cs34

EIGENVECTORS OF A REAL MATRIX BY INVERSE ITERATION

BY

J. M. VARAH

TECHNICAL REPORT CS34
FEBRUARY 8, 1966

COMPUTER SCIENCE DEPARTMENT.
School of Humanities and Sciences
STANFORD UNIVERSITY

Eigenvectors of a Real Matrix by Inverse Iteration

J. M. Varah

1. - Theoretical Background

Calculation of the Eigenvectors

We are given a real $n \times n$ matrix A. We first reduce A to upper Hessenberg form, and find approximations to its eigenvalues. Then each eigenvector is calculated by inverse iteration on the upper Hessenberg form using an approximation to the corresponding eigenvalue.

For a real eigenvalue λ, the iteration is defined by

$$(A-\lambda I)y^{(k)} = x^{(k-1)}$$

$$x^{(k)} = \frac{y^{(k)}}{\epsilon_k \cdot \max\limits_{1\leq j\leq n} |y_j^{(k)}|}$$

where $\epsilon_k = \operatorname{sgn}(y_{jmax}^{(k)})$, $y_{jmax}^{(k)} = \max\limits_{1\leq j\leq n} |y_j^{(k)}|$.

Thus the largest component of $x^{(k)}$ in modulus is 1. The iteration is started with a given initial vector $x^{(0)}$.

For a complex eigenvalue $\lambda = \xi + i\eta$, the iteration is defined by

$$(A-\lambda I)(A-\bar{\lambda} I)y^{(k)} = x^{(k-1)}$$

$$\eta z^{(k)} = u^{(k-1)} - (A-\xi I)y^{(k)}$$

$$u^{(k)} + iv^{(k)} = \frac{y^{(k)} + iz^{(k)}}{\epsilon_k \max\limits_{1\leq j\leq n} |y_j^{(k)} + iz_j^{(k)}|}$$

$$x^{(k)} = (A-\xi I)u^{(k)} - \eta v^{(k)} ,$$

Prepared under Contract Nonr-225(37) (NR-044-211) at Stanford University.

where again ϵ_k is ± 1, chosen so that a largest component of $u^{(k)} + iv^{(k)}$ in modulus is $1 + 0i$. The iteration is started with a given initial vector $x^{(0)}$ and with $u^{(0)} = 0$.

The iteration here is such that all computations can be done in real arithmetic, For a complete description of inverse iteration, see Wilkinson [1]. The iteration above for a complex eigenvalue is his method (iv), page 630.

2.- Applicability

The algorithm will accept any real matrix. However, the accuracy of the results will depend on the condition of the eigenvalues and eigenvectors, so that no a priori estimate of the accuracy can be given.

3.- Formal Parameter List

N	order of matrix A
A	given matrix. On output, the upper Hessenberg form is stored in A, with the transformations used in the lower sub-triangle,
RTR,RTI	real and imaginary parts of the eigenvalues, respectively, with complex conjugate pairs consecutive,,
U	matrix of column eigenvectors, stored by columns. If λ_k is complex, with $\lambda_{k+1} = \bar{\lambda}_k$ then columns k and k+1 of U contain the real and imaginary parts, respectively, of the eigenvector corresponding to eigenvalue λ_k. The eigenvector corresponding to λ_{k+1} is then the complex conjugate of this.

MACHEPS machine precision, i.e. the smallest floating point number δ such that $1+\delta>1$.

EXPLIMIT largest floating point number represented in the machine .

