On Lehmer's Method for Finding the Zeros of a Polynomial

By G. W. Stewart III*

Abstract. Lehmer's method for finding a zero of a polynomial is a procedure for searching the complex plane in such a way that a zero is isolated in a sequence of disks of decreasing radii. In this paper modifications of the method that improve its stability are given. The convergence of the method and the use of the resulting approximate zero to deflate the polynomial are discussed.

1. Introduction. Lehmer's method [4] for finding the zeros of a polynomial

$$f(z) = a_0 + a_1 z + \cdots + a_n z^n , \qquad (a_0 a_n \neq 0) ,$$

is based on a procedure for determining if $f(z)$ has a zero in the closed disk

$$D(s; \rho) = \{z : |z - s| \leqq \rho\} .$$

This procedure is used to search the complex plane in such a way that a zero of $f(z)$ is isolated in a sequence of disks of decreasing radii. When a sufficiently small disk containing a zero is found, the center of that disk is accepted as an approximate zero to be divided out of the polynomial. The process is then restarted, using the reduced polynomial. Of course, at any point in the process an iterative method such as Newton's method may be applied in an attempt to find a zero contained in the current disk.

The method as given by Lehmer tends to be numerically unstable. Specifically, in the procedure for determining if $f(z)$ has a zero in a disk, numbers must be computed that may easily underflow or overflow the floating-point range of most computers. In addition, the searching procedure is organized so that there is some possibility of the method breaking down prematurely. It is the object of this paper to show how these difficulties may be eliminated by suitable modifications in the method.

In the next section the modified method will be described and its differences from the original method pointed out. In Section 3 the problem of convergence will be discussed, with particular attention being paid to the consequences of dividing out the approximate zero. Finally, in Section 4 a PL/I program, appearing in the microfiche section of this journal, that incorporates the results of this paper will be described.

2. The Modified Method. Lehmer's method uses the basic procedure for determining if $f(z)$ has a zero in a disk to search the complex plane for a zero of $f(z)$. One step of the search pattern goes roughly as follows.

Received August 19, 1968, revised May 12, 1969.

* Oak Ridge Graduate Fellow from the University of Tennessee under appointment from the Oak Ridge Associated Universities, operated by Union Carbide Corporation for the U. S. Atomic Energy Commission.

Starting with a disk $D(s; \rho)$ containing a zero of $f(z)$, an annulus

$$A(s'; \rho') = \{z : \rho' < |z - s'| \leqq 2\rho'\}$$

inside $D(s; \rho)$ containing a zero of $f(z)$ is determined. This annulus is then covered by disks and one of them, $D(s''; \rho'')$, containing a zero of $f(z)$ is found. The process is then restarted using the disk $D(s''; \rho'')$. Except perhaps for the first step, each annulus $A(s'; \rho')$ is contained in $D(s; \rho)$. Moreover, after the first step

$$\rho' \leqq \rho/2$$

and

$$\rho'' = 7\rho'/8 , \tag{2.1}$$

so that the process must converge.

Specifically, given the disk $D(s; \rho)$, determine if it contains a zero of $f(z)$. If it does, determine the first positive integer i such that the disk $D(s; 2^{-i}\rho)$ does not contain a zero of $f(z)$, and set

$$\rho' = 2^{-i}\rho .$$

If $D(s; \rho)$ does not contain a zero of $f(z)$, determine the first positive integer i such that $D(s; 2^{i}\rho)$ does contain a zero of $f(z)$, and set

$$\rho' = 2^{i-1}\rho .$$

In either case, if $s' = s$, the annulus $A(s'; \rho')$ contains a zero of $f(z)$.

