cs 43

TWO WORKING ALGORITHMS FOR THE EIGENVALUES OF A SYMMETRIC TRIDIAGONAL MATRIX

BY

W. KAHAN and J. VARAH

TECHNICAL REPORT NO. CS43

AUGUST 1, 1966

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

TWO WORKING ALGORITHMS FOR THE EIGENVALUES OF A SYMMETRIC TRIDIAGONAL MATRIX

by

W. Kahan and J. Varah*

ABSTRACT

Two tested programs are supplied to find the eigenvalues of a symmetric tridiagonal matrix. One program uses a square-root-free version of the QR algorithm. The other uses a compact kind of Sturm sequence algorithm. These programs are faster and more accurate than the other comparable programs published previously with which they have been compared.

The first author's present address is Mathematics Department, University of Toronto.

*Stanford University, Computer Science Department.

Prepared under Contract Nonr-225(37)(NR-044-211) Office of Naval Research.

I. Description of procedure STDQR

STDQR finds all N eigenvalues E[1],E[2],...,E[N] of the symmetric tridiagonal matrix with A[1],A[2],...,A[N] on the diagonal and B[1],B[2], ● o=rB[N-1] on the superdiagonal. The eigenvalues are not found in any particular order. The input data A and B are not changed.

Accuracy:

In our experience, the absolute error in each value E[i] has not exceeded a few units in the last place of $\max_{1 \le j \le N} |E[j]|$. The larger is N , the larger the error can be. But our best rigorous error bounds are far larger than the error observed in practice. Turning the matrix end-for-end (by exchanging A[i] with A[N+1-i] and B[i] with B[N-i]) can change the errors significantly because the eigenvalues nearest the elements at the bottom of the matrix tend to be found first. For best results when the matrix contains significant elements A[i] and B[i] of very different magnitudes, the smaller elements should appear at the bottom, in which case the errors in their corresponding eigenvalues may be as much as 100 times smaller than if the matrix were reversed.

The program contains provisions for scaling to prevent trouble with premature over/underflow. It assumes that the computer replaces underflowed arithmetic results by zero. Then each computed eigenvalue E[i] will be correct to the accuracy described above unless it overflows or underflows.

Timing:

Roughly proportional to N^2. This is the fastest program known to date for computing all the eigenvalues E[i]. If only a few eigenvalues are wanted, then our RECURSECTION program may be faster. Because the QR iteration used here is cubically convergent, little time can be saved by relaxing the accuracy requirement. In our experience, the time required for the whole program corresponds to roughly N^2 circuits of the inner loop (see label LOOP in ALGOL 60 program).

Method:

The QR iteration used here is based upon a square-root free version of the original Francis algorithm [4], published by Ortega and Kaiser [5]. However, the algorithm described by the latter authors, and published in ALGOL 60 by Businger [2], is numerically unstable. (See example 1 of our test results and Welsch [7].) Revisions proposed by Rutishauser [6] and Wilkinson ([10], pg. 567) do not cure the problem. Our version appears to be stable. We are indebted to J.H. Wilkinson for a 2 X 2 example containing the first intelligible evidence that the Ortega-Kaiser, and also the Wilkinson-Rutishauser version, might be numerically unstable.

The origin-shift strategy (the choice of LAMBDA) is an important contributor to the rapid convergence of the algorithm. We set LAMBDA to . that eigenvalue of the bottom 2 X 2 principal submatrix which is closer to the last diagonal element, except when this choice is not unique, in which case the eigenvalue of smaller magnitude is selected.