4. - Algol Program (see next page)

-

```
PROCEDURE EIGENVALUESANDEIGENVECTORS(N,A,RTR,RTI,U,MACHEPS,EXPLIMIT);
    VALUE N,MACHEPS,EXPLIMIT;
    INTEGER N ; REAL MACHEPS,EXPLIMIT; ARRAY A,RTR,RTI,U;
BEGIN
COMMENT THIS PROCEDURE FINDS ALL EIGENVALUES AND COLUMN EIGENVECTORS
    OF THE N×N MATRIX A, THE EIGENVALUES ARE STORED IN(RTR[K]+RTI[K]×I)
    WITH COMPLEX CONJUGATE PAIRS CONSECUTIVE. THE EIGENVECTORS ARE
    STORED BY COLUMNS IN U, IF THE K-THEIGENVALUE IS COMPLEX, COLUMN
    K IS THE REAL PART AND COLUMN K+1 THE IMAGINARY PART OF THE EIGEN-
    VECTOR CORRESPONDING TO EIGENVALUE K, MACHEPS IS THE MACHINE
    PRECISION, AND EXPLIMIT THE LARGEST FLOATING POINT NUMBER CARRIED
    BY THE MACHINE;

COMMENT FIRST DECLARE OTHER PROCEDURES;

REAL PROCEDURE MAX(A,B);
    VALUE A,B;
    REAL A,B;
    MAX:=IF A>B THEN A ELSE B;

REAL PROCEDURE MIN(A,B);
    VALUE A,B;
    REAL A,B;
    MIN:=IF A<B THEN A ELSE B;

REAL PROCEDURE ABSC(A,B);
    VALUE A,B;
    REAL A,B;
```

```
COMMENT GIVES MODULUS OF COMPLEX NUMBER A+BI;
BEGIN
    A:=ABS(A); B:=ABS(B);
    ABSC:=IF A<B THEN B×SQRT(1+(A/B)↑2) ELSE
          IF A>B THEN A×SQRT(1+(B/A)↑2) ELSE
          A×1.41421356237; COMMENT SQRT(2);
END ABSC;

REAL PROCEDURE INNERPRODUCT(I,M,N,A,B,C);
    VALUE M,N,C;
    INTEGER I,M,N; REAL C;
BEGIN COMMENT BODY OF PROCEDURE SHOULD BE REPLACED BY
    DOUBLE PRECISION MACHINE CODE;
    FOR I:=M STEP 1 UNTIL N DO C:=C+A×B;
    INNERPRODUCT:=C
END INNERPRODUCT;

PROCEDURE HESSENBERG(N,A,INT);
    VALUE N;
    INTEGER N; ARRAY A; INTEGER ARRAY INT;
BEGIN COMMENT HESSENBERG REDUCES A TO UPPER HESSENBERG FORM USING
    ELEMENTARY ROW AND COLUMN OPERATIONS WITH INTERCHANGES. THE
    INTERCHANGES ARE STORED IN THE INTEGER ARRAY INT IN SUCH A WAY
    THAT, ON EXIT FROM HESSENBERG, INT[I] IS THE INDEX OF THE ROW OF
    THE ORIGINAL MATRIX NOW IN THE I-TH ROW. PROCEDURE HESSENBERG
    IS A SLIGHTLY MODIFIED VERSION OF PROCEDURE TRINGLE, GIVEN BY
    B.PARLETT IN [1];
    INTEGER I,J,K,L; REAL S,T,EPS;
```

```
EPS:=SQRT(MACHEPS);
FOR I:=1 STEP 1 UNTIL N DO INT[I]:=I;
FOR J:=N STEP -1 UNTIL 2 DO
BEGIN
    S:=ABS(A[J,J-1]); L:=J-1;
    FOR K:=J-2 STEP -1 UNTIL 1 DO
    BEGIN COMMENT FIND ELEMENT OF LARGEST MAGNITUDE IN J-THROW;
        T:=ABS(A[J,K]);
        IF T>S THEN
        BEGIN
            S:=T; L:=K
        END
    END;
    IF L≠J-1 THEN
    BEGIN COMMENT INTERCHANGE COLUMNS AND ROWS J-1 AND L;
        T:=INT[J-1]; INT[J-1]:=INT[L]; INT[L]:=T;
        FOR K:=1 STEP 1 UNTIL N DO
        BEGIN
            T:=A[K,J-1]; A[K,J-1]:=A[K,L]; A[K,L]:=T
        END;
        FOR K:=1 STEP 1 UNTIL N DO
        BEGIN
            T:=A[J-1,K]; A[J-1,K]:=A[L,K]; A[L,K]:=T
        END
    END OF CONDITIONAL;
    IF S≤EPS×MIN(ABS(A[J,J]),ABS(A[J-1,J-1])) THEN
    BEGIN
        FOR K:=1 STEP 1 UNTIL J-1 DO A[J,K]:=0
```

```
    END ELSE
      FOR K:=1 STEP 1 UNTIL J-2 DO A[J,K]:=A[J,K]/A[J,J-1];
    FOR I:=N STEP -1 UNTIL 1 DO
    BEGIN COMMENT CHANGE ROW J-1;
      T:=IF A[J,J-1]=0 THEN 0 ELSE INNERPRODUCT(K,1,J-2,A[J,K],