If $s \neq 0$, let

$$u = -s/|s| ; \tag{2.2}$$

otherwise let u be chosen so that $|u| = 1$. If

$$s_k' = s + \frac{13}{8}\rho' u \exp\left(\frac{k-1}{4}\pi i\right), \qquad (k = 1, 2, \cdots, 8) ,$$

and ρ'' is defined by (2.1), then the disks

$$D_k = D(s_k'; \rho'')$$

cover the annulus $A(s'; \rho')$. Examine the disks D_k for zeros of $f(z)$ in the order $D_1, D_8, D_2, D_7, D_3, D_6, D_4, D_5$. Let D_j be the first of these disks containing a zero of $f(z)$ and let $s'' = s_j'$. This completes one step of the search.

The choice of a starting disk depends on whether a zero has already been found. If one has, let $s = 0$ and ρ be equal to the outer radius of the first annulus obtained in the search for the last zero. If no previous zero has been found, take $s = 0$ and

$$\rho = 1.1|a_0/a_n|^{1/n} .$$

This last choice insures that the starting disk $D(0; \rho)$ contains a zero of $f(z)$.

No disk after the first one can contain the origin. Hence the number u is well defined by (2.2) except in the first step of the search. For the first step the choice of u again depends on whether a zero has already been found. If none has, take $u = 1$. If the last zero found is r, take

$$u = \bar{r}/|r| . \tag{2.3}$$

This choice of u is motivated by the expectation that $f(z)$ will usually have real coefficients and hence conjugate pairs of zeros. If u is defined by (2.3), then, having found the zero r, the search immediately attempts to find a conjugate zero.

The procedure for determining whether $f(z)$ has a zero in $D(s; \rho)$ consists of three steps. First note that $f(z)$ has a zero in $D(s; \rho)$ if and only if

$$h(z) = f(\rho z + s)$$

has a zero in the unit disk $D(0; 1)$. Thus the procedures can be broken up into the following three steps.

1. Calculate the coefficients of

$$g(z) = b_0 + b_1 z + \cdots + b_n z^n = f(z + s) .$$

2. Calculate the coefficients of

$$h(z) = c_0 + c_1 z + \cdots + c_n z^n = g(\rho z) . \tag{2.4}$$

3. Determine whether $h(z)$ has a zero in the unit disk.

The polynomial $g(z)$ is obtained from $f(z)$ by *shifting*, and $h(z)$ from $g(z)$ by *scaling*.

The shifting step can be accomplished by iterated synthetic division:

$$\begin{aligned} b_{n-i}^{(i-1)} &= a_{n-i} , \qquad (i = 0, 1, \cdots, n) , \\ b_n^{(k)} &= b_n^{(k-1)} , \qquad (k = 0, 1, \cdots, n) , \\ b_{n-i}^{(k+1)} &= b_{n-i}^{(k)} + s b_{n-i+1}^{(k)} , \qquad (i = 1, 2, \cdots, k+1; k = 0, 1, \cdots, n-1) . \end{aligned} \tag{2.5}$$

The coefficients of $g(z)$ are given by

$$b_i = b_i^{(n)} , \qquad (i = 0, 1, \cdots, n) .$$

This straightforward scheme offers no special computational difficulties.

More care must be taken with the scaling step. Mathematically, the coefficients of $h(z)$ are given by

$$c_i = \rho^i b_i . \tag{2.6}$$

However, if n is large and $\rho > 1$, the absolute values of the c_i may exceed, or overflow, the largest number representable in the computer performing the calculations. Likewise, if $\rho < 1$, then the absolute values of the c_i may underflow the smallest positive number representable in the computer. Most computers have provisions for setting the results of an underflow producing operation to zero. The following scaling algorithm uses this feature.

Let Ω be the largest positive number that can be represented in the computer. Then a set of c_i, different from those of (2.6), are defined as follows:

1. Determine the largest number σ satisfying

$$\begin{aligned} &0 < \sigma \leqq \Omega , \\ &\sigma |b_i| \leqq \Omega , \qquad (i = 0, 1, \cdots, n) . \end{aligned}$$

2. If $\rho < 1$, set

$$c_i = (\sigma \rho^i) b_i , \qquad (i = 0, 1, \cdots, n) ,$$

where it is understood that $c_i = 0$ if underflow occurs in its computation.