The criteria for deciding when an off-diagonal element B[i] is negligible are discussed in reference [13].

```
PROCEDURE STDQR(A,B,E,N);
   VALUE N; INTEGER N; ARRAY A,B,E;
K&GI N COMMENT: STDQR FINDS ALL N EIGENVALUES E[1],E[2],...,E[N] OF THE
   SYMMETRIC TRIDIAGUNAL MATRIX WITH A[1],A[2],...,A[N] ON THE DI AGONAL
   AND B[1],B[2],...,B[N-1] UN THE SUPER-DI AGONAL, THE EIGENVALUES ARE
   FUUND IN NU PARTICULAR URDER.;

COMMENT: NQR IS A GLOBAL INTEGER VARIABLE, USED TO COUNT THE NUMBER OF
   QR STEPS MADE.;

COMMENT: WE ASSUME GIVEN THE FOLLOWING MACHINE QUANTI TI ES:
   BASE      = NUMBER BASE OF THE MACHINE
   MACHI NF  = LARGEST EXACT POWER OF THE BASE LESS THAN 1/4 OF THE
               MACHI NE UVERFLOW LI MI T
   MACHNEGL = SMALLEST NUKMALI ZED POSI TI VE NUMBER HEPRESENTABLE ON
               THE MACHI NE
   MACHPREC = FLOATING-POINT RELATIVE MACHI NE PRECISION.;

INTEGER I,K,M;
REAL R,S,T,C,G,P,W,SCALE,EPS,DELTA,LAMBDA,EK1;
ARRAY BB[0:N];

COMMENT: FIRST SCALE MATRIX SO THAT A[I]↑2 AND B[I]↑2 DO NOT OVERFLOW
   AND A[I]↑(-2) AND B[I]↑(-2) DO NOT UNDERFLOW, FIRST FI ND MAXI MUM
   ELEMENI OF THE MATRIX.;
R:=ABS(A[N]);
FOR I:=N-1 STEP -1 UNTIL 1 DO
BEGIN S:=ABS(A[I]); IF S>R THEN R:=S;
   S:=ABS(B[I]); IF S>R THEN R:=S
END;
IF R=0 THEN
BEGIN COMMENT: MATRIX IS ZERO.;
   FOR I:=1 STEP 1 UNTIL N DO E[I]:=0;
   GO TO RETURN
END;

COMMENT: FOR SCALI NG, WE ASSUME GI VEN THE MACHI NE QUANTI TI ES
   MACHNO1 = MIN(MACHINF,1/MACHNEGL) (EXACT POWER OF THE BASE)
   MACHNO2 = SMALL&ST EXACT POWER OF THE BASE LARGER THAN
             SQRT(MACHNO1)/MACHINF.
   THUS MACHNO2 = BASE↑(ENTIER((0.5×LN(MACHNO1)-LN(MACHINF))/LN(BASE))
                        +1).;

SCALE:=IF R≤MACHNO2 THEN MACHI NF
                    ELSE BASE↑(ENTIER((0.5×LN(MACHNO1)-LN(R))/
                                      LN(BASE)));

COMMENT: SCALE IS THE LARGEST EXACT POWER OF THE BASE REPRESENTABLE
   SUCH THAT (R×SCALE)↑2<MACHINF AND (R×SCALE)↑(-2)>MACHNEGL.
   THIS COMPUTATION SHOULD BE DONE IN MACHI NE CODE,
   IT IS PUSSIBLE THAT SCALE COULDUNDERFLOW IF THE MACHI NE IS SUCH
   THAT MACHINF×MACHNEGL > SQRT(MACHNO1) BUT WE KNOW OF NO MACHI NE
   WHEKE THIS IS TRUE.;

E[N]:=A[N]×SCALE;
```

```
FORH I:=N-1 STEP -1 UNTIL 1 DU
GIN   E[I]:=A[I]×SCALE;
   BB[I]:=(B[I]×SCALE)↑2
END;
BB[0]:=BB[N]:=0;
DELTA:=R×SCALE×MACHPREC/(100×N); COMMENT:N×DELTA Is SMALL
   COMPARED WITH THE EXPECTED ERROR OF A UNIT IN THE LAST PLACE OF
   THE LARGEST EIGENVALUE (SCALED).;
EPS:=DELTA↑2; COMMENT: EPS IS USED TO TEST FUR THE NEGLIGIBILITY Of
   BB[I].;
K:=N;
FOR M:=K WHILE M>0 DU
BEGIN COMMENT: SCAN FUR NEGLIGIBLE BB[K] IN ROWS AND COLUMNS M BACK
      TO 1.;
   FOR K:=K-1 WHILE TRUE DU IF BB[K]<EPS THEN GO TO NEXT;
NEXT:
   IFK=M-1 THEN BB[K]:=0 ELSE
   BEGIN CUMMENT: DEAL WITH BOTTOM 2x2 BLOCK, ;
TWOBY2:
      T:=E[M]-E[M-1];
      R:=BB[M-1];
      IF K<M-2 THEN
      BEGIN COMMENT: WEAKER TEST FUR NEGLIGIBLEBB[M-1].;
         W:=BB[M-2];
         C:=T↑2; S:=R/(C+W);
         IF S×(W+S×C)<EPS THEN BEGIN M:=M-1;
                                   BB[M]:=0;
                                   GO TO TWOBY2
                               END
      END NEGLIGIBLE BB;

      IF ABS(T)<DELTA THEN S:=SQRT(R)
      ELSE BEGIN W:=2/T;
           S:=W×R/(SQRT(W↑2×R+1)+1)
        END;
      IF K=M-2 THEN
      BEGIN COMMENT: A 2×2 BLOCK HAS BEEN SEPARATED, SO WE STORE THE
         EIGENVALUES.;
         E[M]:=E[M]+S;
         E[M-1]:=E[M-1]-S;
         BB[K]:=0
      END
      ELSE
      BEGIN COMMENT: DU A QR STEP UN ROWS ANO COLUMNS K+1 THROUGH M,
            USING K AS THE INCREMENT VARIABLE, IN THE NOTATION OF
            URTEGA AND KAISEN, C = C[K]↑2, S = S[K]↑2, P = P[K],
            G = GAMMA[K], T = P[K]↑2+B[K]↑2, W = WORK SPACE, ;
         NQR:=NQR+M-K;
         COMMENT: FIRST CHOOSE THE SHIFT PARAMETER LAMBDA.;
         LAMBDA:=E[M]+S;
         IF ABS(T)<DELIA THEN
         BEGIN W:=E[M-1]-S;
            IF ABS(W)<ABS(LAMBDA) THEN LAMBDA:=W
         END;
         S:=0; G:=E[K+1]-LAMBDA; C:=1;
```

```
            GO TO ENTRY;
LOOP:       C:=P/I; S:=W/I; W:=G;
            EK1:=E[K+1];
            G:=C×(EK1-LAMBDA) - S×W;
            E[K]:=(W-G)+EK1;
ENTRY:      IF ABS(G)<DELTA THEN
               G:=G+(IF G≥0 THEN C ELSE -C)×DELTA;
            P:=G↑2/C;
            K:=K+1;
            W:=BB[K];
            I:=W+P;
            BB[K-1]:=S×T;
            IF K<M THEN GO TO COUP;
         E[K]:=G+LAMBDA
      END QR STEP
   END OF CONDITIONAL
END.M;
FOR I:=1 STEP 1 UNTIL N DO E[I]:=E[I]/SCALE;
NQR:=NQR/N; COMMENT: NQR GIVES THE NUMBER OF EQUIVALENT FULL QR STEPS.;
RETURN:
END STDQR;
```

II* Description of procedure RECURSECTION

RECURSECTION finds

if K > 0 then the greatest K eigenvalues E[1] ≥ E[2] ≥ .*. ≥ E[K]
else if K < 0 then the least -K eigenvalues E[1] ≤ E[2] ≤.** ≤ E[-K]
of the given N X N symmetric tridiagonal matrix with A[1],A[2],...,A[N] on the diagonal and B[1],B[2],...,B[N-1] on the superdiagonal. The input data A and B are not changed.

Accuracy:

Each computed E[i] differs by a unit or two in its last place from the i-th eigenvalue of some tridiagonal matrix which differs from that given by a few units in the last place of each off-diagonal element. All told, no computed E[i] can be in error by more than a few units in the last place of the largest eigenvalue of the given matrix. The error bound depends upon the details of the machine arithmetic units, but is independent of N and K .

