A Numerical Method for Locating the Zeros of an Analytic Function*

By L. M. Delves† and J. N. Lyness‡

1. Introduction. In this paper we study the following problem: How can we locate all the zeros of a given analytic function $f(z)$ which lie in a given region R.

A number of methods are currently available for the determination of the zeros of an analytic function $f(z)$. The best known and certainly the simplest class of methods are iterative, as exemplified by the familiar Newton iteration:

$$z_{n+1} = z_n - f(z_n)/f'(z_n) . \tag{1.1}$$

Such methods work very well if the approximate location of the zeros is known in advance, and hence one- or two-step iterative methods are often used as the final stage in the calculation of the zeros, to refine approximations obtained by other means. However, it is a difficult task to attempt to obtain all the zeros of $f(z)$ within a given finite region using only iterative methods. If the number of zeros is not known at the outset, the result may be unreliable in the sense that some of the zeros may not be discovered. On the other hand, if the region contains no zeros one may search long and fruitlessly for something which does not exist. Even if the number of zeros in the region is known, it may be difficult to coax the iterations into converging, or to stop them always converging to the same zero. Of course, this last difficulty can be obviated by successively removing the zeros as they are found, but while deflation is a convenient procedure for polynomials, and results in a simpler polynomial, for analytic functions in general the division cannot be carried out explicitly, and hence the resulting functional form becomes progressively more complicated.

If the exact power series expansion of $f(z)$ is available, methods based on the *qd* table are available (Rutishauser [8]). So far as the authors are aware these methods have not been developed to the extent of producing an automatic method for an automatic computer (Henrici and Watkins [1]). The existence of arbitrarily large terms in parts of the table, which give rise to cancellation errors which are not predictable, is an undesirable feature of an otherwise very powerful technique. However, a more powerful practical objection to these methods is that in general the power series expansion may not be available without a prohibitive amount of work.

If $f(z)$ is a polynomial, there exists a large number of special methods to determine its roots. An extensive summary of the methods available in 1951 is given by Olver [7]. He discusses among others the Aitken-Bernoulli process and the Graeffe (root squaring) method with their use with a hand calculating machine in

Received November 9, 1966.

* Research sponsored by the U. S. Atomic Energy Commission under contract with the Union Carbide Corporation.

† On leave of absence from the University of Sussex, Falmer, Brighton, England.
‡ On leave of absence from the University of New South Wales, Australia.
Present address: Argonne National Laboratory, Argonne, Illinois.

mind. Each of the methods he describes will solve a low-degree polynomial (degree six or less) without difficulty; but for higher-degree polynomials each involves constant supervision to avoid pitfalls connected with possible ill conditioning and the building up of rounding errors or of cancellation errors.

Since 1951 two alternative methods for polynomials have been developed. One, based on the *qd* algorithm, is mentioned above. The other, now known as Lehmer's method (Lehmer [3]) contains the following two features. First, there is a basic algorithm which determines whether or not there is a root of $f(z)$, a polynomial of degree n, within a given circle. The algorithm involves constructing a Sturm sequence of up to n elements, and noting the number of sign changes. Second, a search program constructs circles of successively smaller radius until a root is sufficiently well isolated. It is then divided out, and the process repeated until all n roots have been located. Although this method is designed for automatic computers, it is very susceptible to machine underflow or overflow. This difficulty is practically nonexistent for polynomials of degree six or less, but gets rapidly worse with increasing degree n. It can be alleviated, but not removed, by performing multiple-precision arithmetic within the routine, including a complete machine word for the floating-point exponent.

One possible (and commonly advocated) approach if $f(z)$ is an analytic function is to replace it by a polynomial $p(z)$ which approximates $f(z)$ in the region under consideration. The polynomial $p(z)$ may be an interpolating polynomial or simply a truncated power series. This procedure has the obvious advantage that many methods are available to find the zeros of the polynomial. Once these are found they may perhaps be used as starting values for an iterative process to find the corresponding zeros of $f(z)$. The major disadvantage is that the zeros of $f(z)$ may bear little or no relation to those of $p(z)$. An extreme case is given by the choice

$$f(z) = e^z \tag{1.2}$$

and

$$p(z) = \sum_{j=0}^{n} z^j/j! \,. \tag{1.3}$$

Here $f(z)$ has no zeros in any finite region of the complex plane while $p(z)$ clearly has n zeros. An approach of this sort is described in Lehmer [2]. It is clear from the example given in that paper that it requires alert human supervision to carefully avoid this kind of pitfall.

The method we describe in the remainder of this paper is designed specifically to remove any need for human supervision, and to give a reliable result in the sense that all the zeros of the given analytic function within the given region are determined to a given accuracy.

The method bears a strong family resemblance to Lehmer's method, in that it involves the reiteration of what Lehmer refers to as a Basic Question for a region, followed by a subdivision of the region. For Lehmer, the Basic Question is: Does a given region contain at least one root? and if the answer is yes, the region is covered by smaller subregions and the Question asked in each of these, the process being repeated until a root is sufficiently tightly bracketed. The procedure of this paper asks a somewhat stronger Basic Question:

How many zeros (of the analytic function) does the region contain?

If the answer is greater than a preassigned number M, the region is subdivided suitably and the question is asked again. If there are as few as M zeros in the region, a polynomial is constructed having the same zeros within the region as $f(z)$, and these zeros obtained by solving the polynomial. Hence the method isolates up to M zeros simultaneously rather than one as does Lehmer; and the Basic Question is asked very few times, since these M roots are localized by solving a polynomial rather than by successive subdivision. On the other hand, the construction of the polynomial is relatively expensive.

Both the determination of the number of zeros and the construction of the polynomial depend on the explicit use of Cauchy's theorem; specifically, we proceed as follows.

If C is a closed curve in the complex plane which does not pass through a zero of $f(z)$ and R is the interior of C, it is well known from the theory of complex variables that

$$s_N = \frac{1}{2\pi i} \int_C z^N \frac{f'(z)}{f(z)}\, dz = \sum_{i=1}^{\nu} z_i^N \tag{1.4}$$

where z_i ($i = 1, 2, \cdots, \nu$) are all the zeros of $f(z)$ which lie in R. (A multiple zero is counted according to its multiplicity in this formula.) Thus in principle if we are considering a region R, and carry out the contour integral numerically for several values of N, we may determine approximations to $s_0, s_1, s_2, \cdots$. The true value of s_0 is an integer ν, the number of zeros of $f(z)$ in R. Using these approximations we may write down a polynomial $p(z)$ of degree ν whose zeros coincide with the zeros of $f(z)$ in R. We should perhaps note that $p(z)$ is in no sense required to be an approximating polynomial to $f(z)$; it merely shares zeros with $f(z)$, and has the computational advantage that, provided ν is sufficiently small, we may regard the calculation of its zeros as a standard procedure. Of course, $f(z)$ may have many zeros in the given region R. In this case, the polynomial $p(z)$ is of high degree, and may well (in fact, life being what it is, probably will) be very ill conditioned. We would have to determine the s_N to high accuracy for the root of the resulting polynomial to adequately approximate those of $f(z)$, and then maintain this accuracy in the solution of $p(z)$. We avoid this difficulty as it arises by subdividing R into regions each of which contains only a few zeros. Thus the method falls into four sections:

(1) Evaluate the number of roots $s_0 = \nu$ in the region. If this is few enough to handle conveniently, evaluate also $s_1, s_2, \cdots, s_\nu$ and carry on to section (3). If not,

(2) Subdivide the region into smaller subregions and do (1) on each in turn.