         A[K,I],0);
      A[J-1,I]:=-INNERPRODUCT(K,MAX(J,I+2),N,A[J-1,K-1],A[K,I],
         -A[J-1,I]-T)
    END I
  END J
END HESSENBERG;

PROCEDURE TRANSFORM(N,X,A,INT);
  VALUE
  INTEGER ARRAY X,A; INTEGER ARRAY INT;,
BEGIN COMMENT THIS PROCEDURE TRANSFORMS AN EIGENVECTOR X OF THE
  HESSENBERG MATQIX INTO AN EIGENVECTOR OF THE ORIGINAL MATRIX;
  INTEGER I,J; AQRAY Y[1:N];
  FOR I:=1 STEP 1 UNTIL N-1 DO
    Y[I]:=-INNERPRODUCT(J,1,I-1,A[I+1,J],Y[J],-X[I]);
    Y[N]:=X[N];
  FOR I:=1 STEP 1 UNTIL N DO X[INT[I]]:=Y[I];
END TRANSFORM;

PROCEDURE EIGENVALUES(NSTART,NFINISH,A,RTR,RTI);
  VALUE NSTART,NFINISH;
  INTEGER NSTART,NFINISH; ARRAY A,RTR,RTI;
BEGIN COMMENT THE BODY OF PROCEDURE EIGENVALUES IS LEFT UNDEFINED.
```

```
    IT IS ASSUMED EIGENVALUES FINDS ALL THE EIGENVALUES OF THE
    PRINCIPAL SUBMATRIX OF THE UPPER HESSENBERG MATRIX A CONSISTING
    OF ELEMENTS A[NSTART,NSTART],...,A[NFINISH,NFINISH] AND PLACES
    THE EIGENVALUES IN RTR[NSTART]+RTI[NSTART]×I,...,RTR[NFINISH]
    +RTI[NFINISH]×I, WITH COMPLEX CONJUGATE PAIRS CONSECUTIVE),
END EIGENVALUES;

  PROCEDURE NORMREAL(N,V,VNORM);
    VALUE N;
    INTEGER N; ARRAY V; REAL VNORM;
  BEGIN COMMENT THIS PROCEDURE NORMALIZES THE REAL VECTOR V OF DIMENSION
    N SO THAT ITS COMPONENT OF LARGEST MAGNITUDE IS 1.0, AND SETS
    VNORM EQUAL TO THE MAGNITUDE OF THE LARGEST COMPONENT BEFORE
    NORMALIZATION;
   REAL S,S1,V1; INTEGER I,J;
   S:=0;
   FOR I:=1 STEP 1 UNTIL N DO
   BEGIN
      S1:=ABS(V[I]);
      IF S1>S THEN
      BEGIN
         S:=S1; J:=I
      END
   END;
   VNORM:=S;
   IF VNORM ≠ 0 THEN
   BEGIN
      V1:=V[J]; V[J]:=1;
```

```
      FOR I:=1 STEP 1 UNTIL J-1,J+1 STEP 1 UNTIL N DO
         V[I]:=V[I]/V1
   END
END NORMREAL;

PROCEDURE NORMCOMPLEX(N,U,V,VNORM);
   VALUE N;
   INTEGER N ; ARRAY U,V; REAL VNORM;
BEGIN COMMENT THIS PROCEDURE NORMALIZES THE COMPLEX VECTOR U+VI OF
   DIMENSION N SO THAT ITS COMPONENT OF LARGEST MAGNITUDE IS 1+0I,
   AND SETS VNORM EQUAL TO THE MAGNITUDE OF THE LARGEST COMPONENT
   BEFORE NORMALIZATION;
   INTEGER I,J; REAL S,S1,U1,U2,V1,V2,R,DEN;
   S:=0;
   FOR I:=1 STEP 1 UNTIL N DO
   BEGIN
      S1:=ABSC(U[I],V[I]);
      IF S1>S THEN
      BEGIN
         S:=S1; J:=I
      END
   END;
   VNORM:=S;
   IF VNORM≠0 THEN
   BEGIN
      U1:=U[J]; V1:=V[J];
      U[J]:=1; V[J]:=0;
      IF ABS(U1)≥ABS(V1) THEN
```

```
BEGIN
    R:=V1/U1; DEN:=U1+R×V1;
    FOR I:=1 STEP 1 UNTIL J-1,J+1 STEP 1 UNTIL N DO
    BEGIN
        U2:=U[I]; V2:=V[I];
        U[I]:=(U2+R×V2)/DEN; V[I]:=(V2-R×U2)/DEN
    END
END ELSE
BEGIN
    R:=U1/V1; DEN:=V1+R×U1;
    FOR I:=1 STEP 1 UNTIL J-1,J+1 STEP 1 UNTIL N DO
    BEGIN
        U2:=U[I]; V2:=V[I];
        U[I]:=(V2+R×U2)/DEN; V[I]:=(R×V2-U2)/DEN
    END
END
END OF CONDITIONAL
END NORMCOMPLEX;

PROCEDURE GAUSSM(N,M,A,EPS,PIVOT);
    VALUE N,M,EPS;
    INTEGER M,N; ARRAY A; REAL EPS; INTEGER ARRAY PIVOT;
BEGIN COMMENT THIS PROCEDURE REDUCES THE N×N MATRIX A, HAVING M
    SUB-DIAGONALS, TO LU FORM BY GAUSSIAN ELIMINATION WITH INTERCHANGES.