3. If $\rho > 1$, set

$$c_i = (\sigma\rho^{i-n})b_i\,, \qquad (i = n, n-1, \cdots, 0)\,,$$

with $c_i = 0$ if underflow occurs in its computation.

The nonzero c_i defined by this algorithm stand in constant proportion to the c_i defined by (2.6). Overflows cannot occur in the course of the algorithm. The effect of setting underflows to zero is to produce a polynomial slightly perturbed from some constant multiple of $h(z)$ as defined by (2.6). To these perturbations in the coefficients there correspond perturbations in the zeros of $h(z)$. The perturbations in the zeros may be large; for if $\rho < 1$, the degree of the polynomial produced by the scaling algorithm may be less than n. However, the searching procedure only requires that the zeros of $h(z)$ in and about the unit disk be well determined, and it is just these zeros that are least sensitive to the perturbations generated by the scaling algorithm.

The algorithm for determining whether a polynomial has a zero in the unit disk is based on the following theorem due to Schur [5] and Cohn [3].

THEOREM. *With the polynomial*

$$h_0(z) = c_0 + c_1 z + \cdots + c_n z^n\,, \qquad (c_0 \neq 0)\,,$$

associate the polynomial

$$h_0^*(z) = z^n\overline{h}_0(z^{-1}) = \bar{c}_n + \bar{c}_{n-1}z + \cdots + \bar{c}_0 z_n\,.$$

Let $m_0 = c_n/\bar{c}_0$. *Then, if* $|m_0| \geqq 1$, $h_0(z)$ *has a zero in the unit disk. On the other hand, if* $|m_0| < 1$, *the polynomial*

$$h_1(z) = h_0(z) - m_0 h_0^*(z) \tag{2.7}$$

is of degree less than n *and has the same number of zeros in the unit disk as* $h_0(z)$. *Moreover,* $h_1(0) \neq 0$.

The theorem may be applied repeatedly to generate a sequence of polynomials $h_i(z)$, all having the same number of zeros in the unit disk as $h_0(z)$, and a sequence of associated constants m_i. The process terminates either when some $m_i \geqq 1$, in which case $h_0(z)$ has a zero in the unit disk, or when some $h_i(z)$ is constant, in which case $h_0(z)$ has no zeros in the unit disk. This is the basic algorithm for determining if $h_0(z)$ has a zero in the unit disk.

After some value s has been accepted as an approximate zero, it must be divided out of the polynomial:

$$f(z) = (z - s)f_1(z) + f(s)\,.$$

The search is then restarted with the deflated polynomial $f_1(z)$. It is given by

$$f_1(z) = b_1^{(n-1)} + b_2^{(n-2)}z + \cdots + b_n^{(0)}z^{n-1}\,,$$

where the $b_i^{(k)}$ are defined by (2.5).

The method proposed in this section differs from Lehmer's original method in a number of ways. In the search pattern the orientation of the disks covering an annulus and the order in which they are examined have been changed to enhance the tendency of the method to find smaller zeros first. This tends to increase the

stability of the deflation process [7, pp. 56–59]. However, the resulting improvement is marginal, and other patterns may be preferable. The referee has suggested two which may save machine computations. The first pattern takes $u = 1$ and examines the disks in the order 1 5 3 7 2 4 6 8, which tends to capture real zeros first. The second always examines the disks in the order 1 4 7 2 5 8 3 6, so that the fourth disk examined is the first to overlap with its predecessors.

The procedure for determining if $f(z)$ has a zero in a given disk must of course be carried out with rounding error. This means that if a zero of $f(z)$ lies very near the boundary of the disk, the procedure may locate it inside the disk when it is actually outside, and *vice versa*. In particular, if the covering disks do not overlap the annulus sufficiently, a zero lying near the boundary of the annulus may be located in the annulus but fail to appear in any of the covering disks. To avoid such a premature breakdown in the search, the size and position of the disks have been adjusted so that any point of the annulus lying near the boundary of one disk lies well within another disk.