(3) Given a suitable region and the evaluated $s_1 \cdots s_\nu$, construct and solve the equivalent polynomial $p(z)$.

(4) An optional section which takes the roots of $p(z)$ as approximations to the zeros of $f(z)$ and refines these using an iterative method on the original function $f(z)$.

In the following paragraphs we describe these sections in some detail. We note that the speed with which the method operates depends critically upon the efficiency with which the contour integrals are evaluated. This depends on having adequate quadrature rules for the curve C. We have considered two kinds of region R, circles and squares. In the case of circles we have developed three methods (which we refer to as A, B and C) for determining s_N (Eq. (1.4)).

2. The Basic Subroutine; Method A.

The Organization. We refer to the subroutine which carries out the contour integration as the basic subroutine. This routine is given:

(i) a contour C;

(ii) a function $f(z)$ and $f'(z)$ analytic on and within this contour;

(iii) a list of the known zeros $z_1, z_2, \cdots, z_k$ of $f(z)$ which have so far been obtained;

(iv) constants M, K and ϵ (see below).

The routine attempts to calculate the number of zeros of $f(z)$ within C using trapezoidal rule approximations to the contour integral

$$s_0 = \frac{1}{2\pi i}\int_C \frac{f'(z)}{f(z)}\,dz\,. \tag{2.1}$$

Since the exact result s_0 is known to be an integer, the accuracy required is low. We need only determine unambiguously which integer is involved. There are three possible outcomes. These are

(a) It finds that $f(z)$ becomes unduly small on the contour and takes this to imply that there is a zero of $f(z)$ close to the contour. In this case the integration method, if continued, would converge slowly. The routine does not continue the integration, but returns control to the search routine, which in turn chooses a different contour.

(b) It finds a value of s_0. On checking the list of known zeros it finds that q of these lie within the region C, and hence there are $s_0 - q$ unknown zeros within C. If $s_0 - q > M$ it returns control to the search routine.

(c) It proceeds as in (b) but finds a value $s_0 - q \leqq M$. It then evaluates the $s_0 - q$ unknown zeros as follows. It evaluates approximations to the sums of powers of zeros

$$s_N = \sum_{i=1}^{s_0} z_i^N\,, \qquad N = 0, 1, \cdots, s_0 - q\,, \tag{2.2}$$

using trapezoidal rule approximations to the integral

$$s_N = \frac{1}{2\pi i}\int_C z^N \frac{f'(z)}{f(z)}\,dz\,. \tag{2.3}$$

The method of integration is described in more detail below. Since the locations of the known zeros $z_1\, z_2 \cdots z_q$ are available, the sums $\bar{s}_N$ of the powers of the unknown zeros are

$$\bar{s}_N = \sum_{i=q+1}^{s_0} z_i^N = s_N - \sum_{i=1}^{q} z_i^N\,, \qquad N = 0, 1, \cdots, s_0 - q\,. \tag{2.4}$$

Based on these numbers, a polynomial of degree $s_0 - q$ may be constructed, using Newton's formulas, which has z_i ($i = q + 1, \cdots s_0$) as zeros. This polynomial is solved using the polynomial root-finding subroutine.

Local Deflation. We term the process of subtracting the already known zeros described by (2.4), *local deflation*. It performs much the same function as does deflation for a polynomial; that is, it avoids finding any root twice. However, since deflation is carried out only over a given region, it does not suffer from the danger of accumulation of round-off errors, as does polynomial deflation (Wilkinson [9]).**

Local deflation is not an essential feature of the method described here; if it is not used, the only consequence is that additional regions may have to be searched. However, the total time taken depends on the number of regions searched rather than on the number of zeros. Thus the use of local deflation results in a considerable saving of time.

Programming Details. The integration is abandoned under heading (a) if there is a zero close to the contour, as manifested by the occurrence of a large value of $|f'(z)/f(z)|$. The value of this function is of order $1/\rho$, where ρ is the distance from the contour to the nearest zero. Thus if $r|f'(z)/f(z)|$ exceeds some pre-set value K, indicating that ρ is less than r/K, the current integration is abandoned. Section 7 of (Lyness and Delves) [5], which we refer to as Paper B, is devoted to a discussion of this point.

It is necessary to give the integration routine some convergence criteria for the integrals s_N. We have already noted that s_0 need only be evaluated quite crudely. The routine we have written accepts a set of values $s_1, s_2, \cdots, s_\nu$ if each agrees with a previous iterate to within a pre-set constant ϵ.

The routine also requires a number M, the maximum number of zeros which it is allowed to handle. The choice of M involves a compromise. The effect of increasing M is that fewer regions need be scanned. However, if we choose M too large, the resulting equivalent polynomial may be ill conditioned. We would then have to evaluate $s_1, s_2, \cdots, s_\nu$ to too high an accuracy. We have generally chosen $M = 5$.

Integration Method; Circles. We define $C(z_0, r)$ as the circle, center z_0 and radius r, in the complex plane. We translate the origin to the center of this circle and introduce the integration parameter t given by

$$z = z_0 + r \exp(2\pi i t) . \tag{2.5}$$

The zeros of $f(z)$ are denoted by

$$z_j = z_0 + r_j \exp(2\pi i t_j) , \qquad j = 1, 2, \cdots , \tag{2.6}$$

and we find

$$\frac{s_N}{r^N} = \int_0^1 \exp(2\pi i(N+1)t) \frac{rf'(z_0 + r\exp(2\pi i t))}{f(z_0 + r\exp(2\pi i t))} dt , \quad N = 0, 1, 2 \cdots . \tag{2.7}$$

Since the integrand is a periodic function of t with period 1, an appropriate rule is the trapezoidal rule. This is discussed in Paper B. Denoting by $\phi_N(t)$ the integrand in (2.7) and using standard functional notation, we define

$$R^{[m,1]}\phi_N = \frac{1}{m}\sum_{j=1}^{m} \phi_N\left(\frac{j}{m}\right) = \frac{1}{m}\sum_{j=1}^{m} \phi_0\left(\frac{j}{m}\right) \exp(2\pi i N j/m) . \tag{2.8}$$

It follows that

$$R^{[2m,1]}\phi_N = \frac{1}{2}R^{[m,1]}\phi_N + \frac{1}{2m}\sum_{j=1}^{m}\phi_N\left(\frac{2j-1}{2m}\right) . \tag{2.9}$$

** We note that deflation as carried out in Lehmer's method is stable.

Thus the integral in each of the $(\nu + 1)$ calculations depends on the same set of function evaluations of $f'(z)/f(z)$ and, as is usual in trapezoidal rule calculations, increasing the number of points for function evaluation from m to $2m$ requires only m additional function evaluations.