    ANY ZERO PIVOTAL ELEMENTS ARE REPLACED BY EPS;
    INTEGER I,J,K,IMAX; REAL T,SUM,QUOT;
    FOR K:=1 STEP 1 UNTIL N DO
    BEGIN
```

```
      SUM:=0;
      FOR I:=K STEP 1 UNTIL MIN(K+M,N) DO
      BEGIN
         T:=ABS(A[I,K]);
         IF T>SUM THEN
         BEGIN
            SUM:=T; IMAX:=I
         END
      END;
   IF SUM=0 THEN
      BEGIN COMMENT K-TH COLUMN IS ZERO - REPLACE DIAGONAL ELEMENT
         BY EPS;
         A[K,K]:=EPS; IMAX:=K
      END;
      PIVOT[K]:=IMAX;
      IF IMAX≠K THEN
      FOR J:=1 STEP 1 UNTIL N DO
      BEGIN COMMENT INTERCHANGE ROWS IMAX AND K;
         T:=A[K,J]; A[K,J]:=A[IMAX,J]; A[IMAX,J]:=T
      END;
      FOR I:=K+1 STEP 1 UNTIL MIN(K+M,N) DO
      BEGIN
         A[I,K]:=QUOT:=A[I,K]/A[K,K];
         FOR J:=K+1 STEP 1 UNTIL N DO
            A[I,J]:=A[I,J]-QUOT×A[K,J]
      END
   END K
END GAUSSM;
```

```
PROCEDURE SOLVE(N,A,X,Y,PIVOT,FIRST);
    VALUE N,FIRST;
    INTEGER ARRAY A,X,Y; INTEGER ARRAY PIVOT; BOOLEAN FIRST;
BEGIN COMMENT THIS PROCEDURE SOLVES A×Y=X GIVEN A IN L U FORM.
    IT DOES NOT MAKE USE OF THE FACT THAT THE ORIGINAL A HAS ONLY
    M SUB-DIAGONALS, SO THAT THE L OF LU IS SOMEWHAT SPARSE.
    THE SOLUTION Y IS TESTED FOR OVERFLOW BEFORE BEING COMPUTED,
    AND SCALED DOWN IF OVERFLOW WOULD HAVE OCCURRED;
    INTEGER I,K; REAL T,SUM;
    COMMENT FORM (L-INVERSE)×X UNLESS THIS IS FIRST INVERSE ITERATION;
    IF ¬ FIRST THEN
    FOR K:=1 STEP 1 UNTIL N DO
    BEGIY
        T:=X[PIVOT[K]]; X[PIVOT[K]]:=X[K];
        X[K]:=-INNERPRODUCT(I,1,K-1,A[K,I],X[I],-T)
    END;
    COMMENT NOW SOLVE U×Y=X;
    FOR K:=N STEP UNTIL 1 DO
    BEGIN
        COMMENT AVOID OVERFLOW IN INVERSE ITERATE BY FIRST CALCULATING
        LN(ABS(Y[K])) AND, IF Y[K] WOULD BE TOO LARGE FOR MACHINE,
        SCALING DOWN THE COMPONENTS OF Y BY MACHEPS UNTIL SMALL ENOUGH.
        THIS SHOULD BE DONE BY EXAMINING EXPONENTS IN MACHINE CODE;
    SCALE:
        SUM:=0;
        FOR I:=K+1 STEP 1 UNTIL N DO
            SUM:=MAX(SUM,LN(MAX(1,ABS(A[K,I]×Y[I]))));
```

```
        IF LN(N-K+1)+SUM-LN(ABS(A[K,K]))>LN(EXPLIMIT) THEN
        BEGIN
            FOR I:=K+1 STEP 1 UNTIL N DO Y[I]:=Y[I]×MACHEPS;
            GO TO SCALE
        END;
        Y[K]:=-INNERPRODUCT(I,K+1,N,A[K,I],Y[I],-X[K])/A[K,K]
    END K
END SOLVE;

PROCEDURE EIGENVECTORS(NSTART,NFINISH,A,U);
    VALUE NSTART,NFINISH;
    INTEGER NSTART,NFINISH; ARRAY A,U;
BEGIN COMMENT THIS PROCEDURE FINDS THE EIGENVECTORS OF THE PRINCIPAL
    SUBMATRIX Of THE UPPER HESSENBERG MATRIX A CONSISTIYG Of ELEMENTS
    A[1,1],...,A[NFINISH,NFINISH] CORRESPONDING TO THE EIGENVALUES
    OF THE PRINCIPAL SUBMATRIX CONSISTING Of ELEMENTS A[NSTART,NSTART],
    ...,A[NFINISH,NFINISH]. IT ASSUMES COMPLEX CONJUGATE EIGENVALUES
    ARE CONSECUTIVE. THE EIGENVECTORS ARE STORED IN THE ARRAY U,BY
    COLUMNS. NON-LOCAL ARRAYS RTR AND RTI ARE USED;
    INTEGER I,J,K,L,M,ITNS;
    REAL ANORM,RABSQ,NORM,VNORM,RR,RI,EPS,NORMTOLERANCE,T;
    ARRAY B[1:NFINISH,1:NFINISH],X,US,UT,VT[1:NFINISH];
    INTEGER ARRAY PIVOT[1:NFINISH]; BOOLEAN FIRST;