The scaling algorithm has been modified as described above to deal with the problem of overflows and underflows.

The Schur-Cohn algorithm has been changed from its original form [3], which, instead of forming the polynomial $h_1(z)$ of Eq. (2.7), works with the polynomial

$$\tilde{h}_1(z) = \bar{c}_0 h_0(z) - c_n h_0^*(z) . \tag{2.8}$$

While each of the polynomials $\tilde{h}_i(z)$ is a constant multiple of the $h_i(z)$ obtained from (2.7), their coefficients can increase or decrease so rapidly that overflow or underflow becomes a serious problem. On the other hand, the coefficients of the polynomials $\tilde{h}_i$ can at most double in size at each step. Note also the computation of $\tilde{h}_i$ requires only half as many multiplications as the computation of $\tilde{h}_i$.

Finally, some comments on the stability of the Schur-Cohn algorithm are in order. An error analysis of the algorithm in [6] shows that the algorithm can be numerically unstable in the sense that the rounding errors made in computing the h_i and m_i may correspond to large perturbations in the coefficients of h_0. Such an instability is signaled by the emergence of an m_i with modulus very near unity. In somewhat limited experiments, the author has not encountered a gross failure of the algorithm; moreover, the overlapping of the disks mentioned above provides some protection against mild instabilities. If, however, greater security is desired, one can monitor the m_i and, following Lehmer, enlarge the offending disk when an instability occurs.

3. Convergence. In determining a convergence criterion for the method, two points must be kept in mind. The first is that there is a limit to the accuracy obtainable by the method, for when the disks become small enough rounding errors will cause the search to fail to find a zero in the current annulus. The second point is that the approximate zero finally accepted must be used in the deflation process. If it is in error, its inaccuracies may be transmitted to the remaining zeros. Thus we must answer two questions. First, how accurate must an approximate zero be before it can be safely used in the deflation process? Second, can the method outlined in the last section attain that accuracy?

Wilkinson [7] has analyzed the deflation process in detail. The following is a summary of his results.

The accuracy attainable in any zero is limited by the sensitivity of the zero to small relative perturbations in the coefficients of a size corresponding to changing the low-order portions of the machine representation of the coefficients. Each perturbation in the coefficients causes a corresponding perturbation in the zero, and as the perturbations vary the perturbed zero traces out a region of indeterminacy about the original zero. The best that can be expected of a numerical method operating at a fixed precision is that it locate an approximate zero within this region of indeterminacy. A zero having a small region of indeterminacy is called a well-conditioned zero; one having a large region is called an ill-conditioned zero.

Wilkinson has shown that an approximate zero may be used in the deflation process without unduly affecting the other well-conditioned zeros, provided first that the zero divided out is among the smallest in absolute value and second that the approximate zero lies in the region of indeterminacy. The search pattern of Lehmer's method tends to find smaller zeros first, so that the first condition is satisfied. In [6] arguments are given to indicate that for a simple zero Lehmer's method will break down only when the center of the current annulus is near the region of indeterminacy for the zero being located. There is numerical evidence for believing that this is also true of multiple zeros. Thus it is recommended that Lehmer's method be allowed to proceed until it breaks down, at which point the center of the offending annulus is to be accepted as an approximate zero.

4. A PL/I Program. The program listed in the microfiche section of this journal is a straightforward implementation of the method described in Section 2. The program returns a set of approximate zeros and a set of condition numbers

$$\text{cond}\ (z) = f_a(|z|)/|f'(z)|\ ,$$

where

$$f_a(z) = |a_0| + |a_1|z + \cdots + |a_n|z^n\ .$$

For a simple zero z, η cond (z) gives an estimate of the perturbation induced in z by relative perturbations in the coefficients of size η. This is treated in more detail in [6].

Since the program is quite slow, the user may wish to apply an iteration such as Newton's method in an attempt to find a zero in the current disk. When this is done, the iteration should be continued until the limiting accuracy described in Section 3 has been reached (see [8, pp. 461–464]; also [1] and [2]).