We describe the error analysis of this process in Paper B. There it is shown that asymptotically

$$|R^{[m,1]}\phi - I\phi| \sim O(A^m) \tag{2.10}$$

where $|A| < 1$. More precisely

$$|A| = \max (A_1, A_2, A_3) \tag{2.11}$$

where

$$A_1 = \max_{r_j<r} r_j/r\,, \qquad A_2 = \max_{r_j>r} r/r_j\,, \qquad A_3 = r/R$$

and R is the distance from z_0 of the nearest point at which $f(z)$ is not analytic. The convergence of the sum to the integral is linear in the usual sense; that is, each additional function evaluation reduces the error by a constant factor.

Integration Method; Squares. We define $s(z_0, r)$ as the square whose vertices are $z_0 + (\pm 1 \pm i)r$. We use an m-point trapezoidal rule on each side separately, and eliminate early terms of the asymptotic expansion of the discretization error, following the method of Romberg. It is shown in Paper B that

$$R^{(m)}f - If \sim c_2/m^2 + c_6/m^6 + c_{10}/m^{10} + \cdots\,, \tag{2.12}$$

where If stands for the true contour integral and $R^{(m)}f$ for the discretization using the m-point trapezoidal rule for each side. Using $m = 1, 2, 4, 8, \cdots, 2^p$, it is shown that after using 2^{p+2} points the error E is bounded by

$$|E| < r\,\frac{(4p + 2)!4^{-p^2}}{(2\pi\rho)^{2p+2}}\,K \tag{2.13}$$

where $2r\rho$ is the shortest distance between any zero of $f(z)$ and a side of the square and K is a constant. Asymptotically, in terms of the number of points n, the discretization error is shown in Paper B to be dominated by a term $1/n^{2\log_2 n}$ as $n \to \infty$. This is a slower rate of convergence than for the circle.

3. The Search Routine. The search routine is that part of the program which splits a given region R containing too many zeros, into smaller regions $R_1, \cdots, R_j$. We have investigated only very simple schemes in which $R_1, R_2, \cdots, R_j$ have the same shape as R. The method of subdivision we have used is as follows:

(a) *Squares.* The square is divided into four as shown in Fig. 1 (a).

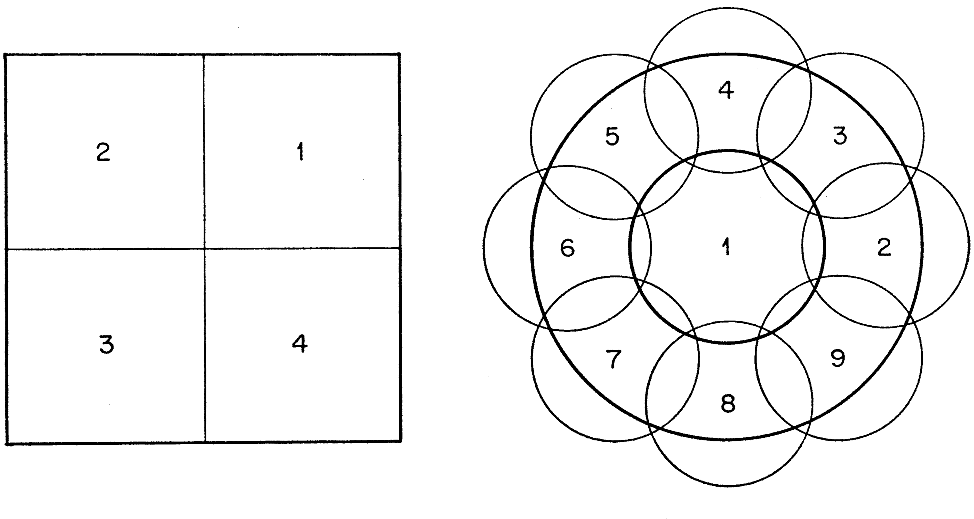
(b) *Circles.* We have adopted Lehmer's subdivision; that is, we cover the circle of radius R with nine smaller circles, namely

(1) a concentric circle of radius $R/2$,

(2) eight circles of radius $5R/12$ regularly spaced around the remaining annulus.

This scheme is shown in Fig. 1 (b).

The subregions are then treated in turn in the order indicated by the subroutine. This order is broken if at any stage the number of zeros in R remaining to be located becomes less than or equal to M. In this case the region R is searched again.



(a) (b)

FIGURE 1. The method used to subdivide squares and circles. In each case the original region is represented by the heavy line. Square regions have the advantage that they can be subdivided without redundancy. On the other hand, more efficient integration rules can be obtained for circles. The numbers in the subregions denote the order in which they are treated. This order may be altered if it is apparent that all the roots have been found, or that those remaining can be found by a faster alternative procedure (see text).

If the basic subroutine indicates that a region R_j is unsuitable (because a zero lies close to the boundary) the region is extended.

For the circle this extension is simple; the radius of the current circle is increased. The procedure for squares is more complicated, since these possess common sides and the program looks ahead to avoid future trouble. A number of different cases arise which are clumsy to describe but easy to implement; a typical case is shown in Fig. 2, which portrays the rescue operation when trouble is struck (for the first time) on (an (inside) edge of) the second subregion of a given square.

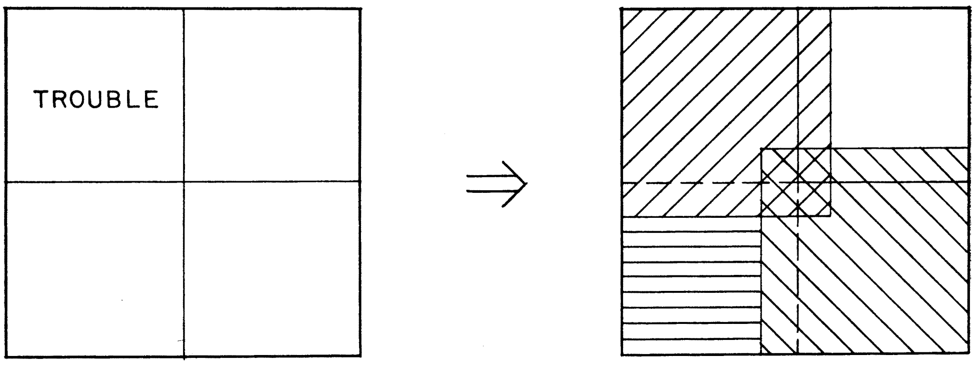


FIGURE 2. An example of the algorithm used for avoiding roots near the edge of the square subregions. In this example, square 1 was successfully treated, and the trouble encountered on an inner edge of square 2. The program readjusted squares 2, 3, and 4 as shown. As a result a small area is covered more than once.