The program contains provisions for scaling to prevent trouble with premature over/underflow. It assumes the computer replaces underflowed arithmetic results by zero. The program is such that any underflows which do occur in intermediate results do not cause serious errors in the final results E[i] . In fact, intermediate over/underflows can contribute an absolute error no larger than

$$(3\times(\text{MACHNEGL}\uparrow 0.25)/(\text{MACHINF}\uparrow 0.5))\times\text{NORM}$$

where MACHNEGL and MACHINF are, respectively, the smallest and largest positive numbers normally representable on the machine, and

$$\text{NORM} = \max_i \max.(|A[i]|\ ,\ |B[i]|)\ .$$

Such an error is smaller than 10^{-10} units in the last double precision digit of the biggest eigenvalue of the matrix on any computer we know. This is in marked contrast with the Wilkinson Sturm sequence - bisection algorithm [9], where premature over/underflow can cause disastrous errors in the results. Then the user may be unaware of those errors if underflows are replaced by zero with no message output from the machine telling him of the underflow. For examples of this, see our test results. But for our program, each computed eigenvalue will be correct to the accuracy described above unless it overflows or underflows. The program also assumes that each arithmetic operation (+, -, x, /) is monotonic in its two operands despite roundoff, which is the case on most machines in single precision arithmetic For a detailed error analysis, see reference [12].

Timing:

Roughly proportional to $|K| \times N$. This program is the fastest known to date when $1 \leq |K| \ll N$. When $|K| \cong N$, our QR program is several times faster in some cases. In particular, RECURSECTION is slowest to find those eigenvalues of the matrix which remain almost unchanged when the last row and column of the matrix are deleted, because a binary chop technique is used to find those eigenvalues. The other (and normally most) eigenvalues are found more quickly by a superlinearly convergent iteration. For this reason, RECURSECTION sometimes works faster after the matrix is turned end-for-end via the replacement of A[i] by A[N+1-i] and B[i] by B[N-i] . Also, if any B[i] = 0 , time can be saved by feeding the matrix to RECURSECTION in two or more bites, although one

must subsequently sort the eigenvalues of each bite to obtain the desired ordering of the eigenvalues of the whole matrix.

In any case, RECURSECTION is substantially faster than programs which apply a binary chop technique to a Sturm sequence, and is intended to supersede such programs.

Method:

The basic idea was first put forth at the University. of Toronto by Dr. Boris Davison in 1959, and follow6 from Sylvester's inertia theorem:

If A is a symmetric matrix,, D is diagonal, and L is non-singular, and if

$$A-xI = LDL^T ,$$

then the number of A's eigenvalues less than or equal to x is the same as the number of negative or zero elements of D.

We apply this theorem in procedure SYLVESTER to our symmetric tridiagonal matrix A by performing Gaussian elimination without interchange6 on A-xI , obtaining

$$A-xI = LU = LDL^T .$$

However, since we do not need L explicitly, we only compute the diagonal elements u_i of D and record the number of $u_i \leq 0$. The recurrence relation for these u_i is particularly simple:

$$u_1 = a_1-x$$

$$u_i = (a_i-b_{i-1}^2/u_{i-1})-x , \quad i=2,\ldots,N .$$

Provided the time required for a single precision division is not appreciably longer than that for a single precision multiplication, this takes about 1/3 as long as the usual Sturm sequence recurrence, partly because no serious scaling problems are encountered in SYLVESTER. Also, provided the machine arithmetic is monotone, the recurrence for the $\{u_i\}$ is such that the number $m(x)$ of $u_i \leq 0$ is a monotone non-decreasing function of x despite roundoff. This simplifies the logic of the program: For a similar reason, we compute u_1 as shown rather than from

$$u_1 = (a_i - x) - b_{i\,1}^2 / u_{i\,1} ,$$

to preserve the strict monotonicity of $u_N(x)$ near its zeros.

Procedure SECTION chooses a sequence of values x to feed to SYLVESTER in order to find the eigenvalues of A. This procedure is always entered with two abscissa LO and HI which are known to bracket the eigenvalues we are seeking. We then proceed to find points x between LO and HI , using a method described below, in order to converge to the eigenvalues. Whenever a value x is found which separates (LO, HI] into two subintervals (LO,x] and (x,HI] , each known to contain at least one eigenvalue, SECTION calls itself recursively to deal with each subinterval separately. Mr. Michael D. Green suggested this recursive calling of SECTION, and this seems to be the simplest way of coding the program so that the best bounds are used for each eigenvalue, though stack-overflow may be encountered in some cases if too many recursive calls are made. The depth of recursion cannot exceed |K|.

To form the sequence of values x , a binary chop method would work in principle, but in practice that can be slow. To accelerate convergence of the iterates x to the eigenvalues, we use a modified secant iteration,

patterned after D. J. Wheeler's program F2 (see [8], pg. 84 and 130).

This iteration is applied to the function $u_N(x)$, where u_N is the last element of D defined above. Now,

$$u_N(x) = \frac{\det(A-xI)}{\det(A^{(N-1)}-xI)} \quad , \text{ where } A^{(N-1)} \text{ is the } (N-1)\times(N-1) \text{ matrix}$$

formed from the first (N-1) rows and columns of A. Thus $u_N(x)$ is a rational function with slope < -1 at all points, whose zeros are the zeros of det(A-x1) , except for those zeros which are also zero6 of the denominator to an equal or greater multiplicity. These zero6 are called "hidden eigenvalues".