In the author's experience, the program produces a set of approximate zeros that belong to a polynomial with coefficients very near those of the original polynomial, a strong indication that the modified Lehmer's method attains the limiting accuracy of a zero before it breaks down. Of course, the zeros themselves may be quite inaccurate.

5. Acknowledgement. I am indebted to Professor A. S. Householder for suggesting this work and for his valuable comments and criticisms.

University of Texas
Department of Computer Sciences
Austin, Texas 78712

→ D. A. ADAMS, "A stopping criterion for polynomial root finding," *Comm. ACM*, v. 10, 1967, pp. 655–658.
2. B. W. BOEHM, Review No. 14,748 of Adams' "A stopping criterion for polynomial root finding," *Comput. Rev.*, v. 9, 1968, pp. 395–396.
3. A. COHN, "Über die Anzahl der Wurzeln einer algebraischen Gleichung in einem Kreise," *Math. Z.*, v. 14, 1922, pp. 110–148.
4. D. H. LEHMER, "A machine method for solving polynomial equations," *J. Assoc. Comput. Mach.*, v. 8, 1961, pp. 151–162.
5. J. SCHUR, "Über algebraische Gleichungen, die nur Wurzeln mit negativen Realteilen besitzen," *Z. Angew. Math. Mech.*, v. 1, 1920, pp. 307–311.
6. G. W. STEWART III, *Some Topics in Numerical Analysis*, Oak Ridge National Laboratory Report ORNL-4303, Sept. 1968.
7. J. H. WILKINSON, *Rounding Errors in Algebraic Processes*, Prentice-Hall, Englewood Cliffs, N. J., 1963. MR **28** #4661.
8. J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965. MR **32** #1894.

```
      IF ØLDR=0
         THEN DØ;
            R = 1.1*ABS(A(0))**(1/FLØAT(N));
            CALL ANNULUS(A,R,'1'B,N);
         END;
         ELSE DØ;
            R = ØLDR;
            CALL ANNULUS(A,R,'0'B,N);
         END;
      ØLDR = R;
      IF R=0 THEN GØ TØ DEFLATE;

FIND: /* CØVER THE ANNULUS OF INNER RADIUS R AND CENTER S
         WITH EIGHT DISKS AND ATTEMPT TØ FIND A DISK CØNTAINING
         A ZERØ.  IF THE ANNULUS IS CENTERED ABØUT THE ØRIGIN,
         PLACE THE FIRST DISK IN THE DIRECTIØN ØF THE CØNJUGATE
         OF THE LAST ZERØ FØUND.  IF IT DØES NØT, PLACE THE
         FIRST DISK AS NEAR AS PØSSIBLE TØ THE ØRIGIN AND TEST
         THE NEARER DISKS FIRST.                                  */

      DELTAS = CØNS*R:
      IF S¬=0
         THEN DELTAS = - DELTAS*(S/ABS(S));
         ELSE IF N¬=NN
            THEN IF Z(NN-N)¬=0
               THEN DELTAS = DELTAS*(CØNJG(Z(NN-N))/ABS(Z(NN-N)));
      RP = CØNR*R;
      DØ K=1 TO 8;
         SK = S + DELTAS*UNIT(K);
         IF SK = S THEN GØ TØ NØZERØ;
         CALL FUNDER(A,B,SK,N,N-1);
         CALL SCALE(B,C,RP,N);
         IF CØHN(C,N) THEN GØ TØ ZERØ;
      END;

NØZERØ: /* NØNE ØF THE CØVERING DISKS IS FØUND TØ CØNTAIN A ZERØ.
         THIS FAILURE IS CAUSED BY RØUNDING ERRØR AND INDICATES
         THAT THE METHØD CANNØT PRØCEED WITHØUT INCREASING