An Example. The working of the search routine is illustrated by a practical example taken from Lehmer [2]. We choose

$$f(z) = J_1^2(z) - J_0(z)J_2(z) .$$

This is a real analytic function of z containing a double zero at the origin and four other complex zeros of the form $\pm\alpha_1 \pm i\alpha_2$ inside a square of side $2R = 12$, center the origin. Fig. 3 illustrates how the program located these zeros when it was allowed to deal with groups of up to three zeros at a time.

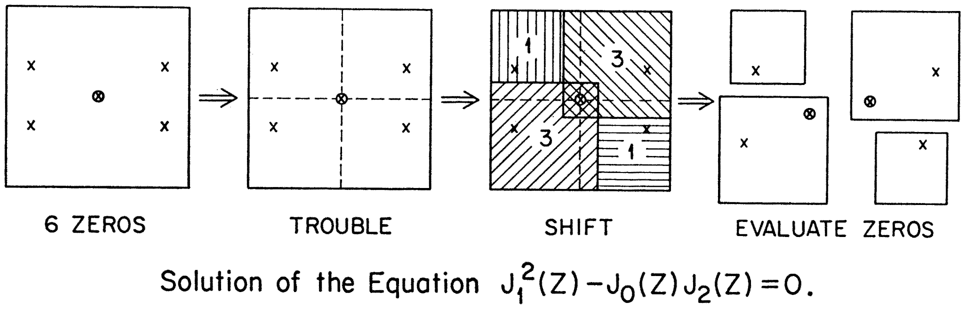


FIGURE 3. Steps in the location of the zeros of the function $J_1^2 - J_0J_2$. The original region is a square of side $2R = 12$, centered at the origin. The position of the zeros is marked with a cross. That at the origin is a double zero.

After obtaining the equivalent polynomials $p(z)$ the routine calculated the zeros of these to be

$$\alpha_1 = 4.466298 , \qquad \alpha_2 = 1.46747037 ,$$

which agree with the zeros of $f(z)$ to the number of significant figures shown. In this example the convergence parameter ϵ for the integrals was set at 10^{-4}.

The Buffer Zone. The user initially specifies some region which he wishes to be searched for zeros; if this region is square it is most likely that the search will be limited to this region. Exceptions will occur if there is a zero close to the boundary of the square, when the region may be extended to avoid a difficult integration.

If the region is a circle of radius R, it is not true that the specified circle will be the only region searched. Fig. 1 shows that the process of subdividing the circle will cause a search over a region outside the required circle. The extent of this 'Buffer Zone' depends on the number of subdivisions made; that is, on the number of zeros in the region and its extension. If there are a large number of zeros, the region actually searched may include points up to a distance $235R/172 \simeq 10R/7$ from the center of the required circle. This feature of the search routine must be kept in mind. For the methods described here to be valid, the function must be analytic over the entire region searched, which may exceed the original region.

4. Comparison with Lehmer's Method. The algorithm presented in this paper may be applied to an arbitrary analytic function. However, an inevitable question is: if it is given a polynomial, how does it compare with existing methods? In particular, a comparison with Lehmer's method is of interest. We first look at the relative speed of the methods. Since our algorithm spends most of its time evaluating $f(z)$ for the numerical integration, its speed is determined by the number of evaluations needed to attain a given accuracy ϵ in these integrations. We saw in Section 2 that the dependence of ϵ on the number of function evaluations n has the form

$$\epsilon \sim \epsilon_0 A^n\,, \qquad |A| < 1\,. \tag{4.1}$$

That is, the process converges *linearly* in the number n.

This is the same order of convergence as obtained in Lehmer's method, which converges linearly in the number of major steps. However, for a polynomial of degree ν, each major step in Lehmer's method involves the tabulation of a Sturm sequence of $O(\nu)$ steps, taking of order ν^2 operations in all. A single function evaluation takes only of order ν operations, and hence the time taken by the method of this paper increases less rapidly with the degree of the polynomial than does Lehmer's method, although for polynomials of low degree it is considerably slower. Such low-degree examples are, of course, rather unkind to the method, since if it is allowed to treat all the zeros at once, it will spend its time re-constituting the original polynomial coefficients and then hand these on to a polynomial-solving routine!

On the other hand, it may often be convenient to use the routine for polynomials of high degree ($\gtrsim 12$). Two difficulties can arise with polynomials of high degree. The first difficulty lies in the polynomial itself, which may be very ill conditioned with respect to its coefficients. These must then be carried to multiple precision in order to define the roots adequately, independently of the method of solving the polynomial. The second difficulty is the possible occurrence, in a given method of solution, of extremely large or extremely small numbers which may underflow or overflow the machine word. This difficulty exists in Lehmer's method; for any given choice of precision in the routine, it is possible to construct polynomials for which this will occur. The routine then fails completely to give a solution. This difficulty does *not* occur with the method given here, for which all numbers generated are automatically of a reasonable size.

We illustrate these two different points with the following example of a badly ill-conditioned polynomial of degree 16, taken from Wilkinson [9] and Olver [7].

$$\begin{aligned} f(z) = {} & 12501\,62561\,x^{16} + 3854\,55882\,x^{15} + 8459\,47696\,x^{14} + 2407\,75148\,x^{13} \\ & + 2479\,26664\,x^{12} + 642\,49356\,x^{11} + 410\,18752\,x^{10} + 94\,90840\,x^{9} \\ & + 41\,78260\,x^{8} + 8\,37860\,x^{7} + 2\,67232\,x^{6} + 44184\,x^{5} \\ & + 10416\,x^{4} + 1288\,x^{3} + 224\,x^{2} + 16\,x + 2 \end{aligned} \tag{4.2}$$

Table 1 shows the result when this polynomial is solved to varying degrees of precision, using the method of this paper, but with no final iteration. The results agree with those given by Wilkinson, to the precision stated by him. We see that, indeed it is necessary to define the polynomial to double precision (about 24 decimal places on our machine) before the more closely spaced roots become stabilized. However, an accuracy of about 9 significant figures is obtained using only single-precision working in the routine itself. We gave the polynomial also to a single-precision version of Lehmer's method, which was available. This routine failed to solve the equation, due to the overflow difficulty mentioned above, although we used it as