We use the modified secant iteration on $u_N(x)$ when our current bounds LO and HI are such that $u_N(LO) > 0$ and $u_N(HI) < 0$. Because of the nature of the function $u_N(x)$, this ensures that there is at least one zero of $u_N(x)$ between LO and HI . Otherwise we use binary chop to find the next point x = (LO + HI)/2 . Thus for cluster6 of eigenvalues and the "hidden eigenvalues" mentioned above, the binary chop strategy will be used a large part of the time. But once a zero of u_N is isolated, the secant strategy will be used from then on, giving superlinear convergence to this eigenvalue, with average asymptotic order $3^{1/3} \cong 1.44$.

```
PROCEDURE RECURSECTION(A,B,E,N,K);
   VALUE N,K; INTEGER N,K; ARRAY A,B,E;
BEGIN COMMENT: RECURSECTION FINDS
   IF K>0 THEN THE GREATEST K EIGENVALUES E[1]≥E[2]≥...E[K]
   ELSE IF K<0 THEN THE LEAST (-K) EIGENVALUES E[1]≤E[2]≤...≤E[-K]
   OF THE N×N SYMMETRIC TRIDIAGONAL MATRIX WITH A[1],A[2],...,A[N] ON
   THE DIAGONAL AND B[1],B[2],...,B[N-1] ON THE SUPER-DIAGONAL. THE
   INPUT DATA A AND B ARE NOT CHANGED,;

COMMENT: WE ASSUME GIVEN THE FOLLOWING MACHINE QUANTITIES;
   BASE      = NUMBER BASE OF THE MACHINE
   MACHINF   = LARGEST EXACT POWER OF THE BASE LESS THAN 1/4 OF THE
               MACHINE OVERFLOW LIMIT
   MACHNEGL  = SMALLEST NORMALIZED POSITIVE NUMRER REPRESENTABLE ON
               THE MACHINE.;
INTEGER I;
LO,HI,LU,HU,C,R,R1,S,T,SCALE;
ARRAY AA,BB[1:N];

PROCEDURE SYLVESTER(X,U,M);
   VALUE X; INTEGER M; REAL X,U;
BEGIN COMMENT: SYLVESTER SETS M TO THE NUMBER OF EIGENVALUES OF THE
   N×N SYMMETRIC TRIDIAGONAL MATRIX WITH DIAGONAL AA[1],...,AA[N], AND
   SUPER-DIAGONAL SQRT(BB[2],...,BB[N]), WHICH ARE ≤ X. U IS SET TO THE
   VALUE OF THE LAST PIVOT IN THE GAUSSIAN ELIMINATION OF ((THE MATRIX)
   -X×I), WITH THE CONSTRAINT THAT XL-XH ≤ U(X) ≤ XH-XL, WHERE XL AND
   XH ARE THE BEST BOUNDS WE HAVE FOR THE LEAST AND GREATEST
   EIGENVALUES.;
INTEGER I; OWN REAL XL,XH; DEAL
U:=AA[1]-X; M:=0; I:=1; GO TO L;
LOOP:
   I:=I+1; U:=(AA[I]-BB[I]/U)-X;
L: IF U≤0 THEN BEGIN M:=M+1;
                  IF U=0 THEN U:=-MACHNEGL
               END;
   COMMENT: THIS CODE ASSUMES OVERFLOWS ARE ALLOWED, AND THAT WHEN
      THEY OCCUR, THE ARGUMENT IS REPLACED BY THE LARGEST MAGNITUDE
      WITH THE SAME SIGN. IF THIS IS NOT AVAILABLE TO THE USER, HE CAN
      REPLACE THE CODE AFTER LABEL L: BY THE FOLLOWING, MORE TIME'"
      CONSUMING CODE:
      L: IF U < RTMACHNEGL THEN BEGIN M:=M+1
                                IF U>-RTMACHNEGL THEN U:=-RTMACHNEGL
                             END
      WHERE RTMACHNEGL = SQRT(MACHNEGL),;

   IF I<N THEN GO TO LOOP;
IF M=N THEN XH:=X
ELSE IF M=0 THEN XL:=X
ELSE BEGIN D:=XH-XL;
        IF ABS(U)>D THEN U:=D×SIGN(U)
     END CONSTRAINING U
END SYLVESTER;

REAL PROCEDURE NEXT(X,Y);
   VALUE X,Y; REAL X,Y;
```

```
BEGIN COMMENT: NEXT(X,Y) IS THE NEXT VALUE AFTER X BETWEEN X AND Y
      INCLUSIVE THAT DIFFERS FROM X BY AN AMOUNT WHICH IS AT LEAST
      AS LARGE AS 1 UNIT IN THE LAST PLACE OF Y-X . THIS
      PROCEDURE SHOULD BE WRITTEN IN MACHINE CODE, THE COOING
      GIVEN HERE IS JUST AN EXAMPLE, AND CANNOT BE EXPECTED TO BE
      OPTIMAL FOR ALL MACHINES. THE MACHINE QUANTITY ULP IS ASSUMED
      GIVEN TO BE THE SMALLEST POSITIVE NUMBER SUCH THAT
      1.0+ULP×BASE ≠ 1.0 IN THE MACHINE, ;
   D,E,F;
   E:=ABS(Y-X); F:=ABS(X);
   F:=D:=(IF E>F THEN E ELSE F)×ULP×SIGN(Y-X);
   IF D≠0 THEN FOR E:=X+D WHILE E=X DO D:=D+F
          ELSE E:=X+(Y-X)/2;
   NEXT:=E
END NEXT;

PROCEDURE SECTION(L,H,LU,HU,LO,HI,LM,HM);
   L,H,LU,HU,LO,HI,LM,HM;