    OWN BOOLEAN ASQCALC;
    IF NFINISH=N THEN ASQCALC:=FALSE; COMMENT THIS GIVES ASQCALC A VALUE
        ON FIRST ENTRY;
    COMMENT NOW FIND MAXIMUM ROW SUM OF SUBMATRIX OF A USED;
    ANORM:=0;
```

```
FOR I:=1 STEP 1 UNTIL NFINISH DO
BEGIN
    T:=0;
    FOR J:=MAX(1,I-1) STEP 1 UNTIL NFINISH DO T:=T+ABS(A[I,J]);
    ANORM:=MAX(ANORM,T)
END;
NORMTOLERANCE:=1000×ANORM/MACHEPS; COMMENT SOMEWHAT ARBITRARY;
EPS:=MAX(ANORM,1)×MACHEPS; COMMENT USED IN GAUSSM IN PLACE OF
    ZERO PIVOT;
COMMENT NOW FIND EIGENVECTORS;
M:=1; COMMENT SO THE INCREMENT VARIABLE IN THE FOR LOOP HAS A
    VALUE ON ENTRY;
FOR K:=NSTART STEP M UNTIL NFINISH DO
BEGIN
    COMMENT FIRST PERTURB EIGENVALUE I'F IDENTICAL TO SOME PREVIOUS;
    J:=0;
    FOR I:=NSTART STEP 1 UNTIL K-1 DO
        IF RTR[I]=RTR[K] ∧ RTI[I]=RTI[K] THEN J:=J+1;
    RR:=RTR[K]+3×J×MACHEPS×MAX(1,ABS(RTR[K]));
    RI:=RTI[K];
    IF RI≠0 ∧ (¬ ASQCALC) THEN
    BEGIN COMMENT COMPUTE SUBMATRIX OF A↑2 REQUIRED WHEN FIRST'
        COMPLEX EIGENVALUE ENCOUNTERED;
        FOR I:=1 STEP 1 UNTIL NFINISH DO
            FOR J:=1 STEP 1 UNTIL NFINISH DO
                ASQ[I,J]:=INNERPRODUCT(L,MAX(I-1,1),MIN(J+1,NFINISH),
                A[I,L],A[L,J],0);
        ASQCALC:=TRUE
```

```
END;
COMMENT NOW GENERATE B,THE MATRIX TO S     E REDUCED;
IF RI=0 THEN
BEGIN COMMENT REAL EIGENVALUE-B=A-RR×IDENTITY;
    M:=1;
    FOR I:=1 STEP 1 UNTIL NFINISH DO
    BEGIN
        FOR J:=1 STEP 1 UNTIL I-2 DO B[I,J]:=0;
        FOR J:=MAX(I-1,1) STEP 1 UNTIL NFINISH DO
            B[I,J]:=A[I,J]-(IF J=I THEN RR ELSE 0)
    END
END ELSE
BEGIN COMMENT COMPLEX EIGENVALUE-
    B=(A-(RR+RI×I)×IDENTITY)×(A-(RR-RI×I)×IDENTITY);
    M:=2; RABSQ:=RR↑2+RI↑2;
    FOR I:=1 STEP 1 UNTIL NFINISH DO
    BEGIN
        FOR J:=1 STEP 1 UNTIL I-3 DO B[I,J]:=0;
        IF 123 THEN B[I,I-2]:=ASQ[I,I-2];
        FOR J:=MAX(I-1,1) STEP 1 UNTIL NFINISH DO
            B[I,J]:=ASQ[I,J]-2×RR×A[I,J]+(IF J=I THEN RABSQ ELSE 0)
    END
END GENERATIDN Of B;
COMMENT NOW REDUCE B TO LU FORM -M GIVES THE' NUMBER Of
    SUBDIAGONALS O f B;
GAUSSM(NFINISH,M,B,EPS,PIVOT);
ITNS:=0; FIRST:=TRUE; NORM:=1;
FOR I:=1 STEP 1 UNTIL NFINISH DO
```

```
    BEGIN
        X[I]:=1; US[I]:=0
    END;
    COMMENT INITIAL VECTOR X IS SUCH THAT L×X=E,THE VECTOR WITH
        EACH COMPONENT =1.0. NOW SOLVE B×UT=X;
RHS:
    SOLVE(NFINISH,B,X,UT,PIVOT,FIRST);
    FIRST:=FALSE; ITNS:=ITNS+1;
    IF RI=0 THEN NORMREAL(NFINISH,UT,VNORM) ELSE
    BEGIN
        COMMENT CALCULATE VT FROM (A-RR×IDENTITY)×UT+RI×VT=US;
        FOR I:=1 STEP 1 UNTIL NFINISH DO
            VT[I]:=-INNERPRODUCT(J,MAX(I-1,1),NFINISH,A[I,J],UT[J],
                -RR×UT[I]-US[I])/RI;
        NORMCOMPLEX(NFINISH,UT,VT,VNORM)
    END;
    COMMENT HERE ONE COULD PRINT OUT THE HESSENBERG ITERATES
        UT OR UT+VT×I;
    COMMENT NOW TEST NORM OF INVERSE ITERATE-FIRST TEST FOR OVERFLOW;
    IF LN(NORM)+LN(VNORM)>LN(EXPLIMIT) THEN NORM:=NORMTOLERANCE
        ELSE NORM:=NORM×VNORM; COMMENT TEST SHOULD BE MADE ON