TABLE 1

Col.	(1) double precision result	(2)	(3)	(4)
1	−.13244 72469 90246 20179 ± .13600 55079 51377 63786i	9 + i9	10 + i11	10 + i9
2	−.01869 49953 44576 20767 ± .25034 56818 77088 4804 i	5 + i7	11 + i11	10 + i11
3	−.00032 09446 10861 38 ± .29258 37451 03366 82 i	5 + i4	11 + i11	11 + i11
4	−.00049 14535 99303 83 ± .30418 23930 25528 14 i	0 + i0	7 + i7	7 + i7
5	−.00014 26410 89728 97 ± .30861 21242 15863 54 i	—	10 + i11	10 + i10
6	−.00004 71311 10293 77 ± .31066 18478 80804 22 i	—	10 + i10	10 + i10
7	−.00001 48384 57209 34 ± .31169 63046 87558 09 i	—	12 + i11	12 + i11
8	−.00000 30529 83523 32 ± .31219 69683 754 i	—	10 + i9	10 + i9

TABLE 1. The zeros of the polynomial (4.2). Column (1) gives the zeros as obtained by a complete double-precision program based on the method of this paper. Columns (2), (3), and (4) refer to evaluations with a single-precision program, but with the function evaluation carried out in alternative ways. The number of decimal places obtained accurately is shown.
Column (2) single-precision evaluation of the polynomial.
Column (3) double-precision evaluation of the polynomial.
Column (4) single-precision evaluation of the equivalent analytic function (4.3).

TABLE 2

Θ	*Lower bound*	*Calculated zero*		*Calculated zero of derivative*	
$\pi/12$	18.26	19.14864645	16	23.896196	64
$\pi/6$	8.92	9.37328298	32	11.73753767	128
$\pi/4$	5.82	6.13876304	16	7.7039936	128
$\pi/3$	4.29	4.54215142	32	5.7039986	128
$5\pi/12$	3.48	3.60417681	32	4.5203892	128
$\pi/2$	2.81	2.99999999965	32	3.7488827	256

TABLE 2. The smallest zeros of the function $P_v^{-2}(\cos\theta)$ and of $(d/dv)P_v^{-2}(\cos\theta)$ for various values of θ. Column 2 gives the lower bound obtained by Low [4]. Column (3) gives the zeros evaluated using the routine described here, while Column (5) gives the smallest zero of the first derivative using the method described in Section 7. Columns (4) and (6) give the number of function evaluations used by the program in each case.

the final polynomial-solving routine in our program (for which it had to handle polynomials of degree 5 or less) with every success.

Column (4) of the table is of special interest. The polynomial (4.2) was originally formed from the analytic function $f(z)$ given by Olver [7].

$$\begin{aligned} f(z) &= 2\lambda P(z) + 2Q(z)\,, \\ P(z) &= z(dz^2+1)^{n-1}\sinh n\gamma \operatorname{cosech}\gamma\,, \\ Q(z) &= (dz^2+1)^n\cosh n\gamma\,, \\ \cosh\gamma &= (dz^2+1)^{-1}\{(d+1/2)z^2+1\}\,, \end{aligned} \tag{4.3}$$

with parameters $n = 8$, $d = 10$, $\lambda = 1.0$.

Moreover, it is quite *well conditioned* with respect to these parameters, although the polynomial form is badly conditioned. Column (4) shows that we obtain the same accuracy in the zeros by evaluating (4.3) single precision, as by evaluating the polynomial double precision. This is to be expected from the condition numbers, which for the higher numbered roots are of order unity for (4.3) but 10^7 for (4.2). The ability to accept the more natural form (4.3) rather than the derived polynomial is a useful feature of our method; one is given the choice of presenting a function in the way which makes the zeros the least sensitive to the parameters.

We have demonstrated this ability in an even more striking, although rather artificial example. The polynomial of degree 20 having as zeros the integers from 1 to 20, has condition number of order 10^{10} for the larger roots (Wilkinson [9]) and hence is not represented at all even by a double-precision calculation. The (of course trivial) representation of this polynomial in terms of its linear factors was given to the single-precision program, which satisfactorily computed all 20 zeros to at least nine significant figures.

5. The Basic Subroutine; Method B.

The algorithm given in the previous section requires the evaluation of the single function $f'(z)/f(z)$ at a number of points, to produce the zeros of $f(z)$. In practice, of course, this will usually involve the separate evaluation of $f'(z)$ and $f(z)$; and it may happen that the calculation of $f'(z)$ is unduly difficult or takes very much longer than that of $f(z)$. In these circumstances it would be convenient to have an algorithm which required only function evaluation of $f(z)$. We have developed two such methods. The first is derived from Eq. (5.1) by integrating by parts, and involves the evaluation only of $\ln f(z)$; it is described in this section. An alternative method, involving a numerical approximation to $f'(z)$, is described in a subsequent section.

We start from Eq. (2.3), which we restate here

$$S_N \equiv \sum z_i^N = \frac{1}{2\pi i}\int_C z^N \frac{f'(z)}{f(z)}\,dz\,. \tag{5.1}$$

Since

$$\frac{f'(z)}{f(z)} = \frac{d}{dz}\ln f(z)\,, \qquad \arg f(z) \neq -\pi\,, \tag{5.2}$$

we see that we can replace the factor $f'(z)/f(z)$ by the derivative of $\ln f(z)$ and then integrate by parts. The initial difficulty in doing this is that $\ln z$ is not analytic within the circle C_r: $|z| = r$, but has a branch point at the origin. We therefore have to be able to keep track of the appropriate sheet on which $\ln f(z)$ lies as z passes round the contour C. The possible points of discontinuity in $\ln (z)$ occur where $\arg z = -\pi$ and the main effort in the following analysis goes into ways of numerically keeping track of these points. We define the principal value of $\ln (z)$ as:

$$\ln (r \exp (2\pi i t)) = \ln r + 2\pi i t\,, \qquad -\tfrac{1}{2} < t \leqq \tfrac{1}{2}\,,$$

and in numerical calculations we assume that the subroutine for evaluating the logarithm of a complex number calculates this quantity. In what follows, the function $\ln (z)$ refers to this principal value.

We assume that no zero of $f(z)$ lies on the circle $|z| = r$. If we define a new w complex plane, the mapping $w = f(z)$ maps the circle $|z| = r$ into a closed curve $\overline{C}$ in the w plane. This curve does not pass through the origin. We suppose $\overline{C}$ cuts the negative real axis at μ distinct points $q_1, q_2, \cdots, q_\mu$. We assign to each point an index $\bar{\sigma}_j$ which takes the value $+1$ or -1 according as $\bar{C}$ cuts the axis in a positive (counter clockwise) or in a negative direction. Thus, if the points p_j in the z plane correspond to q_j in the w plane

$$q_j = f(p_j)\,, \qquad \arg q_j = -\pi\,, \tag{5.5}$$

and for all $0 < \epsilon_j \leqq h_j$, $\arg f(p_j \exp(i\epsilon_j)) \neq -\pi$

$$\bar{\sigma}_j = -\operatorname{sgn}(\operatorname{Im} f(p_j \exp(ih_j)))\,. \tag{5.6}$$