   INTEGER L,H; REAL LU,HU,LO,HI,LM,HM;
BEGIN COMMENT: SECTION IS A RECURSIVE PROCEDURE WHICH SEEKS EIGENVALUES
   E[L]≤E[L+1]≤...≤E[H] OF THE N×N SYMMETRIC TRIDIAGONAL MATRIX WITH
   DIAGONAL AA[1],...,AA[N] AND SUPER-DIAGONAL SQRT(BB[2]),...,BB[N]).
   WHEN CALLED, IT IS ASSUMED THAT LO < ALL DESIRED EIGENVALUES ≤HI,
   AND THAT U(LO)/LU > 1 AND U(HI)/HU > 1, WHERE U(X) IS THE OUTPUT OF
   SYLVESTER(X,U,M). LM AND HM ARE ACCELERATION PARAMETERS, ;
REAL X,U; INTEGER M;
START:
IF LU≤0 V HU≥0 THEN
BEGIN COMMENT: DO A BISECTION STEP.;
   X:=LO+(HI-LO)/2; COMMENT: THIS SHOULD BE DONE IN SUCH A WAY THAT
      THE CONSEQUENCES OF UNDERFLOW TO ZERO IN (HI-LO)/2 OR IN X ARE
      CONSONANT WITH THE TREATMENT OF UNDERFLOW IN NEXT(LO,HI).;
END
ELSE X:=LU+(LU/(LU-HU))×(HI-LO); COMMENT: DO A SECANT STEP, ;

COMMENT: THE NEXT SIX LINES GUARANTEE THAT LO < X < HI.;
U:=NEXT(LO,HI); IF X<U THEN X:=U;
U:=NEXT(HI,LO); IF X>U THEN X:=U;
IF X=HI V X=LO THEN
BEGIN COMMENT: THERE ARE (H-L+1) EIGENVALUES AT X.;
   FOR M:=L STEP 1 UNTIL H DO E[M]:=X
END
ELSE
BEGIN SYLVESTER(X,U,M);
   IF M<L THEN
   BEGIN COMMENT: INCREASE LOWER BOUND.;
      LO:=X; LU:=U; LM:=2;
      HM:=0.5×HM; HU:=HU×HM;
      GO TO START
   END;
   IF M≥H THEN
   BEGIN COMMENT: DECREASE UPPER BOUND, ;
      HI:=X; HU:=U; HM:=2;
      LM:=0.5×LM; LU:=LU×LM;
      GO TO START
```

```
    END;
    COMMENT: AT THIS POINT L≤M<H SO WE CAN FIND EIGENVALUES L THROUGH M
       AND M+1 THROUGH H SEPARATELY.;
    SECTION(L,M,LO,U,LO,X,LM,2);
    SECTION(M+1,H,U,HO,X,HI,2,HM);
END
END SECTION;

COMMENT: NOW BEGIN MAIN PROCEDURE RECURSECTION.;
IF K=0 THEN GO TO RETURN;
IF ABS(K)>N THEN K:=N×SIGN(K);

COMMENT: NOW SCALE MATRIX SO THAT EACH SCALED ABS(A[I]) AND ABS(B[I])
    IS LESS THAN MACHNO, A MACHINE QUANTITY DEFINED BELOW. FIRST FIND
    MAXIMUM ELEMENT OF THE MATRIX.;
R:=ABS(A[N]);
FOR I:=N-1 STEP -1 UNTIL 1 DO
BEGIN S:=ABS(A[I]); IF S>R THEN R:=S;
    S:=ABS(B[I]); IF S>R THEN R:=S
END;
IF R=0 THEN
BEGIN COMMENT: MATRIX IS ZERO.;
    FOR I:=1 STEP 1 UNTIL ABS(K) DO E[I]:=0;
    GO TO RETURN
END;

COMMENT: FOR SCALING, WE ASSUME GIVEN THE MACHINE QUANTITY
    MACHNO = LARGEST EXACT POWER OF THE BASE SMALLER THAN
             SQRT(MACHINF×SQRT(MACHNEGL)).
    THUS MACHNO = BASE↑(ENTIER((0.5×LN(MACHINF)+0.25×LN(MACHNEGL))
                                /LN(BASE))).;

SCALE:=SIGN(-K)×(IF R ≤ MACHNO/MACHINF THEN MACHINF
                 ELSE BASE↑(ENTIER((LN(MACHNO)-LN(R))/LN(BASE))));

COMMENT: ABS(SCALE) IS NOW THE LARGEST EXACT POWER OF THE BASE
    REPRESENTABLE SUCH THAT ABS(SCALE)×R<MACHNO. THIS COMPUTATION
    SHOULD BE DONE IN MACHINE CODE.;

COMMENT: NOW SCALE MATRIX AND FIND UPPER AND LOWER GERSCHGORIN BOUNDS
    FOR THE EIGENVALUES.;
C:=A[1]×SCALE;
R1:=ABS(B[1]×SCALE);
LO:=C-R1; HI:=C+R1;
C:=AA[N]:=A[N]×SCALE; R:=0;
FOR I:=N-1 STEP -1 UNTIL 1 DO
0×SCALE)}ABS(B[I
    R1:=R+S;
    T:=C-R1; IF LO>T THEN LO:=T;
    T:=C+R1; IF HI<T THEN HI:=T;
    C:=AA[I]:=A[I]×SCALE;
    R:=S;
    BB[I+1]:=S 2
END;
R:=ABS(LO)+ABS(HI);
```

```
LO:=NEXT(LU,LO-R); HI:=NEXT(HI,HI+R); COMMENT: TO INCLUDE KOUNOOFF
  - ERROR IN GERSCHGORIN BOUNDS.;

COMMENT: NOW MAKE THE INITIAL CALL OF PROCEDURE SECTION. THIS INITIAL
   CALL IS SET UP TO FIND THE GREATEST OR LEAST ABS(K) EIGENVALUES. IF
   SOME OTHER CONFIGURATION OF EIGENVALUES IS DESIRED, THE USER CAN
   CHANGE THIS INITIAL CALL ACCORDINGLY.;
BB[1]:=0;
SYLVESTER(LO,LU,I);
SYLVESTER(HI,HU,I);
SECTION(1,ABS(K),LU,HU,LU,HI,2,2);
COMMENT: NOW UNSCALE THE EIGENVALUES.;
FOR I:=1 STEP 1 UNTIL ABS(K) DO E[I]:=E[I]/SCALE;
RETURN:
END RECURSECTION;
```