        EXPONENTS IN MACHINE CODE;
    IF NORM < NORMTOLERANCE ∧ ITNS<10 THEN
    BEGIN COMMENT CALCULATE NEXT RIGHT-HAND SIDE;
        IF RI=0 THEN
        BEGIN
            FOR I:=1 STEP 1 UNTIL NFINISH DO X[I]:=UT[I]
        END ELSE
```

```
        FOR I:=1 STEP 1 UNTIL NFINISH DO
        BEGIN
           US[I]:=UT[I]; COMMENT SAVE OLD REAL PART Of ITERATE1
           COMMENT NOW CALCULATE NEW X=(A-RR×IDENTITY)×UT-RI×VT;
           X[I]:=INNERPRODUCT(J,MAX(I-1,1),NFINISH,A[I,J],UT[J],
              -RR×UT[I]-RI×VT[I]);
        END;
        GO TO RHS
     END;
     IF NORM≥NORMTOLERANCE THEN
     BEGIN COMMENT ITERATE HAS CONVERGED-STORE IN U;
        FOR I:=1 STEP 1 UNTIL NFINISH DO U[I,K]:=UT[I];
        IF RI≠0 THEN
        FOR I:=1 STEP 1 UNTIL NFINISH DO U[I,K+1]:=VT[I]
     END ELSE
     BEGIN COMMENT ITERATE DID NOT CONVERGE IN 10ITERATIONS- SET
        VECTOR =0;
        FOR I:=1 STEP 1 UNTIL NFINISH DO U[I,K]:=0;
        IF RI≠0 THEN
        FOR I:=1 STEP 1 UNTIL NFINISH DO U[I,K+1]:=0
     END
  END K
END EIGENVECTORS;

COMMENT NOW BEGIN MAIN PROCEDURE EIGENVALUESANDEIGENVECTORS;
INTEGER NSTART,NFINISH;
REAL VNORM; ARRAY UT,VT[1:N],ASQ[1:N,1:N]; INTEGER ARRAY INT[1:N];
COMMENT FIRST REDUCE A TO UPPER HESSENBERG FORM,USINGPROCEDURE
```

```
    HESSENBERG, WHICH KEEPS THE TRANSFORMATIONS USED IN THE
    LOWER SUB-TRIANGLE OF A SO THAT PROCEDURE TRANSFORM, USING MATRIX A,
    WILL TRANSFORM THE EIGENVECTORS FROM HESSENBERG BASIS TO ORIGINAL
    BASIS - I.E. IF HESS(A)=(S-INVERSE)×A×S, THEN TRANSFORM(N,A,UT)
    CHANGES UT INTO S×UT;
HESSENBERG(N,A,INT);
COMMENT NOW SEARCH SUB-DIAGONALS OF HESSENBERG MATRIX FOR ZERO ELEMENTS
    THUS FINDING A IN SPLIT FORM;
NSTART:=N+1;
NEWBLOCK:
NFINISH:=NSTART-1;
FOR K:=NFINISH+1,K-1 WHILE(IF K≤1 THEN FALSE ELSE A[K,K-1]≠0) DO L:=K;
NSTART:=L-1;
COMMENT NOW FIND EIGENVALUES OF PRINCIPAL SUBMATRIX OF A CONSISTING
   OF ELEMENTS A[NSTART,NSTART],...,A[NFINISH,NFINISH] USING
    PROCEDURE EISENVALUES;
EIGENVALUES(NSTART,NFINISH,A,RTR,RTI);
COMMENT NOW FIND EIGENVECTORS OF PRINCIPAL SUBMATRIX OF A CONSISTING
    OF ELEMENTS A[1,1],...,A[NFINISH,NFINISH] CORRESPONDING TO THESE
    EIGENVALUES;
EIGENVECTORS(NSTART,NFINISH,A,U);
COMMENT NOW AUGMENT HESSENBERG VECTORS BY ZEROS IN POSITIONS
    NFINISH+1,...,N AND TRANSFORM TO ORIGINAL BASIS;
L:=1;
FOR K:=NSTART STEP L UNTIL NFINISH DO
IF RTI[K]=0 THEN
BEGIN
    L:=1;
```

```
    FOR I:=1 STEP 1 UNTIL NFINISH DO UT[I]:=U[I,K];
    FOR I:=NFINISH+1 STEP 1 UNTIL N DO UT[I]:=0;
    TRANSFORM(N, JT,A,INT);
    NORMREAL(N,UT,VNORM);
    FOR I:=1 STEP 1 UNTIL N DO U[I,K]:=UT[I];
END ELSE
BEGIN
    L:=2;
    FOR I:=1 STEP 1 UNTIL NFINISH DO
    BEGIN
        UT[I]:=U[I,K]; VT[I]:=U[I,K+1]
    END;
    FOR I:=NFINISH+1 STEP 1 UNTIL N DO UT[I]:=VT[I]:=0;
    TRANSFORM(N,JT,A,INT); TRANSFORM(N,VT,A,INT);
    NORMCOMPLEX(N,UT,VT,VNORM);
    FOR I:=1 STEP 1 UNTIL N DO
    BEGIN
        U[I,K]:=UT[I]; U[I,K+1]:=VT[I]
    END;
END TRANSFORMATION;
IF NSTART>1 THEN GO TO NEWBLOCK
END EIGENVALUESANDEIGENVECTORS;
```