In order to carry out the integration we divide the contour integral C into sections $p_1p_2, p_2p_3, \cdots$. Within each of these sections $\arg f(z) \neq -\pi$ and so $\ln f(z)$ is continuous. Thus

$$\begin{aligned}\frac{1}{2\pi i}\int_{C_r} z^N \frac{f'(z)}{f(z)}\,dz &= \sum_{j=1}^{\mu} \frac{1}{2\pi i}\int_{p_j}^{p_{j+1}} z^N \frac{f'(z)}{f(z)}\,dz \\ &= \sum_{j=1}^{\mu} \frac{1}{2\pi i} z^N \ln f(z)\Big|_{p_j}^{\overset{(-)}{p_{j+1}}}{}^{\!\!(+)} - \frac{1}{2\pi i}\int_{C_r} N z^{N-1} \ln f(z)\,dz\,, \qquad N \geqq 1\,,\end{aligned} \tag{5.7}$$

where the sign on p_j indicates that the appropriate limit is used. The first term on the right-hand side may be written

$$\sum_{j=1}^{\mu} \bar{\sigma}_j (p_j)^N\,.$$

In the case in which $N = 0$ we find

$$\nu = \frac{1}{2\pi i}\int_{C_r} \frac{f'(z)}{f(z)}\,dz = \sum_{j=1}^{\mu} \bar{\sigma}_j\,. \tag{5.8}$$

This corresponds to the well-known result that the number of zeros in C_r is the number of times $\bar{C}$ encircles the origin.

For $N > 0$, the expression

$$S_N = \sum_{j=1}^{\mu} \bar{\sigma}_j (p_j)^N - \frac{1}{2\pi i}\int_{C_r} N z^{N-1} \ln f(z)\,dz \tag{5.9}$$

is clearly quite unsuitable as a basis for discretization. The locations of p_j would need to be determined and the integral of a function with several discontinuities would have to be evaluated. To obviate this difficulty we obtain an expression for $\sum \bar{\sigma}_j p_j^N$ in the form of an integral whose discontinuities exactly eliminate those in the second term. We set

$$p_j = r \exp(2\pi i T_j) \tag{5.10}$$

and we define a function $[t]$ by

$$\begin{aligned}[t] &= t\,, \qquad -\tfrac{1}{2} < t \leqq \tfrac{1}{2}\,, \\ [t+1] &= [t]\,.\end{aligned} \tag{5.11}$$

An elementary calculation shows that

$$(p_j)^N = 2\pi i N \int_0^1 (r \exp(2\pi i t))^N [t - T_j - \tfrac{1}{2}]\,dt\,. \tag{5.12}$$

Thus we find from (5.9) that

$$S_N = \int_0^1 N(r \exp (2\pi i t))^N \phi(t) dt \tag{5.13}$$

where

$$\phi(t) = \sum_j \bar{\sigma}_j 2\pi i [t - T_j - \tfrac{1}{2}] - \ln f(r \exp (2\pi i t)) . \tag{5.14}$$

We note that $\phi(t)$ is a continuous function of t with period 1. Hence if the values of T_j were known exactly, the integral would be a priori quite as straightforward as the original (2.7), the principal difference in the amount of work being that $\ln [f(z)]$ is calculated instead of $f'(z)$. We show below that it is not necessary to know T_j exactly to obtain accurate numerical results. The information that T_j is chosen so that $\phi(t)$ is continuous is sufficient.

Discretization. The discretization may be accomplished using an n-point trapezoidal rule

$$R^{[n,1]}\phi(t) = \frac{1}{n} \sum_{k=1}^{n} \phi\left(\frac{k}{n}\right) . \tag{5.15}$$

We first define what we term a "*sufficiently fine discretization.*" We require the computer to determine the number of times the curve $\bar{C}: w = f(z),\ |z| = r$ cuts the negative real w axis, and it has at its disposal only a finite number of points on this curve. Thus if the curve behaves in an unexpected manner between two successive points, the routine may fail to indicate this. We define a sufficiently fine discretization as follows:

(i) The curve cuts the negative real axis only once or not at all between any pair of neighboring points.

(ii) The straight line segment connecting these points cuts the negative real axis the same number of times as the corresponding section of the curve.

The analytic results obtained below all depend on the discretization being sufficiently fine.

We suppose that $\arg z$ is defined so that

$$-\pi < \arg z \leqq \pi . \tag{5.16}$$

For a particular discretization we have available the values of $f(r \exp (2\pi i t_j))$ where $t_j = j/n$ and $j = 1, 2, \cdots, n$. We may assign to each point a value of σ defined as follows:

$$\begin{aligned} \sigma_j &= +1 , & \arg f_j - \arg f_{j-1} &\leqq -\pi , \\ \sigma_j &= 0 , & -\pi < \arg f_j - \arg f_{j-1} &\leqq \pi , \\ \sigma_j &= -1 , & \pi < \arg f_j - \arg f_{j-1} & . \end{aligned} \tag{5.17}$$

This has the effect that, if T_j lies between t_k and t_{k+1}, σ_{k+1} is set equal to $\bar{\sigma}_j$. In this respect the discretization transfers properties pertaining to points at which $\bar{C}$ cuts the negative real axis to the next calculated point on the curve. We now consider the trapezoidal sum

$$R^{[n,1]}[t - T_j - \tfrac{1}{2}] \exp (2\pi i N t) ,$$

where

(5.18) $$t_k < T_j < t_{k+1}$$
and

$$n > N \geqq 0\,.$$

We find

(5.19) $$R^{[n,1]}([t - T_j - \tfrac{1}{2}] \exp(2\pi iNt)) = \frac{1}{n}\sum_{l=0}^{k} (t_l - T_j + \tfrac{1}{2}) \exp(2\pi iNt_l) + \frac{1}{n}\sum_{l=k+1}^{n-1} (t_l - T_j - \tfrac{1}{2}) \exp(2\pi iNt_l)$$

where

(5.20) $$t_l = l/n\,.$$

Since

(5.21) $$\frac{1}{n}\sum_{l=0}^{n-1} \exp(2\pi iNt_l) = 0\,, \qquad n > N\,,$$

it follows that the terms in T_j in expression (5.19) eliminate each other. The value of $R^{[n,1]}[t - T_j - \frac{1}{2}] \exp(2\pi iNt)$ does not depend on the exact value of T_j, but only depends on k, that is on the interval in which T_j is situated. A simple calculation gives

(5.22) $$R^{[n,1]}[t - T_j - \tfrac{1}{2}] \exp(2\pi iNt) = \frac{\exp(2\pi iNt_{k+1})}{n(\exp(2\pi iN/n) - 1)}\,,$$

where T_j lies in the interval $t_k < T_j \leqq t_{k+1}$ and $n > N$.