III. Test Results

Several tridiagonal matrices were fed to RECURSECTION and STDQR, and the results produced compared with those from some other programs, as shown below. Except in a few cases where the eigenvalues could be computed in closed form or were otherwise known, we were unable to verify our claims to accuracy because RECURSECTION is the most accurate program we have. The differences between RECURSECTION's results and those from the other programs were never in excess of the known error bounds for the other programs.

The other programs compared were:

"WBIS2" – Wilkinson's binary chop Sturm sequence algorithm [9].

"OKBQR" – Ortega and Kaiser's QR method, published by Businger [2]. (The version proposed by Rutishauser and Wilkinson was also tested).

"FJLLT" – Sturm sequence - LL^T algorithm proposed by Fox and Johnson [3].

In the results listed below, we let

T = time in seconds to produce $|K|$ eigenvalues (K given). However the actual time taken depends on the machine used, so we also let

F = (number of full passes (i.e. N times) through the inner loop(s))/$|K|$,

and tabulate "T sec."/"F passes/eigenvalue".

However a direct comparison of the numbers F is still unfair because the inner loop for each program requires a different number of operations. For convenience, we give here a table listing the number of operations in each inner loop.

		divisions	multiplications	additions - subtractions	array references	comparisons
STDQR		3	4	5	4	1
OKBQR		2	3	7	4	1
Wilkinson- Rutishauser version		3	3	6	4	1.
RECURSECTION		1	0	2	2	$1^{1/2}$
WBIS2		0	2	2	2	3
FJLLT	#1	1	0	2	4	1
	#2	0	2	1	4	0

The FJLLT program really has two separate inner loops, each of which is described separately above. In the counting of inner loops executed, each was counted separately and then the results were added. Note that we count the number of "full passes" through the inner loop. In the QR methods, this is not the same as the number of QR steps made, since we do not always work with the full matrix. A similar consideration affects FJLLT. For RECURSECTION, the count is just the number of calls of procedure SYLVESTER per eigenvalue, and for WBIS2 just the number of calls of procedure sturms sequence per eigenvalue.

All results were obtained on a Burroughs' B5500 with 13 octal digits of significance in floating-point (i.e. about 11 decimal digits). Division on this machine takes twice or thrice as long as multiplication, so the procedure RECURSECTION appears in its least favourable light compared with WBIS2. Timing on this machine is unreliable because of multiprocessing,

so the times tabulated below should be regarded merely as rough indications.

To assure as fair a comparison as possible, all programs were set up to yield results of comparable accuracy. The following adjustments were required:

In WBIS2 , the user is expected to state how many binary chops (t) he wants done for each eigenvalue. This means that each eigenvalue will be in absolute error by at most about $2^{-t} \times |\text{largest eigenvalue}|$. If t is chosen just large enough to yield a desired relative accuracy in the larger eigenvalues, the smaller eigenvalues may suffer unacceptable relative errors. Therefore we set t = 50 even though our machine uses only 39 binary digits of significance. To save time, we also modified Wilkinson's program to stop chopping as soon as the computed bounds for an eigenvalue differed by no more than a unit or two in their last place. Thus the actual code changed was the j-loop in the procedure tridibisection 2:

```
for j := 1 step 1 until t do
begin lambda := h + (g-h)/2 ;
    if lambda = h or lambda = g then go to continue;
    sturms sequence ;
    if al > d then h := lambda else g := lambda
end j ;
continue: ml := ml + 1 ;
w[ml]:= h + (g-h)/2 ;
```

This modification can only improve the program.

The Businger version of Ortega and Kaiser's QR method was found to be numerically unstable in certain cases. We modified the loop in the way suggested by Rutishauser and Wilkinson without curing the instability. (See example 1.) Even when the answers were correct, the program usually took somewhat longer than our STDQR despite the fact that our program has an extra multiplication in its inner loop. (See example 2.) We attribute the speed of our program to a better strategy for choosing the acceleration parameter lambda than was used by Businger.

The Fox and Johnson program was amended slightly, mainly to correct a few syntactic errors in the ALGOL listing and to add a scaling block. This program combines a Sturm sequency-binary chop method with a secant iteration applied to the characteristic polynomial of the matrix, and uses the Q-D transformation, organized like Ortega and Kaiser's LL^T algorithm, to deflate successive eigenvalues out of the matrix. In order to guarantee accuracy comparable to that of our STDQR, we found it necessary to set eps2 = 10^{-11} and epsl = 10^{-21} in this program.

TEST NO. 1

Matrix:
$$\begin{pmatrix} x & 1 & & \\ & 1 & 1 & \\ & 1 & -x & 1 \\ & & 1 & -1 \end{pmatrix}$$

This matrix was run with different small values of x to test the QR programs. The results for $x = 10^{-5}$ and $x = 10^{-12}$ were particularly interesting. We believe the true eigenvalues are as follows to 10 figures, since our most accurate programs gave results agreeing to 10 significant figures:

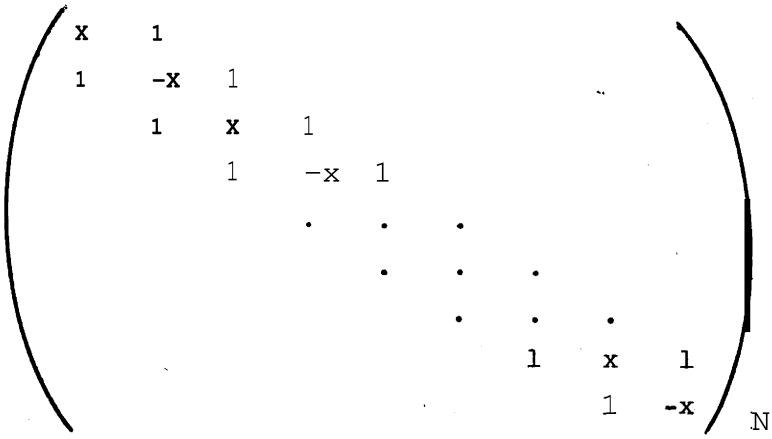
$x = 10^{-5}$:		$x = 10^{-12}$:	
λ_1 =	2.061498246	λ_1 =	2.061498851
λ_2 =	0 [illegible]	λ_2 =	0 .3963385310
λ_3 =	-0 .6938171874	λ_3 =	-0.6938224565
λ_4 =	-1.764018050	λ_4 =	-1.764014925

T sec./F. passes/eigenvalue	
STDQR	0.05/1.5
OKBQR	0.05/1.8
RECURSECTION	0.13/14
WBIS2	0.42/41
FJLLT	0.10/6.0

With $x = 10^{-5}$, the original Ortega-Kaiser QR, as published by Businger, gave results accurate to only 2 decimal places. And with $x = 10^{-12}$, the Rutishauser-Wilkinson amendment to this gave results accurate to at best one decimal place. For both matrices, our STDQR gave results accurate to 10 figures.

TEST NO. 2.

Matrix:



NXN

$$\lambda_k = [x^2 + 4\cos^2(\tfrac{\pi k}{N+1})]^{1/2}$$

$$\lambda_{N+1-k} = -\lambda_k \ , \ k=1,\ldots,[N/2].$$

$$\lambda_{\frac{N+1}{2}} = 0 \text{ (if N is odd).}$$

for x = 1:

	T sec./F passes/eigenvalue		
	K = N = 30	K = +5	K = -5
STDQR	1.2/1.3		
OKBQR	1.6/1.9		
RECURSECTION	3.8/12	0.8/16	0.8/16
WBIS2	13/38	2.3/39	2.6/39
FJLLT	3.0/7.2		

The timing was nearly the same for each x tried, except as noted below.