5. - Organizational and Notational Details

The input matrix A is first reduced to a similar, upper Hessenberg form by elementary row and column operations with interchanges using procedure HESSENBERG, a modified version of procedure TRINGLE, given by B. Parlett in [2]. Then the sub-diagonals of this Hessenberg matrix A is examined for zeros, thus finding A split into smaller Hessenberg submatrices.

For each Hessenberg submatrix, first the eigenvalues are found using procedure EIGENVALUES. This is left undefined here, but the user must insert some applicable eigenvalue procedure. See section 7 for further information, Then the eigenvectors of the Hessenberg matrix corresponding to these eigenvalues are found using procedure EIGENVECTORS. Finally, the eigenvectors are transformed from Hessenberg basis back to the basis of the original matrix, using procedure TRANSFORM, and stored in the columns of U.

6. - Discussion of Numerical Properties

Calculation of the Eigenvectors

On the iteration for the eigenvectors, the equations $(A-\lambda I)y^{(k)} = x^{(k-1)}$ and $(A-\lambda I)(A-\bar{\lambda} I)y^{(k)} = x^{(k-1)}$ are solved by Gaussian elimination, decomposing the matrix into LU form, In both real and complex cases, the initial vector $x^{(0)}$ used is Le, where e is the vector of all ones, Thus on the first iteration, we solve $Uy^{(1)} = e$. To ensure that we can always solve these matrix equations, any zero pivots in U are replaced by small numbers.

Eigenvectors of a Real Matrix by Inverse Iteration

The convergence criterion for the iterations is that the "norm" of the inverse iterate be larger than some given tolerance, where

"norm" $(x^{(k)}) = \prod_{m=1}^{k} \max_{1\le j\le n} |y_j^{(m)}|$ in the real case, and

"norm" $(u^{(k)} + iv^{(k)}) = \prod_{m=1}^{k} \max_{1\le j\le n} |y_j^{(m)} + iz_j^{(m)}|$ in the complex case.

As Wilkinson shows in [1], this ensures that the residual $\|Ax^{(k)} - \lambda x^{(k)}\|$ is small, so that the eigenvector is accurate unless it is ill-conditioned. Unless the eigenvalue approximation is very inaccurate, this criterion is almost always met after two inverse iterations.

Within each Hessenberg submatrix, if any eigenvalues are calculated as identical, they are perturbed slightly so as to give convergence to independent eigenvectors, if they exist. If the matrix has a double eigenvalue with a quadratic elementary divisor, so that it has only one eigenvector, the iterations for both eigenvalue approximations are found by experience to converge to the eigenvector. Similarly, for eigenvalues of higher multiplicities - i.e. the iterations will converge to a set of vectors generating the eigenspace of that eigenvalue. Thus, no principal vectors can be determined by this method so that the whole invariant subspace of the eigenvalue will not be found in the case of non-linear elementary divisors. In this case, because of the ill-conditioning, the eigenvalue approximations are likely to be inaccurate, so that three or four inverse iterations may be needed to give convergence to the eigenvectors, and the eigenvectors obtained will probably also be less accurate than in the case of well-conditioned eigenvectors.

7. - Use of the Program

As we have said, some eigenvalue procedure must be inserted, set up so that it finds all eigenvalues of a given principal submatrix of an upper Hessenberg matrix. The procedure used by the author was that given by Parlett in [2] (with corrections in [3]), using Laguerre's method, although it had to be modified slightly because it uses the matrix reduced to lower, rather than upper, Hessenberg form. However, the user may prefer to use the QR method to find the eigenvalues if such a procedure is available to him, since it seems to be faster, in general, than that of Laguerre.

