliskunde, deel 2: Kansberekening*, 1965. ISBN 90 6196 015 0.

MCS 1.3 J. HEMELRIJK & J. KRIENS, *Leergang Besliskunde, deel 3: Statistiek*, 1966. ISBN 90 6196 016 9.

MCS 1.4 G. DE LEVE & W. MOLENAAR, *Leergang Besliskunde, deel 4: Markovketens en wachttijden*, 1966. ISBN 90 6196 017 7.

MCS 1.5 J. KRIENS & G. DE LEVE, *Leergang Besliskunde, deel 5: Inleiding tot de mathematische besliskunde*, 1966. ISBN 90 6196 018 5.

MCS 1.6a B. DORHOUT & J. KRIENS, *Leergang Besliskunde, deel 6a: Wiskundige programmering 1*, 1968. ISBN 90 6196 032 0.

MCS 1.6b B. DORHOUT, J. KRIENS & J.TH. VAN LIESHOUT, *Leergang Besliskunde, deel 6b: Wiskundige programmering 2*, 1977. ISBN 90 6196 150 5.

MCS 1.7a G. DE LEVE, *Leergang Besliskunde, deel 7a: Dynamische programmering 1*, 1968. ISBN 90 6196 033 9.

MCS 1.7b G. DE LEVE & H.C. TIJMS, *Leergang Besliskunde, deel 7b: Dynamische programmering 2*, 1970. ISBN 90 6196 055 X.

MCS 1.7c G. DE LEVE & H.C. TIJMS, *Leergang Besliskunde, deel 7c: Dynamische programmering 3*, 1971. ISBN 90 6196 066 5.

MCS 1.8 J. KRIENS, F. GÖBEL & W. MOLENAAR, *Leergang Besliskunde, deel 8: Minimaxmethode, netwerkplanning, simulatie*, 1968. ISBN 90 6196 034 7.

MCS 2.1 G.J.R. FÖRCH, P.J. VAN DER HOUWEN & R.P. VAN DE RIET, *Colloquium Stabiliteit van differentieschema's, deel 1*, 1967. ISBN 90 6196 023 1.

MCS 2.2 L. DEKKER, T.J. DEKKER, P.J. VAN DER HOUWEN & M.N. SPIJKER, *Colloquium Stabiliteit van differentieschema's, deel 2*, 1968. ISBN 90 6196 035 5.

MCS 3.1 H.A. LAUWERIER, *Randwaardeproblemen, deel 1*, 1967. ISBN 90 6196 024 X.

MCS 3.2 H.A. LAUWERIER, *Randwaardeproblemen, deel 2*, 1968. ISBN 90 6196 036 3.

MCS 3.3 H.A. LAUWERIER, *Randwaardeproblemen, deel 3*, 1968. ISBN 90 6196 043 6.

MCS 4 H.A. LAUWERIER, *Representaties van groepen*, 1968. ISBN 90 6196 037 1.

MCS 5 J.H. VAN LINT, J.J. SEIDEL & P.C. BAAYEN, *Colloquium Discrete wiskunde*, 1968.
ISBN 90 6196 044 4.

MCS 6 K.K. KOKSMA, *Cursus ALGOL 60*, 1969. ISBN 90 6196 045 2.

MCS 7.1 *Colloquium Moderne rekenmachines, deel 1*, 1969. ISBN 90 6196 046 0.

MCS 7.2 *Colloquium Moderne rekenmachines, deel 2*, 1969. ISBN 90 6196 047 9.

MCS 8 H. BAVINCK & J. GRASMAN, *Relaxatietrillingen*, 1969.
ISBN 90 6196 056 8.

MCS 9.1 T.M.T. COOLEN, G.J.R. FÖRCH, E.M. DE JAGER & H.G.J. PIJLS, *Elliptische differentiaalvergelijkingen, deel 1*, 1970.
ISBN 90 6196 048 7.

MCS 9.2 W.P. VAN DEN BRINK, T.M.T. COOLEN, B. DIJKHUIS, P.P.N. DE GROEN, P.J. VAN DER HOUWEN, E.M. DE JAGER, N.M. TEMME & R.J. DE VOGELAERE, *Colloquium Elliptische differentiaalvergelijkingen, deel 2*, 1970.
ISBN 90 6196 049 5.

MCS 10 J. FABIUS & W.R. VAN ZWET, *Grondbegrippen van de waarschijnlijkheidsrekening*, 1970. ISBN 90 6196 057 6.