The calculation of S_N may be arranged as follows:

(5.23) $$S_N \simeq NR^{[n,1]}(\phi(t) \exp(2\pi iNt)r^N) = T_N^{(n)} - L_N^{(n)}$$

where

(5.24) $$L_N^{(n)} = NR^{[n,1]}((r \exp(2\pi it))^N \ln f(r \exp(2\pi it)))$$

and

(5.25) $$T_N^{(n)} = N \sum_j 2\pi iR^{[n,1]}\bar{\sigma}_j[t - T_j - \tfrac{1}{2}](r \exp(2\pi it))^N\,.$$

The advantage of doing this is that $T_N^{(n)}$ need not be calculated as a trapezoidal sum. We note that in the discretization a nonzero value of $\bar{\sigma}_j$ is assigned to the next calculated point; moreover, replacing T_j by the next calculated point does not alter the value of the trapezoidal sum. Consequently

(5.26) $$T_N^{(n)} = N \sum_{k=1}^{n} 2\pi i\sigma_k \exp(2\pi iNt_k)/n(\exp(2\pi iN/n) - 1)\,.$$

This relatively simple analytic expression for $T_N^{(n)}$ is clearly much more convenient than (5.25).

Programming Details. The final results (5.24) and (5.26) give an alternative way of evaluating the power sums S_N, and a routine based on these equations can replace the basic subroutine described in Section 2. The user need not then evaluate $f'(z)$ at all. In order to evaluate the σ_i it is necessary to tabulate and store the argument θ_j of the function evaluations $f_j(z) = (r_j, \theta_j)$. Further, to allow for the possibility that during the evaluation of the S_N, the apparent number of zeros (and hence the required number of powers S_N) changes, it is necessary to store also the σ_j and r_j. This possibility arises because in order to obtain the correct number of zeros it is necessary for the routine to be able to recognize a "sufficiently fine discretization." Unless some care is taken, it is possible to be misled, since the apparent number of zeros in the region as obtained from (5.8) is always an *integer*; then, convergence to the correct integer is not smooth as the number of points used increases. This contrasts with the situation described by (2.1) for Method A, where the apparent number of zeros *approximates* an integer, and in general converges smoothly towards an integer. There are two simple ways of avoiding this difficulty.

(a) We calculate S_1 and observe its behavior.

(b) We accept three consecutive equal evaluations of the number of zeros.

We have used method (b) successfully as giving a sufficiently good indication of the number of unknown zeros. If this is less than M (see Section 2) the coefficients S_N are evaluated; if during this evaluation the apparent number of zeros changes, the evaluations are adjusted accordingly.

6. The Basic Subroutine; Method C.

We describe in this section an alternative method which also avoids the necessity for the user to calculate $f'(z)$. This method is identical with that given in Section 2, except that $f'(z)$ is replaced by an approximation derived only from function evaluations of $f(z)$. The function evaluations used for this are the same as those required to approximate the contour integrals, and hence the method is comparatively economical of points. Moreover, it has the advantage that it can be very simply extended to locate the zeros of any given derivative $f^{(p)}(z)$ of $f(z)$, again using only function evaluations of $f(z)$. This extension is described briefly in Section 7.

The numerical approximation of derivatives of analytic functions when function evaluations in the complex plane are available, is described in Paper B, Section 2. The sth Taylor coefficient a_s of $f(z)$ expanded about the point z_0, is given by

$$a_s \equiv \frac{f^{(s)}(z_0)}{s!} = \frac{1}{2\pi i}\int_C \frac{f(z)}{(z-z_0)^{s+1}}\,dz \tag{6.1}$$

where the closed contour C surrounds the point z_0. In (Lyness and Moler) [6] it was shown that methods of evaluating a_s based on Eq. (6.1) converge very rapidly, and are free from the round-off errors normally associated with numerical differentiation. We proceed as follows. The evaluation of the contour integrals (2.2) in Section 2 requires $f'(z)$ at N regularly spaced points round the circle in question, which we may take to be centered at the origin. We approximate $f'(z)$ by a truncated Taylor series $\tilde{f}'(z)$

$$\tilde{f}'(z) = \sum_{j=0}^{N-1} j\tilde{a}_j^{(N)} z^{j-1}\,. \tag{6.2}$$

We obtain the approximation $\tilde{a}_j^{(N)}$ to the Taylor coefficients a_j using the N-point trapezoidal rule to evaluate the integrals in (6.1). In this way we need function evaluations only at a single set of points. Defining these function values by

$$f_{j/N} = f(r\omega_N^{\,j})\ ; \qquad \omega_N = \exp\,(2\pi i/N)\ , \tag{6.3}$$

the required formulas are

$$r^s\tilde{a}_s^{(N)} = \frac{1}{N}\sum_{j=1}^{N} \omega_N^{-sj} f_{j/N} \tag{6.4}$$

and

$$r\tilde{f}'^{(N)}_{j/N} = \sum_{s=0}^{N-1} sr^s\tilde{a}_s^{(N)}\omega_N^{j(s-1)} . \tag{6.5}$$

The trapezoidal rule (6.4) used for approximating a_s corresponds to the first term in the expansions given in Lyness and Moler. The truncation error of the approximation (6.5) for f' is considered in Paper B. This error stems from two sources —the truncation of the Taylor series at the Nth term, and the approximation used for the coefficients a_s which are retained. In Paper B it is shown that the total truncation error using N points has the form

$$|rf'(z) - r\tilde{f}'(z)^{(N)}| < \tilde{E}_N; \qquad |z| \leqq r, \tag{6.6}$$

where

$$\tilde{E}_N \sim \text{const } N\rho^N \text{ as } |N| \to \infty \ ; \qquad 0 < |\rho| < 1 . \tag{6.7}$$

A more detailed bound is given in Paper B, Eq. (2.6). Eq. (6.7) shows that the accuracy of the approximation to the derivatives at each point has the same exponential dependence on N, as the truncation error involved in performing the final contour integrations using Eq. (2.6).

As in (2.7), the work may be arranged so that in proceeding from N to $2N$ points we need only perform operations with the added points. We define coefficients $b_j^{(N)}$ involving only these added points:

$$r^jb_j^{(N)} = \frac{1}{N}\sum_{k=1}^{N} \omega_{2N}^{-j(2k-1)} f_{(2k-1)/2N} , \qquad j = 1 \cdots N . \tag{6.8}$$

Then in terms of these coefficients we have

$$\begin{aligned} r^s\tilde{a}_s^{(2N)} &= \tfrac{1}{2}(r^s\tilde{a}_s^{(N)} + r^sb_s^{(N)}) , \\ r^{s+N}\tilde{a}_{s+N}^{(2N)} &= \tfrac{1}{2}(r^s\tilde{a}_s^{(N)} - r^sb_s^{(N)}) , \end{aligned} \qquad s = 0 \cdots N-1 , \tag{6.9}$$

and

$$\begin{aligned} rf'^{(2N)}_{t/2N} &= rf'^{(N)}_{t/N} + N\sum_{s=0}^{N-1} r^{s+N}\tilde{a}_{s+N}^{(2N)}\omega_{2N}^{t(s-1)} , && t \text{ even} , \\ &= \sum_{s=0}^{N-1} sr^sb_s^{(N)}\omega_{2N}^{t(s-1)} - N\sum_{s=0}^{N-1} r^{s+N}\tilde{a}_{s+N}^{(2N)}\omega_{2N}^{t(s-1)} , && t \text{ odd} . \end{aligned} \tag{6.10}$$