The errors in RECURSECTION and WBIS2 were at most 2 units in the last place and for STDQR and the other programs at most 2 units in the last place of the largest eigenvalue.

With $x = 10^{-5}$, OKBQR again gave very bad results, with errors in the third decimal place. For x = 0 , that program gave acceptable results, but took 11 seconds to find them.

With x = 10,000, no results were obtained from WBIS2 because floating-point overflow occurred, causing the program to be terminated. Also, we obtained no results from FJLLT for x = 10000 , even after allowing it to run for 4 minutes, because that program is very slow to find near-repeated eigenvalues, especially when underflows intervene.

TEST NO. 3.

Zeros of the Bessel Functions $J_m(x)$

Because of the three-term recurrence relation satisfied by the Bessel functions, the non-trivial zeros $\xi_{k,m}$ ($k = 1,2,\ldots,m > 0$) of $J_m(x)$ are given by

$$\xi_{k,m} = 2/\sqrt{\mu_k} \quad , \text{ where } \mu_1 > \mu_2 > \mu_3 > \cdots$$

are the eigenvalues of the following infinite symmetric tridiagonal matrix:

$$\begin{pmatrix} a_1 & b_1 & & & \\ b_1 & a_2 & b_2 & & \\ & b_2 & a_3 & b_3 & \\ & & \ddots & \ddots & \ddots \end{pmatrix} \quad \text{with} \quad \begin{cases} a_n = \dfrac{2}{(m+2n-1)(m+2n+1)} \\ b_n = \dfrac{1}{(m+2n+1)\sqrt{(m+2n)(m+2n+2)}} \end{cases} , \quad n=1,2,\ldots .$$

Furthermore, the first several μ_k are closely approximated by the eigenvalues of the matrix formed by taking the first N (say) rows and columns of the above matrix, provided N is large enough.

In particular, we took N = 50 to obtain approximations to the first 20 zeros of $J_0(x)$ (and $J_1(x)$), using both the matrix as given above (matrix A), then flipped end-for-end (matrix B).

T sec./F passes eigenvalue

	matrix A (50×50)			matrix B (50×50)		
	K = 50	K = 20	K = 5	K = 50	K = 20	K = 5
STDQR	2.2/0.9			2.9/1.2		
OKBQR	2.7/1.2			3.1/1.4		
RECURSECTION		10/37	2.6/39		4.2/15	0.9/13
WBIS2		22/43	5.2/42		20/43	4.5/42
FJLLT	12/10			18/16		

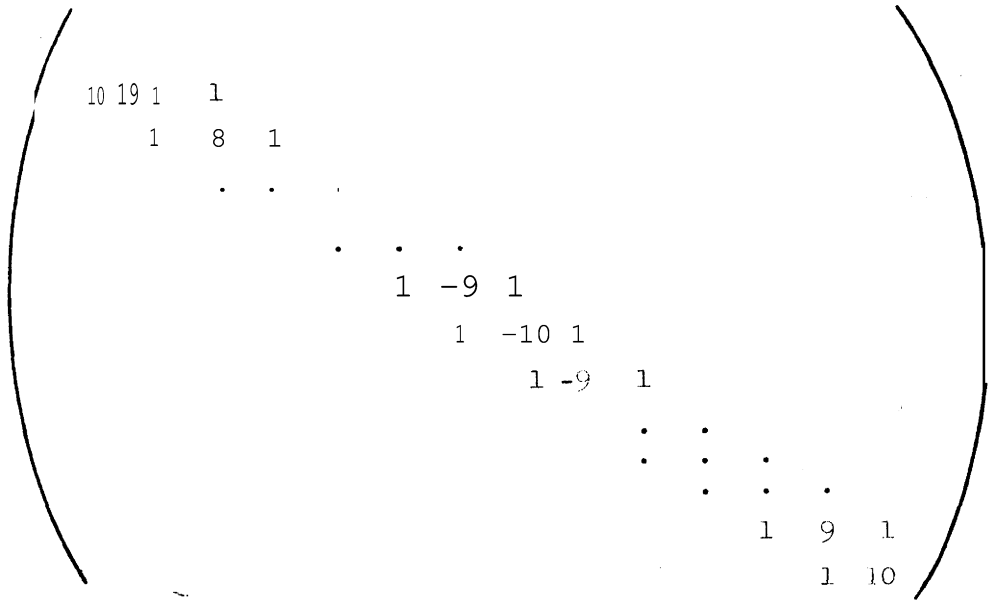
To examine the accuracy, we compared the results with the tables given in [1], pg.409-411. The results from RECURSECTION for both matrices A and B agreed with the tables to the machine limit of 11 decimal digits. The results from WBIS2 agreed to 11 digits for the first two zeros, but the others were progressively more inacurrate, with some incorrect in every digit, because of machine underflow.

The results from STDQR for matrix A were in error by at most 30 units in the last place (for the larger zeros), and by at most 300 units in the last place for matrix B with the small elements at the top of the matrix. However, these errors in the zeros ξ were reflections of absolute errors in the eigenvalues μ of only a few units in the last place of the largest eigenvalue. The results from OKBQR were comparable, and those from FJLLT were somewhat more in error in all cases.

The results for $J_1(x)$ were comparable. The errors did not change when the matrix size was increased from N = 50 to N = 100 , but times were about doubled for RECURSECTION and WBIS2 and quadrupled for STDQR, OKBQR, and FJLLT, since all 100 eigenvalues μ were found in the latter cases.

TEST NO. 4.

Matrix:



N = 41.

T sec./F passes/eigenvalue

	K-N= 41	K = +5	K = -5
YTDQR	1.7/1.0		
OKBQR	2.8/1.8		
KECURSECTION	12/23	0.5/7.0	2.0/35
WILKINSON	23/41	2.5/38	2.7/39
FJLLT	6.7/9.1		

The results from all methods differed by at most 2 units in the last place of the largest eigenvalue. We do not know the eigenvalues exactly, but we list here the results obtained for some of the eigenvalues. Because of the agreement among the methods, we feel these results are correct to the 10 figures given.