MCS 11 H. BART, M.A. KAASHOEK, H.G.J. PIJLS, W.J. DE SCHIPPER & J. DE VRIES, *Colloquium Halfalgebra's en positieve operatoren*, 1971.
ISBN 90 6196 067 3.

MCS 12 T.J. DEKKER, *Numerieke algebra*, 1971. ISBN 90 6196 068 1.

MCS 13 F.E.J. KRUSEMAN ARETZ, *Programmeren voor rekenautomaten; De MC ALGOL 60 vertaler voor de EL X8*, 1971. ISBN 90 6196 069 X.

MCS 14 H. BAVINCK, W. GAUTSCHI & G.M. WILLEMS, *Colloquium Approximatietheorie*, 1971. ISBN 90 6196 070 3.

MCS 15.1 T.J. DEKKER, P.W. HEMKER & P.J. VAN DER HOUWEN, *Colloquium Stijve differentiaalvergelijkingen, deel 1*, 1972. ISBN 90 6196 078 9.

MCS 15.2 P.A. BEENTJES, K. DEKKER, H.C. HEMKER, S.P.N. VAN KAMPEN & G.M. WILLEMS, *Colloquium Stijve differentiaalvergelijkingen, deel 2*, 1973. ISBN 90 6196 079 7.

MCS 15.3 P.A. BEENTJES, K. DEKKER, P.W. HEMKER & M. VAN VELDHUIZEN, *Colloquium Stijve differentiaalvergelijkingen, deel 3*, 1975.
ISBN 90 6196 118 1.

MCS 16.1 L. GEURTS, *Cursus Programmeren, deel 1: De elementen van het programmeren*, 1973. ISBN 90 6196 080 0.

MCS 16.2 L. GEURTS, *Cursus Programmeren, deel 2: De programmeertaal ALGOL 60*, 1973. ISBN 90 6196 087 8.

MCS 17.1 P.S. STOBBE, *Lineaire algebra, deel 1*, 1974. ISBN 90 6196 090 8.

MCS 17.2 P.S. STOBBE, *Lineaire algebra, deel 2*, 1974. ISBN 90 6196 091 6.

MCS 17.3 N.M. TEMME, *Lineaire algebra , deel 3*, 1976. ISBN 90 6196 123 8.

MCS 18 F. VAN DER BLIJ, H. FREUDENTHAL, J.J. DE IONGH, J.J. SEIDEL & A. VAN WIJNGAARDEN, *Een kwart eeuw wiskunde 1946-1971, Syllabus van de Vakantiecursus 1971*, 1974. ISBN 90 6196 092 4.

MCS 19 A. HORDIJK, R. POTHARST & J.Th. RUNNENBURG, *Optimaal stoppen van Markovketens*, 1974. ISBN 90 6196 093 2.

MCS 20 T.M.T. COOLEN, P.W. HEMKER, P.J. VAN DER HOUWEN & E. SLAGT, *ALGOL 60 procedures voor begin- en randwaardeproblemen*, 1976. ISBN 90 6196 094 0.

MCS 21 J.W. DE BAKKER (red.), *Colloquium Programmacorrectheid*, 1975. ISBN 90 6196 103 3.

MCS 22 R. HELMERS, F.H. RUYMGAART, M.C.A. VAN ZUYLEN & J. OOSTERHOFF, *Asymptotische methoden in de toetsingstheorie; Toepassingen van naburigheid*, 1976. ISBN 90 6196 104 1.

MCS 23.1 J.W. DE ROEVER (red.), *Colloquium Onderwerpen uit de biomathematica, deel 1*, 1976. ISBN 90 6196 105 X.

MCS 23.2 J.W. DE ROEVER (red.), *Colloquium Onderwerpen uit de biomathematica, deel 2*, 1976. ISBN 90 6196 115 7.

MCS 24.1 P.J. VAN DER HOUWEN, *Numerieke integratie van differentiaalvergelijkingen, deel 1: Eenstapsmethoden*, 1974. ISBN 90 6196 106 8.

MCS 25 *Colloquium Structuur van programmeertalen*, 1976. ISBN 90 6196 116 5.

MCS 26.1 N.M. TEMME (ed.), *Nonlinear analysis, volume 1*, 1976. ISBN 90 6196 117 3.

MCS 26.2 N.M. TEMME (ed.), *Nonlinear analysis, volume 2*, 1976. ISBN 90 6196 121 1.