An Example. Both this method and that of the previous section are particularly easy to use, since the user has only to provide a routine for evaluating $f(z)$. We demonstrate their power by means of a simple example which was the subject of a recent paper (Low [4]). This was concerned with the evaluation of the smallest zeros of the associated Legendre function P_v^{-m} (cos θ) for fixed m and θ, as a function of v. These zeros are known to be all simple and real, and Low obtained bounds on the smallest zeros for $m = 2$ and a number of values of θ. A program based on the method of this paragraph was used to search for the smallest zeros, and gave the results in Table 2.

7. The Zeros of $f^{(p)}(z)$. Method C may be adapted in an obvious manner to obtain the zeros of the pth derivative $f^{(p)}(z)$ of a given analytic function $f(z)$. The calculation of the Taylor coefficients $\tilde{a}_j^{(N)}$ takes place as before, but $f'(z)/f(z)$ is replaced whenever it occurs by the approximation to $f^{(p+1)}(z)/f^{(p)}(z)$ based on these Taylor coefficients, using

$$f^{(p)}(z) \simeq \tilde{f}^{(p)}(z) = \sum_{j=0}^{N-1} j(j-1) \cdots (j-p+1)\tilde{a}_j^{(N)} z^{j-p}. \tag{7.1}$$

The truncation error involved in Eq. (7.1) is considered in Paper B. There it is shown that using N function evaluations

$$r^p |f^{(p)}(z) - \tilde{f}^{(p)}(z)| < \tilde{E}_N^{(p)}; \qquad |z| \leqq r, \tag{7.2}$$

where

$$\tilde{E}_N^{(p)} \sim \text{const } N^p \rho^N, \qquad N \to \infty;\ 0 < |\rho| < 1. \tag{7.3}$$

An Example. As an example of the use of this technique, we consider a problem due to W. L. Morris (private communication). The following function arises in the consideration of algorithms for solving sets of ordinary differential equations:

$$f(z) = \frac{1 - 1.005e^{-z} + 0.525e^{-2z} - 0.475e^{-3z} - 0.045e^{-4z}}{2.27e^{-z} - 2.19e^{-2z} + 1.86e^{-3z} - 0.38e^{-4z}} \tag{7.4}$$

for which the characteristic of interest is the location of the zero of smallest modulus of $f'(z)$. A program based on the method described here was used to search for this smallest zero, and located it to be

$$z_0 = 0.3430042 + i1.0339458. \tag{7.5}$$

This example is interesting since the function (7.4) has a pole at z_p:

$$z_p = -0.2275004 \pm i1.1152220$$

with modulus only slightly greater than that of z_0. This serves to emphasize the remarks made in Section 3. The methods given in this paper require a knowledge that the function $f(z)$ is analytic in the region searched.

8. The Final Iteration. A feature of almost any global method for locating zeros is that it is uneconomic to find the zeros to high accuracy. Rather, these should be located to a relatively low accuracy first and then refined by an iterative technique. This is also true of the method given here, since it is time consuming to evaluate the contour integrals to high accuracy. We have therefore allowed for a final iterative stage in all of the methods.

For Method B we have used the secant rule, while for methods A and C we have used Newton iteration, with the required derivative in Method C being obtained from the Taylor expansion (6.2). For all of these methods the final accuracy attained depends only on the accuracy with which the function can be evaluated near the zero. We have also used Newton's method to iterate the pth derivative, obtaining both $f^{(p)}(z)$ and $f^{(p+1)}(z)$ from the Taylor expansion. In this case, the attainable accuracy depends on the accuracy of the truncated Taylor series.

9. Discussion of the Methods. The methods given in this paper give a practical means of finding the zeros of an analytic function or of its derivatives, in a given region. Each method has its own advantages and disadvantages, and it does not seem possible to single out one as being always preferable. However, we attempt here to make some comments on their relative virtues.

These methods give the user the choice of searching in squares or in circles. The search routine for squares is rather more efficient than for circles, yielding much less overlap (usually, none) when a subdivision occurs. But the integration rules given are more efficient for circles than for squares. For circles, we have presented three methods:

Method A. Explicit evaluation of $f'(z)$.

Method B. Evaluation of $\ln(f(z))$.

Method C. Evaluation of $f'(z)$ by interpolation.

Of these three methods, A is by far the most economical of storage space and is about as fast as Method B. If no derivatives are available, then in general Method B is faster than Method C, since for large numbers of points the evaluation of the Taylor coefficients and then the derivatives in Method C is time consuming. However, Method C has the advantage that it is readily adapted to find the zeros of the derivatives or the zeros of several derivatives at the same time.

Acknowledgment. We are indebted to Dr. W. B. Gragg for several stimulating discussions on these topics.

Oak Ridge National Laboratory
Oak Ridge, Tennessee

1. P. Henrici & B. O. Watkins, "Finding zeros of a polynomial by the Q–D algorithm," *Comm. ACM*, v. 8, 1965, pp. 570–574. MR **31** #4172.

2. D. H. Lehmer, "The Graeffe process as applied to power series," *MTAC*, v. 1, 1945, pp. 377–383. MR **7**, 84.

3. D. H. Lehmer, "A machine method for solving polynomial equations," *J. Assoc. Comput. Mach.*, v. 8, 1961, pp. 151–162.

4. R. D. Low, "On the first positive zero of $P_{\gamma-1/2}^{-m}(\cos\theta)$, considered as a function of γ," *Math. Comp.*, v. 20, 1966, pp. 421–424.

5. J. N. Lyness & L. M. Delves, "On numerical contour integration round a closed contour," *Math. Comp.*, v. 21, 1967, pp. 561–577.

6. J. N. Lyness & C. B. Moler, "Numerical differentiation of analytic functions," *J. SIAM Numer. Anal.*, v. 4, 1967, pp. 202–210.

7. F. W. J. Olver, "The evaluation of zeros of high-degree polynomials," *Phil. Trans. Roy. Soc.* A, v. 244, 1952, pp. 385–415. MR **14**, 209.

8. H. Rutishauser, "Der Quotienten-Differenzen-Algorithmus," *Z. Angew. Math. Phys.*, v. 5, 1954, pp. 233–251. MR **16**, 176.

9. J. H. Wilkinson, *Rounding Errors in Algebraic Processes*, Prentice-Hall, Englewood Cliffs, N. J., 1963. MR **28** #4661.