The computed eigenvalues are:

10.74619418 (twice)	⋮
9.210678647 (twice)	
8.038941119 (twice)	-7.869790781
7.003952003 (twice)	-8.210678647
6.000225680 (twice)	-9.052465632
.	-9.746194183
	-11.12544152

This matrix is interesting for two reasons. First, its twenty algebraically larger eigenvalues occur in almost indistinguishable pairs, while its ten lesser eigenvalues are well separated. None the less, the well separated eigenvalues are "hidden" to RECURSECTION, which must therefore use the slow binary chop to find them (see under K = -5). The nearly double eigenvalues look like simple zeros of $u_N(x)$ to **RECURSECTION,** which therefore converges to them superlinearly (see under K = +5).

Second, the nearly double eigenvalues do not retard the convergence of the QR algorithms at all. But this is not surprising in view of the known theoretical results about QR (see Wilkinson [11]). What is surprising is that the theoretically nettlesome phenomenon of "disordered latent roots", exhibited by this example, seems to have no more practical significance than that the eigenvalues are not computed in any predictable order.

TEST NO. 5.

Matrix:

$$\begin{pmatrix} 1/2 & 4^{-1} & & & & & \\ 4^{-1} & 4^{-1} & 4^{-2} & & & & \\ & 4^{-2} & 4^{-2} & 4^{-3} & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & & \\ & & & & 4^{2-N} & 4^{2-N} & 4^{1-N} \\ & & & & & 4^{1-N} & \frac{1}{2}(4^{1-N}) \end{pmatrix} \qquad N = 30$$

This matrix was input first as above (matrix A), then turned end-for-end (matrix B).

T sec./F passes/eigenvalue

	Matrix A		Matrix B	
	K = N = 30	K = 5	K = N = 30	K = 5
STDQR	0.2/0.2		0.8/0.8	
OKBQR	0.2/0.2		0.4/0.4	
RECURSECTION	11/39	1.9/39	7.7/25	0.6/12
WBIS2	30/47	5.0/42	27.2/47	3.5/43
FJLLT	0.7/1.6		1.3/2.7	

For matrix B, no results were obtained for RECURSECTION at first, because stack overflow caused the program to be terminated. However, when the stack length was doubled, RECURSECTION ran as usual.

Although we do not know the exact eigenvalues of this matrix, the results from RECURSECTION agree except for the last place in both modes of input. The results from WBIS2 agree comparably only for the largest

two eigenvalues, the error in the others presumably due to underflow. The results from STDQR, OKBQR, and FJLLT differed from those of RECURSECTION by at most 2 units in the last place of the largest eigenvalue.

Changing each element of the matrix by a unit in its last place causes a change in each eigenvalue of at most a few units in its last place, except that the eigenvalue zero may change by an absolute amount comparable to the change in the smaller elements. Only RECURSECTION computed the eigenvalues as accurately as they are determined by the data.

We list here some of the eigenvalues computed by RECURSECTION. Because of the agreement of these results in the two modes of input, we feel these eigerivalues are correct to the 10 places given.

$6.56343370 \times 10^{-1}$

$1.346533638 \times 10^{-1}$

$3.187715678 \times 10^{-2}$

$7.852609156 \times 10^{-3}$

$1.955655725 \times 10^{-3}$

$\vdots$

$\vdots$

$4.440892099 \times 10^{-16}$

$1.110222890 \times 10^{-16}$

$2.774313194 \times 10^{-17}$

$6 \bullet \square \times 10^{-18}$

0.0

Conclusions:

Our analyses and test results indicate that STDQR is the fastest program known to date for finding all the eigenvalues of a symmetric tri-diagonal matrix. The absolute error in each eigenvalue has never exceeded a few units in the last place of the largest. Conceivably, the program could be speeded up, in those cases where only a few eigenvalues are wanted, if some way were found to force the desired eigenvalues to come out first. Until that is accomplished, RECURSECTION is the fastest method known to date for computing a few specified eigenvalues of a very large matrix. This program is also at least as accurate as any general purpose program can be expected to be. With very few changes RECURSECTION can be generalized to cope with the more general eigenproblem

$$\det(A - \lambda B)$$

with symmetric tridiagonal matrices A and B, and positive definite B.

References:

1. Abramowitz, M., and Stegun, I.A. (ed). Handbook of Mathematical Functions. National Bureau of Standards AMS 55 (1965).
2. Businger, P.A. Algorithm 253, Comm. ACM 8(1965), 217-218.
3. Fox, A.J., and Johnson, F.A. On finding the eigenvalues of real symmetric tridiagonal matrices. Comp. J. V9(1966), 98-105.
4. Francis, J.G.F. The QR transformation: A unitary analogue to the LR transformation. Part I Comp. J. V4(1961), 265-271. Part II Comp. J. V4 (1962), 332-345.
5. Ortega, J.M., and Kaiser, H.F. The LL^T and QR methods for symmetric tridiagonal matrices. Comp. J. V6(1963), 99-101.
6. Rutishauser, H. Letter to the Editor. Comp. J. V6(1963), 133.
7. Welsch, J.H. Certification of Algorithm 253, Comm. ACM (to appear).
8. Wilks, Wheeler, and Gill. The Preparation of Programs for an Electronic Digital Computer. Addison-Wesley, 1951.
9. Wilkinson, J.H. Calculation of the eigenvalues of a symmetric tridiagonal matrix by the method of bisection. Num. Math. V4 (1962), 362-367.
10. Wilkinson, J.H. The Algebraic Eigenvalue Problem. Oxforx Press, 1965.
11. Wilkinson, J.H. The QR algorithm for real symmetric matrices with multiple eigenvalues. Comp. J. V8(1965), 85-87.
12. Kahn, W. Accurate eigenvalues of a symmetric tridiagonal matrix. Stanford U. Comp. Sci. Dept. Tech. Rept. CS41.
13. Kahan, W. When to neglect off-diagonal elements of symmetric tri-diagonal matrices. Stanford U. Comp. Sci. Dept. Tech. Rept. CS42.