Also, other methods could be used for reducing the matrix to upper Hessenberg form, such as Householder transformations. If this is done, of course, procedure transform must be changed accordingly, to give the correct change of basis for the eigenvectors.

8. - Test Results

The program has been tested on dozens of matrices on the Burroughs B5500 at Stanford University. One example is given here, a 6x6 matrix with a double eigenvalue at 1 with a quadratic elementary divisor, a double eigenvalue at 3 with linear divisors, and a complex eigenvalue $2 \pm i$. In the computation, the inner product procedure was changed to double precision code.

$$A = \begin{bmatrix} -9 & 21 & -15 & 4 & 2 & 0 \\ -10 & 21 & -14 & 4 & 2 & 0 \\ -8 & 16 & -11 & 4 & 2 & 0 \\ -6 & 12 & -9 & 3 & 3 & 0 \\ -4 & 8 & -6 & 0 & 5 & 0 \\ -2 & 4 & -3 & 0 & 1 & 3 \end{bmatrix}$$

CORRECT EIGENVALUES

3.0000000000E+00	3.0000000000E+00	2.0000000000E+00 + 1.0000000000E+00 I	1.0000000000E+00	1.0000000000E+00

CORRECT EIGENVECTORS

1.0000000000E+00	0.0000000000E+00	1.0000000000E+00 + 0.0000000000E+00 I	1.0000000000E+00	1.0000000000E+00
1.0000000000E+00	0.0000000000E+00	9.0163934426E-01 + 8.1967213115E-02 I	1.0000000000E+00	1.0000000000E+00
1.0000000000E+00	0.0000000000E+00	7.2131147541E-01 + 6.5573770492E-02 I	1.0000000000E+00	1.0000000000E+00
1.0000000000E+00	0.0000000000E+00	5.4098360656E-01 + 4.9180327869E-02 I	7.5000000000E-01	7.5000000000E-01
1.0000000000E+00	0.0000000000E+00	3.6065573770E-01 + 3.2786885246E-02 I	5.0000000000E-01	5.0000000000E-01
1.0000000000E+00	1.0000000000E+00	1.8032786885E-01 + 1.6393442623E-02 I	2.5000000000E-01	2.5000000000E-01

CALCULATED EIGENVALUES

3.0000000000E+00	3.0000000000E+00	2.0000000000E+00 + 9.9999999999E-01 I	9.9999618530E-01	1.0000016784E+00

CALCULATED EIGENVECTORS

1.0000000000E+00	-2.0736479195E-11	1.0000000000E+00 + 0.0000000000E+00 I	1.0000000000E+00	1.0000000000E+00
1.0000000000E+00	-2.0736479195E-11	9.0163934426E-01 + 8.1967213114E-02 I	1.0000000000E+00	1.0000000000E+00
1.0000000000E+00	-2.0736479196E-11	7.2131147541E-01 + 6.5573770490E-02 I	1.0000000000E+00	1.0000000000E+00
1.0000000000E+00	-2.0736479196E-11	5.4098360656E-01 + 4.9180327866E-02 I	7.4999966472E-01	7.5000013443E-01
1.0000000000E+00	-2.0736479197E-11	3.6065573770E-01 + 3.2786885243E-02 I	4.9999977648E-01	5.0000008962E-01
0.0000000000E+00	1.0000000000E+00	1.8032786885E-01 + 1.6393442624E-02 I	2.4999988824E-01	2.5000004481E-01

Note that for the double eigenvalue 1 with a quadratic elementary divisor (so there is only one eigenvector), both eigenvector iterates converged to the eigenvector, although the approximations obtained are both only accurate to about 6 decimal digits. This accuracy is to be expected, as 6 digits is about half of the machine precision for the B5500. For the double eigenvalue 3 with linear divisors, the eigen-space is a 2-space generated by the orthogonal vectors $(1,1,1,1,1,0)^T$ and. $(0,0,0,0,0,1)^T$. In this example, the iterates did converge to these orthogonal eigenvectors, but in general, we cannot expect the eigenvectors obtained to 'be orthogonal, although they will be linearly independent.

Acknowledgements

This work was supported by the United States Office of Naval Research Contract Nonr-225(37). The author would like to thank Professor G. E. Forsythe for his assistance and supervision,

References

1. - Wilkinson, J. H.: The Algebraic Eigenvalue Problem, Oxford Press 1965.

2. - Parlett, B. N.: Laguerre's Method Applied to the Matrix Eigenvalue Problem, Math. of Comp. 18, 464-485 (1964).

3. - Varah, J. M.: Certification of Parlett's ALGOL Eigenvalue Procedure Eig3. Math. of Comp. (to appear).

Computation Center
Stanford University
Stanford, California