         THE PRECISIØN ØF THE ARITHMETIC.  HERE THE FAILURE IS
         TREATED AS EFFECTIVE CØNVERGENCE, AND THE CENTER ØF
         THE ANNULUS IS ACCEPTED AS AN APPRØXIMATE ZERØ.

      GØ TØ DEFLATE;
```

```
   ZERØ: /* A ZERØ HAS BEEN LØCATED IN THE DISK WITH CENTER SK AND
            RADIUS RP.  DETERMINE A NEW ANNULUS AND PRØCEED WITH
            THE SEARCH.  AT THIS PØINT THE USER MAY WISH TØ IMPRØVE
            THE EFFICIENCY ØF THE PRØGRAM BY ATTEMPTING TØ FIND
            A ZERØ IN THE DISK USING AN ITERATIVE METHØD.  IF THE
            ATTEMPT IS SUCCESSFUL, THE RESULT SHØULD BE PLACED IN
            S AND CØNTRØL TRANSFERRED TØ DEFLATE.                  */

         S = SK;
         R = RP;
         CALL FUNDER(A,B,S,N,N-1);
         CALL ANNULUS(B,R,'1'B,N);
         IF R=0
            THEN GØ TØ DEFLATE;
            ELSE GØ TØ FIND;

DEFLATE: /* DIVIDE THE APPRØXIMATE ZERØ S ØUT ØF THE PØLYNØMIAL.     */

         CALL FUNDER(A,B,S,N,0);
         N = N-1;
         Z(NN-N) = S;
         DØ I=0 TØ N; A(I)=B(I+1); END;
         IF N =1 THEN GØ TØ START;
         Z(NN) = -A(0);

 FINISH: /* FØR EACH ZERØ Z CØMPUTE THE CØNDITIØN NUMBER

                        CØND(Z) = FA(ABS(Z))/ABS(FP(Z)),

            WHERE FP IS THE DERIVATIVE ØF F AND FA IS THE PØLYNØMIAL
            WHØSE CØEFFICIENTS ARE THE ABSØLUTE VALUE ØF THØSE
            ØF F.                                                  */

         ØN ØVERFLØW GØ TØ ENDF;
         ØN ZERØDIVIDE GØ TØ ENDF;
         DØ I=0 TØ NN; A(I)=ABS(AA(I)); END;
         DØ K=1 TØ NN;
            CØND(K) = -1;
            S = ABS(Z(K));
            CALL FUNDER(A,B,S,NN,0);
            CALL FUNDER(AA,C,Z(K),NN,1);
            CØND(K) = B(0)/ABS(C(1));
   ENDF: END;
         RETURN;
```

```
FUNDER: PRØCEDURE(A,B,S,N,NC);

        /* FUNDER CØMPUTES BY ITERATED SYNTHETIC DIVISIØN THE
           FIRST NC+1 CØEFFICIENTS ØF THE TAYLØR EXPANSIØN ABØU
           S ØF THE PØLYNØMIAL WHØSE CØEFFICIENTS ARE CØNTAINED
           IN THE ARRAY A.  THE RESULTS ARE RETURNED IN THE
           FIRST NC+1 LØCATIONS ØF THE ARRAY B.

        DECLARE ((A,B)(*),S) CØMPLEX(16),
                (I,J,NNC) STATIC FIXED BINARY;
        DØ I=0 TØ N; B(I)=A(I); END;
        NNC = MIN(NC,N-1);
        DØ I=0 TØ NNC;
           DØ J=N TØ I+1 BY -1;
              B(J-1) = B(J-1) + S*B(J);
           END;
        END;
        RETURN;
        END FUNDER;

SCALE:  PRØCEDURE(A,B,SCLFAC,N);

        /* SCALE RETURNS IN B A CØNSTANT MULTIPLE ØF THE NUMBERS
           A(I)*SCLFAC**I.  THIS IS DØNE IN SUCH A WAY THAT
           ØVERFLØWS CANNØT ØCCUR.  WHEN UNDERFLØWS ØCCUR, THE
           RESULT IS SET TO ZERØ.                                  */

        DECLARE (A,B)(*) CØMPLEX(16), (SCLFAC,(R,MX) STATIC) REAL(16),