MCS 27 M. BAKKER, P.W. HEMKER, P.J. VAN DER HOUWEN, S.J. POLAK & M. VAN VELDHUIZEN, *Colloquium Discretiseringsmethoden*, 1976. ISBN 90 6196 124 6.

MCS 28 O. DIEKMANN, N.M. TEMME (EDS), *Nonlinear Diffusion Problems*, 1976. ISBN 90 6196 126 2.

MCS 29.1 J.C.P. BUS (red.), *Colloquium Numerieke programmatuur, deel 1A, deel 1B*, 1976. ISBN 90 6196 128 9.

MCS 29.2 H.J.J. TE RIELE (red.), *Colloquium Numerieke programmatuur, deel 2*, 1976. ISBN 90 6196 144 0

* MCS 30 P. GROENEBOOM, R. HELMERS, J. OOSTERHOFF & R. POTHARST, *Efficiency begrippen in de statistiek*, . ISBN 90 6196 149 1.

MCS 31 J.H. VAN LINT (red.), *Inleiding in de coderingstheorie*, 1976. ISBN 90 6196 136 X.

MCS 32 L. GEURTS (red.), *Colloquium Bedrijfssystemen*, 1976. ISBN 90 6196 137 8.

MCS 33 P.J. VAN DER HOUWEN, *Differentieschema's voor de berekening van waterstanden in zeeën en rivieren*, 1977. ISBN 90 6196 138 6.

MCS 34 J. HEMELRIJK, *Oriënterende cursus mathematische statistiek*, ISBN 90 6196 139 4.

MCS 35 P.J.W. TEN HAGEN (red.), *Colloquium Computer Graphics*, 1977. ISBN 90 6196 142 4.

MCS 36 J.M. AARTS, J. DE VRIES, *Colloquium Topologische Dynamische Systemen*, 1977. ISBN 90 6196 143 2.

MCS 37 J.C. van Vliet (red.), *Colloquium Capita Datastructuren*, 1978. ISBN 90 6196 159 9.

MCS 38.1 T.H. KOORNWINDER (ED.), *Representations of locally compact groups with applications*, 1979. ISBN 90 6196 161 0.

MCS 38.2 T.H. KOORNWINDER (ED.), *Representations of locally compact groups with applications*, 1979. ISBN 90 6196 181 5.

MCS 39 O.J. VRIEZE & G.L. Wanrooij, *Colloquium Stochastische Spelen*, 1978. ISBN 90 6196 167 X.

MCS 40 J. VAN TIEL, *Convexe Analyse*, 1979. ISBN 90 6196 187 4.

MCS 41 H.J.J. TE RIELE (ED.), *Colloquium Numerical Treatment of Integral Equations*, 1979. ISBN 90 6196 189 0.

MCS 42 J.C. VAN VLIET (RED.), *Colloquium Capita Implementatie van Programmeertalen*, 1980. ISBN 90 6196 191 2.

MCS 43 A.M. COHEN & H.A. WILBRINK, *Eindige groepen (Een inleidende cursus)*, 1980. ISBN 90 6196 203 X

MCS 44 J.G. VERWER (ED.), *Numerical solution of partial differential equations*, 1980. ISBN 90 6196 205 6.

MCS 45 P. KLINT (red.), *Colloquium hogere programmeertalen en computerarchitectuur*, 1980. ISBN 90 6196 206 4.

MCS 46.1 P.M.G. APERS (RED.), *Colloquium Databankorganisatie*, 1981. ISBN 90 6196 212 9.

MCS 47.1 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part I: *General information and indices*, 1981. ISBN 90 6196 217 X.

MCS 47.2 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part II: *Elementary procedures, algebraic evaluation*, 1981, ISBN 90 6196 217 X.

MCS 47.3 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part III: *Linear algebra, part I*, 1981. ISBN 90 6196 217 X.

MCS 47.4 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part IV: *Linear algebra, part II*, 1981. ISBN 90 6196 217 X.

MCS 47.5 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part V: *Analytical evaluations, analytical problems, part I*, 1981. ISBN 90 6196 217 X.

MCS 47.6 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part VI: *Analytical problems, part II*, 1981. ISBN 90 6196 217 X.

MCS 47.7 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part VII: *Special functions and constants, interpolation and approximation*, 1981. ISBN 90 6196 217 X.

De met een * gemerkte uitgaven moeten nog verschijnen.