                I STATIC FIXED BINARY;
        MX = 0;
        DØ I=0 TØ N; MX = MAX(MX,ABS(A(I))); END;
        R = BIGØMEGA/MX;
        IF SCLFAC<1
           THEN DØ I=0 TØ N;
              B(I) = R*A(I);
              R = R*SCLFAC;
           END;
           ELSE DØ I=N TØ 0 BY -1;
              B(I) = R*A(I);
              R = R/SCLFAC;
           END;
        RETURN;
        END SCALE;
```

```
ANNULUS: PRØCEDURE(A,R,RØØT,N);

         /* ANNULUS DETERMINES AN ANNULUS ABØUT THE ØRIGIN Ø! INNER
            RADIUS R AND ØUTER RADIUS 2*R CØNTAINING A ZERØ
            ØF THE PØLYNØMIAL WHØSE CØEFFICIENTS ARE CØNTAINED
            IN A.  IF THE PØLYNØMIAL HAS A ZERØ AT THE ØRIGIN,
            THE VALUE ZERØ IS RETURNED IN R.  INITIALLY R CØNTAINS
            AN ESTIMATE OF THE ØUTER RADIUS ØF THE ANNULUS.  IF
            RØØT/ IS TRUE, THE CIRCLE ABØUT THE ØRIGIN ØF RADIUS
            R IS ASSUMED TØ CØNTAIN A ZERØ.

         DECLARE A(*) CØMPLEX(16), (R,R2 STATIC) REAL(16),
                 RØØT BIT(1);
         IF A(0)=0
            THEN DØ;
               R = 0;
               RETURN;
            END;
         IF RØØT THEN GØ TØ CNTRCT;
         R2 = R;
         CALL SCALE(A,C,R,N);
         IF CØHN(C,N)
            THEN GØ TØ CNTRCT;
            ELSE GØ TØ EXPAND;
CNTRCT:  R2 = R;
         R = R/2;
         CALL SCALE(A,C,R,N);
         IF CØHN(C,N)
            THEN GØ TØ CNTRCT;
            ELSE RETURN;
EXPAND:  R = R2;
         R2 = 2*R;
         CALL SCALE(A,C,R2,N);
         IF CØHN(C,N)
            THEN RETURN;
            ELSE GØ TØ EXPAND;
         END ANNULUS;

  CØHN:  PRØCEDURE(A,N) BIT(1);

         /* CØHN RETURNS A VALUE OF '1'B IF THE PØLYNØMIAL WHØSE
            CØEFFICIENTS ARE CØNTAINED IN A HAS A ZERØ IN
            THE CLØSED UNIT DISK.  CØHN DESTRØYS THE ARRAY A.
```

```
        DECLARE (A(*),(M,TEMP) STATIC) COMPLEX(16),
                (MX REAL(16), (I,L) FIXED BINARY) STATIC;
        MX = 0;
        DO I=0 TO N; MX = MAX(MX,ABS(A(I))); END;
        L = 0;
        DO I=N TO 0 BY -1;
           A(I) = A(I)/MX;
           IF A(I) =0 & L=0 THEN L=I;
        END;
        IF A(0)=0 THEN RETURN('1'B);
LOOP:   M = A(L)/CONJG(A(0));
        IF ABS(M)>=1 THEN RETURN('1'B);
        IF L=1 THEN RETURN('0'B);
        A(0) = A(0) - M*CONJG(A(L));
        L = L-1;
        DO I=1 BY 1 WHILE(I<=L-I+1);
           TEMP = A(L-I+1) - M*CONJG(A(I));
           A(I) = A(I) - M*CONJG(A(L-I+1));
           A(L-I+1) = TEMP;
        END;
        GO TO LOOP;
        END COHN;
        END LEHMER;
```

The editorial committee would welcome readers' comments about this microfiche feature. Please send comments to Professor Eugene Isaacson, MATHEMATICS OF COMPUTATION, Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, New York 10012.

Mathematics of Computation

TABLE OF CONTENTS

OCTOBER 1969